SRH Hochschule Heidelberg

Masters Thesis

A Layout-Aware Metadata Extraction Framework for Digital Libraries

Author:
Sanath Vijay Haritsa
11038004

Supervisor:
Prof. Dr.-Ing. Binh Vu
Sina Mehraeen

A thesis submitted in fulfilment of the requirements
for the degree of MSc Applied Data Science and Analytics

in the

School of Information, Media and Design

September 2025

Declaration of Authorship

I hereby declare that my herewith submitted paper is my own original work. I have written it independently without outside help and have not used any sources other than those indicated - in particular, no sources not named in the references.

I have appropriately indicated any direct quotations or passages taken from literature, and the use of intellectual property from other authors, by providing the necessary citations within the work. This applies equally to the sources used for text generation by Artificial Intelligence (AI).

I hereby declare that the paper was not previously presented to another examination board, and I also confirm that the PDF version of this paper is exactly identical in content to the hard copy. Signed:

_____

Date: _____

Acknowledgements

First of all, I would like to express my deep gratitude to Prof. Dr. Binh Vu and Sina Mehraeen, who have directly guided and supported me throughout the process of writing this thesis. Their dedication, extensive knowledge, and valuable suggestions helped me to complete this thesis in the best possible way.

Furthermore, I would also like to express my sincere gratitude to all the teachers in Applied Data Science and Analytics of SRH Hochschule Heidelberg who have provided me with the fundamental and in-depth knowledge in this subject throughout the learning process at school.

Additionally, I would also like to thank my family and friends who have always encouraged, supported and provided the best conditions for me during my study and research period.

Finally, I would like to thank all the individuals and organizations that provided me with documents, data and the necessary support to complete this thesis.

Abstract

This thesis investigates the problem of reliable metadata extraction from scholarly PDF documents, a task complicated by diverse layouts, structural variability, and the absence of semantic markup in PDFs. To address these challenges, the study systematically benchmarks open-source PDF parsing tools and compares two complementary paradigms for metadata extraction: layout-aware systems and instruction-tuned language models. The research introduces a two benchmark frameworks; one consisting of page-level free text ground truth created from DocBank annotations and the other is a heterogeneous collection of publisher PDFs. A novel multi-model consensus and adjudication pipeline was developed to construct high-quality gold-standard metadata, leveraging multiple small models with selective adjudication by a stronger model and schema validation to ensure consistency. Empirical results reveal that layout-aware tools such as GROBID excel on structured fields like titles and keywords, while small language models—particularly Qwen-3B—achieve competitive or superior accuracy on more variable fields such as affiliations and email addresses, albeit with higher computational demands. Large generative models deliver the strongest results overall but face scalability constraints. The findings highlight the complementary strengths of both approaches, suggesting that hybrid strategies can balance efficiency, accuracy, and adaptability. This work contributes new benchmarks, methodological innovations, and comparative insights that advance the state of scholarly document processing and inform the design of future metadata extraction systems.

Keywords: Metadata extraction, scholarly documents, PDF parsing, layout-aware models, large language models, GROBID, small language models, transformer models, consensus annotation, schema validation, digital libraries, information retrieval.

Contents

List of Figures

List of Tables

Abbreviations

IR	Information Retrieval
LLM	Large Language Model
SLM	Small Language Model
DOI	Digital Object Identifier
CRF	Conditional Random Fields
CER	Character Error Rate
WER	Word Error Rate
OCR	Optical Character Recognition
BLEU	Bilingual Evaluation Understudy
MT	Machine Translation
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
HMM	Hidden Markov Models
LSA	Latent Semantic Analysis
SVD	Singular Value Decomposition
CBOW	Continuous Bag of Words
NLP	Natural Language Processing
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory networks
RAG	Retrieval-Augmented Generation
RLHF	Reinforcement Learning from Human Feedback
DNN	Deep Neural Network
NLP	Natural Language Processing
CV	Computer Vision
GROBID	GeneRation Of BIbliographic Data
LAME	Layout-Aware Metadata Extraction
TEI	Text Encoding Initiative
XML	Extensible Markup Language
YOLO	You Only Look Once

Chapter 1. Introduction

Academic publications serve as the primary medium for disseminating scientific knowledge, typically appearing as journal articles, conference proceedings, or technical reports. These works are distributed through a variety of publishers, including IEEE, Springer, ACM, Elsevier, and many open-access platforms, each adhering to distinct formatting conventions. While some outlets adopt highly structured formats such as XML or HTML for digital archives, the majority of scholarly communication is disseminated in PDF. This format ensures fidelity in human-readable presentation but offers little semantic structure, leading to substantial variability in how bibliographic metadata is presented across venues.

This thesis addresses these challenges by formulating metadata extraction as a comparative problem between two distinct paradigms: layout-aware systems that exploit structural and visual cues within documents, and small-scale language models that rely on general-purpose representation learning. Through systematic benchmarking of PDF parsers, construction of ground-truth resources, and evaluation of both extraction strategies, the study seeks to identify approaches that balance accuracy, robustness, and computational efficiency. The goal is to provide actionable insights for real-world deployment in digital libraries and indexing platforms while contributing to the academic discourse on metadata extraction.

1.1 Motivation

Academic publications serve a dual purpose: they present novel research contributions and simultaneously carry a comprehensive set of bibliographic metadata that is essential for the organization and dissemination of scholarly knowledge. This metadata includes critical information such as the title, list of authors, author affiliations, email addresses, keywords, abstract, publication date, publisher, and Digital Object Identifier (DOI). Such metadata forms the backbone of the entire scholarly communication ecosystem, supporting a wide array of functions including indexing in digital libraries, facilitating article discovery, enabling citation tracking and analysis, powering bibliometric studies, enhancing recommendation systems, and constructing large-scale knowledge graphs (47, 84, 97). The quality and accuracy of this metadata are therefore directly linked to the visibility, accessibility, and impact of research outputs; well-structured and reliable metadata ensures that research is easily discoverable and correctly attributed. Conversely, poor or inconsistent metadata can propagate errors across interconnected systems, leading to fragmented scholarly records, misattribution, and decreased trustworthiness of digital repositories, ultimately hindering the progress of scientific communication (97).

The predominance of the Portable Document Format (PDF) in scientific publishing exacerbates these challenges. Unlike structured digital formats such as XML or HTML, which explicitly encode semantic boundaries and hierarchical relationships within the document, PDFs are designed primarily for human visual consumption. They preserve the visual layout and formatting intended by publishers, often at the expense of explicit semantic information. Consequently, metadata embedded within PDFs is frequently represented in visually distinct regions—titles may appear in bold and centered, author names may be accompanied by superscripted affiliations, and email addresses are often located in footnotes or sidebars—making automated extraction a complex task. This visual diversity and lack of semantic markup complicate programmatic interpretation, requiring sophisticated algorithms to accurately identify and extract relevant metadata fields from unstructured layouts (8, 50).

The challenge of reliable extraction is further intensified by the scale and heterogeneity of modern research outputs. Leading digital repositories such as arXiv, PubMed Central, and SpringerLink collectively host millions of articles originating from a multitude of publishers, each adhering to different formatting standards and evolving guidelines. Without robust, automated extraction methods, maintaining interoperability across these diverse sources would be practically impossible. Ensuring that metadata remains complete, accurate, and consistent across large-scale, multi-source collections is critical for supporting advanced search, data mining, and analytics tasks. Therefore, advancing the state of metadata extraction is not merely a technical pursuit but a fundamental requirement for preserving the integrity, scalability, and usability of scholarly knowledge infrastructures in the digital age.

1.2 Problem Statement

Despite its importance, accurate metadata extraction from scholarly PDFs remains an unsolved problem due to three main challenges: *structural variability, noisy encodings, and limitations of existing extraction paradigms.*

First, structural variability stems from diverse publishing workflows. Articles may appear in single- or multi-column layouts, interleave metadata with footnotes or headers, and vary in ordering of bibliographic fields. Some publishers present affiliations prominently, while others distribute them sparsely across the page. This heterogeneity means that metadata cannot be reliably located using uniform rules (55, 88, 113).

Second, the underlying PDF encoding compounds these difficulties. Unlike structured formats, PDFs lack semantic tags; instead, they encode positions, fonts, and glyphs. Variations in font rendering, ligatures, and character encodings frequently mislead parsers, causing tokens to be split, reordered, or dropped (8, 50). Text extraction tools, although improving, still produce noisy or disordered outputs—especially for scanned or low-quality PDFs—which hampers downstream metadata recognition tasks (8, 66).

Third, existing extraction paradigms exhibit fundamental limitations. Rule-based and template-driven systems achieve precision for specific formats but are brittle when exposed to new layouts, demanding constant manual updates. Classical machine learning models improve adaptability but rely heavily on annotated corpora that are expensive to construct at scale. Deep neural approaches focusing on textual sequences often disregard layout cues, while vision-based methods capture layout but incur additional computational cost due to image processing and multimodal fusion (6, 58, 102, 111).

The result is a persistent gap between the theoretical need for scalable, accurate metadata extraction and the practical shortcomings of available tools. Addressing this gap requires rethinking how structural cues and semantic reasoning are balanced. This thesis tackles the problem by comparatively evaluating two complementary paradigms: *layout-aware systems*, which leverage structural features and spatial positioning, and *language model–based approaches*, which exploit contextual reasoning to infer metadata fields from unstructured text. In addition, because reliable metadata extraction depends on the quality of raw text extracted from PDFs, this work systematically benchmarks parsing libraries to identify which tools best preserve character fidelity and logical reading order (8, 66). Together, these investigations aim to clarify the trade-offs between paradigms and propose strategies for practical, real-world deployment in digital libraries and academic indexing platforms.

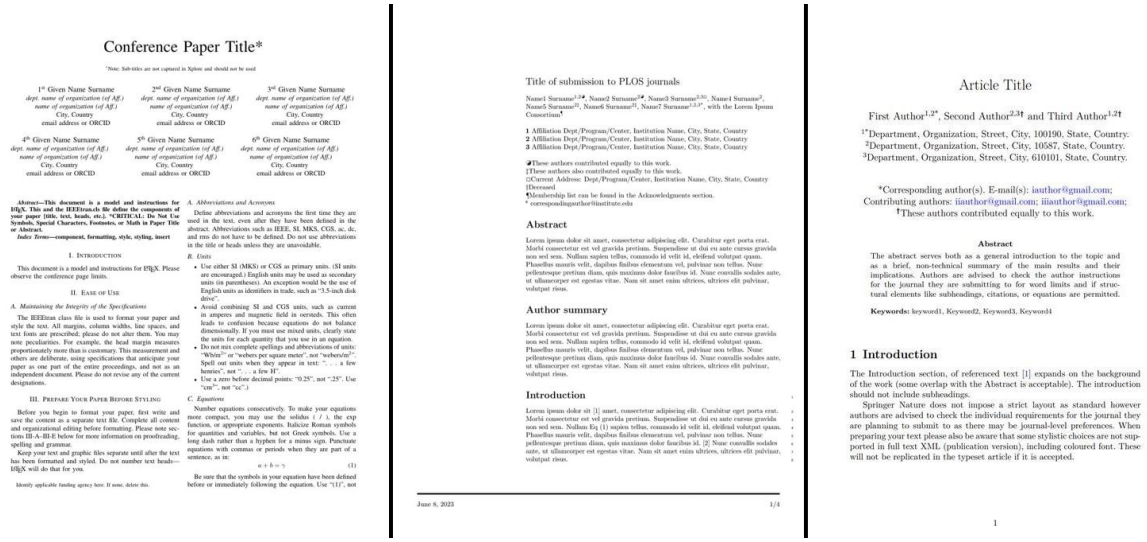

FIGURE 1.1: Manuscript examples: IEEE, PLOS ONE, Springer

1.3 Research Questions

The investigation is guided by the following research questions:

R1 **How accurately and robustly can existing open-source, layout-aware extraction systems identify bibliographic metadata from PDFs across diverse formatting styles, while remaining computationally efficient for large-scale deployment?**

This question aims to evaluate the performance of current open-source systems that leverage layout-aware techniques—methods that analyze structural and visual cues such as text coordinates, font styles, font sizes, column boundaries, and spatial arrangements. The focus is on understanding whether these systems can reliably extract accurate bibliographic metadata across a wide variety of document formats, publishers, and layout styles without the need for retraining or extensive customization. An essential aspect is assessing their robustness in handling differences such as multi-column layouts, varying font conventions, or non-standard formatting encountered in real-world publications. Additionally, since large-scale digital repositories process millions of documents, the question emphasizes the importance of computational efficiency: Can these systems perform fast enough to be integrated into production environments without excessive resource consumption? The investigation considers trade-offs between extraction accuracy, robustness to diverse layouts, and processing speed, aiming to identify solutions that are both effective and scalable.

R2 **Which PDF text extraction strategies provide the most complete, accurate, and logically ordered raw text as input for downstream metadata recognition?** Since the quality of the subsequent metadata extraction heavily depends on the initial raw text obtained from PDFs, this question focuses on comparing different text extraction approaches. It examines various parsing strategies—such as line-by-line extraction, block-based parsing, or more sophisticated algorithms—that aim to preserve the logical reading order, textual integrity, and completeness of the extracted content. The goal is to identify methods that minimize errors like broken sentences, misplaced line breaks, or missing sections, which can significantly impair the performance of downstream tasks. The evaluation considers how well each approach handles complex layouts, multi-column formats, embedded figures, footnotes, and other challenges inherent to scholarly PDFs. Ultimately, the research seeks to determine which extraction strategies best serve as reliable, high-quality inputs for automated metadata recognition pipelines.

R3 **To what extent can small-scale, general-purpose language models extract structured bibliographic metadata from unstructured text, and how do they compare with layout-aware systems in terms of accuracy and efficiency?** This question explores the potential of lightweight, general-purpose language models, such as smaller transformer-based models or sequence classifiers, that are not specifically trained for metadata extraction but possess broad language understanding. It investigates whether these models can be effectively adapted, for example through prompt engineering, to identify and extract structured metadata fields (like authors, titles, affiliations) from unstructured or semi-structured text extracted from PDFs. A key aspect is comparing their performance against dedicated layout-aware systems: Are they capable of achieving comparable accuracy? How do they perform in terms of processing speed, resource

consumption, and ease of deployment? The investigation aims to understand the trade-offs involved, especially in scenarios where deploying large, resource-intensive models is impractical, and to assess whether small-scale models can offer a viable, cost-effective alternative for metadata extraction tasks.

R4 **What are the relative advantages and limitations of layout-aware approaches versus language model–based methods for metadata extraction from scholarly publications?**

This question aims to compare and contrast the two primary paradigms for metadata extraction. It evaluates the strengths of layout-aware approaches, such as their ability to leverage explicit structural cues and their effectiveness across diverse formatting styles, against the advantages of language model–based methods, including their flexibility and potential to infer metadata from unstructured text without strict dependence on layout cues. The analysis also considers limitations—for instance, the potential for reduced accuracy and rigidity of layout-aware systems versus the computational demands or lack of context understanding in language models. The goal is to identify the conditions under which each approach excels or falls short, such as the complexity of the document layout, the availability of high-quality training data, or resource constraints. Ultimately, this comparison aims to guide researchers and practitioners in selecting or combining methods that best suit specific use cases, fostering the development of more robust, efficient, and adaptable metadata extraction pipelines for scholarly publications.

1.4 Approach

The overall approach of this thesis is structured as a comparative framework that evaluates metadata extraction from scholarly PDFs across two paradigms: *layout-aware systems* and *small-scale language models (SLMs)*. The methodology follows a pipeline consisting of literature grounding, ground truth construction, system design, and comparative evaluation.

Literature Review and Problem Framing

A comprehensive review of prior work in PDF analysis, document segmentation, and metadata extraction established the theoretical foundation. This step clarified the limitations of rule-based, vision-based, and deep learning systems, while motivating the need to systematically compare layout-aware baselines with lightweight, instruction-tuned language models.

Ground Truth for Parsing

To ensure fair evaluation of PDF parsing tools, a high-quality reference dataset was created. A subset of 101 pages from the DocBank corpus was selected and enriched: token-level annotations were aggregated into section-level text and then reordered into logical reading sequences. This

produced a robust benchmark against which parsing accuracy, completeness, and order fidelity could be quantified.

Benchmarking of PDF Parsers

Five widely used open-source parsing tools (PyMuPDF, pypdfium2, pdfminer.six, PyPDF2, and pdfalto) were benchmarked using the constructed ground truth. Their outputs were assessed for character-level fidelity, word-level accuracy, and preservation of reading order. This stage established how the quality of low-level parsing propagates into downstream metadata extraction.

Gold Standard for Metadata Extraction

In parallel, a gold standard dataset was curated for evaluating metadata extraction itself. A diverse set of 61 born-digital scholarly PDFs was collected across multiple publishers. For each file, bibliographic fields (title, authors, affiliations, emails, publication date, publisher, DOI, keywords, and abstract) were manually curated with the assistance of reasoning-enabled LLMs in a voting and adjudication workflow. This produced a heterogeneous and reliable ground truth covering a wide range of layouts.

Design of the SLM Pipeline

A metadata extraction pipeline based on small-scale, general-purpose language models was developed. Using prompt engineering with schema-constrained outputs, the pipeline extracted structured, machine-readable metadata without any model retraining or fine-tuning. This approach emphasized adaptability and deployment simplicity.

Baseline and Model Comparison

For a fair comparison, an established layout-aware framework (GROBID) was applied to the same dataset, serving as the reference baseline. Multiple SLMs were then tested on identical inputs, and their performance was compared across dimensions such as accuracy, robustness to layout variation, and reliability of structured output.

Evaluation and Analysis

Both paradigms were evaluated against the gold standard metadata using field-specific accuracy metrics (Levenshtein distance, F1 score, and cosine similarity). In addition to correctness, runtime and memory consumption were measured to assess computational efficiency. The resulting analysis clarifies trade-offs between layout-aware and language model–based pipelines, providing insights into their relative strengths, limitations, and practical suitability for large-scale deployment.

1.5 Thesis Contributions

This thesis makes contributions along two complementary axes: advancing the academic understanding of metadata extraction methods and providing practical guidance for their deployment in real-world settings.

In terms of academic research, this work establishes a rigorous and reproducible evaluation framework for metadata extraction. A central component of this framework is the systematic benchmarking of various PDF parsing tools, including PyMuPDF, pypdfium2, pdfminer, PyPDF2, and pdfalto. The evaluation focuses on key metrics such as text fidelity, completeness, and the preservation of reading order, offering a detailed understanding of how low-level parsing quality influences the reliability of downstream metadata systems. Additionally, the thesis introduces two high-quality ground-truth datasets: one is a page-level, free-text benchmark derived from DocBank designed specifically for parser evaluation, and the other is a manually curated collection of 61 heterogeneous scholarly publications. The latter was augmented by model-assisted consensus validation, ensuring accuracy and robustness. These datasets serve as foundational resources that support reproducible and comparable research in the field. Moreover, this work investigates the potential of schema-constrained small-scale language models (SLMs) for structured metadata extraction. Through carefully engineered prompts and constrained response formats, it demonstrates that general-purpose models can achieve effective structured data extraction without the need for retraining or fine-tuning. Complementing these efforts, the evaluation framework balances field-level accuracy with considerations of computational efficiency and robustness. This dual focus highlights important trade-offs between resource consumption, inference time, and extraction quality, thus addressing a notable gap in prior research that often prioritized accuracy at the expense of deployability.

On the practical side, the thesis provides valuable insights for practitioners seeking effective metadata extraction strategies. The comparative results inform decision-making by clarifying when to favor layout-aware tools like GROBID versus lightweight language model pipelines, depending on specific accuracy, efficiency, and resource constraints. The demonstration that a training-free language model pipeline can achieve competitive extraction quality without annotated data or fine-tuning offers a cost-effective and accessible solution, especially for environments with limited resources. Testing across diverse publishers and document layouts underscores the generalization potential of both paradigms, offering actionable insights for digital libraries, open-access repositories, and bibliometric platforms that aim to maintain consistent metadata at scale. Furthermore, the analysis of runtime and memory profiles delineates the scalability limits of various systems, providing essential guidance for their integration into production environments where throughput and stability are as critical as accuracy.

Together, these contributions advance both the scientific discourse on metadata extraction and the operational capabilities of institutions dedicated to maintaining high-quality scholarly infrastructures.

1.6 Thesis Outline

The remainder of this thesis is structured as follows.

`Chapter 2` provides a comprehensive review of related work on metadata extraction, covering traditional rule-based methods, machine learning approaches, and modern deep learning techniques. This chapter highlights the strengths and limitations of existing methods and motivates the need for a comparative evaluation of layout-aware and language model–based approaches.

`Chapter 3` describes the process of data collection and ground truth construction. It details the preparation of datasets used for benchmarking PDF parsers as well as the creation of gold-standard metadata for evaluating extraction methods, ensuring a reliable foundation for experimentation.

`Chapter 4` outlines the methodological framework adopted in this research. It explains the experimental design, the tools and models employed at each stage, and the design of the small language model pipeline, along with the layout-aware baseline for comparison.

`Chapter 5` presents the experimental results and analysis. It reports the performance of different small-scale language models in metadata extraction, compares them against layout-aware systems, and evaluates both paradigms in terms of accuracy, robustness, and computational efficiency.

`Chapter 6` concludes the thesis with a summary of key findings, synthesizes answers to the research questions, and reflects on the contributions of the work. It also discusses limitations encountered during the study and outlines directions for future research in automated metadata extraction.

Chapter 2. State Of The Art

Automated metadata extraction from scholarly documents can be understood in terms of two core stages: text extraction and metadata identification. Text extraction refers to converting the visual PDF representation into a machine-readable sequence of characters arranged in the correct logical reading order (21). Accuracy at this stage is critical, since errors such as broken ligatures, missing tokens, or misinterpreted column structures propagate directly into downstream metadata identification (96). Metadata identification, in turn, involves mapping text segments to semantic categories such as "title" or "abstract," accomplished through heuristic rules, supervised learning, or model-driven approaches (18).

A variety of open-source libraries, like PyMuPDF, pypdfium2, pdfminer.six, PyPDF2, and pdfalto are available for text extraction. These tools form the basis for higher-level metadata pipelines and are commonly benchmarked with metrics such as Character Error Rate (CER), Word Error Rate (WER), BLEU, and ROUGE-L, which respectively assess character-level fidelity, word-level correctness, and sentence-level fluency (114). For evaluation, ground truth is often constructed from datasets such as DocBank (55), with tokens grouped into logical sections using DocLayNet layout classes (89), and ordered for readability with systems like LayoutReader (34).

Two main methodological paradigms are widely adopted for metadata extraction. Layout-aware systems combine textual content with spatial layout features to classify and extract bibliographic fields. Among them, GROBID has gained prominence as an open-source system that integrates deep learning with conditional random fields (CRFs) to parse documents into structured TEI-XML (59). Extending such systems, however, often requires costly retraining on annotated corpora.

In contrast, language model-based approaches treat metadata extraction as a text understanding task. Leveraging the reasoning and pattern recognition capabilities of large-scale language models, these methods often bypass explicit layout features and instead rely on instruction tuning or schema-constrained prompting (106). Recent studies show that lightweight, instruction-tuned models can produce structured metadata outputs directly from raw text with competitive performance (112). Evaluation in this setting relies on field-level metrics such as the F1 score, Levenshtein distance for measuring string similarity, and cosine similarity for semantic overlap in free-text fields such as abstracts (63).

In this thesis, instruction-tuned models including Qwen2.5-3B-Instruct, Phi-4-Mini-Instruct, Llama-3.2-3B-Instruct, and gpt-oss-20b are prompted with schema constraints to generate

structured outputs from raw text. For benchmarking, ground-truth metadata was assembled through a consensus-based mechanism among openai/o3-mini, google/gemini-2.5-flash-lite, and x-ai/grok-3-mini, with openai/gpt-4.1 adjudicating conflicting fields.

Together, these developments delineate the technical landscape in which this work is situated. Layout-aware frameworks offer strong accuracy on structured fields but remain rigid and annotation-intensive, while language model pipelines provide flexibility and adaptability at the expense of higher computational costs. A balanced assessment of these paradigms requires both critical literature analysis and systematic empirical evaluation, which is the central objective of this research.

2.1 Background

This section provides an overview of the key technologies and resources that underpin this research. It begins with a discussion of prominent PDF parsing tools, highlighting their capabilities and differences. Subsequently, it introduces the DocBank dataset, a large-scale benchmark for document layout analysis, which serves as a foundational resource for training and evaluating layout detection models. The section then explores the critical role of layout detection and reading order prediction in document understanding workflows, emphasizing recent developments and tools that enhance accuracy and efficiency. Finally, it reviews evaluation metrics and foundational models. This background establishes the context for our approach and underscores the technological landscape in which our work is situated.

2.1.1 PDF Parsing Tools

PyMuPDF is the Python binding of the MuPDF library, originally called Fitz. MuPDF was initiated in 2002, and by 2005 the Fitz rendering engine powered its core functionality. The project has since been maintained by Artifex Software, with licensing moved to AGPLv3 in 2013 (21). PyMuPDF exposes MuPDF's C API, enabling efficient extraction of text, images, and layout information. It supports multiple output formats, including JSON and SVG, and provides bounding box data at both word and block levels. Additionally, PyMuPDF integrates with OCR modules and offers basic document manipulation features. Its high performance and low memory footprint make it a reliable choice for large-scale extraction tasks (21).

pypdfium2 is a Python interface to the PDFium rendering engine, a project originally developed and hosted by Google. The library provides a modern Python-native API that leverages PDFium's performance in rendering and layout preservation (32). It offers robust handling of page structures, supporting precise positioning of text and layout elements. With its fast, layout-aware extraction capabilities, pypdfium2 is gaining adoption in academic and industrial pipelines where accurate representation of complex PDF layouts is essential.

pdfminer.six is a maintained fork of the original PDFMiner project, extending its capabilities to modern Python environments while ensuring continued Unicode support and fine-grained text analysis (23). Unlike simpler libraries, pdfminer.six analyzes PDF objects at a low level, reconstructing text through offsets, font markers, and layout positioning. This allows detailed control over extraction and accurate parsing of complex page layouts. However, the library is slower and more memory-intensive compared to newer alternatives, making it better suited for scenarios requiring fine-grained, rule-based analysis.

PyPDF2 is a longstanding Python library that provides tools for PDF editing and basic text extraction. While widely adopted for simple document operations such as splitting and merging files, its text extraction features are relatively limited. It parses PDF streams but lacks advanced layout analysis or bounding box support, leading to weaker accuracy and inconsistent reading order preservation (24). Nevertheless, its ease of use and broad adoption in the Python ecosystem make it a popular choice for lightweight tasks.

pdfalto, developed as part of the XRCE (Xerox Research Centre Europe) initiative, is a command line tool derived from the pdf2xml project. It generates output in ALTO XML format, capturing both textual and layout structure, including fonts, spacing, and style information (59). Based on the Xpdf engine, pdfalto provides extensive options for parsing visually significant elements, annotating reading order, and preserving metadata such as outlines and embedded images. Its ability to produce ALTO schema outputs makes it especially well-suited for integration with downstream tools such as GROBID, which rely on detailed XML encodings for metadata extraction.

2.1.2 DocBank Dataset

DocBank is a large-scale benchmark dataset for document layout analysis, introduced by Li et al. in 2020 (55). The dataset spans diverse academic domains and incorporates multiple years of arXiv submissions, thereby improving generalizability across document formats. It contains 500,000 PDF pages with token-level annotations, where each token is labeled with fine-grained layout categories such as title, author, abstract, section, equation, figure, table, caption, reference, footer, list, and paragraph. Each annotation entry includes textual content, bounding box coordinates (x_0, y_0, x_1, y_1), font properties, RGB color codes, and the associated semantic label.

DocBank employs a weak supervision approach by exploiting LaTeX source files of arXiv papers. The LaTeX markup was modified to color-code semantic units, after which the compiled PDFs were mapped back to tokens and assigned layout labels according to the color encoding (55). This pipeline enabled the automated yet accurate creation of large-scale annotations, circumventing the high cost of manual labeling.

The dataset was designed for both natural language processing and computer vision tasks. Text-based models can directly use the labeled text, whereas visual models can leverage bounding box information to exploit layout features. The dataset provides a balanced split of 400,000 pages for training, and 50,000 pages each for validation and testing. Baseline experiments demonstrated that multimodal models such as LayoutLM—which jointly incorporate text and layout features—significantly outperform text-only (e.g., BERT, RoBERTa) and image-only (e.g., Faster R-CNN) baselines, highlighting the benefits of multimodal integration in document understanding tasks.

2.1.3 Layout Detection

Effective layout detection is a prerequisite for robust document parsing pipelines, especially in applications relying on retrieval-augmented generation (RAG). Errors in layout understanding propagate to downstream modules, leading to disorganized or misinterpreted input. The combination of the DocLayNet dataset (89) with the YOLO (You Only Look Once) family of object detection models (11, 103) addresses this challenge by delivering both speed and accuracy in layout detection.

The `hantian/yolo-doclaynet` project provides an implementation that converts the DocLayNet dataset into YOLO training format and offers tooling for model training, evaluation, and inference. DocLayNet itself comprises 80,863 annotated pages from a diverse set of document types, labeled across 11 semantic categories including text, picture, caption, section-header, footnote, formula, table, list-item, page-header, page-footer, and title. This diversity makes it more representative than earlier datasets such as PubLayNet (113) and DocBank (55).

The repository supports YOLO models across multiple versions, from YOLOv8 through YOLOv12, and across model scales ranging from Nano to Extra. Benchmarking results indicate mean average precision (mAP50–95) scores of approximately 0.756 for YOLOv12-Nano and up to 0.794 for YOLOv12-Extra, demonstrating both scalability and accuracy improvements across configurations. This makes the project suitable for high-throughput document parsing pipelines that require real-time inference.

2.1.4 Reading Order Prediction

The `hantian/layoutreader` project focuses on reading order prediction, an essential step in reconstructing the logical sequence of text from either native PDFs or OCR-derived bounding boxes. This task is particularly valuable in digitization workflows where the natural order of content is lost during extraction (34).

This implementation introduces several improvements over the original Microsoft LayoutReader. First, it refactors the architecture to use the HuggingFace `transformers` library with

`LayoutLMv3ForTokenClassification`, resulting in a more structured and maintainable training pipeline. Second, it replaces the earlier sequence-to-sequence design with a single-pass token classification framework, reducing inference latency by avoiding iterative decoding. Third, it emphasizes span-level rather than token-level inputs, aligning better with outputs typically produced by OCR systems and PDF parsers. Finally, a layout-only variant is trained using purely bounding-box geometry while omitting textual features, thereby achieving strong multilingual generalization with minimal performance degradation.

Empirical evaluations demonstrate the effectiveness of this approach. On span-level inputs, the layout-only variant achieves BLEU Index 95.3 and BLEU Token 97.8 on unshuffled sequences, with performance remaining nearly identical under shuffled conditions. At the token level, it reaches BLEU Index 98.0 and BLEU Token 98.2. The public release model, which integrates both text and layout features, further improves token-level BLEU to 98.3. These results highlight its superiority over heuristic baselines, which perform substantially worse (BLEU Index $\approx$ 44.4).

In summary, `hantian/layoutreader` provides a streamlined, high-performance solution for reading order prediction. Its LayoutLMv3-based architecture, span-level optimization, and layout-only variant make it both accurate and practical for real-world digitization pipelines.

2.1.5 Evaluation Metrics

Character Error Rate (CER) measures transcription accuracy at the character level by quantifying the proportion of character-level errors, insertions, deletions, and substitutions, relative to the total number of characters in the reference text (71). It is formally defined as:

$$\text{CER} = \frac{S + D + I}{N}$$

where S is the number of substitutions (characters), D the number of deletions, I the number of insertions, and N the total number of characters in the ground-truth text.

Word Error Rate (WER) is a widely adopted metric for evaluating the accuracy of automatic speech recognition, optical character recognition, and transcription systems. Unlike CER, WER evaluates errors at the word level, making it more reflective of semantic intelligibility (71). Its formulation is identical to CER but applied to words:

$$\text{WER} = \frac{S + D + I}{N}$$

where S, D, and I denote word-level substitutions, deletions, and insertions, respectively, and N is the total number of words in the reference.

In practice, CER is better suited for OCR tasks and languages with complex word segmentation, while WER is preferred for speech recognition and other tasks where meaning is conveyed primarily at the word level. Using both metrics in parallel allows error analysis to distinguish between low-level spelling deviations and higher-level recognition failures.

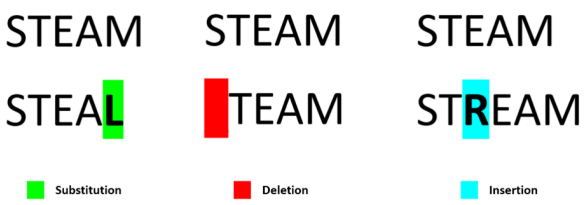

FIGURE 2.1: Examples of errors in text

The **BLEU (Bilingual Evaluation Understudy) score**, introduced by Papineni et al. (2002), is one of the most widely used metrics for assessing the quality of automatically generated text (83). Originally designed for machine translation, it has since been applied to summarization, dialogue generation, and other natural language generation tasks. BLEU evaluates candidate text by comparing its n-gram overlap with one or more reference texts, focusing on surface-level precision rather than semantic similarity.

It is formally defined as:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

where p_n is the modified precision for n-grams of size n, w_n is the weight for each n-gram (commonly uniform, e.g., 0.25 for $n = 1$ to 4), c is the candidate length, and r is the reference length. The brevity penalty (BP) penalizes outputs that are shorter than the reference.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation), introduced by Lin (2004), is a family of metrics for evaluating automatically generated text by comparing it to human-authored references (56). Unlike BLEU, which emphasizes precision, ROUGE is recall-oriented and measures how much of the reference content is captured by the candidate output.

ROUGE-L focuses on the Longest Common Subsequence (LCS) between the candidate and the reference. This allows it to capture in-sequence matches without requiring strict adjacency, making it suitable for evaluating sentence-level fluency.

The F-measure for ROUGE-L is defined as:

$$F_{LCS} = \frac{(1 + \beta^2) \cdot R_{LCS} \cdot P_{LCS}}{R_{LCS} + \beta^2 \cdot P_{LCS}}$$

$$R_{LCS} = \frac{LCS(X,Y)}{|Y|}, \quad P_{LCS} = \frac{LCS(X,Y)}{|X|}$$

where $LCS(X,Y)$ is the length of the longest common subsequence between candidate X and reference Y, $|X|$ is the length of the candidate (in words), and $|Y|$ is the length of the reference. R_{LCS} represents recall, P_{LCS} precision, and F_{LCS} the harmonic mean of recall and precision. The parameter β adjusts the relative weighting of recall and precision (commonly set to 1, giving equal weight).

The **F1 score** is a widely used evaluation metric in information retrieval and classification tasks. It represents the harmonic mean of *precision* and *recall*, balancing both false positives and false negatives. A high F1 score indicates both high precision and high recall.

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Cosine similarity measures the cosine of the angle between two non-zero vectors in a multi-dimensional space. It is commonly used to evaluate text similarity by representing documents as vectors. Values range from -1 (opposite) to 1 (identical), with 0 indicating orthogonality.

$$\text{CosineSim}(A,B) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Levenshtein distance quantifies the minimum number of single-character edits (insertions, deletions, or substitutions) required to transform one string into another. It is widely used in spell checking, approximate string matching, and OCR error correction.

$$d_{lev}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} d_{lev}(i-1,j) + 1, \\ d_{lev}(i,j-1) + 1, \\ d_{lev}(i-1,j-1) + \mathbb{1}_{(a_i \neq b_j)} \end{cases} & \text{otherwise,} \end{cases}$$

where a_i and b_j are the i-th and j-th characters of the two strings, and $\mathbb{1}_{(a_i \neq b_j)}$ is 0 if the characters are equal and 1 otherwise.

2.1.6 Traditional Language Models

Language modeling is the task of estimating probability distributions over sequences of words, forming the foundation of applications such as speech recognition, machine translation, and information retrieval. Prior to the development of large-scale neural architectures, statistical

approaches—often referred to as traditional language models—dominated the field. These models capture linguistic regularities through probabilistic formulations, usually under simplifying assumptions of independence and limited context. Prominent examples include n-gram models, Markov models, Hidden Markov Models (HMMs), class-based models, and maximum entropy models (52, 64).

An n-gram model estimates the probability of a word given the preceding $n-1$ words:

$$P(w_1, w_2, \ldots, w_T) \approx \prod_{t=1}^{T} P(w_t \mid w_{t-n+1}, \ldots, w_{t-1})$$

Probabilities are typically estimated via Maximum Likelihood Estimation (MLE):

$$P(w_t \mid w_{t-n+1}^{t-1}) = \frac{C(w_{t-n+1}^{t})}{C(w_{t-n+1}^{t-1})}$$

where $C(\cdot)$ denotes counts in the training corpus. Unigram models ($n = 1$) assume complete independence, bigram models ($n = 2$) condition on the previous word, and trigram models ($n = 3$) extend this to two preceding words. Larger n yields richer context but exacerbates sparsity, necessitating smoothing techniques (19). The Markov property assumes that the future

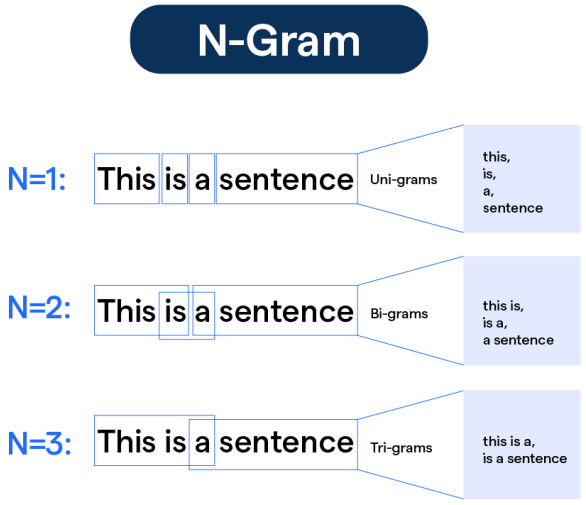

FIGURE 2.2: N-Gram

depends only on a limited history. In its simplest form:

$$P(w_t \mid w_1, \ldots, w_{t-1}) \approx P(w_t \mid w_{t-1})$$

This bigram formulation corresponds to a first-order Markov assumption, with higher-order models aligning with n-gram approaches. The assumption improves tractability but limits the ability to capture long-range dependencies.

While n-gram models focus on observable word sequences, many tasks require modeling latent structure (e.g., part-of-speech tags or phonetic states). Hidden Markov Models (HMMs) introduce a set of hidden states $S = \{s_1, \ldots, s_N\}$ with transition probabilities $a_{ij} = P(s_j \mid s_i)$, emission probabilities $b_j(o_t) = P(o_t \mid s_j)$, and initial state distribution $\pi_i = P(s_i \text{ at } t = 1)$. The joint probability of an observation sequence O and hidden states S is:

$$P(O, S) = \pi_{s_1} b_{s_1}(o_1) \prod_{t=2}^{T} a_{s_{t-1}, s_t} b_{s_t}(o_t)$$

HMMs became central to early speech recognition systems, enabling probabilistic modeling of sequential phenomena (91).

To alleviate sparsity in n-gram models, class-based approaches cluster words into classes $C(w)$ and factor probabilities as:

$$P(w_t \mid w_{t-1}) \approx P(C(w_t) \mid C(w_{t-1})) \cdot P(w_t \mid C(w_t))$$

This reduces the parameter space while preserving generalization (16). Maximum entropy (MaxEnt) models generalize language modeling by leveraging feature-based representations:

$$P(w_t \mid h) = \frac{1}{Z(h)} \exp\left(\sum_i \lambda_i f_i(w_t, h)\right)$$

where f_i are indicator features, λ_i are learned weights, and $Z(h)$ is the normalization constant. MaxEnt models remove strict independence assumptions, allowing rich contextual features to influence predictions (10).

`Strengths of Traditional Models.` Traditional language models are mathematically interpretable, computationally efficient for small contexts, and served as the foundation for speech recognition, information retrieval, and early NLP tasks.

`Limitations of Traditional Models.` They handle long-range dependencies poorly, suffer from exponential parameter growth with increasing context size, and require extensive smoothing or approximations. Furthermore, they capture only surface-level co-occurrence statistics and lack deeper semantic understanding, motivating the transition to neural and attention-based models.

2.1.7 Word Embeddings

Word embeddings are vector representations of words in a continuous, low-dimensional space, designed to capture semantic and syntactic relationships (38). Unlike traditional one-hot encodings, which represent words as sparse, high-dimensional vectors with no inherent similarity,

embeddings provide dense, distributed representations. In such spaces, semantically similar words are located near each other, enabling models to generalize across contexts.

Formally, a word embedding is a mapping:

$$f : V \to \mathbb{R}^d \tag{2.1}$$

where V is the vocabulary of size $|V|$, and $d \ll |V|$ is the embedding dimension.

The motivation for embeddings arises from limitations of traditional representations. One-hot encodings yield vectors of size $|V|$, making them memory-inefficient for large corpora, while also failing to capture word similarity. Embeddings overcome this by placing semantically related terms—such as "king" and "queen"—closer together than unrelated words like "king" and "car." They also exhibit compositional properties useful for analogical reasoning, for example:

$$\text{vec}(\text{king}) - \text{vec}(\text{man}) + \text{vec}(\text{woman}) \approx \text{vec}(\text{queen}).$$

These representations serve as compact, informative input features for tasks such as machine translation, text classification, and question answering.

`Latent Semantic Analysis (LSA)` was one of the earliest methods for generating distributed word representations (17, 29). It constructs a word–document co-occurrence matrix and applies Singular Value Decomposition (SVD) to uncover latent semantic structure. By truncating to a lower dimension, LSA produces compact embeddings that capture global semantics. However, LSA is computationally expensive for large corpora and cannot model word order.

`Word2Vec`, introduced by Mikolov et al. (2013), advanced embedding learning by training predictive neural models (70). It includes two architectures: Continuous Bag-of-Words (CBOW), which predicts a word from its surrounding context, and Skip-Gram, which predicts surrounding words from a target. Techniques such as negative sampling and hierarchical softmax allow efficient scaling to large vocabularies. Word2Vec embeddings capture semantic and syntactic regularities and support analogical reasoning.

`Global Vectors (GloVe)`, proposed by Pennington et al. (2014), combined co-occurrence statistics with predictive modeling (86). GloVe learns embeddings such that vector dot products approximate the logarithm of word co-occurrence probabilities. Its weighted least-squares objective balances rare and frequent pairs, yielding embeddings that encode both statistical and semantic information.

`FastText`, developed by Facebook AI Research, extended Word2Vec by incorporating subword information (12). Each word is represented as a bag of character n-grams, allowing embeddings to be generated for rare or unseen words. This makes FastText particularly effective for morphologically rich languages and open-vocabulary tasks.

Despite their strengths, static embeddings exhibit limitations. A key issue is polysemy: a single vector cannot represent multiple senses of a word, leading to ambiguity. Moreover, embeddings are context-free, assigning the same representation regardless of usage, which restricts their ability to capture nuanced meaning. Bias amplification is another concern, as embeddings inherit social and cultural biases from training corpora (13). These limitations motivated the development of contextual embeddings, where deep architectures such as ELMo (87), BERT (33), and GPT produce context-dependent representations. Unlike static embeddings, these dynamically adjust based on surrounding text, addressing polysemy and substantially improving downstream task performance.

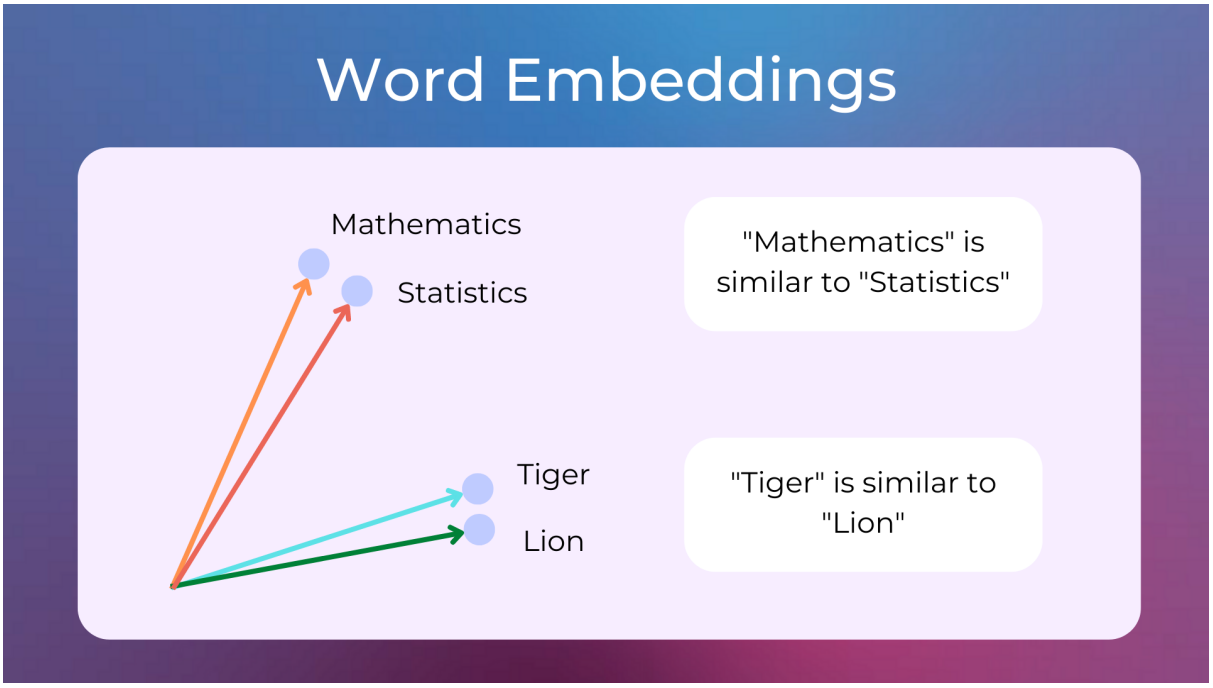

FIGURE 2.3: Word Embeddings: Representing text as vectors

2.1.8 Attention

The attention mechanism has become a cornerstone of modern natural language processing and sequence modeling. Originally introduced as a means to address the limitations of recurrent neural networks (RNNs) and long short-term memory (LSTM) networks in handling long-range dependencies (7), attention enables models to dynamically focus on the most relevant parts of the input when generating an output. The landmark paper *Attention Is All You Need* by Vaswani et al. (2017) generalized this idea into the Transformer architecture, discarding recurrence entirely and relying on stacked attention and feed-forward layers (104). This innovation improved scalability and parallelizability, and quickly redefined the state of the art in machine translation, ultimately leading to large-scale language models such as BERT, GPT, and LLaMA.

Traditional sequence models—RNNs and LSTMs—encode inputs sequentially, which makes them inherently slow to train and prone to vanishing or exploding gradients when dealing

with long sequences (48). While gating mechanisms improved memory retention, these models still struggled to capture distant dependencies. Attention mitigates this by enabling direct interactions between any two positions in a sequence, regardless of distance, thereby improving both efficiency and expressiveness.

The fundamental operation of attention maps a query (Q) and a set of key–value pairs (K, V) to an output. The attention score between Q and each key K is computed using a dot product, divided by the key vector dimension, and normalized through the softmax operation.:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where $Q \in \mathbb{R}^{n \times d_k}$ are the query vectors, $K \in \mathbb{R}^{m \times d_k}$ the key vectors, $V \in \mathbb{R}^{m \times d_v}$ the value vectors, and d_k is the dimensionality used for scaling (104).

A single attention mechanism may be insufficient to capture the diverse types of relationships within a sequence. To address this, multi-head attention projects the inputs into multiple subspaces and computes attention in parallel:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$

with each head defined as:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

where W_i^Q, W_i^K, W_i^V are learned projection matrices, W^O is the output projection, and h is the number of heads. This allows the model to attend to different types of dependencies jointly (104).

Since the Transformer processes tokens in parallel and lacks inherent sequential structure, positional encodings are added to input embeddings. These sinusoidal encodings inject order information:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right), \quad PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

where pos is the position, i is the dimension index, and d_{model} is the embedding size. This formulation enables the model to generalize to longer sequences than those encountered during training (104).

The Transformer adopts an encoder–decoder structure, where both are built from stacked layers of multi-head attention and feed-forward sublayers.

`Encoder layers` consist of multi-head self-attention and position-wise feed-forward networks, each wrapped with residual connections and layer normalization.

`Decoder layers` add masked multi-head self-attention to prevent access to future tokens, along with encoder–decoder attention to integrate encoder outputs.

The feed-forward network within each block is defined as:

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

where W_1, W_2, b_1, b_2 are learned parameters.

Advantages

- `Parallelization`: Unlike RNNs, all tokens are processed simultaneously, accelerating training.
- `Long-range dependencies`: Attention allows direct connections between distant tokens.
- `Expressivity`: Multi-head mechanisms enable modeling diverse linguistic dependencies.
- `Scalability`: The design is well-suited for distributed training on GPUs and TPUs.

Despite its success, the vanilla Transformer scales quadratically with sequence length ($O(n^2)$) because of the full attention matrix. This makes it inefficient for very long sequences. Numerous variants have been proposed to mitigate this, including sparse attention (Longformer (9)), linear attention (Linformer (105), Performer (20)), and memory-augmented models (Transformer-XL (27)).

The Transformer revolutionized NLP by providing the foundation for pretrained language models such as BERT (33), GPT (92), and T5 (93). Its generality has also extended to computer vision (Vision Transformers, ViT (35)), speech recognition, and multimodal tasks, establishing attention as a universal mechanism for representation learning.

2.1.9 Modern Language Models

OpenAI's `o3-mini` is a small, cost-efficient member of the o-series of reasoning models trained with large-scale reinforcement learning to think before answering. The model exposes configurable *reasoning effort* (low / medium / high), and a higher-intelligence variant (`o3-mini-high`) is available for tasks that benefit from deeper deliberation (81). The API supports function calling, Structured Outputs, developer messages, and streaming; vision is not supported (81). The accompanying system card documents training with reinforcement learning and safety filtering (78, 82). Benchmark evidence emphasizes STEM and logic performance with effort-scalable accuracy–latency trade-offs. In high-effort mode, `o3-mini` reports 83.6% on AIME 2024 and 77.0% on GPQA Diamond (81).

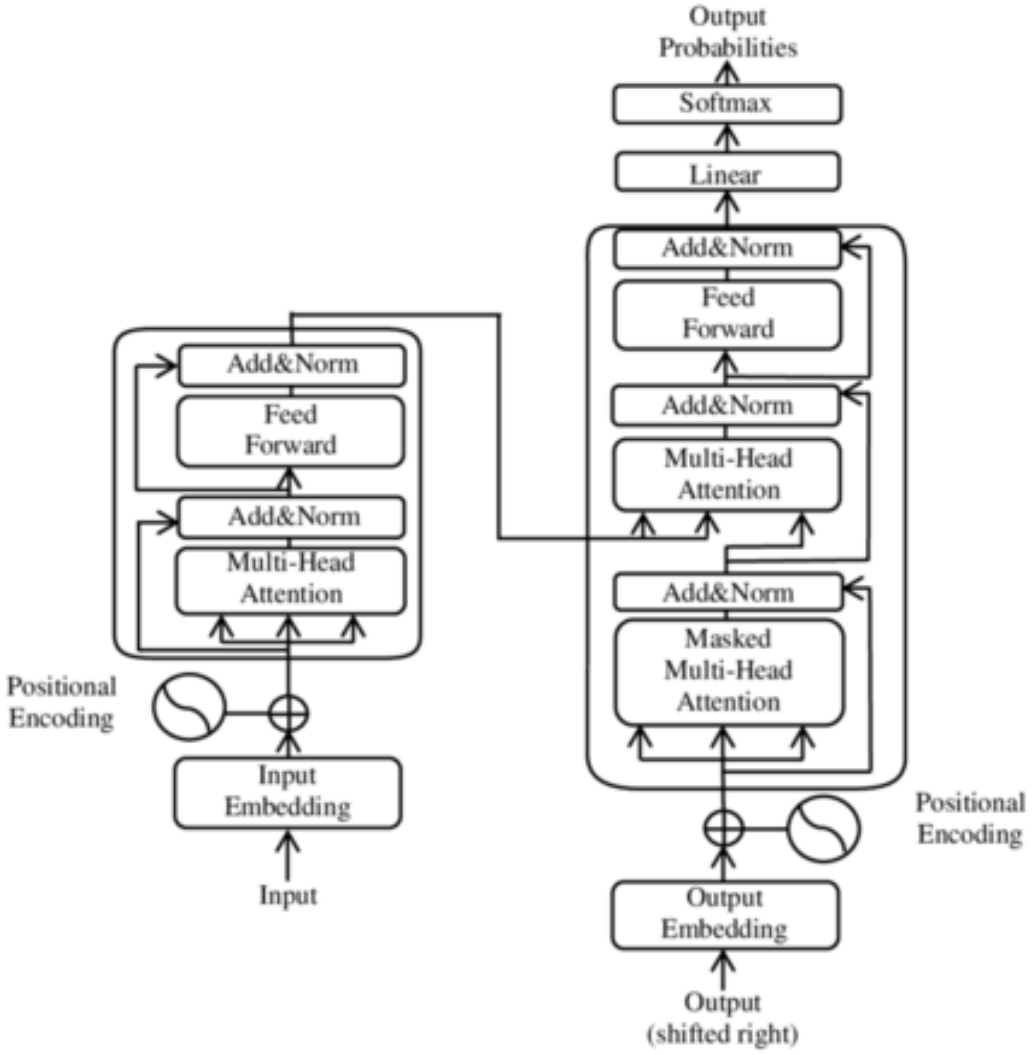

FIGURE 2.4: Architecture of a Transformer Model

`Gemini 2.5 Flash-Lite` is Google's most cost-efficient, low-latency model in the Gemini 2.5 family, designed for high-throughput production use. It supports multimodal inputs (text, image, audio, video), an extended context window (up to one million tokens), optional "thinking"/reasoning effort, function calling, structured outputs, retrieval, code execution, and grounding via search (40). Pricing is listed at $0.10 per million input tokens and $0.40 per million output tokens (39). Vertex AI documentation specifies model limits of up to 1,048,576 input tokens and up to 65,536 output tokens (22). Google highlights real-world adoptions across diverse domains (e.g., Satlyt, HeyGen, DocsHound, Evertune), illustrating latency and cost benefits in production (40).

xAI's `grok-3-mini` is a lightweight reasoning model with a 131,072-token context window and per-token pricing of roughly $0.30 (input) and $0.50 (output) via the xAI API (109). The model supports an explicit reasoning mode in which the API *returns* a separate reasoning trace alongside the final answer, enabling adjustable "thinking budgets" to trade latency for accuracy

(108). Documentation lists text modality and standard developer capabilities (function calling, structured outputs) (109).

The `gpt-4.1` family (including `gpt-4.1`, `gpt-4.1-mini`, and `gpt-4.1-nano`) was introduced with a 1 million-token context window, improved instruction following, and stronger coding performance relative to prior GPT-4 variants (74). OpenAI reports gains on benchmarks such as SWE-bench Verified and instruction-following evaluations (74). At the same time, independent testing (e.g., SplxAI) and media coverage (e.g., TechCrunch) raised concerns that `gpt-4.1` can be less aligned or more vulnerable to certain attacks than `GPT-4o` under some prompting setups, motivating caution and careful prompt/system design (99, 107).

`Qwen2.5-3B-Instruct` is an instruction-tuned, dense, decoder-only model with ∼3.1B parameters in the Qwen 2.5 family. Qwen 2.5 models were pretrained up to ∼18 trillion tokens, support long context (commonly up to 128K), multilingual usage (29+ languages), and emphasize structured outputs and instruction adherence (90, 101). The official model card documents training details, architecture choices (e.g., RoPE, RMSNorm, SwiGLU), and instruction tuning for robust structured output generation and tool use (90, 101).

`Phi-4-mini-instruct` (3.8B parameters), released in early 2025 as part of Microsoft's Phi-4 family, is an instruction-tuned decoder-only transformer targeting efficient reasoning, code, and multilingual tasks. Microsoft highlights a substantially expanded vocabulary, long context (commonly reported at up to 128K tokens), function calling support, and improved instruction following relative to prior Phi versions (67–69). Public model cards and ecosystem listings likewise report a 128K context length for the mini variant and broad availability across Azure AI Foundry, Hugging Face, and other platforms (67–69).

`Llama 3.2 3B Instruct` is a compact, instruction-tuned model in Meta's multilingual Llama 3.2 lineup. Official materials and model cards describe an autoregressive transformer with Grouped-Query Attention and a 128K-token context window, tuned for multilingual dialogue, retrieval, summarization, and tool use, with efficient deployment on edge hardware (4, 65). Community distributions and partner tooling provide optimized inference paths across CPUs/GPUs (e.g., TensorRT-LLM) and popular inference frameworks (65).

`gpt_oss_20b` is an open-weight, Apache 2.0–licensed, mixture-of-experts (MoE) reasoning transformer released by OpenAI in August 2025 alongside a larger 120B variant. The 20B model uses 24 layers (about 21B parameters total) with ∼3.6B active parameters per token, grouped multi-query attention, RoPE, and alternating dense/sparse attention, and natively supports 128K context. Weights are provided in MXFP4 quantization, enabling ∼16 GB VRAM deployments (75, 77). OpenAI's model card reports results on reasoning, coding, and health benchmarks (e.g., Codeforces Elo and SWE-bench Verified), with the 20B model generally competitive with `o3-mini` on several tasks (75). OpenAI's availability notes list broad ecosystem support (e.g.,

Model	Context Window	Parameters	Key Features / Notes
OpenAI `o3-mini` (81)	Input: 200k; Output: 100k	∼3B	Reasoning-effort levels (low/medium/high), structured outputs, function calling, developer messages; optimized for STEM reasoning.
Google `Gemini-2.5-Flash-Lite` (28)	Input: 1,048,576; Output: 65,536	(undisclosed)	Multimodal (text, image, video, audio, PDF), search grounding, code execution, structured outputs, "thinking budgets."
xAI `Grok-3-mini` (110)	131,072	(undisclosed)	Lightweight reasoning model, exposes reasoning traces, function calling, tuned for enterprise documents.
OpenAI `GPT-4.1` (79)	Input: 1M	(undisclosed)	Long-context (1M tokens), strong coding and instruction following, multimodal, faster/cheaper than GPT-4o.
Qwen `Qwen2.5-3B-Instruct` (44)	32k	3B	Instruction-tuned, compact edge model; strong math/coding for its size.
Microsoft `Phi-4-mini-instruct` (94)	128k	3.8B	Dense Transformer with grouped-query attention, shared embeddings; strong reasoning.
Meta `Llama-3.2-3B-Instruct` (5)	Default 8k, up to 128k	3B	Multilingual, instruction-tuned; strong summarization, retrieval, dialogue.
OpenAI `gpt_oss_20b` (80)	128k (up to 1M claimed)	21B total, ∼3.6B active	Open-weight MoE (Apache 2.0); structured outputs, reasoning config, agentic capabilities; deployable locally.

TABLE 2.1: Comparison of language models used in this research.

Azure, Hugging Face, vLLM, Ollama, llama.cpp, LM Studio, AWS, Fireworks, Databricks) to facilitate local or hosted inference (76, 77).

2.2 Related Work

The field of document metadata extraction has evolved through several distinct phases, starting from early rule-based systems to sophisticated multimodal and deep learning approaches. Initial methods relied heavily on handcrafted heuristics that encoded domain-specific rules, such as font size or positional cues, to identify metadata fields. While straightforward, these were fragile and required constant manual updates to accommodate new formats (61). This limitation spurred the adoption of supervised machine learning techniques, notably sequence labeling models like CRFs and SVMs, which leveraged annotated datasets to learn patterns in text and layout (37, 85). Although more adaptable, these models still depended on extensive manual feature engineering and struggled with highly variable or complex document formats.

The advent of deep learning in the 2010s marked a turning point, enabling models such as BiLSTM-CRFs and character-aware neural networks to automatically learn rich representations from raw data. These approaches significantly improved accuracy and generalization, especially for structured tasks like citation parsing and author attribution (49, 53, 62). However, they often treated text as a linear sequence, limiting their robustness across diverse layouts and out-of-distribution documents (30).

To overcome layout variability, researchers turned to computer vision techniques, framing document pages as images and applying segmentation models like U-Net and Mask R-CNN to detect structural components (14, 31, 46). While effective at capturing geometric layout cues, these vision-only models faced challenges in fine-grained textual understanding and often failed to generalize to subtle layout differences or complex multi-element pages.

The integration of visual and textual modalities gave rise to multimodal approaches, which combined layout-aware features with deep NLP models such as LayoutLM and LAME. These systems jointly encode spatial, typographic, and semantic information, resulting in improved robustness and adaptability across diverse document formats (57, 100). Such models mark a significant advancement in achieving format-independent metadata extraction, reducing reliance on handcrafted rules or rigid templates.

Meanwhile, tools like GROBID have established strong baselines for scholarly document processing, utilizing cascaded CRF models and PDF layout analysis to extract bibliographic metadata with high accuracy (41). More recently, the emergence of large language models (LLMs) like GPT-4 has opened new avenues for direct, prompt-driven extraction of comprehensive metadata schemas, potentially reducing the need for task-specific training (98). However, challenges related to hallucination, computational cost, and consistency remain, prompting ongoing research into hybrid and fine-tuned systems.

This body of work underscores the transition from rule-based heuristics to deep, multimodal, and now LLM-driven methods, each offering different trade-offs in flexibility, accuracy, and scalability.

2.2.1 Early Rule-Based and Machine Learning Approaches

The earliest attempts at metadata extraction relied on rule-based systems encoding handcrafted heuristics about document structure. Such systems identified fields like titles or author names using cues such as font size, style, or position on the page (61). While effective on the specific formats for which they were designed, these approaches were brittle: even minor layout variations could break the rules, necessitating constant manual updates for new publishers or journals. Their lack of adaptability motivated a shift toward data-driven machine learning techniques, which offered more flexibility across diverse document formats.

By the mid-2000s, systems such as CiteSeerX began applying supervised learning for large-scale metadata extraction (37). Classical models including Hidden Markov Models (HMMs), Conditional Random Fields (CRFs), and Support Vector Machines (SVMs) were trained on labeled examples of metadata fields such as titles, authors, and affiliations. For instance, Peng and McCallum (2006) demonstrated that CRFs could significantly improve extraction accuracy by modeling contextual dependencies in research paper text (85). Similarly, SVM-based approaches classified lines of text into categories (e.g., title, author, abstract) with reasonable performance on constrained datasets (45). CRFs became particularly popular due to their effectiveness in sequence labeling, and they were often combined with domain-specific features to incorporate rule-based knowledge—for example, encoding the heuristic that email addresses typically follow author names.

Notable systems of this era include ParsCit, an open-source CRF-based toolkit for parsing citation strings, which demonstrated that learning from annotated data outperformed rigid rule sets for bibliographic metadata extraction (26). These approaches were more robust than purely rule-based methods, particularly when handling moderately varying layouts. However, they required substantial annotated training corpora, since token-level labeling of PDFs into categories such as title, author, or abstract is labor-intensive and costly. As a result, early datasets often covered only a narrow range of publication formats, limiting generalizability to unseen layouts.

Moreover, even data-driven models of this generation struggled with highly complex or out-of-distribution documents that differed substantially from their training examples. Researchers noted that these models, while more flexible, still required frequent retraining or feature updates as document templates evolved—reflecting a similar maintenance burden to rule-based methods (72). This limitation set the stage for the emergence of deep learning methods capable of automatically learning richer, multimodal representations of both document text and layout.

2.2.2 Advances with Deep Learning for Metadata Extraction

The 2010s marked the rise of deep learning, which revitalized research on metadata extraction. Deep neural networks became attractive due to their ability to automatically learn latent features from raw data, capturing subtle patterns that handcrafted features might miss. Early successes in natural language processing (NLP) tasks suggested that sequence models, particularly recurrent neural networks (RNNs), were able to significantly improve extraction accuracy. Indeed, studies soon demonstrated that neural architectures outperformed the previous generation of rule-based and classical machine learning approaches in both effectiveness and efficiency (54).

One influential line of work applied Bidirectional LSTM-CRF models to metadata extraction. In this setup, a bidirectional Long Short-Term Memory (BiLSTM) network encodes contextual information in both forward and backward directions, while a Conditional Random Field (CRF) layer enforces globally consistent label sequences. Originally proposed by Huang et al. for sequence tagging tasks (49), this architecture proved highly effective for labeling bibliographic fields such as Title, Author, and Affiliation. Other studies extended these ideas by incorporating character-level convolutional neural networks (CNNs) to produce subword representations. Combined with LSTMs, this allowed models to capture orthographic patterns—particularly useful for detecting acronyms, names, or domain-specific terms (53).

When trained on sufficiently large datasets, these neural models established new benchmarks for metadata extraction. For instance, Luo et al. introduced a deep segment-by-segment labeling model for citation metadata, reporting substantial improvements over older systems such as ParsCit and BibPro when evaluated on standard corpora like UMass and Cora (62). Importantly, deep learning methods demonstrated stronger generalization to unseen formats, as they could automatically learn structural cues—for example, that titles are often displayed in larger fonts or that author names are listed in comma-separated groups—without requiring explicit hand-crafted rules.

Despite these advances, text-only neural approaches continued to face limitations. Recurrent models could fail on highly atypical layouts, such as when journals reordered title and author blocks or interspersed metadata with decorative elements. Even state-of-the-art sequence models struggled to capture layout-specific cues, since they process text as a linear sequence and disregard the two-dimensional structure of documents. As a result, high performance within a particular template did not guarantee robustness across radically different ones (30). This limitation highlighted the need to move beyond purely text-based deep learning, leading researchers to explore layout-aware and multimodal methods that integrate visual and spatial information from document structure.

2.2.3 Layout and Vision-Based Techniques

To address the variability of page layouts, researchers explored computer vision (CV) methods, reframing each page of a PDF as an image to be segmented or classified. The motivation was that if metadata fields could be detected visually by their spatial arrangement or styling, an image-based model might generalize across languages and complex formatting.

A notable example of this direction is DeepPDF, which reconceptualized PDF parsing as an image segmentation problem. It employed a U-Net convolutional network (95), originally developed for biomedical segmentation, to partition a page into distinct regions corresponding to blocks of text while ignoring non-textual elements such as figures or headers (31). By focusing on geometric layout, DeepPDF demonstrated that visual segmentation could indirectly aid metadata extraction by localizing core components such as titles or abstracts.

Building on similar principles, Boukhers et al. introduced MexPub, a system for extracting metadata from German academic publications using an object-detection paradigm (14). MexPub leveraged the Mask R-CNN architecture (46) with a ResNeXt backbone and a feature pyramid network, enabling pixel-level classification of regions on the page. In practice, the system learned to "see" the document and detect areas corresponding to metadata elements (e.g., title box, author list) without directly reading the textual content.

While effective on its training set, MexPub revealed key limitations. It struggled to generalize to layouts outside of its training distribution—for instance, papers with unusually formatted abstracts or multi-column first pages often went unrecognized. Moreover, small or subtle elements such as email addresses in fine print frequently eluded detection. These shortcomings illustrate that vision-only models, while powerful for structural layout detection, are insufficient for fine-grained textual tasks.

Overall, CV-based methods showed the value of exploiting layout as a visual signal, but also suggested that combining vision and text representations would yield more robust and generalizable metadata extraction. This insight paved the way for multimodal models that integrate natural language processing with document image analysis.

2.2.4 Multimodal Approaches

Multimodal metadata extraction methods aim to fuse textual and visual information, leveraging the complementary strengths of both. Natural language processing excels at interpreting textual content (e.g., recognizing that a sequence resembles a person's name or a date), while vision-based processing captures layout structure (e.g., identifying bold centered lines or distinguishing multi-column layouts). By combining these modalities, systems can localize candidate regions based on layout while confirming the semantic plausibility of the text, thus improving robustness across varied formats.

An early demonstration of this synergy was presented by Liu et al., who designed a deep learning model that jointly processed textual and visual features of scientific documents (57). Their architecture applied a recurrent neural network (RNN) to the text sequence and a convolutional neural network (CNN) to an image rendering of the page. The outputs were concatenated and passed through a BiLSTM–CRF model to predict metadata labels. This multimodal design outperformed unimodal counterparts, as layout cues provided critical disambiguation. For example, if an author name appeared in an unusual location, the text-only model might misclassify it, but the visual branch could still recognize that the region corresponded to a conventional "author block."

A particularly challenging scenario is presented by highly heterogeneous layouts, such as those encountered in German social science publications, where small publishers adopt a wide variety of styles. Boukhers et al. addressed this by combining their MexPub vision model (Mask R-CNN with a ResNeXt backbone) (14) with a BiLSTM text model in a two-stage architecture. The fused representation allowed the system to both "read" the textual content and "see" the page structure, significantly improving generalization to idiosyncratic layouts. For instance, even when metadata was arranged non-linearly on the page, the visual module could guide the sequence model in reconstructing the logical reading order.

In parallel, researchers proposed layout-aware text models that incorporate spatial and typographic features without explicitly relying on image inputs. A representative framework is LAME (Layout-Aware Metadata Extraction), which first applies PDF parsing tools such as PDFMiner to segment content by position and style (100). The resulting layout-annotated dataset is then used to train a specialized model, Layout-MetaBERT, which extends BERT with positional encodings reflecting document structure. LAME achieved strong robustness on unseen journal layouts, with Macro-F1 scores exceeding 93% even when evaluated on publishers absent from the training distribution.

These findings underscore the promise of multimodal and layout-informed models for achieving format-independent metadata extraction. Unlike earlier systems that were brittle to template variation, multimodal solutions adapt more flexibly to novel layouts, requiring minimal re-training. By jointly encoding textual semantics and structural layout, they represent a significant step toward unified, domain-agnostic metadata extraction.

2.2.5 Technical Overview of GROBID

GROBID (GeneRation Of BIbliographic Data) is an open-source machine learning library for extracting structured information from scientific and technical publications in PDF form. It converts raw PDFs into XML/TEI documents, enabling metadata extraction, document segmentation, and reference parsing for digital libraries and NLP pipelines (41, 60).

GROBID provides modular capabilities, including:

- `Header Parsing:` Extraction of bibliographic fields such as title, authors, affiliations, abstract, and keywords.
- `References Parsing:` Parsing of bibliographic references with benchmark F1-scores near 0.90 (42, 43).
- `Citation Context Resolution:` Linking in-text citations to references.
- `Full-Text Structuring:` Segmentation into sections, paragraphs, figures, tables, and footnotes.

The system follows a cascaded model architecture, where specialized models sequentially process document segments. It primarily uses Conditional Random Fields (CRFs) for sequence labeling, with optional neural models via DeLFT. PDFs are first processed with `pdfalto` to preserve layout features, before segmentation and extraction. Results are output in TEI XML, supporting fine-grained tagging and interoperability.

Benchmark evaluations consistently place GROBID ahead of comparable tools. For instance, it achieved F1 scores of 0.91 (title), 0.82 (abstract), and 0.79 (reference extraction), outperforming alternatives such as CERMINE and Science Parse (66). Its segmentation also supports reliable extraction of narrative structures such as sections and captions.

GROBID is available as a JAR, RESTful service, or Docker container, with throughput of about 10 PDFs/s on a 16-core server (60). It supports Unicode, multi-script content, and integration into large-scale ingestion pipelines such as CORE (25).

`Strengths:` High accuracy in metadata extraction, modular design, interoperable TEI outputs, and proven scalability.
`Limitations:` Dependence on CRFs reduces ability to capture long-range dependencies, and performance can degrade on scanned or highly complex PDF layouts.

GROBID remains one of the most reliable open-source systems for structured metadata extraction from PDFs, balancing accuracy, scalability, and extensibility. While newer deep learning approaches are emerging, it continues to serve as a robust foundation for scholarly document processing.

2.2.6 Emerging Use of Large Language Models

The latest frontier in metadata extraction is the application of large language models (LLMs) for direct parsing of scholarly documents. Transformer-based LLMs such as GPT-4 have demonstrated remarkable ability to interpret and generate human-like text, owing to their pretraining on vast, diverse corpora (73). A key advantage is that LLMs can be repurposed for metadata extraction with minimal task-specific supervision. Instead of building dedicated sequence models, researchers investigate whether LLMs can parse full scientific papers and output structured metadata directly.

One recent example is the MOLE framework (Metadata extraction using Open-domain LLMs for Enhanced coverage), which employs prompting strategies to guide LLMs in extracting a rich schema of metadata attributes (98). MOLE processes entire documents, sometimes chunked or reformatted for long context handling, and instructs the model to produce outputs in structured JSON. Beyond conventional fields such as title, authors, and abstract, MOLE targets dozens of attributes including license information, dataset links, and funding acknowledgments. By exploiting single-shot prompting and in-context examples, MOLE reduces the need for task-specific training data. Early evaluations showed that with iterative refinement and self-validation (e.g., instructing the model to re-check its outputs), extraction of up to 30 metadata attributes is feasible with encouraging levels of accuracy. This expands the scope of automated metadata collection toward more comprehensive research documentation.

Despite these promising results, opinions on LLM-based methods remain divided. General-purpose LLMs are not inherently optimized for structured extraction: they may hallucinate plausible-looking outputs or disregard formatting constraints (51). For instance, when asked to identify the title, an LLM might generate a fluent but fabricated variant if the true title appears in an unusual format. Boukhers et al. argue that without task-specific fine-tuning, models like GPT-4 can yield inconsistent metadata and struggle with the rigid precision required for bibliographic tasks (15). Scalability also presents challenges, since running large models on thousands of papers is computationally expensive, and occasional failures undermine reliability.

To mitigate these issues, current frameworks employ validation mechanisms, hybrid error-checking pipelines, and are exploring fine-tuning smaller, domain-specific transformers as a middle ground between generality and precision. While LLM-based metadata extraction has not yet supplanted specialized pipelines, it represents a compelling direction for future systems capable of handling diverse formats on demand through prompt-driven inference.

FIGURE 2.5: Progression of Metadata Extraction Tools

2.3 Comparative Analysis

Over the evolution of metadata extraction techniques, each generation of approaches has introduced improvements in handling diverse publication formats, alongside new trade-offs. Early rule-based systems were straightforward to implement for known layouts, relying on handcrafted heuristics such as font size or positional cues (61). However, these methods failed to generalize: even minor variations in style could render the rules ineffective, and extensive maintenance was required to adapt to new publishers or disciplines.

Classical machine learning approaches improved robustness by learning from annotated examples. Methods such as Support Vector Machines (SVMs) (45) and Conditional Random Fields (CRFs) (85) achieved reasonable accuracy in recognizing fields like titles, authors, and affiliations, provided sufficient labeled data. ParsCit, a CRF-based system for citation parsing, exemplified the gains of these models over purely rule-based pipelines (26). Yet, their reliance on costly labeled corpora limited their scalability, and performance degraded on unseen formats.

Deep learning in NLP marked a turning point by automatically learning latent feature representations. BiLSTM-CRF models (49), as well as hybrid character-level CNN-LSTM architectures (53), significantly boosted accuracy on benchmark datasets such as UMass and Cora (62). These approaches reduced dependence on handcrafted features and generalized better than earlier methods. Nonetheless, text-centric models assumed a relatively linear reading order, struggling with highly unconventional layouts that disrupted the sequence of tokens.

Vision-based methods reframed metadata extraction as a document image analysis task. Systems such as DeepPDF (31) and MexPub (14) used segmentation and object detection to locate structural components independent of language. This made them more resilient to format variation, but without semantic interpretation, vision-only models often missed fine-grained cues, such as small author footnotes or irregular placements of abstracts.

Multimodal approaches emerged as a powerful compromise, fusing textual and visual modalities to capture both content and layout. Liu et al. demonstrated the effectiveness of combining CNN-based vision features with RNN-based text models (57), while Boukhers et al. extended MexPub with a BiLSTM text branch to improve robustness on heterogeneous German publications (14). Layout-aware frameworks such as LAME (100) further encoded structural cues directly into language models, achieving Macro-F1 scores above 93% on unseen publisher layouts. These advances underscored the value of multimodality for format-independent extraction.

The latest frontier is the use of large language models (LLMs). Frameworks such as MOLE leverage GPT-4 (73) to extract rich schemas of metadata—including licensing, dataset links, and funding information—through prompt engineering and single-shot examples (98). While these systems extend the breadth of retrievable attributes, challenges remain. General-purpose LLMs may hallucinate, produce inconsistent outputs, or ignore strict formatting requirements (15, 51). Moreover, their high computational cost raises questions of scalability for large bibliographic databases.

In comparing these approaches, a clear trajectory emerges. Adaptability to diverse formats has steadily increased—from brittle rule-based methods to multimodal deep learning and LLM-driven systems. Accuracy has similarly improved: early rule-based tools might correctly identify a title 70–80% of the time, while modern multimodal or layout-informed systems routinely exceed 90% F1 scores on challenging corpora. At the same time, the burden has shifted from

manual rule-writing to the preparation of annotated data, large-scale model training, and computationally intensive inference. Another axis of progress lies in granularity: earlier systems focused narrowly on titles and authors, while recent frameworks aim to extract dozens of attributes automatically, reflecting broader ambitions for metadata coverage.

In conclusion, the field has progressed toward increasingly general and powerful solutions for metadata extraction. Early rule-based and machine learning approaches established the foundations, while deep learning and multimodal systems improved resilience to layout variability. LLM-based methods represent the newest paradigm, promising flexible, schema-driven extraction but still facing challenges in reliability and scalability. The comparative insight from the literature is that consistency and accuracy are best achieved by methods that embrace document heterogeneity—through the integration of textual, visual, and contextual signals—rather than constraining extraction to predefined templates. Future systems will likely converge these paradigms, combining multimodal deep learning with the generative flexibility of LLMs, thereby enabling robust, scalable, and comprehensive metadata extraction for the ever-growing body of scientific literature.

2.4 Summary

The evolution of metadata extraction techniques reflects a clear progression from early rule-based systems to sophisticated multimodal and large language model (LLM) approaches. Initial rule-based methods, relying on handcrafted heuristics, were simple to implement but lacked robustness across diverse formats. Classical machine learning models such as SVMs and CRFs improved accuracy and adaptability but depended heavily on extensive labeled data and still struggled with layout variability. The advent of deep learning introduced models like BiLSTM-CRFs and hybrid neural architectures, which significantly enhanced generalization by automatically learning features from text, yet remained limited by their linear processing of document content. Vision-only methods, treating pages as images, offered greater resilience to layout differences but lacked semantic understanding, while multimodal approaches combining visual and textual signals achieved notable robustness and flexibility. The latest trend leverages LLMs like GPT-4, capable of schema-driven, prompt-based extraction of rich metadata attributes, though challenges such as hallucination, inconsistency, and computational cost persist. Overall, each new paradigm has improved the ability to handle diverse formats and complex layouts, with recent methods focusing on integrating multimodal signals and leveraging the flexibility of LLMs. Future systems are poised to unify these approaches, balancing robustness, accuracy, and scalability for comprehensive metadata extraction across the rapidly expanding corpus of scientific literature.

Chapter 3. Data and Methodology

This chapter presents the methodological foundation of the research, focusing on two complementary dimensions of ground truth construction: establish a reliable ground truth for evaluating open-source PDF parsing tools and the creation of a curated gold standard for metadata extraction. By integrating advanced layout analysis, token organization, and multi-model validation, we aim to create a robust benchmark that reflects the diversity and complexity of real-world academic documents.

3.1 Ground Truth for PDF Text Extraction

Achieving highly accurate metadata extraction from born-digital academic PDFs fundamentally relies on obtaining a precise, complete, and correctly ordered textual representation of each document's content. However, this seemingly straightforward task is fraught with numerous challenges stemming from the inherent heterogeneity and complexity of academic publication formats. To address these challenges and facilitate a rigorous evaluation of open-source PDF parsing libraries, it is imperative to develop a robust, high-quality ground-truth dataset that accurately reflects real-world document variability.

The primary obstacles in extracting faithful text representations can be categorized into three broad categories. First, heterogeneity in publisher templates introduces substantial variability in layout conventions. Different publishers such as IEEE or Springer adopt distinct font styles, heading hierarchies, and artifacts—like decorative lines or logos—that can confound naive extraction algorithms. Second, variable page layouts, including single-column, multi-column, or hybrid formats, often violate the assumption of a straightforward top-to-bottom reading order, leading to fragmented or misaligned text sequences if naive parsing is employed. Third, parser-specific limitations further complicate extraction fidelity; many open-source tools struggle with encoded characters, ligatures, embedded figures, footnotes, and incremental updates within complex PDF structures, resulting in incomplete or distorted textual outputs.

Given these complexities, the core objective is to identify which among the available open-source libraries—namely, `PyMuPDF`, `pypdfium2`, `pdfminer.six`, `PyPDF2`, and `pdfalto`—delivers the highest fidelity in extracting textual content, particularly from the first pages of arXiv papers that exhibit diverse layouts and styles. To achieve this, a meticulously curated, gold-standard dataset must be assembled. This dataset will serve as an authoritative benchmark against which the performance of each parser can be quantitatively assessed.

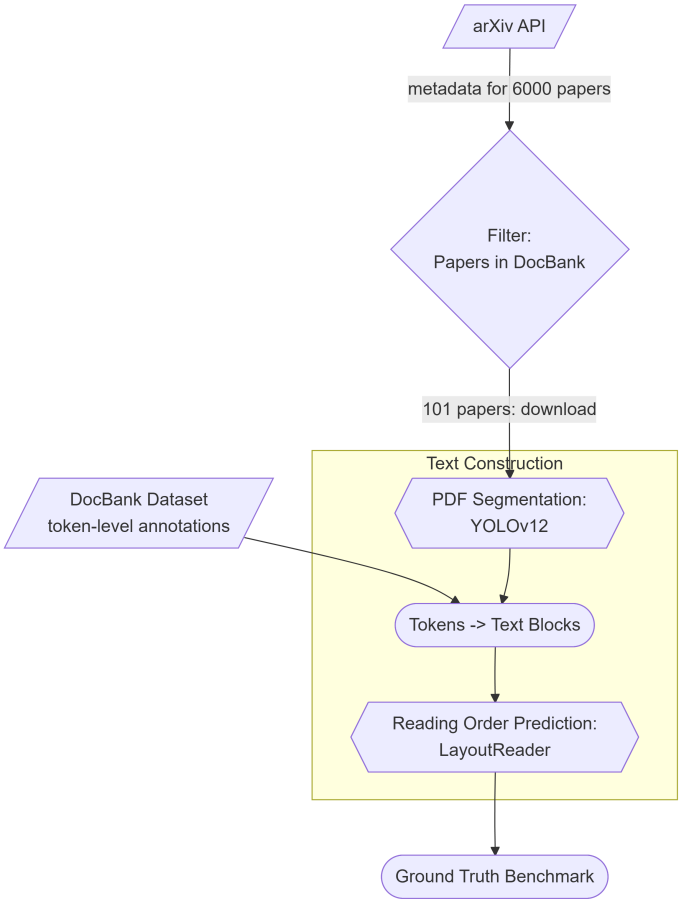

FIGURE 3.1: Creation of Ground Truth to Evaluate PDF Text Extraction.

3.1.1 Source Dataset

The cornerstone of this evaluation is the **DocBank** dataset, a recently constructed large-scale corpus designed explicitly for layout analysis tasks in academic documents. It provides token-level annotations for a broad spectrum of scholarly PDFs, including precise bounding boxes, font attributes, and the exact textual content for each token. Unlike traditional image-based layout datasets—primarily tailored for computer vision applications—DocBank offers detailed, fine-grained annotations compatible with NLP tasks. The dataset encompasses approximately 500,000 document pages, stratified into training (400K pages), validation (50K pages), and testing (50K pages), ensuring comprehensive coverage of layout variability (however only a carefully picked subset of 101 pages were used in this research).

The utility of DocBank in this context stems from its ability to provide high-quality, token-level labels obtained via a weak supervision approach, which circumvents the extensive manual labeling typically required for such fine-grained annotations. By leveraging this dataset, the evaluation can be grounded in a robust, scalable benchmark that encapsulates the diversity of academic document layouts, font styles, and structural conventions.

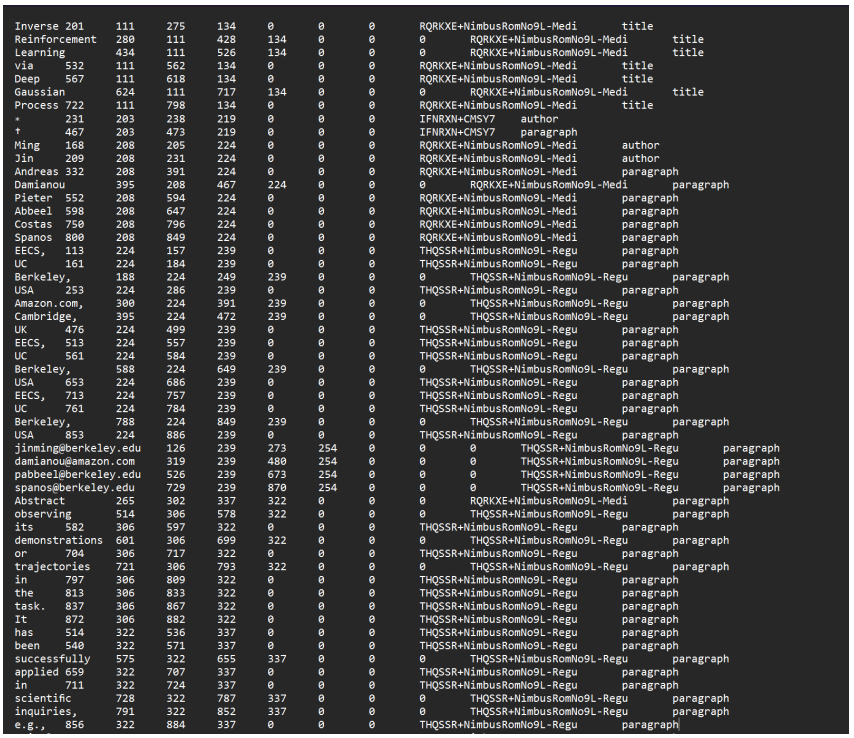

FIGURE 3.2: Example Annotation from DocBank Dataset

3.1.2 Sampling Strategy

To ensure that the evaluation reflects real-world variability, a representative subset of arXiv articles published between 2014 and 2018 was selected. The arXiv platform provides a comprehensive API that enables open, programmatic access to both metadata and full-text resources, which allowed systematic sampling across disciplines. In total, metadata for more than 6,000 papers was retrieved, spanning computer science, statistics, mathematics, electrical engineering and systems science, and economics. This multidisciplinary coverage was chosen to capture a wide range of layout styles, formatting conventions, and structural complexities.

However, not all of these candidate papers could be used directly. To construct a reliable benchmark, it was necessary to retain only those publications already present in the DocBank dataset, since DocBank provides the token-level annotations required for precise layout evaluation. The final corpus, of the first pages of 101 publications, therefore represents the **intersection of arXiv API metadata and DocBank annotations**. This filtering step ensured that each chosen document not only reflected authentic arXiv publishing practices but also came with high-quality ground-truth labels.

After applying this constraint, a curated subset of the first pages of 101 publications was obtained, distributed across disciplines to maintain diversity: 31 from computer science, 22 from statistics, 18 from mathematics, 18 from electrical engineering and systems science, and 12 from economics. This dataset size strikes a balance between statistical representativeness and feasibility for downstream processing and human oversight.

Category	Number of Files
Computer Science	31
Statistics	22
Mathematics	18
Electrical Engineering and Systems Science	18
Economics	12

TABLE 3.1: Distribution of files across disciplines.

Year	Number of Files
2014	23
2015	21
2016	28
2017	27
2018	2

TABLE 3.2: Distribution of files across years.

3.1.3 Text Construction

Once the dataset was assembled, a crucial step was to organize the token annotations into a canonical reading order. The fidelity of this ordering directly impacts the validity of subsequent evaluations, as it serves as the reference transcript against which the output of PDF parsers is compared. To this end, a multi-stage process integrating state-of-the-art layout analysis and reading order prediction models was employed.

Initially, each page was segmented into logical structural blocks—such as titles, section headings, paragraphs, figures, and tables—using a YOLOv12 object detection model trained explicitly for document layout analysis. YOLO (You Only Look Once) models are renowned for their real-time object detection capabilities, and YOLOv12 offers improved accuracy and efficiency for complex layout recognition tasks. The trained model was applied to each page, producing bounding boxes for various structural regions.

Following segmentation, the token-level annotations from DocBank were aggregated according to these identified segments. This grouping resulted in span-level text blocks that reflect semantically coherent regions. However, the correct reading order across these blocks is non-trivial, especially in multi-column or hybrid layouts. To infer this order, the grouped segments, along with any unassigned tokens, were processed through `LayoutReader`, a sequence-to-sequence model based on LayoutLM architecture designed explicitly for reading order detection.

LayoutReader leverages the spatial and textual features of layout elements to predict a sequence of segments that most closely resembles natural reading progression. Since LayoutReader is trained on span-level data derived from the Word dataset, which contains thousands of documents with annotated reading sequences, it effectively captures complex layout patterns. Applying this model yields an ordered sequence of segments, and concatenating the tokens within this sequence produces a high-fidelity, ground-truth textual transcript for each page.

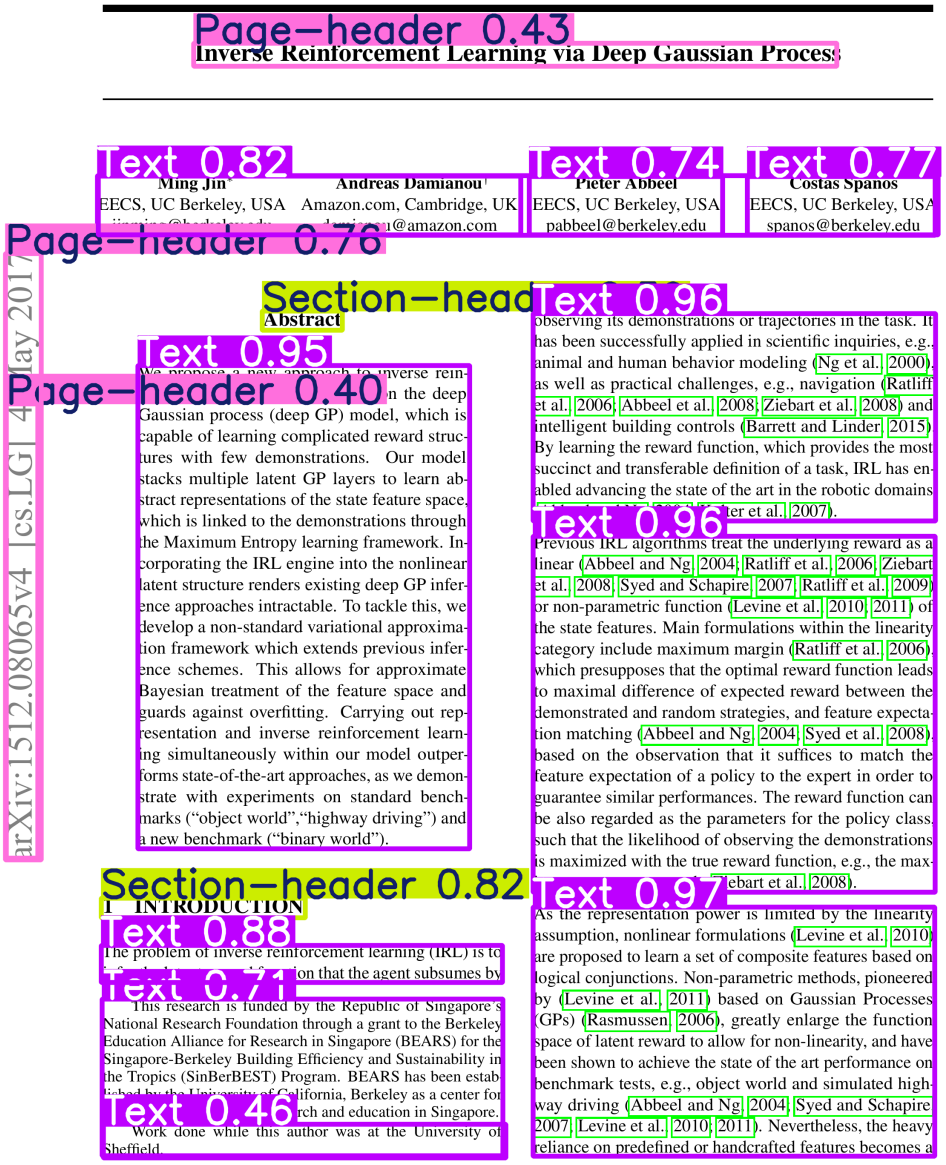

FIGURE 3.3: Example Document Layout Detection from YOLOv12

This organized, token-precise reading order forms the benchmark reference, enabling subsequent quantitative evaluation of the text extraction fidelity of various open-source parsing tools. The entire pipeline—from structural segmentation via YOLO, token grouping, to reading order prediction via LayoutReader—constitutes a comprehensive methodology for establishing an authoritative ground truth, essential for rigorous, reproducible evaluation in this research.

3.2 Novel Gold Standard Metadata Curation Approach

The development of a high-quality, reliable gold standard dataset was a foundational pillar of this research, underpinning the rigorous evaluation of various metadata extraction techniques. Given the inherent diversity and complexity of academic publications—characterized by a wide array of publisher formatting styles, column layouts, typographic conventions, and structural nuances—it was essential to construct a dataset that comprehensively captures this heterogeneity. Such a dataset ensures that the evaluation metrics are meaningful and that the proposed extraction methods are robust and generalizable across the broad spectrum of scholarly documents encountered in real-world scenarios.

Traditional methods for dataset creation often rely heavily on manual annotation or automated extraction from a limited set of models, which can inadvertently introduce biases, miss subtle variations, or lack sufficient diversity to test the limits of extraction systems. Recognizing these limitations, this research introduces a deliberately designed multi-model consensus framework, which represents a significant methodological contribution in its own right. Rather than treating the multi-model process as merely a preprocessing step, it is integrated as a core component of the data curation pipeline—systematically leveraging the complementary strengths of different language models to produce more accurate, reliable, and robust annotations. This approach exemplifies a principled strategy for harnessing AI diversity to improve ground truth quality, a concept that can inform future dataset creation efforts across various NLP and document analysis domains.

3.2.1 Multi-Model Consensus

The process begins with the careful selection of a representative sample of digital-born PDF documents. A total of 61 PDFs were chosen from a variety of reputable publishers, including PLOS, Elsevier, arXiv, Springer, PMLR, MDPI, and Frontiers Media. The selection process prioritized layout heterogeneity, ensuring inclusion of documents with varied stylistic features such as differing column configurations, heading hierarchies, font styles, and graphical artifacts. This diversity was crucial to ensure that the resulting dataset would serve as a resilient benchmark capable of evaluating extraction techniques across the full spectrum of academic publishing styles.

Once the sample was selected, content extraction was performed using the PyMuPdf library, which efficiently converts PDF pages into plain text. This conversion creates a standardized textual base suitable for interaction with large language models (LLMs). Before running the extraction models, a deliberate *prompt engineering* stage was introduced. Prompts were iteratively refined to enforce schema adherence, minimize hallucinations, and encourage JSON-formatted responses that explicitly mapped to the bibliographic fields. This stage ensured consistency in

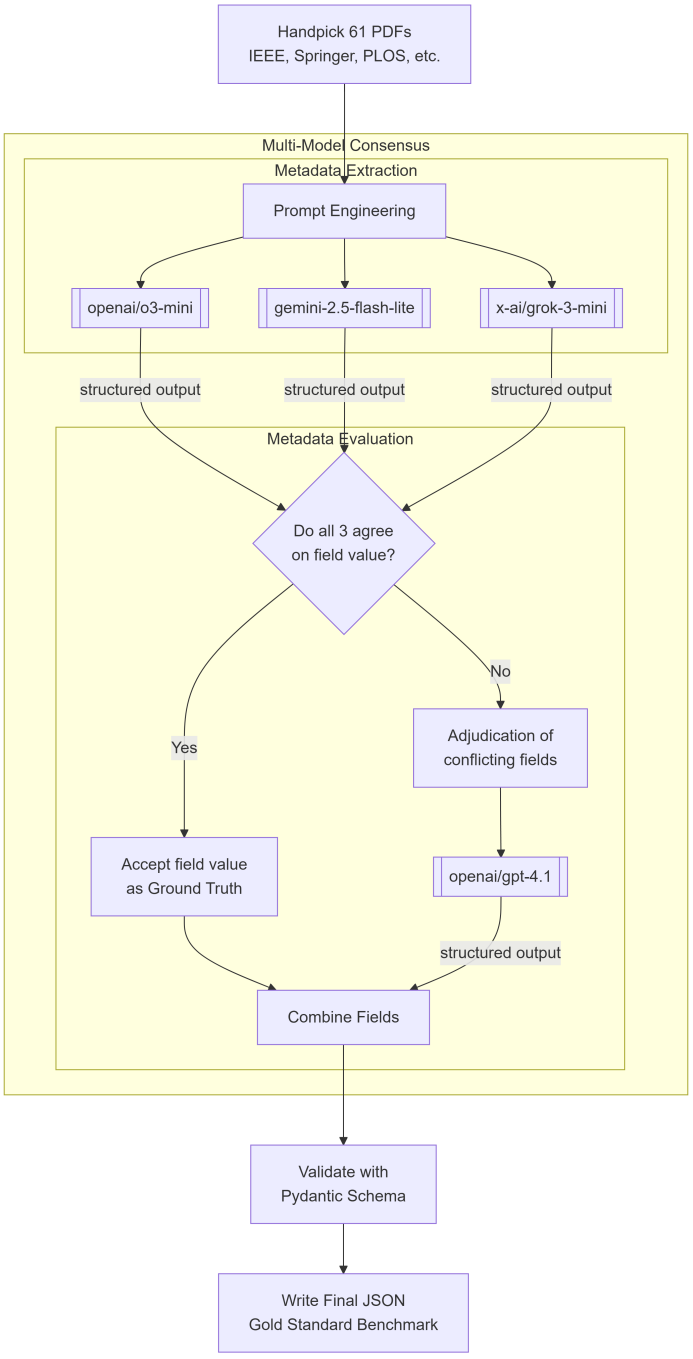

FIGURE 3.4: Novel Approach for Curation of Gold Standard Metadata.

the way different models approached the task, reducing noise in downstream consensus formation.

To generate metadata annotations, three diverse LLMs—DeepAI's GPT-3.5-turbo, Google's Gemini-2.5-flash-lite, and X-AI's Grok-3-mini—were employed. These models were chosen intentionally for their architectural differences, provider origins, and optimization strategies, thereby maximizing the diversity of potential outputs. For instance, Gemini is optimized for speed, GPT-3.5 emphasizes reliability, and Grok offers explicit reasoning capabilities. This diversity enables the system to capture a broad spectrum of potential annotations, facilitating

the identification of consensus and highlighting areas of uncertainty or ambiguity.

At the heart of the framework lies a deterministic *agreement checkpoint*. If all three models returned identical values for a given field, that value was accepted directly as ground truth. In cases where disagreement arose, the contested field was escalated to an adjudication stage. Here, GPT-4.1 was employed as a heavyweight, domain-aware evaluator. Crucially, GPT-4.1 did not regenerate the entire record but focused only on resolving disputed fields. Provided with both the extracted text and the conflicting candidate values, it selected the most accurate option based on contextual reasoning. This selective adjudication reduced computational overhead while avoiding the homogenization of all records to a single model's perspective.

Following either consensus or adjudication, the accepted field values were combined into a unified metadata record. This aggregation step ensured that the final annotation integrated both "no-conflict" and "adjudicated" fields into a single, internally consistent record.

The responses from all models were collected via the OpenRouter API, which enforced strict JSON compliance. The final annotation was further validated using Pydantic models, acting as a schema gatekeeper to ensure type correctness, mandatory field coverage, and structural uniformity. By preventing malformed or incomplete outputs from propagating into the ground truth dataset, this step safeguarded annotation integrity.

The fully validated metadata records were finally written to a machine-readable JSON-based benchmark. This design not only guaranteed scalability and reproducibility but also ensured compatibility with downstream metadata extraction evaluations. By producing a rigorously validated, diverse, and representative dataset, the framework delivers a new standard for high-fidelity ground truth construction.

This multi-model consensus and adjudication framework marks a key methodological advancement. By explicitly designing a system that leverages the strengths of different models—each with unique capabilities—and systematically resolving conflicts through a domain-aware evaluator, the approach sets a new benchmark for dataset creation. It emphasizes that reliable gold standard data can be constructed not solely through manual annotation but through a principled, AI-driven collaboration that combines diversity, specialization, and contextual reasoning. The resulting benchmark is robust, transferable, and scalable, providing a valuable resource for the advancement of metadata extraction research.

3.2.2 The Response Schema

`Pydantic model for structured metadata extraction and validation`

```
from pydantic import BaseModel, Field, ConfigDict
from typing import List
```

```python
class ExtractMetadata(BaseModel):
    """Structured metadata for an academic publication."""
    model_config = ConfigDict(extra="forbid")
    title: str = Field(
        ...,
        description="The full name identifying the academic publication.",
    )
    authors: List[str] = Field(
        ...,
        description="The names of individuals who wrote the publication.",
    )
    affiliations: List[str] = Field(
        ...,
        description="Institutions or organizations associated with the authors.",
    )
    email_ids: List[str] = Field(
        ...,
        description="Contact email IDs of the authors.",
    )
    publication_date: str = Field(
        ...,
        description="The date when the publication was officially published in
        DD-MM-YYYY format.",
    )
    publisher: str = Field(
        ...,
        description="The organization responsible for publishing the document.",
    )
    doi: str = Field(
        ...,
        description="A unique digital object identifier linking directly to the
        publication online.",
    )
    keywords: List[str] = Field(
        ...,
        description="Specific terms highlighting the main topics of the
        publication.",
    )
    abstract: str = Field(
        ...,
        description="A brief summary outlining the publication's content, methods,
        and findings.",
    )
```

3.2.3 Prompts

System Prompt to Extract Metadata

```
You are a helpful assistant extracting publication metadata from an academic paper's text.
Identify the following fields from the paper: Title, Authors, Affiliations, Email IDs, Publication Date, Publisher, DOI, Keywords, Abstract.
Provide the answer in JSON format with exactly these keys: title, authors, affiliations, email_ids, publication_date, publisher, doi, keywords, abstract.
Use an array of strings for "authors", "affiliations", "email_ids", and "keywords".
Use strings for the other fields. If a field is not found in the text, return an empty string or empty list for it.

Do NOT include any explanation, only output valid JSON.
```

System Prompt to Evaluate Metadata

```
You are a strict evaluator comparing metadata extracted by three different models from an academic paper.
You will be given the paper text and the these metadata fields: {{fields}}.
Your tasks:
1. Identify discrepancies or differences between the models' outputs for each field.
2. Check the actual paper text to determine which output (if any) is correct for each field.
3. Based on relevant text from the paper, decide the correct value for each metadata field. If the field is not found in the text, return an empty string or empty list for it.
4. After the analysis, output a JSON object with the correct values the fields.

Do NOT include any explanation, only output valid JSON.
```

3.3 Summary

This chapter outlines the comprehensive methodological framework established to evaluate and enhance the extraction of textual content and metadata from academic PDFs. It highlights two key components: the creation of a precise ground truth for PDF text extraction and the development of a high-quality gold standard for metadata annotation.

First, the chapter discusses the challenges inherent in accurately capturing the text from diverse scholarly PDFs—including layout heterogeneity, multi-column formats, and parser limitations—and introduces a rigorous sampling strategy that leverages the DocBank dataset. By

selecting a representative subset of arXiv papers across multiple disciplines, the researchers ensure their evaluation reflects real-world variability. They detail a sophisticated multi-stage process involving layout segmentation with YOLOv12 and reading order prediction with LayoutReader, culminating in a canonical, token-level ground truth that serves as a benchmark for parser performance.

Second, the chapter presents an innovative, AI-driven approach to constructing a reliable gold standard for metadata extraction. This multi-model consensus framework employs diverse large language models (LLMs)—such as GPT-3.5, Gemini, and Grok—to generate candidate annotations from PDF content. Discrepancies among these models are resolved through a domain-aware adjudication by GPT-4.1, ensuring high accuracy and consistency. The process incorporates strict validation steps, schema enforcement via Pydantic, and systematic conflict resolution, resulting in a scalable, robust, and representative dataset.

Overall, the chapter emphasizes that combining advanced layout analysis, state-of-the-art NLP models, and principled consensus mechanisms creates a solid foundation for evaluating and improving open-source PDF parsing tools and metadata extraction methods, advancing the reliability and reproducibility of scholarly document analysis.

Chapter 4. Implementation

The implementation chapter delineates the comprehensive methodologies and technical workflows employed to extract metadata from scholarly documents. It provides an in-depth account of the deployment strategies, data processing pipelines, and optimization techniques that underpin the system's functionality. The chapter emphasizes reproducibility and robustness by detailing the deployment configurations, processing workflows, and validation procedures.

Initially, the chapter discusses the deployment of GROBID via Docker containers, highlighting the differences between the full and lightweight images and their respective roles in balancing accuracy and computational efficiency. It describes the use of RESTful API clients, the process of parsing TEI-XML outputs, and the normalization routines implemented to standardize extracted metadata fields. The modular design of the pipeline ensures scalability and facilitates future enhancements, such as GPU support.

Subsequently, the chapter elaborates on the use of advanced transformer based language models for metadata extraction. It explains the selection and configuration of models, including the application of quantization techniques to optimize inference on limited hardware resources. The process of text extraction from PDFs using PyMuPDF, coupled with prompt engineering strategies, is described in detail. The importance of structured prompts, attention masks, and response parsing into JSON format is underscored, alongside validation mechanisms using Pydantic models.

4.1 Metadata Extraction Using GROBID

This section provides a comprehensive overview of the implementation details of the metadata extraction system utilizing GROBID. It elaborates on the deployment strategies, client configurations, PDF processing workflows, and data handling mechanisms employed throughout the research. The focus is placed on technical specifics to ensure reproducibility and clarity regarding the system architecture and operational procedures.

4.1.1 Deployment and Python Client

The deployment of GROBID is achieved through Docker, enabling consistent, isolated, and portable environments for processing scholarly documents. Two distinct Docker images are employed, each optimized for different operational requirements and resource constraints.

The first image, designated as the "full" image, with the tag `grobid/grobid:0.8.2`. This image, approximately 8GB in size, incorporates all necessary dependencies for advanced document processing, including deep learning models, CRF-based models, and associated libraries. It supports

GPU acceleration, leveraging TensorFlow or PyTorch frameworks, which significantly enhances the accuracy of reference parsing and citation context extraction. The inclusion of pre-trained embeddings—over 5GB of data—and comprehensive Python libraries (totaling approximately 3GB) facilitates high-precision extraction tasks. Due to current limitations in Docker's support for GPU acceleration on Windows and MacOS platforms, GPU resources are not utilized in this implementation. Consequently, all containers operate using CPU-only mode, which may impact processing speed, especially when utilizing the full image with deep learning models.

The second image, referred to as the "lightweight" image (`lfoppiano/grobid:0.8.2`), is approximately 500MB in size. It contains only the Conditional Random Field (CRF) models necessary for fundamental metadata extraction, such as title, authors, and publication date. This image is optimized for environments with limited computational resources, offering faster processing times and reduced memory footprint but at the expense of some extraction accuracy, particularly in citation parsing and relation extraction.

The system employs the GROBID Python client, a lightweight and efficient wrapper facilitating interaction with GROBID's RESTful API services. The client is configured to support concurrent processing, allowing multiple requests to be handled in parallel, thereby optimizing throughput when processing large document batches.

The client supports configurable parameters, including server URL, timeout durations, and processing options such as layout analysis, segmentation, and reference extraction. It enables layout-aware sentence segmentation, which is crucial for maintaining contextual accuracy during text parsing. The client's concurrency features harness Python's asyncio or multi-threading modules, depending on implementation, to process multiple PDFs or reference strings simultaneously.

The client initiates REST API requests to the GROBID server instances running within Docker containers. Proper error handling mechanisms are implemented to manage network failures, timeout exceptions, and server errors, ensuring robustness during batch processing.

4.1.2 PDF Processing Workflow

The core of the implementation revolves around the systematic processing of PDF documents to extract structured metadata. The workflow comprises several sequential steps, detailed as follows.

A dedicated Python script iterates over a specified directory containing the PDFs. For each document, the script invokes the GROBID client's `processHeader` method, which submits the PDF to the GROBID server for header and metadata extraction. The process leverages asynchronous calls to maximize throughput, especially when processing large corpora.

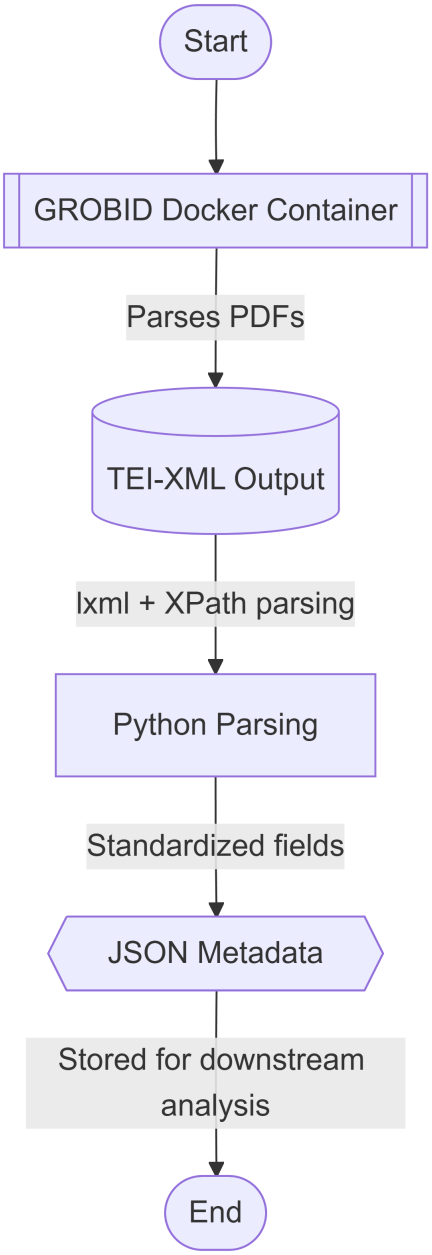

FIGURE 4.1: GROBID Workflow

GROBID returns the processed data in TEI-XML format, encapsulating metadata such as the document's title, list of authors, affiliations, publication date, publisher details, DOI, keywords, and abstract. The TEI-XML structure adheres to the standards defined by the Text Encoding Initiative, facilitating reliable parsing and data extraction.

Using the `lxml` Python library, the TEI-XML content is parsed to locate specific XML elements corresponding to each metadata field. XPath expressions are employed to accurately retrieve the relevant tags. The extracted data undergoes normalization procedures to standardize formats, particularly for the publication date. The date string is parsed using Python's `datetime` library, accommodating various date representations. It is noteworthy that GROBID's extraction capabilities do not include email addresses, which are omitted from the metadata. Missing

fields are handled by assigning empty strings in the JSON output, ensuring data consistency.

The extracted metadata for each document is stored in structured JSON files, facilitating downstream analysis. Each JSON object contains key-value pairs representing the metadata fields, with consistent key naming conventions. The process includes error logging for failed extractions, enabling debugging and iterative improvements.

The implementation employs a modular architecture, combining Docker-based deployment of GROBID with a Python-based processing pipeline. The choice between the full and lightweight images provides flexibility depending on accuracy requirements and resource availability. The use of a robust Python client and structured XML parsing ensures precise metadata extraction, while normalization routines prepare the data for subsequent analysis. Future enhancements may include GPU support.

`Python Client for GROBID`

```python
from grobid_client.grobid_client import GrobidClient

client = GrobidClient(config_path="./grobid_config.json")
client.process(
    service="processHeaderDocument",
    input_path="./data/pdf",
    output="./outputs/tei",
    consolidate_header=True,
)
```

`Parsing TEI-XML to JSON`

```python
from lxml import etree
from re import sub
from datetime import datetime


def parse_grobid_tei(tei_path) -> dict:
    ns = {"tei": "http://www.tei-c.org/ns/1.0"}
    root = etree.parse(tei_path)
    tex = lambda xp: sub(r"\s+", " ", (root.findtext(xp, namespaces=ns) or
    "").strip())
    many = lambda xp: list(dict.fromkeys([sub(r"\s+", " ", "
    ".join(s.itertext()).strip()) for s in root.findall(xp, namespaces=ns)]))
    def date(xp):
        date_value = root.xpath(xp, namespaces=ns)[0] if root.xpath(xp,
        namespaces=ns) else ""
```

```
        if date_value:
            try:
                date_value = datetime.strptime(date_value,
                "%Y-%m-%d").strftime('%d-%m-%Y')
            except ValueError:
                try:
                    date_value = datetime.strptime(date_value,
                    "%Y-%m").strftime('%m-%Y')
                except ValueError:
                    date_value = datetime.strptime(date_value, "%Y").strftime('%Y')
        return date_value
    data = {
        "title": tex(".//tei:titleStmt/tei:title"),
        "authors": many(".//tei:persName"),
        "affiliations": many(".//tei:affiliation"),
        "publication_date": date(".//tei:publicationStmt/tei:date/@when"),
        "publisher": tex(".//tei:publicationStmt/tei:publisher"),
        "doi": tex('.//tei:idno[@type="DOI"]'),
        "keywords": many(".//tei:profileDesc/tei:textClass/tei:keywords/tei:term"),
        "abstract": " ".join(root.find(".//tei:abstract",
        namespaces=ns).itertext()).strip()
                    if root.find(".//tei:abstract", namespaces=ns) is not None else
                    ""
    }
    return {k: data.get(k, "") for k in FIELDS}
```

4.2 Metadata Extraction Using Language Models

This section provides a comprehensive overview of the technical implementation process for extracting metadata from academic PDFs using advanced language models. The implementation process encompasses the extraction of raw text from PDFs, the utilization of state-of-the-art transformer-based models for metadata extraction, optimization techniques to improve efficiency, and validation procedures to ensure the integrity of the extracted data.

4.2.1 Models and Configurations

For the core task of metadata extraction, multiple transformer-based language models were utilized, each selected based on their demonstrated capabilities in instruction following and contextual understanding. The models included:

1. Qwen/Qwen2.5-3B-Instruct
2. microsoft/Phi-4-mini-instruct
3. meta-llama/Llama-3.2-3B-Instruct

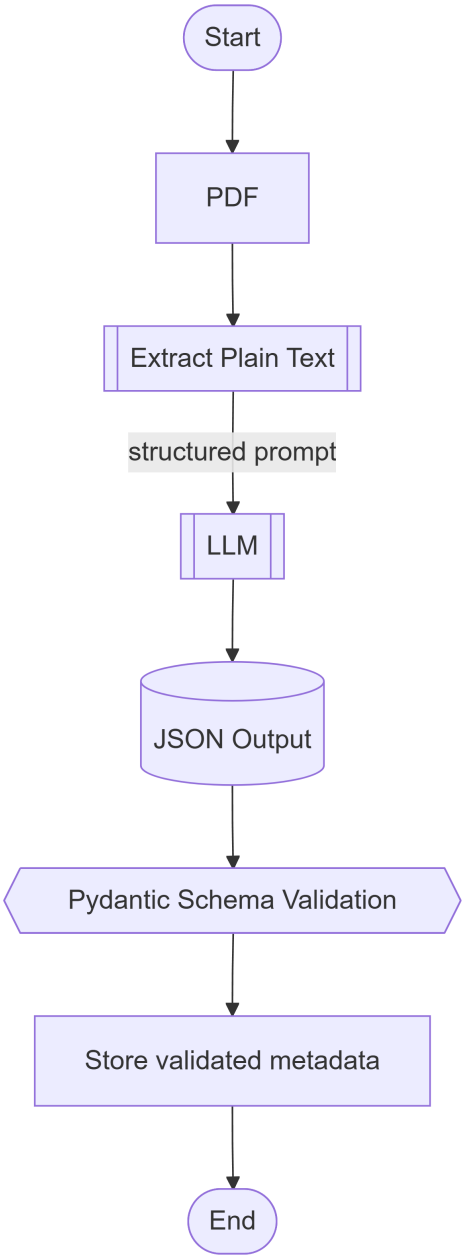

FIGURE 4.2: Language Model Workflow

4. GPT-OSS-20B

The first three models were accessed via the `Hugging Face Transformers` library, employing `AutoModelForCausalLM` for loading the models and `AutoTokenizer` for tokenization. The causal language modeling paradigm was chosen because it aligns well with prompt-based instruction following, allowing the models to generate structured responses based on provided prompts. The GPT-OSS-20B model was incorporated with specific modifications due to compatibility issues.

Given the substantial size of the models, optimization was crucial to ensure feasible inference within limited hardware resources. To this end, 4-bit quantization was implemented utilizing

the `BitsAndBytesConfig` class. This quantization reduces the model's memory footprint and accelerates inference times without significantly compromising model accuracy.

The quantization process involved configuring the `BitsAndBytesConfig` with parameters such as `load_in_4bit=True` and setting appropriate compute data types. During model loading, these configurations enabled the models to operate efficiently on GPU hardware with limited memory capacity.

For the GPT-OSS-20B model, which exhibited incompatibility issues with the quantization configuration on the available GPU hardware, an alternative deployment approach was adopted. This involved serving the model via the `Ollama` platform, which allows for efficient model hosting and inference without the constraints of local GPU memory. This setup bypassed the need for quantization and facilitated the integration of GPT-OSS-20B into the pipeline.

Throughout the implementation, PyTorch was used to monitor memory consumption and execution time. Custom scripts tracked maximum GPU memory utilization via `torch.cuda.max_memory_reserved()` and `torch.cuda.max_memory_allocated()`, providing insights into the model's resource requirements during inference.

`BitsAndBytes Quantization Setup`

```
from transformers import AutoModelForCausalLM, AutoTokenizer, BitsAndBytesConfig
import torch

quant_config = BitsAndBytesConfig(load_in_4bit = True,
bnb_4bit_compute_dtype=torch.bfloat16)
model = AutoModelForCausalLM.from_pretrained("Qwen/Qwen2.5-3B-Instruct",
                                             quantization_config=quant_config,
                                             torch_dtype=torch.float16,
                                             trust_remote_code=False,
                                             device_map="auto")
tok = AutoTokenizer.from_pretrained("Qwen/Qwen2.5-3B-Instruct")
```

4.2.2 Prompt Engineering and Response Extraction

The initial step involved parsing the content of academic PDFs to obtain clean, plain text data suitable for processing by language models. For this purpose, the PyMuPDF library (also known as fitz) was employed. PyMuPDF provides efficient methods for reading PDF files and extracting textual content. During inference it was noticed that including multiple pages not only increased token usage but also degraded the instruction-following capacity of the models, frequently resulting in incomplete or malformed responses that failed downstream

Aspect	**Raw-Text Parsing**	**Schema-Validated Pipeline**
Output Reliability	Inconsistent.	Consistent.
Parsing Complexity	Regex/heuristics.	JSON parsing.
Error Detection	Manual.	Automatic.
Type Safety	Unstructured, ambiguous types.	Typed models (Pydantic).
Pipeline Robustness	Breaks with format changes.	Resilient through enforced schema.

TABLE 4.1: Comparison between raw-text parsing and schema-validated LLM pipeline.

parsing. Restricting the input to the first page, which typically contains most bibliographic metadata, offered a robust compromise between coverage and reliability.

The extraction process involved opening each PDF document using PyMuPDF's fitz.open() function, followed by accessing the first page via `doc.load_page(0)`. The text was then retrieved using the `page.get_text()` method, which extracts the textual content in a structured format. This plain text served as the input for the subsequent metadata extraction stage.

To optimize model performance, a structured prompt template was employed. The prompt incorporated a chat-style format, leveraging the models' superior instruction following capabilities in this format. This template consisted of a system message with a structured response schema outlining the task, followed by the user message containing the extracted PDF text and explicit instructions to identify and extract specific metadata fields such as the title, authors, publication year, and abstract.

An attention mask was created to focus the model's attention on the relevant parts of the input, especially when handling longer texts. Special tokens unique to each model's tokenizer were utilized to demarcate input boundaries and ensure proper parsing of the generated responses.

The models' outputs were obtained in a raw text format, which necessitated structured parsing to extract the metadata fields. The responses were parsed into JSON format, with the expectation that models would follow the prompt instructions to output data in a predefined format. Unlike naive regex extraction from unstructured responses, this approach expects the LLM to output directly in structured key-value JSON format.

To verify the integrity and correctness of the responses, the parsed JSON was validated using Pydantic models. These models defined the expected schema for each metadata field, enforcing data types and required fields. This ensures type correctness, required-field presence, and catches format or structural errors—bringing contract-driven validation into the pipeline. This approach is well-recognized in modern LLM workflows for boosting reliability and reducing parsing brittleness.

The validated metadata fields were saved into JSON files, organized per document, to facilitate further analysis. These files formed the basis for subsequent evaluation stages, including accuracy assessment against ground truth metadata and error analysis. This implementation approach balances the technical challenges associated with large language model inference, token limitations, and hardware constraints. It emphasizes optimization, validation, and structured processing to ensure high-quality metadata extraction from academic PDFs.

`System Prompt to Extract Metadata`

```
You are given the text from an academic publication. Your task is to extract metadata from the given text.
The metadata fields you should extract are: Title, Authors, Affiliations, Email IDs, DOI, Publisher, Publication Date, Keywords, and Abstract.
If any metadata information is missing, leave it blank.
Return a JSON with this schema:
{
    "additionalProperties": false,
    "description": "Structured metadata for an academic publication.",
    "properties": {
        "title": {
            "description": "The full name identifying the academic publication.",
            "title": "Title",
            "type": "string"
        },
        "authors": {
            "description": "The names of individuals who wrote the publication.",
            "items": {
                "type": "string"
            },
            "title": "Authors",
            "type": "array"
        },
        "affiliations": {
            "description": "Institutions or organizations associated with the authors.",
            "items": {
                "type": "string"
            },
            "title": "Affiliations",
            "type": "array"
        },
        "email_ids": {
            "description": "Contact email IDs of the authors.",
```

```
            "items": {
                "type": "string"
            },
            "title": "Email Ids",
            "type": "array"
        },
        "publication_date": {
            "description": "The date when the publication was officially published
            in DD-MM-YYYY or MM-YYYY or YYYY format.",
            "title": "Publication Date",
            "type": "string"
        },
        "publisher": {
            "description": "The organization responsible for publishing the
            document.",
            "title": "Publisher",
            "type": "string"
        },
        "doi": {
            "description": "A unique digital object identifier linking directly to
            the publication online.",
            "title": "Doi",
            "type": "string"
        },
        "keywords": {
            "description": "Specific terms highlighting the main topics of the
            publication.",
            "items": {
                "type": "string"
            },
            "title": "Keywords",
            "type": "array"
        },
        "abstract": {
            "description": "A brief summary outlining the publication\u2019s
            content, methods, and findings.",
            "title": "Abstract",
            "type": "string"
        }
    },
    "required": [
        "title",
        "authors",
        "affiliations",
```

```
        "email_ids",
        "publication_date",
        "publisher",
        "doi",
        "keywords",
        "abstract"
    ],
    "title": "ExtractMetadata",
    "type": "object"
}
Do not add any preamble or explanations.
```

User Prompt Template to Extract Metadata

```
## Text:
{text}

## Output:
```

Extract Metadata Using Language Models

```
def get_pdf_text(file):
    doc = pymupdf.open(file)
    pdf_text = [page.get_text() for page in doc.pages(0, 1)]
    return pdf_text[0]
response_start_token = "<|im_start|>assistant\n"
response_end_token = "<|im_end|>"
user_prompt = user_prompt_template.format(text=get_pdf_text(file))
messages = [{"role": "system", "content": system_prompt},
                {"role": "user", "content": user_prompt}]
inputs = tok.apply_chat_template(messages, add_generation_prompt=True,
return_tensors="pt", padding=True).to(model.device)
attention_mask = torch.ones_like(inputs)
with torch.inference_mode():
    generated = model.generate(
        inputs,
        attention_mask=attention_mask,
        max_new_tokens=1024,
        return_dict_in_generate=True,
        output_scores=True,
        pad_token_id=tok.eos_token_id,
    )
```

```
seq = generated.sequences[0]
text = tok.decode(seq, skip_special_tokens=False)
clean_text = text.split(response_start_token)[1]
                 .split(response_end_token)[0]
                 .replace("\n", "")
                 .replace("```json","").replace("```","")
try:
    op = loads(clean_text)
except JSONDecodeError as e:
    op = {
        "title": "",
        "authors": [],
        "affiliations": [],
        "email_ids": [],
        "doi": "",
        "publisher": "",
        "publication_date": "",
        "keywords": [],
        "abstract": ""
    }
    print("Incomplete Output!", e)
try:
    metadata = ExtractMetadata(**op)
except ValidationError as e:
    print("Pydantic Validation Failed!", e)
```

4.3 Summary

This chapter summarized the development of a comprehensive metadata extraction framework combining layout-aware and language model approaches. It detailed the deployment of GROBID using Docker, including configurations for full and lightweight images, and explained the PDF processing pipeline involving TEI-XML parsing, normalization, and JSON output. The chapter also covered the implementation of transformer-based models, highlighting preprocessing with PyMuPDF, prompt design, inference with instruction-tuned models, and techniques like 4-bit quantization for efficiency. Validation with Pydantic ensured schema consistency, and structured JSON files supported downstream tasks. Overall, the system emphasizes reproducibility, modularity, and robustness, forming a reliable foundation for subsequent evaluation.

Chapter 5. Evaluation

This chapter presents an evaluation of the reliability, order fidelity, and computational cost of PDF text extraction and metadata parsing pipelines. The assessment is organized in two parts. In the first part, plain-text extraction quality from born-digital PDFs is quantified, focusing on character- and word-level fidelity as well as sequence preservation. In the second part, structured metadata extraction is examined for key bibliographic fields, covering both specialized parsers and instruction-tuned language models.

All experiments are based on aligned references, which serve as ground truth. To ensure comparability across tools, both predictions and references were normalized using a uniform pipeline: lowercasing, removal of Unicode control characters, diacritic stripping through Unicode decomposition, trimming of leading and trailing whitespace, and collapsing of multiple internal spaces. For text extraction quality, Character Error Rate (CER) and Word Error Rate (WER) were computed with the `jiwer` library, while order-sensitive metrics were captured using BLEU scores (via `sacrebleu`, scaled to [0,1]) and ROUGE-L (via `rouge_score`). For metadata evaluation, the choice of metric was determined by field type: Levenshtein distance (`strsimpy`) for short string fields such as titles and DOIs, F1 score with `MultiLabelBinarizer` for list-valued fields such as authors or affiliations, and TF–IDF cosine similarity (`scikit-learn`) for abstracts.

Out-of-the-box configurations were employed for the Python libraries `pymupdf`, `pypdfium2`, `pdfminer.six`, and `pypdf2`. The `pdfalto` executable was executed to produce ALTO XML, which was subsequently post-processed to obtain plain text. For metadata extraction, two variants of `GROBID` (CRF and deep learning) were compared against instruction-tuned language models (`microsoft/Phi-4-mini-instruct`, `Qwen/Qwen2.5-3B-Instruct`, `Qwen/Qwen3-4B-Instruct`, `meta-llama/Llama-3.2-3B-Instruct`) and the large generative model `gpt_oss_20b`. In addition to accuracy-based measures, end-to-end wall-clock time and peak memory usage were recorded. For GPU-enabled models, device memory allocation was monitored through the PyTorch runtime. Results are presented both as aggregated metrics and as distributions (box plots) to illustrate central tendencies, variability, and the presence of outliers under uniform, untuned conditions.

5.1 Evaluating PDF Parsing Tools

This section presents a comprehensive evaluation of several open-source PDF parsing tools, focusing on their effectiveness in extracting plain text from PDF documents. The primary goal is to assess the accuracy of the text extraction process, the preservation of reading order, and the consistency of performance across various document layouts and encodings. Five tools

were selected for this purpose: `pymupdf`, `pdfium2`, `pdfminer.six`, `pypdf2`, and `pdfalto`. The first four are Python libraries providing straightforward interfaces for page-wise text extraction, while `pdfalto` is a command-line tool that generates structured XML representations, which are then processed to obtain plain text.

The evaluation involved the extracted and ordered text obtained from DocBank dataset annotations as explained in the third chapter. To ensure fair comparison, a normalization pipeline was applied to all extracted texts and reference texts.

Quantitative metrics, including Character Error Rate (CER) and Word Error Rate (WER), were computed using the `jiwer` library to measure fidelity at the character and word levels. Additionally, order-sensitive metrics such as BLEU and ROUGE-L scores were employed to evaluate the preservation of reading order and semantic coherence. These metrics were calculated using `sacrebleu` and `rouge_score` libraries, respectively. The distribution of scores across documents was visualized through box plots generated with `matplotlib` and `seaborn`, to better understand variability and identify outliers.

5.1.1 Experiment Setup

The tools examined include `pymupdf`, `pdfium2`, `pdfminer.six`, and `pypdf2`. Each was used with default settings to simulate out-of-the-box performance, representing typical usage scenarios without specialized tuning. For `pdfalto`, the command-line executable was employed outside the Python environment, owing to its dependency on system compilers such as GCC and Clang. The output XML files generated by `pdfalto` were parsed with a custom script to extract plain text.

To enable fair comparison across tools, all extracted text was normalized prior to evaluation. Normalization involved multiple stages: first, all text was converted to lowercase to eliminate case sensitivity. Next, Unicode control characters were filtered out to remove formatting artifacts and non-printable symbols. Diacritics and accents were stripped through Unicode decomposition, ensuring that characters such as “é” and “e” were treated equivalently. Excess whitespace at line boundaries was removed, and multiple internal spaces were collapsed into a single space. Identical normalization was applied to the gold-standard reference text sourced from the DocBank dataset, which provides aligned ground truth representations of scholarly PDFs. This process ensured that subsequent evaluation focused purely on content fidelity and order preservation, independent of superficial formatting differences.

Two sets of metrics were computed to capture complementary aspects of extraction quality. For character- and word-level fidelity, Character Error Rate (CER) and Word Error Rate (WER) were calculated using the `jiwer` library. CER quantifies the proportion of characters that differ between extracted text and ground truth, providing sensitivity to even small deviations. WER

extends this notion to the word level, penalizing entire tokens when character-level differences cause word mismatches.

```
metrics["WER"] = round(wer(ref_text, hyp_text), 4)
metrics["CER"] = round(cer(ref_text, hyp_text), 4)
```

To measure preservation of word order and semantic coherence, BLEU and ROUGE-L scores were employed. BLEU was calculated with the `sacrebleu` library, which reports values on a 0–100 scale; these were normalized to the [0,1] range. ROUGE-L, computed using the `rouge_score` library, captures longest common subsequence overlap between extracted and reference text.

```
bleu = sacrebleu.corpus_bleu([hyp_text], [[ref_text]])
metrics["BLEU"] = round(bleu.score / 100.0, 4)

rouge = rouge_scorer.RougeScorer(['rougeL'], use_stemmer=True)
metrics["ROUGE"] = round(
    rouge.score(target=ref_text, prediction=hyp_text)['rougeL'].fmeasure, 4
)
```

Together, these four metrics provide a comprehensive view: CER and WER capture basic textual accuracy, while BLEU and ROUGE-L emphasize order-sensitive and semantic alignment.

For each tool, metric values were collected over the full test set of documents. The distributions were visualized as box plots using `matplotlib` and `seaborn`, allowing comparison of median performance, spread, and outliers. In addition to visual inspection, average scores were computed to summarize overall tendencies, though interpretation is based primarily on observed distributional characteristics.

5.1.2 Error Rates

Figure 5.1 presents a comparative evaluation of text extraction quality across five open-source PDF parsing tools, measured using both Character Error Rate (CER) and Word Error Rate (WER). The results highlight several important trends regarding the relative accuracy and robustness of these tools.

First, it is evident that all tools achieve relatively low median error rates, with CER values remaining well below 0.3% across the board. This suggests that, on average, the character-level fidelity of extracted text is reasonably reliable regardless of the parser employed. However, a consistent increase in error magnitude is observed when shifting from CER to WER, which is

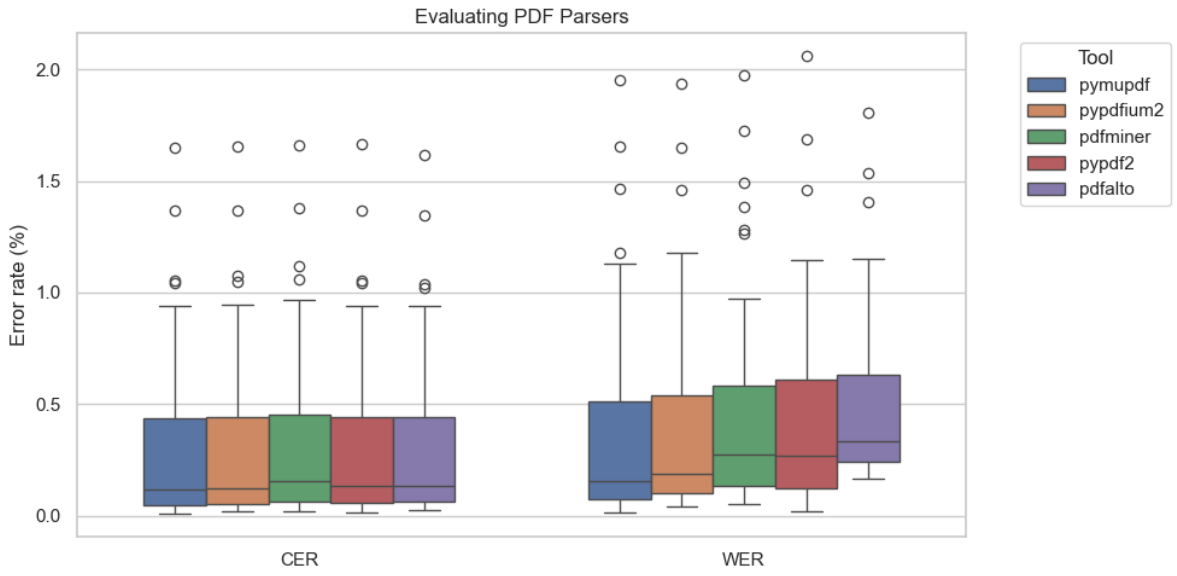

FIGURE 5.1: Error rate in PDF parsers

expected since even a single character-level error can propagate into a complete word mismatch. Consequently, WER provides a stricter measure of extraction quality, reflecting the impact of localized errors on downstream natural language processing tasks.

When comparing tools, the distributions indicate broadly similar performance profiles, yet subtle differences can be discerned. Both `pypdfium2` and `pymupdf` achieve marginally lower median WER values and demonstrate a tighter interquartile range, suggesting greater consistency and reliability across diverse documents. In contrast, `pdfminer` and `pdfalto` exhibit wider spreads, implying more variable performance and a higher likelihood of significant degradation on challenging inputs. Notably, `pdfalto` shows a slightly elevated median WER relative to the other tools, further reinforcing this observation.

Another notable feature of the results is the prevalence of outliers. All five tools display numerous cases with error rates exceeding 1.0, and in some instances approaching or surpassing 2.0. These extreme values suggest that none of the evaluated parsers is universally robust, with certain document layouts or encodings posing substantial difficulties. The higher concentration of outliers for `pdfminer` and `pdfalto` underscores their susceptibility to such problematic cases, whereas `pypdfium2` and `pymupdf` appear relatively more resilient.

5.1.3 Reading Order Measures

Figure 5.2 reports order-sensitive quality measures (BLEU and ROUGE) for five PDF text extraction tools. These metrics complement error-based evaluations (CER, WER) by focusing on the preservation of word sequences and semantic coherence, which are particularly relevant for downstream tasks.

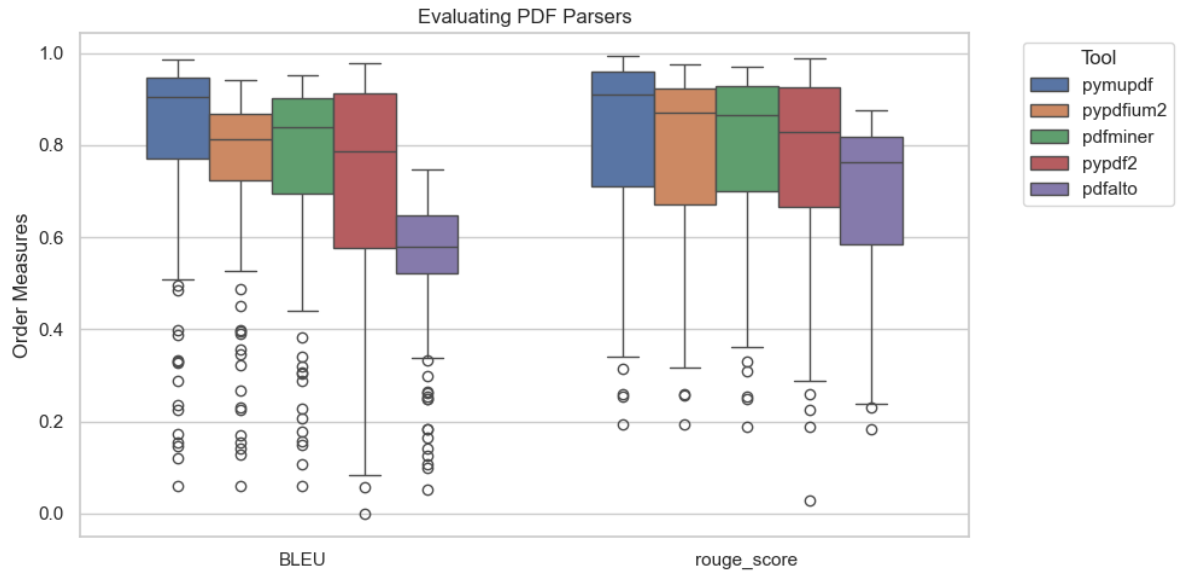

FIGURE 5.2: Measure of reading order in PDF parsers

Overall, the results show that all tools achieve relatively high median BLEU and ROUGE scores, with values typically exceeding 0.7. This indicates that the extracted text generally maintains a high degree of structural and semantic fidelity to the ground truth. However, substantial variability exists across tools and documents, as reflected in the interquartile ranges and the presence of numerous outliers.

Among the evaluated tools, `pymupdf` and `pypdfium2` stand out with consistently high medians across both BLEU and ROUGE, along with relatively tight interquartile ranges. Their distributions remain concentrated near the upper end of the scale, highlighting their reliability in maintaining word order and preserving textual semantics. `pdfminer` shows similar median performance but with a slightly wider spread, suggesting more variability in its robustness across different documents.

In contrast, `pypdf2` and `pdfalto` underperform relative to the other tools. `pypdf2` exhibits a lower median BLEU score with a broad range, indicating less consistent word order preservation, although its ROUGE performance is closer to the mid-range of other tools. `pdfalto`, while achieving reasonable scores on some documents, displays the lowest medians and the widest spread across both metrics, reflecting reduced reliability and greater sensitivity to complex layouts.

Outliers are prevalent across all tools, with BLEU scores occasionally dropping below 0.2. These cases correspond to documents where the extracted text diverges significantly from the reference sequence, confirming that none of the parsers is universally robust. However, the lower frequency of such extreme degradations for `pymupdf` and `pypdfium2` reinforces their superior stability.

5.2 Evaluation of Metadata Extraction

The evaluation in this section examines the ability of different systems to extract structured metadata fields from scholarly PDF documents. The models under consideration span both specialized sequence labeling frameworks and transformer based language models. From the former category, two variants of `GROBID` are included: the deep learning-based implementation, which relies on BiLSTM-CRF architectures with layout features, and the traditional Conditional Random Field (CRF) model trained with Wapiti. From the latter category, four instruction-tuned language models are tested:
`microsoft/Phi-4-mini-instruct`, `Qwen/Qwen2.5-3B-Instruct`, `Qwen/Qwen3-4B-Instruct`, and `meta-llama/Llama-3.2-3B-Instruct`. In addition, the open-source generative model `gpt_oss_20b` is evaluated to represent a large-scale language model with 21 billion parameters. This collection of systems provides a balanced comparison between lightweight, domain-specific models and general-purpose generative models.

The metadata fields evaluated are: *title*, *authors*, *affiliations*, *email addresses*, *publication date*, *publisher*, *digital object identifier (DOI)*, *keywords*, and *abstract*. These fields were selected because they represent the most common bibliographic attributes required for indexing and citation workflows. They also span multiple data types, from short strings such as DOIs to longer free-text segments such as abstracts, necessitating the use of different evaluation metrics.

For fields that consist of short string values— *title*, *publication date*, *publisher*, and *DOI*— the `Levenshtein distance` is employed. This metric computes the minimum number of edit operations (insertions, deletions, substitutions) required to transform the predicted output into the gold-standard reference. It is well-suited to capturing localized transcription errors and is sensitive to even minor deviations in short string fields where exact matching is essential.

For fields that naturally take the form of lists, such as *authors*, *affiliations*, *email addresses*, and *keywords*, the F1 score is used. These fields are evaluated as sets of items rather than continuous text sequences. Precision and recall are first computed to quantify, respectively, how many of the predicted items are correct and how many of the reference items were successfully retrieved. The harmonic mean of these two quantities yields the F1 score, which provides a balanced view of completeness and correctness. Implementation is carried out using `sklearn`'s `MultiLabelBinarizer`, enabling list-type data to be represented as binary vectors for direct metric computation.

For the *abstract* field, which consists of longer continuous text, cosine similarity between TF-IDF vectors is adopted as the evaluation measure. Unlike Levenshtein distance, which becomes less informative for long strings, cosine similarity captures the degree of semantic overlap in terms of word frequency distributions. The TF-IDF representation emphasizes terms that are distinctive in a given abstract, making the similarity score more sensitive to meaningful differences in

content rather than superficial overlaps. This metric is implemented using the `scikit-learn` library.

Prior to evaluation, both system outputs and gold-standard reference texts undergo identical normalization procedures to remove superficial variations. This normalization ensures that differences in formatting, case, or encoding do not affect the evaluation, allowing the metrics to focus solely on substantive differences in extracted content.

Levenshtein distance is computed using the `strsimpy` library. TF-IDF cosine similarity is calculated with the vectorization and similarity modules of `scikit-learn`. F1 scores for list-based fields are computed using `sklearn`'s `MultiLabelBinarizer` in combination with its precision-recall evaluation utilities. For each metric and field, results are averaged over the entire set of documents to obtain aggregate performance values.

In addition to accuracy, system efficiency is also recorded. The total wall-clock time required to process the full set of documents is measured for each model. For the transformer models (`microsoft/Phi-4-mini-instruct`, `Qwen/Qwen2.5-3B-Instruct`, `Qwen/Qwen3-4B-Instruct` and `meta-llama/Llama-3.2-3B-Instruct`), GPU and CPU utilization is monitored during inference. These values represent the peak GPU memory allocation and maximum CPU memory usage observed while processing. Resource monitoring is implemented using the PyTorch runtime, which provides programmatic access to memory consumption statistics.

This evaluation framework thus provides a multifaceted view of model performance, balancing field-specific accuracy metrics with computational efficiency indicators. The combination of Levenshtein distance, F1 score, and cosine similarity allows each type of metadata field to be assessed with an appropriate measure, while runtime and resource profiling highlight the practical requirements of deploying these systems at scale.

5.2.1 Accuracy Metrics by Field

The evaluation reveals clear contrasts between layout-aware parsers, small language models, and the large generative model. Beginning with the *abstract*, measured by cosine similarity, the layout-aware models perform strongly, with `grobid_dl` achieving the highest score at nearly 0.98 and `grobid_crf` also above 0.96. Among the small language models, performance is more varied: `qwen3b` and `phi4_mini` remain competitive at around 0.87–0.89, while `llama3b` falls behind markedly, dropping to 0.67. At the other end of the spectrum, the large language model `gpt_oss_20b` matches the top tier, recording a similarity of 0.96.

In the case of *author extraction*, measured using F1 score, the large model, `gpt_oss_20b`, again performs exceptionally well, scoring around 0.92. The small models show a clearer ranking: `qwen3b` reaches similar heights at 0.91, followed by `llama3b` at 0.89, and then `phi4_mini` at

TABLE 5.1: Performance comparison of metadata extraction models across fields and metrics.

Field	**Metric**	**GROBID-DL**	**GROBID-CRF**	**GPT-OSS**	**Phi4-Mini**	**Qwen3B**	**Qwen4B**	**Llama3B**
title	Lev Dist	0.67	3.04	0.78	4.26	0.75	24.11	0.62
doi	Lev Dist	2.81	2.81	2.04	6.03	4.42	7.45	3.24
pub_date	Lev Dist	1.70	1.68	1.27	6.72	1.31	3.49	1.75
publisher	Lev Dist	14.34	14.34	3.62	8.44	10.88	6.70	7.18
abstract	Cos Sim	0.97	0.96	0.97	0.90	0.87	0.93	0.67
authors	F1 Score	0.95	0.95	0.96	0.91	0.96	0.74	0.92
affiliations	F1 Score	0.82	0.80	0.91	0.87	0.91	0.71	0.89
keywords	F1 Score	0.82	0.78	0.78	0.61	0.63	0.63	0.70
email_ids	F1 Score	0	0	0.98	0.88	0.86	0.69	0.81

0.86. The GROBID variants remain competitive, producing values between 0.86 and 0.87, but without matching the flexibility of the generative models.

Affiliations highlight one of the widest gaps. The GROBID systems score low, both around 0.45, reflecting difficulty in handling heterogeneous affiliation strings. By contrast, the small language models offer clear improvements, with qwen3b leading at 0.73, closely followed by phi4_mini and llama3b at 0.74–0.75. The large model,gpt_oss_20b, once again achieves the best result at 0.84, showing a decisive advantage.

For *email addresses*, GROBID produces no usable outputs, returning zero values. The small models display more variation: phi4_mini and qwen3b each perform strongly with F1 scores between 0.83 and 0.87, llama3b lags slightly at 0.77, while qwen4b underperforms with scores below 0.70. The large model gpt_oss_20b dominates with near-perfect recovery, scoring 0.98.

The extraction of *keywords* yields mixed results across all systems. Here, the layout-aware grobid_dl actually performs best at 0.78, slightly ahead of gpt_oss_20b at 0.76. The small models struggle: llama3b recovers around 0.61, qwen3b about 0.59, and phi4_mini performs lowest at 0.56. This suggests that fixed, layout-based rules still retain an advantage for short, domain-specific lists such as keywords.

For the short string fields evaluated by Levenshtein distance, different trends emerge. On *titles*, most systems perform reliably, with distances often below one. The best results are achieved by llama3b, grobid_dl, and qwen3b, while phi4_mini produces higher error around 4.2 and qwen4b deteriorates dramatically, reaching 24 edits. On *DOIs*, the large model gpt_oss_20b again excels with a distance of 2.3, while the small models show a spread: llama3b and qwen3b remain close to 3, phi4_mini rises to 5.8, and qwen4b performs worst at 7.7. For *publication dates*, most models, including GROBID, remain around 1.3–1.8 edits, but phi4_mini diverges sharply with a distance above 7. In *publishers*, differences are most striking: GROBID models produce very large distances around 14, the small models cluster between 6.7 and 10, and the large model gpt_oss_20b again produces the lowest error at 3.9.

Overall, the layout-aware system, GROBID, demonstrate strength in specific structured fields such as abstracts, titles, and keywords, but consistently struggle with noisy or heterogeneous metadata such as affiliations, publishers, and email addresses. The small language models reveal a ranking where qwen3b performs most consistently, followed by llama3b, then phi4_mini, with qwen4b trailing significantly due to frequent high-error outliers. Finally, the large generative model, gpt_oss_20b, consistently ranks at or near the top across almost all fields, except for keywords where grobid_dl retains a slight edge.

#	Model	Total time taken	Max GPU used	Max CPU used
1	grobid-dl	90s	–	–
2	grobid-crf	15s	–	–
3	qwen4b	3,453s	4673 MiB	227 MiB
4	qwen3b	1,777s	6380 MiB	1848 MiB
5	phi4_mini	3,245s	4263 MiB	225 MiB
6	llama3b	1,713s	3416 MiB	1500 MiB
7	gpt_oss_20b	19,851s	–	–

TABLE 5.2: Runtime and resource usage.

5.2.2 Computational Efficiency

The computational profile of each family of systems underscores stark trade-offs. The layout-aware parsers are extremely lightweight. The CRF-based `grobid_crf` processes the full evaluation set in just fifteen seconds, while the deep learning variant `grobid_dl` requires ninety seconds. Neither variant incurs measurable GPU demand, and their CPU usage is negligible. This efficiency demonstrates why such systems remain attractive in large-scale digital library pipelines.

The small language models, in contrast, are considerably more resource-intensive. `llama3b` and `qwen3b` require approximately half an hour to process the dataset, with runtimes of 1,713 and 1,777 seconds respectively. Their GPU consumption is also notable, with `qwen3b` peaking at 6.3 GB and `llama3b` at 3.4 GB, alongside CPU demands of 1.8 GB and 1.5 GB. The other small models are even slower: `phi4_mini` takes 3,245 seconds with GPU memory near 4.2 GB, while `qwen4b` requires 3,453 seconds and about 4.7 GB. Thus, within this family, `llama3b` and `qwen3b` are the fastest and most memory-hungry, while `phi4_mini` and `qwen4b` are slower despite similar or slightly lower resource consumption.

At the other extreme, the large generative model `gpt_oss_20b` is prohibitively expensive to run in practice. Its runtime exceeds 19,800 seconds, or more than five hours, for the same workload that GROBID completes in under two minutes. While its accuracy scores are among the highest, this comes at a cost of scalability and deployment feasibility. GPU and CPU usage were not captured in this configuration, but given its scale, they are expected to be well above those of the small models.

The efficiency results therefore mirror the accuracy findings: layout-aware systems offer unmatched speed and minimal overhead but are limited in flexibility, small language models strike a middle ground with reasonable accuracy at significant computational cost, and the large model offers the strongest accuracy but at a runtime cost so extreme as to limit its real-world applicability.

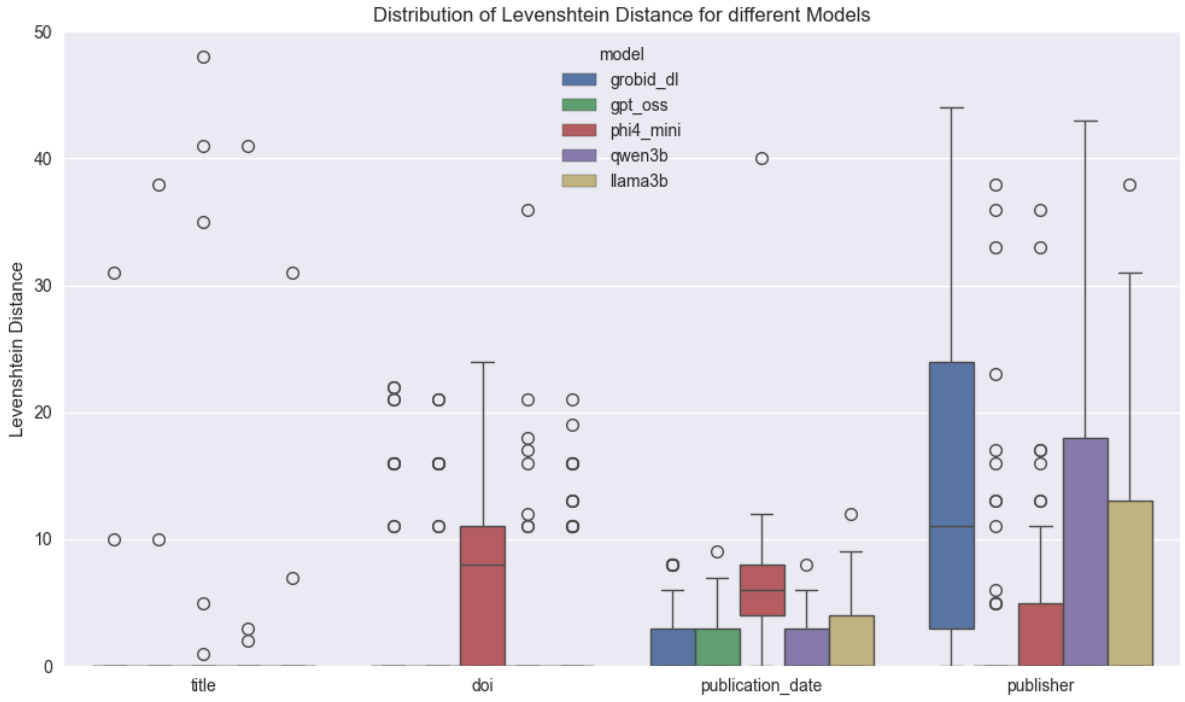

FIGURE 5.3: Evaluation of Title, DOI, Publication Date, Publisher

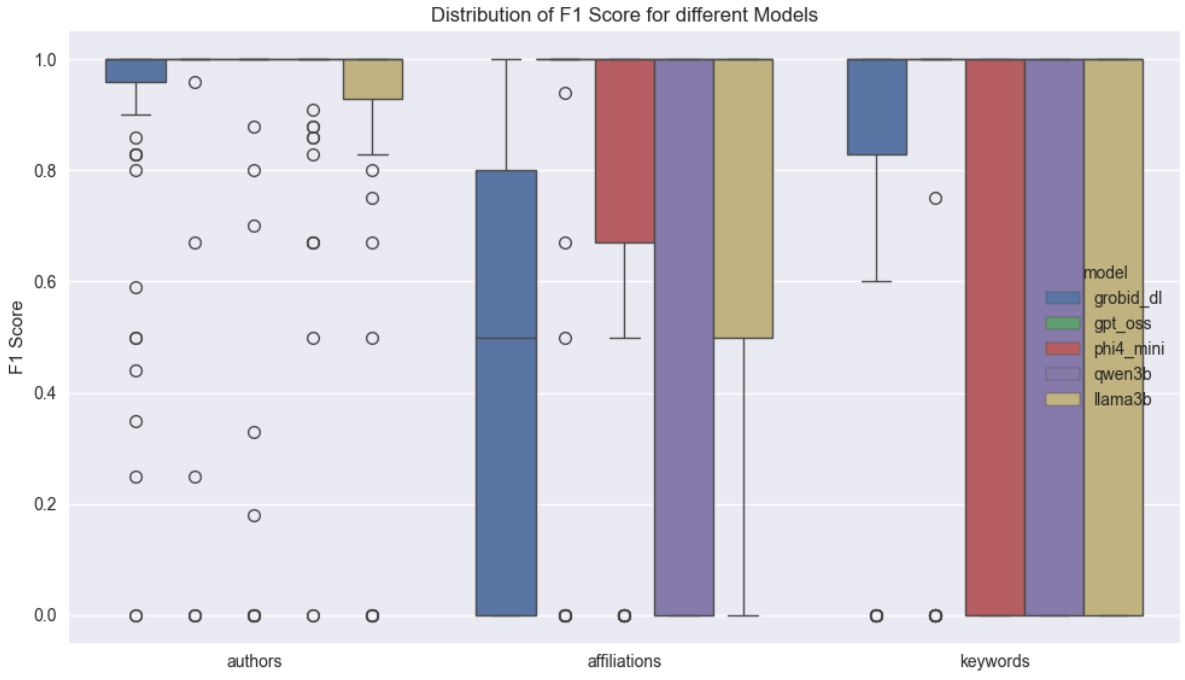

FIGURE 5.4: Evaluation of Authors, Affiliations, Keywords

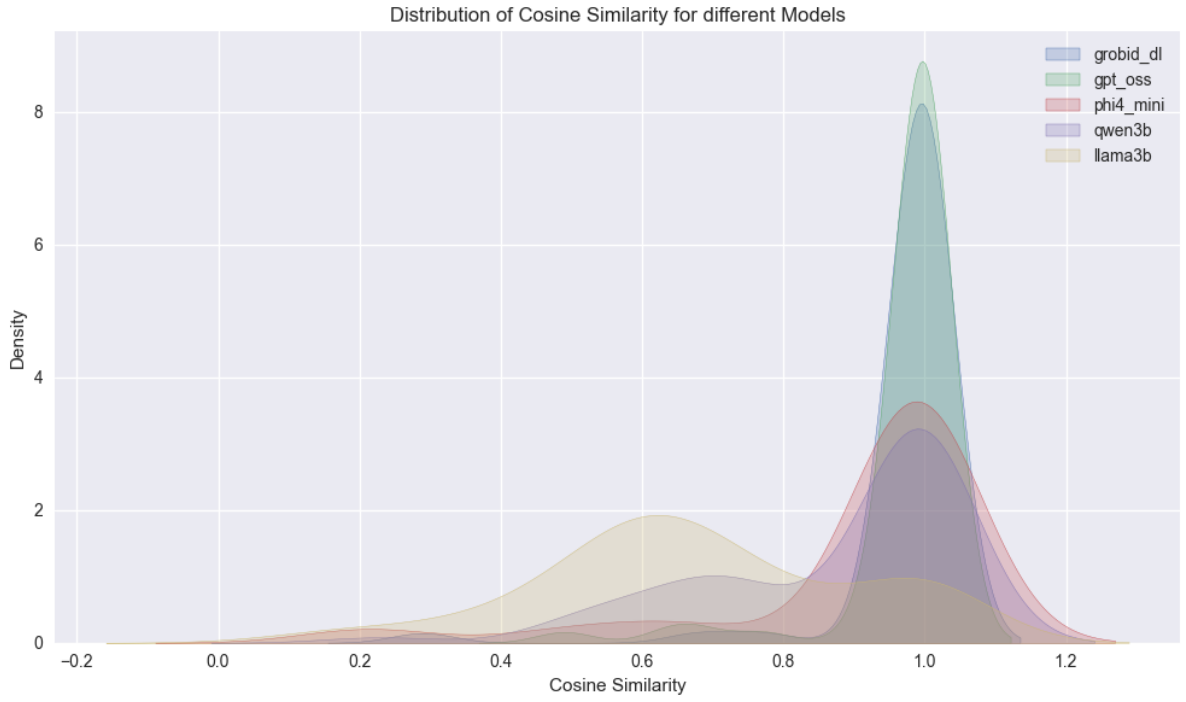

FIGURE 5.5: Evaluation of Abstract

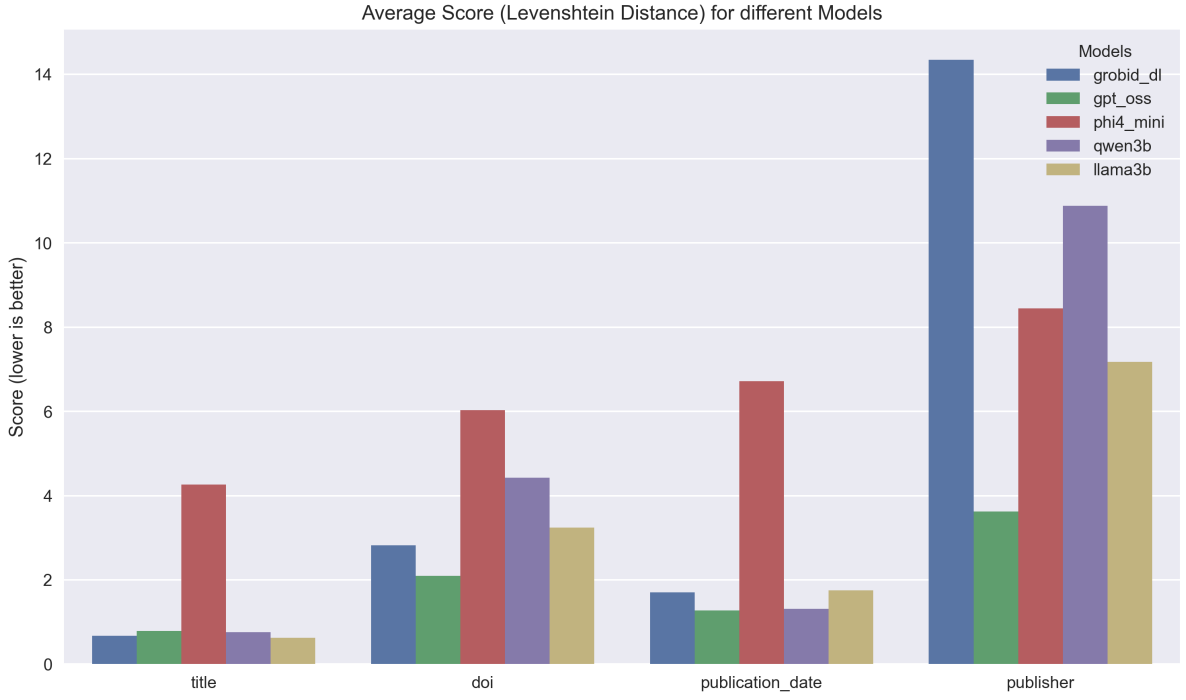

FIGURE 5.6: Overall Performance - Title, DOI, Publication Date, Publisher

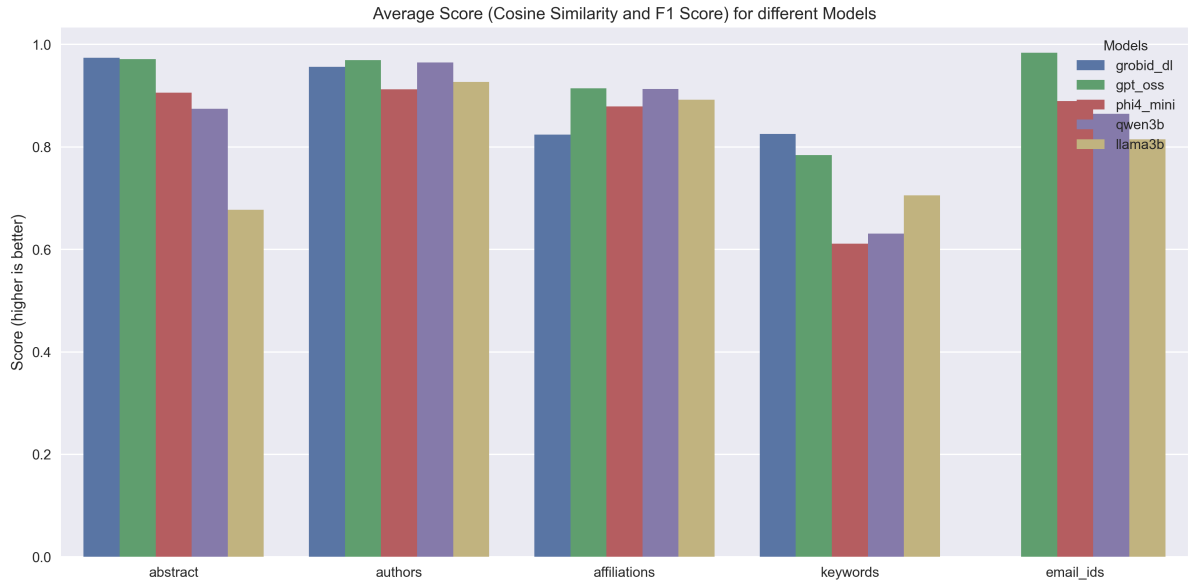

FIGURE 5.7: Overall Performance - Abstract, Authors, Affiliations, Keywords

5.2.3 Error Analysis

During metadata extraction experiments, several recurring error patterns were observed across layout-aware systems and language models. For the small-scale language models—Qwen3B, Phi-4-mini, and Llama3B—only the first page of each article was passed as input. This restriction was necessary to remain within token limits, since providing additional pages frequently led to truncated or malformed outputs that broke the structured JSON schema. In contrast, the larger gpt-oss model was able to process up to two pages without exceeding its capacity, still producing valid and complete structured metadata. The decision to limit smaller models to a single page was further supported by the fact that most publishers present the majority of bibliographic metadata—title, authors, affiliations, DOI, publisher, keywords, and abstract—on the first page, ensuring sufficient coverage while maintaining response reliability.

For consistency in the evaluation graphs, the deep learning variant of GROBID was selected over the CRF-based model, and Qwen3B was preferred over Qwen4B. These choices were based on their marginally superior performance, which made them more representative baselines for layout-aware and small-scale language model categories, respectively.

Publisher extraction exposed a notable weakness in the GROBID systems. Both CRF- and DL-based models produced large Levenshtein distances, often around 14, due to systematic mismatches between the predicted publisher string and the reference. Instead of outputting precise journal titles such as “PLOS ONE” or “PeerJ Computer Science,” GROBID frequently returned expanded forms like “Public Library of Science PLOS” or collapsed variants like “PeerJ.” These discrepancies, though semantically related, were penalized under the exact string match metrics.

Affiliation extraction revealed complementary shortcomings of layout-aware and small language models. GROBID struggled in cases where affiliations were printed in smaller font sizes or were visually embedded without clear typographic distinction, leading to frequent omissions. Similarly, small language models failed whenever affiliations were placed beyond the first page, reflecting their reliance on restricted context. By contrast, the large generative model gpt-oss achieved higher recall by leveraging its ability to process larger context, although at significantly higher computational cost.

Abstract extraction further demonstrated a fundamental limitation of small-scale language models. Because only the first page of each article was passed as context, any abstract content extending beyond that page was systematically truncated or entirely missed. The smaller models strictly confined themselves to the available context, resulting in incomplete or severely shortened abstracts. This behavior highlights their lack of long-context reasoning and reinforces the dependency of small models on input length constraints.

Keyword extraction illustrated the susceptibility of language models to hallucination. Smaller models in particular exhibited unstable behavior: Phi-4-mini and Qwen3B generated keywords in 17 cases where the original publications contained none. These fabricated entries appeared plausible but undermined the integrity of the metadata. In comparison, gpt-oss hallucinated keywords only twice across the entire corpus, indicating better adherence to textual evidence. The GROBID models, while not prone to hallucination, occasionally failed to capture non-standard keyword placements.

Taken together, these error analyses highlight a clear divide between systems. Layout-aware frameworks are sensitive to font size, layout placement, and strict string matching, while small-scale language models are constrained by input length and prone to speculative outputs. Large generative models mitigate some of these limitations but incur substantial runtime and resource costs. These observations reinforce the importance of field-specific evaluation and motivate the hybrid strategies discussed in the conclusion of this thesis.

5.3 Summary

The evaluation compared five open-source PDF parsing tools—`pymupdf`, `pypdfium2`, `pdfminer.six`, `pypdf2`, and `pdfalto`—using the DocBank dataset with a standardized normalization pipeline. Performance was assessed with CER and WER for character- and word-level fidelity, and BLEU and ROUGE-L for order preservation and semantic coherence. Results show that while all tools achieve generally low median CER and reasonably high BLEU/ROUGE scores, `pymupdf` and `pypdfium2` consistently outperform others, delivering tighter distributions and greater robustness. By contrast, `pdfminer` and particularly `pdfalto` show more variability and higher error rates, with frequent outliers indicating sensitivity to complex layouts. `pypdf2`

also underperforms on word order preservation. Overall, no tool is universally robust, but `pymupdf` and `pypdfium2` emerge as the most reliable across metrics.

The study benchmarks layout-aware GROBID (CRF, DL) against small LLMs (Phi-4-mini, Qwen2.5-3B, Qwen3-4B, Llama-3.2-3B) and a large model (gpt_oss_20b) across nine metadata fields using field-appropriate metrics: Levenshtein distance for short strings, F1 for list fields, and TF-IDF cosine similarity for abstracts, with standardized normalization and sklearn/str-simpy tooling. Results show while gpt_oss_20b is consistently top-tier except for keywords where GROBID-DL slightly leads, GROBID excels on structured, layout-anchored content (abstract, titles, keywords) but falters on heterogeneous strings like publishers and affiliations; while small LMs rank Qwen3B $\succ$ Llama3B $\succ$ Phi-4-mini (Qwen4B unstable). Efficiency profiling reveals stark trade-offs: GROBID runs in seconds on CPU with negligible memory, small LLMs require tens of minutes and several GB of GPU RAM, and gpt_oss_20b delivers the best accuracy at prohibitive runtimes ($\approx$5+ hours). Error analysis attributes failures to context limits (small LLMs saw only page 1; gpt_oss_20b handled two pages), GROBID's string mismatches on publisher names (inflated edit distances), missed affiliations beyond page 1 or in low-salience typography, and hallucinated keywords by small LLMs (17 cases vs. 2 for gpt_oss_20b). Overall, layout-aware methods are fast and strong on layout-bound fields, small LLMs offer a middle ground with stability led by Qwen3B, and the large model maximizes accuracy but at impractical cost—motivating hybrid, field-specific strategies.

Chapter 6. Conclusion

This chapter synthesizes the thesis' core outcomes, situating them within the broader literature and drawing implications for scholarly document processing. It first reflects on the construction of reliable ground truth and the design of a schema-constrained, training-free pipeline with small language models, emphasizing accuracy–efficiency trade-offs against layout-aware systems. It then consolidates quantitative findings to answer the stated research questions, clarifies methodological choices (e.g., word-level F1), and acknowledges limitations that bound external validity. Building on these insights, the chapter outlines concrete avenues for future work—ranging from dataset expansion and hybrid, field-specific pipelines to improved metrics and deployment considerations. The goal is to provide a transparent, actionable summary of what was learned, why it matters, and how it can be extended in practice.

6.1 Discussions

The discussions section delves into critical reflections on the methodologies, findings, and broader implications of the research. It begins by examining the construction of ground truth datasets, highlighting the innovative consensus-driven annotation process and the importance of diverse, schema-validated benchmarks for reliable evaluation. The section then explores the development of a schema-constrained, training-free pipeline leveraging small language models, emphasizing its practicality, strengths, and limitations in real-world applications. A comparative analysis of efficiency and accuracy across various open-source tools and models follows, illustrating the trade-offs inherent in different approaches. The discussion continues by situating the work within the evolving landscape of scholarly document processing, referencing recent advances and related systems, and emphasizing the contributions of this research to the field. It then critically addresses the study's limitations—such as dataset scope, language focus, and processing constraints—acknowledging areas for future improvement. Lastly, the section clarifies the evaluation methodology, particularly the nuanced approach to calculating F1 scores, ensuring transparency and interpretability of the reported results.

6.1.1 Ground-truth construction

The first major contribution pertains to the construction of ground truth datasets. Reliable metadata extraction fundamentally depends on accurate text extraction; however, even advanced parsers often struggle with complex multi-column layouts, ligatures, or inconsistent encodings. To address these challenges, two complementary benchmarks were assembled. The first is a page-level, free-text ground truth derived from the extensive DocBank corpus, which provides precise token annotations, and bounding boxes for evaluating PDF parsers. The second

comprises a heterogeneous collection of 61 born-digital PDFs selected from multiple publishers, including PLOS, Elsevier, arXiv, Springer, PMLR, MDPI, and Frontiers, chosen to encompass a wide array of column layouts, fonts, and graphical artifacts. These two datasets facilitate a comprehensive evaluation: the DocBank-based benchmark assesses raw text extraction fidelity at the page level, while the collection of published PDFs enables a holistic assessment of metadata extraction at the document level.

Importantly, the metadata annotation process employed a novel multi-model consensus and adjudication workflow. Candidate metadata for each document were generated by three diverse language models—DeepAI's GPT-3.5-turbo, Google's Gemini-2.5-flash-lite, and X-AI's Grok-3-mini. When all three models concurred on a given field value, that value was automatically accepted. In cases of disagreement, a more capable GPT-4.1 evaluator adjudicated by selecting the most plausible candidate based on contextual understanding. Following this, the entire metadata record was validated against a Pydantic schema, which enforced type correctness, mandatory field coverage, and structural uniformity. This validation step ensured the generation of well-formed, consistent ground truth data, significantly reducing manual annotation effort while maintaining high accuracy.

The significance of these datasets lies in their ability to support reproducible evaluation. By integrating diverse model perspectives, an adjudication process, and schema-driven validation, the resulting benchmarks provide a robust, scalable, and reproducible foundation for assessing extraction systems. Furthermore, because the documents originate from multiple publishers and exhibit marked layout variations, the datasets serve as a comprehensive and high-fidelity standard for evaluating the performance of future extraction methodologies. Overall, this approach enhances the reliability and comparability of research in document parsing and metadata extraction.

6.1.2 Schema-constrained small-language-model pipeline

The second major contribution centers on the development of a schema-constrained, training-free pipeline for metadata extraction utilizing small, general-purpose language models. This pipeline is designed to operate without the need for model retraining or fine-tuning, relying instead on prompt engineering and schema-driven responses to produce structured JSON outputs. The process begins with the extraction of text from the first page of each PDF using PyMuPDF, which supplies high-fidelity textual content. This content is then fed into an instruction-tuned language model, guided by a carefully constructed prompt that specifies the desired metadata fields.

The prompt adopts a chat format, instructing the model to extract specific fields such as the title, authors, affiliations, email addresses, DOI, publisher, publication date, keywords, and abstract. To enhance focus and boundary recognition, the prompt incorporates special tokens

and an attention mask, directing the model to concentrate on relevant input segments. The model's raw JSON response is subsequently parsed and validated against a predefined Pydantic schema, which enforces type correctness, checks for the presence of all required fields, and ensures structural consistency. This automatic validation mechanism not only improves the reliability of the outputs but also facilitates the detection and correction of malformed or incomplete responses. As demonstrated in Table 4.1, this schema-guided approach surpasses naive raw-text parsing in terms of output reliability, error detection, and robustness to changes in document formatting.

This pipeline exemplifies that small language models can effectively extract structured metadata solely through prompt design, without any additional training data. Its adaptability to different model sizes, limited to processing only the first page to conform with token constraints, and its ability to operate on modest hardware, make it particularly suitable for cost-constrained or resource-limited settings. However, certain limitations are acknowledged. Restricting input to the first page may result in missing metadata located on subsequent pages, such as affiliations or abstracts that span multiple pages. Additionally, some smaller models, including Qwen-4B, may produce truncated or malformed outputs when confronted with lengthy or complex inputs, and variability among models can affect stability and accuracy. The absence of training further constrains the pipeline's ability to adapt to domain-specific nuances or idiosyncratic formatting.

In essence, this schema-constrained, instruction-tuned approach reframes metadata extraction as a prompted generation task, leveraging high-fidelity text extraction and structured prompts to produce machine-readable bibliographic data. The automatic parsing and schema validation steps serve as safeguards, ensuring output quality and facilitating error handling. Empirical evidence, particularly in Section 6.2.3, demonstrates that small instruction-tuned models can achieve acceptable accuracy in producing structured metadata. While the approach is inherently limited by processing only the first page and the variability among models, it offers a pragmatic, accessible, and deployable solution that balances complexity, reliability, and resource requirements.

6.1.3 Comparative evaluation and efficiency

Benchmarking the performance of open-source PDF text extraction tools revealed that PyMuPDF and pypdfium2 consistently outperformed other parsers, such as pdfminer.six, PyPDF2, and pdfalto, in terms of median Word Error Rate (WER) and interquartile range, indicating higher accuracy and more consistent preservation of reading order. Both tools demonstrated higher BLEU and ROUGE scores, further confirming their superior ability to maintain the textual sequence. Conversely, pdfminer and pdfalto exhibited wider error distributions and more outliers, suggesting less reliable performance, particularly on complex or challenging layouts. Notably, none of the parsers proved universally robust, as all occasionally produced severe errors when faced with difficult formatting.

Regarding metadata extraction accuracy, performance across nine fields was summarized in Table 5.1. The large generative model gpt-oss 20B achieved the highest accuracy across most fields, with cosine similarity approximately 0.97 for abstracts and F1 scores around 0.96 for author identification. Layout-aware GROBID variants performed exceptionally well on structurally constrained fields such as abstracts, titles, and keywords—achieving cosine similarity of about 0.97 on abstracts and Levenshtein distances below 1 on titles—though they struggled with more heterogeneous fields like affiliations, publishers, and email addresses, where error distances reached up to 14 edits for publisher names. Among smaller models, Qwen-3B consistently ranked highest, matching gpt-oss 20B in author F1 scores (0.96) and achieving approximately 0.91 in affiliations, while Llama-3B and Phi-4-mini followed behind. Qwen-4B, however, lagged significantly, exhibiting frequent outliers and instability.

Efficiency profiling highlighted notable trade-offs. The CRF-based GROBID processing pipeline completed in approximately 15 seconds, with the deep-learning variant taking around 90 seconds, both utilizing minimal GPU and CPU resources. Small language models, such as Llama-3B and Qwen-3B, required over 1,700 seconds, with GPU memory between 3.4 and 6.3 GB and CPU usage of roughly 1.5 to 1.8 GB. Phi-4-mini and Qwen-4B demanded more than 3,200 seconds each. In stark contrast, the gpt-oss 20B model was considerably slower, exceeding five hours for processing the same dataset.

Error analysis identified recurring issues, including the limitation of small models to process only the first page, which resulted in truncated abstracts and missed affiliations that appeared on subsequent pages. GROBID occasionally misidentified publishers, returning expanded or abbreviated variants—such as “Public Library of Science PLOS” instead of “PLOS ONE”—which led to penalties in string-level metrics. Small models sometimes hallucinated keywords, and Qwen-4B produced unstable, inconsistent outputs.

An important practical advantage of transformer-based LLMs is their ease of adaptation: new pieces of information can be extracted simply by modifying the prompt, without retraining or requiring annotated data. This significantly reduces the time and resources typically needed for model retraining, as exemplified by the successful extraction of email addresses—a class of metadata that layout-aware systems like GROBID cannot handle effectively.

Overall, the evaluation underscores that pypdfium2 and PyMuPDF are the most reliable open-source tools for raw text extraction, producing lower error rates and better reading order preservation than pdfminer.six, PyPDF2, or pdfalto. Nonetheless, the presence of outliers highlights that none of these tools is infallible, reinforcing the importance of a page-level ground truth for identifying and addressing difficult cases. In the realm of metadata extraction, the results are more nuanced. The large generative model gpt-oss 20B delivers the highest accuracy but at a computational cost that limits practical deployment at scale. Layout-aware GROBID excels on

structured, layout-dependent fields like titles and keywords but underperforms on more variable fields like affiliations and publishers. Small language models, particularly Qwen-3B, offer a promising middle ground, matching or exceeding GROBID on certain fields and enabling a training-free approach. However, their longer runtimes—tens of minutes—and reliance on GPU resources may pose challenges in resource-constrained environments.

6.1.4 Comparison with existing literature

The results of this thesis can be situated within a rapidly growing body of research on scholarly document processing. A recent multi-domain benchmark by Meuschke et al. evaluates ten open-source PDF extraction tools across 500 k pages and 1.5 M annotated elements. They report that GROBID achieves the best metadata and reference extraction, with CERMINE and Science Parse following, while Adobe Extract delivers the best table extraction but all tools perform poorly on lists, footers and equations (66). Our dual benchmark replicates this hierarchy: GROBID dominates on highly structured fields such as titles and author names, whereas our schema-constrained language models excel on variable fields like keywords and abstracts. Unlike their parser-centric evaluation, we add a page-level ground truth to quantify text-extraction fidelity and report efficiency metrics. These results echo calls from the LAME framework, which builds a large, layout-annotated training set and introduces Layout-MetaBERT to achieve a macro-F1 of 93.27% on unseen journals (100). Our experiments confirm that layout information improves performance on structured fields, yet they extend prior work by coupling layout-aware parsing with schema-validated language models and by constructing a multi-model consensus dataset.

Recent advances in large language models further contextualize our contributions. LayoutLLM, presented at LREC-COLING 2024, argues that existing visually rich document (VrDU) models require expensive fine-tuning for each task and dataset; to overcome this, it couples a layout-aware encoder with an LLM decoder and fine-tunes on multimodal instruction datasets (112). MOLE similarly advocates for a schema-driven pipeline and shows that modern LLMs can process entire documents across multiple formats, but the authors caution that results are merely promising and that further improvements are required for consistent performance (98). Our training-free pipeline aligns with these frameworks by using instruction-tuned models without fine-tuning, but differs in two ways: it uses prompt engineering and Pydantic validation to guarantee type correctness, and it evaluates modestly sized models that can run on commodity hardware.

SpeciMate, a human-AI system for digitised biological specimens, shows that combining OCR, translation and LLMs with expert intervention significantly improves efficiency while maintaining high data quality (15). Our consensus-driven annotation procedure takes a similar approach: we accept field values only when multiple models agree, resolve disagreements using a stronger model, and validate outputs against a schema. This strategy produces reliable ground truth and

extraction results without expensive fine-tuning. Overall, our findings confirm that LLMs can rival or surpass human annotators under certain conditions, extend the literature by orchestrating multiple lightweight models to achieve robust performance, and underscore the continued importance of hybrid systems and human oversight for trustworthy metadata extraction.

6.1.5 Limitations

The study's findings are based on datasets that, while thoughtfully curated, have inherent limitations impacting their broader applicability. The primary metadata dataset includes only 61 PDFs, and the parser evaluation relies on text from 101 first pages from DocBank. Although selected to encompass some heterogeneity, these samples cannot fully reflect the vast diversity found in scholarly PDFs—such as multiple languages, non-Latin scripts, or scanned documents with varying qualities and layouts. Consequently, the results may not generalize to all real-world scenarios.

Both benchmarks focus exclusively on born-digital PDFs, leaving out scanned images, low-quality scans, and historical documents that often pose different and more complex challenges for extraction tools. These unaddressed formats may exhibit issues like degraded text, skewed layouts, or OCR errors, which were not captured in the current evaluation.

The multi-model consensus approach to establishing ground truth relies heavily on outputs from large language models (LLMs). While the use of GPT-4.1 for adjudication improves accuracy, it also introduces systematic biases inherent to the underlying models. Human oversight was limited, meaning potential biases or errors propagated from LLMs could influence the benchmark data.

In the Small Language Model (SLM) pipeline, processing is restricted to the first page due to token length constraints, which leads to missing information such as affiliations or extended abstracts that may appear on subsequent pages. The larger models can process two pages but still cannot handle full-length, multi-page documents, resulting in incomplete metadata capture.

Resource profiling was incomplete; notably, the memory consumption for the gpt-oss 20B model could not be fully captured due to tool limitations, making direct comparisons across models less comprehensive.

The study's scope was limited to English-language scientific papers, and it did not evaluate cross-lingual generalization or performance across specialized domains such as biomedical or legal texts, where vocabulary and document layouts can differ significantly.

Overall, although the study aimed to control variables and ensure fairness, these limitations—small datasets, focus on digital-born PDFs, reliance on LLMs with inherent biases, partial processing of multi-page documents, and the metrics used should temper the generalization of

its conclusions. Real-world scenarios, especially involving scanned or non-English documents, may present additional challenges not fully captured here.

6.1.6 Clarification on Evaluation Methodology

When calculating the F1 score based on exact string matches, certain variations—such as "department of mechanical engineering university of maryland college park md 20742 usa" versus "department of mechanical engineering university of maryland 20742 college park md usa"—would be heavily penalized, potentially resulting in a score of zero despite the entities being semantically equivalent. To mitigate this issue and better reflect the practical accuracy of extraction, the F1 score was not computed on entire item strings. Instead, for author names, affiliations, and keywords, these items were split into individual words, and the F1 score was calculated based on these word-level matches. This approach reduces penalization for minor reordering or formatting differences, offering a more nuanced assessment of extraction quality.

6.2 Answers to the research questions

R1 **How accurately and robustly can existing open-source, layout-aware extraction systems identify bibliographic metadata from PDFs across diverse formatting styles, while remaining computationally efficient for large-scale deployment?**

The evaluation of layout-aware systems, particularly GROBID variants, highlights their strong performance on structured metadata fields that align closely with the page layout. Specifically, the deep-learning GROBID model attains a cosine similarity of approximately 0.97 on abstracts and achieves low Levenshtein distances (less than 1) on titles, indicating near-perfect accuracy in extracting these elements. Additionally, the F1 scores for author lists reach around 0.95, demonstrating reliable recognition and extraction of author information when it is presented in predictable formats.

However, these systems exhibit notable limitations when handling more heterogeneous or variable fields. For example, extracting affiliations, publisher names, and email addresses remains challenging due to the variability in formatting and placement. Levenshtein distances for publisher names tend to hover around 14 edits, reflecting significant discrepancies, while F1 scores for affiliations are approximately 0.82, indicating more frequent inaccuracies. These difficulties stem from the systems' reliance on structural cues, which can be inconsistent or absent in documents with complex layouts or non-standard formatting.

In terms of efficiency, GROBID variants demonstrate exceptional speed and resource economy. The CRF-based GROBID processes the entire dataset in roughly 15 seconds, while the deep-learning GROBID completes in about 90 seconds, both utilizing negligible memory. Such rapid processing makes layout-aware systems highly suitable for large-scale library or repository applications where throughput is critical.

Despite their speed, the robustness of these systems is relative. Error patterns reveal that GROBID can misidentify publisher names—sometimes returning expanded or abbreviated variants—and frequently misses affiliations that are printed in small fonts or embedded within paragraphs. Their performance also declines when documents deviate from expected layouts or contain non-standard formatting, underscoring their dependence on structural cues.

In summary, layout-aware systems like GROBID excel at extracting metadata fields that follow predictable layout patterns, delivering high accuracy and scalability. Nonetheless, their limitations in handling less structured or more variable information mean they cannot fully replace language models, particularly for fields that are embedded deep within the text or formatted irregularly. An optimal approach combines the speed and precision of layout-aware tools for structured fields with the flexibility of language models to capture more heterogeneous or complex metadata elements.

R2 **Which PDF text extraction strategies provide the most complete, accurate, and logically ordered raw text as input for downstream metadata recognition?**
Among the five evaluated parsers, pypdfium2 and PyMuPDF stand out by delivering the lowest median Word Error Rates and the highest BLEU/ROUGE scores. Their interquartile ranges are narrow, indicating consistent and reliable performance across a diverse set of documents.

pdfminer.six and pdfalto exhibit wider error distributions and numerous outliers, reflecting their susceptibility to handling complex layouts and formatting variations. PyPDF2 particularly underperforms in preserving reading order, leading to higher error rates.

All tools demonstrate outlier cases with very high error rates, confirming that even the best open-source parsers can occasionally fail under challenging conditions.

When accurate raw text extraction is crucial, pypdfium2 and PyMuPDF should be preferred. They not only minimize character- and word-level errors but also better maintain the logical reading order—an essential factor for downstream language models. However, the presence of outliers across all parsers highlights the importance of employing multiple fallback strategies and underscores the ongoing need for research to develop more robust text extraction methods. While pdfminer.six, PyPDF2, and pdfalto may still be useful in specific contexts, they cannot be relied upon universally.

R3 **To what extent can small-scale, general-purpose language models extract structured bibliographic metadata from unstructured text, and how do they compare with layout-aware systems in terms of accuracy and efficiency?**
With effective prompt engineering and schema validation, small models can match or even outperform GROBID on several key fields. For example, Qwen-3B achieves F1 scores around 0.96 for author extraction and 0.91 for affiliations. Llama-3B and Phi-4-mini follow closely, while Qwen-4B exhibits more inconsistent performance.

Despite these strengths, small models tend to struggle with keyword extraction, achieving F1 scores around 0.61–0.63. They also often truncate abstracts or affiliations due to limited context windows. Additionally, they are more prone to hallucinating keywords or generating malformed outputs, especially when faced with complex or ambiguous data.

Processing times for these models range from approximately 1,700 seconds to 3,400 seconds, requiring 3–6 GB of GPU memory. While slower than GROBID, they are still faster and more cost-effective than large models like the 20B-parameter variants.

This thesis demonstrates that small, instruction-tuned models can serve as effective metadata extractors when combined with prompt engineering and schema enforcement. Notably, Qwen-3B narrows the accuracy gap with larger models and surpasses GROBID in extracting affiliations and email fields. However, their performance is limited by context window size and computational resources. They produce fewer hallucinations than GPT-3.5-like models but still more than GROBID. Inference times on GPU are measured in minutes rather than seconds. Consequently, small models are a viable option when annotated data are limited and moderate compute resources are available, but they do not yet offer a universal replacement for layout-aware systems.

R4 **What are the relative advantages and limitations of layout-aware approaches versus language model–based methods for metadata extraction from scholarly publications?**

Layout aware systems offer extremely fast runtimes with near-zero resource requirements. They excel in extracting structured fields such as abstracts, titles, and keywords with high accuracy and operate independently of annotated training data.

They struggle with heterogeneous strings—such as affiliations, publisher names, and email IDs—and are sensitive to variations in document layouts, which can hinder consistent extraction.

Language models models are flexible and capable of inferring metadata from unstructured text, often outperforming layout-based tools on variable fields like affiliations and email addresses. They require no extensive training data beyond prompting, making them adaptable to diverse document types.

They are computationally intensive, with runtimes of tens of minutes per document and demands of several gigabytes of GPU memory. Their limited context windows can lead to truncated information, and they may occasionally hallucinate data or violate schema constraints.

Large generative models typically provide state-of-the-art accuracy across nearly all fields but come with significant drawbacks—prohibitively slow processing times (around 5 hours per batch) and high resource consumption, making them impractical for routine use.

Achieving an optimal balance among these paradigms involves recognizing their respective strengths and limitations. Layout-aware systems excel in speed and extracting layout-dependent metadata but falter on unstructured or irregular data. Small instruction-tuned

models offer greater flexibility and can surpass GROBID on variable fields, albeit at higher computational costs and longer runtimes. Large models deliver near-perfect accuracy but are too slow and resource-demanding for everyday scenarios.

6.3 Future Scope

A significant future direction involves scaling and diversifying the underlying datasets to enhance model robustness and generalizability. Expanding the ground-truth corpora to include a wider array of publishers, languages, and document types—such as scanned or OCR-derived texts—will enable models to handle real-world variability more effectively. Incorporating diverse document formats, including conference proceedings, book chapters, and multi-language articles, will broaden the evaluation scope and facilitate the development of multilingual and cross-disciplinary extraction systems. Such efforts will contribute to creating more comprehensive and representative datasets, ultimately improving the accuracy and adaptability of extraction models across diverse scholarly domains.

Enhancing consensus mechanisms can further improve annotation accuracy by integrating a broader spectrum of language models, including multilingual and domain-specific variants, which can better capture the linguistic and terminological nuances present in scholarly documents. Incorporating human adjudication into the annotation pipeline remains essential, particularly for disputed or ambiguous fields, as expert validation can correct systematic errors and guide model refinement. Combining automated consensus with human oversight will strengthen the reliability of labeled data, leading to more robust model training and evaluation.

Addressing the limitations imposed by small model context windows is critical for processing longer, multi-page documents with high fidelity. Techniques such as sliding-window inference, where documents are segmented into overlapping chunks processed sequentially, can mitigate truncation issues while enabling the aggregation of information across segments for holistic metadata extraction. Retrieval-augmented generation (RAG) approaches, which incorporate external knowledge retrieval, can further extend the effective context by dynamically fetching relevant sections during inference, thus bypassing token limits. These strategies collectively aim to preserve contextual integrity and improve extraction accuracy in complex, multi-page scholarly texts.

Fine-tuning and model adaptation techniques, such as Low-Rank Adaptation (LoRA) and prompt tuning, present promising avenues for improving small models' performance without incurring prohibitive computational costs. These methods enable efficient domain adaptation by adjusting only a small subset of parameters or leveraging carefully crafted prompts, which reduces training time and resource requirements. Domain-adaptive pretraining, where models are further trained on specialized corpora—such as discipline-specific literature or publisher-styled documents—can imbue models with deeper contextual understanding of particular terminologies

and formatting conventions. Combining these approaches with high-quality curated datasets will allow small models to achieve higher accuracy on challenging fields like affiliations or funding statements while maintaining the efficiency necessary for large-scale deployment.

Implementing a hybrid, field-specific extraction pipeline offers a practical means of balancing speed and accuracy by leveraging the complementary strengths of layout-aware tools and language models. For layout-dependent fields like titles, keywords, and abstracts, traditional layout-based systems such as GROBID can deliver rapid and precise extractions due to their reliance on spatial and visual cues. Conversely, for unstructured or variable fields like affiliations or email addresses, instruction-tuned large language models can infer metadata from unstructured text with greater flexibility. Such a system would adapt to diverse document formats and complexities, ensuring scalable and reliable metadata extraction in real-world scenarios.

Current evaluation metrics often fail to fully capture the semantic equivalence of extracted metadata, especially when dealing with variations in naming conventions, abbreviations, or typographical errors. Developing more nuanced metrics—such as fuzzy string matching, or embedding-based semantic similarity scores—can provide a more accurate assessment of extraction quality. For example, comparing publisher names via embedding similarity rather than exact string matching might allow for recognition of equivalent or highly similar entries despite minor differences. Additionally, evaluating the downstream impact of extraction errors—such as their effect on citation linking, recommendation accuracy, or bibliometric analyses—can contextualize the practical significance of improvements in metadata quality. These advanced evaluation strategies will enable more meaningful comparisons and facilitate targeted model enhancements.

Finally, transitioning from research prototypes to operational deployment involves rigorous testing and integration into existing digital library workflows. Embedding the hybrid extraction pipeline into production environments will allow continuous monitoring of performance metrics such as throughput, latency, and error rates under real-world loads, revealing operational bottlenecks and robustness issues. Considerations around incremental updates, real-time processing, and privacy compliance must be addressed to ensure scalable and compliant deployment. User feedback mechanisms and explainability features can foster trust and facilitate manual validation, while ongoing performance monitoring can inform iterative improvements. Such deployments will not only validate the practical utility of these systems but also surface new requirements—such as handling dynamic document collections or ensuring data privacy—that will guide future research and development efforts.

6.4 Final Reflection

The unified contributions of this thesis encompass the development of a dual benchmark framework—comprising page-level and document-level evaluations—alongside a schema-validated scholarly language model (SLM) pipeline, and a comprehensive assessment of both accuracy and computational efficiency. These advancements collectively push forward the understanding of metadata extraction, offering both theoretical insights and practical tools for the community. Building upon prior work, the research constructs a consensus-based gold standard that enhances ground truth reliability and demonstrates that prompt-engineered small models can rival dedicated extraction systems without necessitating extensive retraining. This approach extends evaluations beyond accuracy metrics alone, highlighting efficiency and adaptability, and underscores the potential of training-free solutions within scholarly document processing. The broader implications emphasize that improved metadata extraction significantly enhances scholarly discoverability, attribution, and analytical capabilities, illustrating that effective, resource-efficient systems are within reach through strategic integration of layout-awareness and language understanding. The work concludes with a reflection on the centrality of consensus—both in establishing reliable benchmarks and in synthesizing complementary extraction paradigms—urging the community to build systems that are not only accurate but also transparent, scalable, and inclusive. By developing a robust ground truth, a schema-constrained pipeline, and a balanced evaluation framework, this research demonstrates that high-quality metadata extraction is achievable with small, engineered models, challenging the notion that only large, proprietary systems can deliver excellence. The lessons on hybrid strategies, consensus mechanisms, and meticulous evaluation serve as guiding principles for future innovations in document intelligence, ultimately enriching the scholarly record and enabling more robust discovery, analysis, and knowledge synthesis across disciplines.

Appendix A. Data

All the data including PDFs, ground truth, metadata, and output files, can be accessed at Google Drive link

https://tinyurl.com/bddsuhwn

Project Folder Structure

```
Layout Aware Metadata Extraction Data
+-- metadata_extraction_data     # datasets and outputs used for metadata extraction
    +-- gpt_oss_metadata
    +-- grobid_crf_metadata
    +-- grobid_crf_tei           # TEI outputs from GROBID CRF model
    +-- grobid_dl_metadata
    +-- grobid_dl_tei            # TEI outputs from GROBID DL model
    +-- llama3b_metadata
    +-- metadata                 # ground-truth metadata annotations
    +-- pdf                      # selected PDF files used for metadata extraction
    +-- phi4mini_metadata
    +-- qwen3b_metadata
    \-- qwen4b_metadata
\-- pdf_extraction_data          # datasets and outputs used for PDF text extraction
    +-- annotation               # annotations derived from DocBank dataset
    +-- img                      # layout segmentation results from YOLOv12-DocLayNet
    +-- pdf                      # 101 selected PDF files for parser evaluation
    +-- pdfalto_xml              # XML outputs generated by pdfalto
    +-- txt                      # ground-truth plain text for comparison
    +-- arxiv_api_response.json  # API response from arXiv containing metadata
    \-- metadata.csv             # mapping file linking input PDFs to output paths
```

Appendix B. Code

The complete code and full implementation is available at GitHub repository.

https://github.com/sanath95/Layout-Aware-Metadata-Extraction-Framework

Bibliography

[arx] arxiv: About arxiv. https://arxiv.org/about. Accessed 9 Sep 2025.

[pmc] Pubmed central: Tools and resources. https://www.ncbi.nlm.nih.gov/pmc/tools/. Accessed 9 Sep 2025.

[spr] Springerlink: Platforms for librarians. https://www.springernature.com/gp/librarians/products/platforms. Accessed 9 Sep 2025.

[4] AI, M. (2024). Llama 3.2: Advancing open foundation models. Accessed 2025-08-26.

[5] AI, M. (2025). Llama 3.2 models. Accessed: 2025-08-26.

[6] Appalaraju, S., Jasani, B., Kota, B. U., Xie, Y., and Manmatha, R. (2021). Docformer: End-to-end transformer for document understanding. In *Proceedings of ICCV*.

[7] Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *ICLR*.

[8] Bast, H. and Korzen, C. (2017). A benchmark and evaluation for text extraction from pdf. In *Proceedings of the 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*.

[9] Beltagy, I., Peters, M. E., and Cohan, A. (2020). Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

[10] Berger, A. L., Pietra, S. A. D., and Pietra, V. J. D. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

[11] Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.

[12] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

[13] Bolukbasi, T., Chang, K.-W., Zou, J., Saligrama, V., and Kalai, A. (2016). Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *NeurIPS*, pages 4349–4357.

[14] Boukhers, Z., Schindler, D., and Meinel, C. (2019). Mexpub: Metadata extraction from academic publications using mask r-cnn. In *Proceedings of the 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*.

[15] Boukhers, Z., Strotgen, J., and Gipp, B. (2023). Challenges in metadata extraction with large language models: A critical perspective. In *Proceedings of the 2023 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*.

[16] Brown, P. F., deSouza, P. V., Mercer, R. L., Pietra, V. J., and Lai, J. C. (1992). Class-based n-gram models of natural language. In *Computational Linguistics*, volume 18, pages 467–479.

[17] Callin, J. (2017). *Word Representations and Machine Learning Models for Implicit Sense Classification in Shallow Discourse Parsing*. Master's thesis / dissertation.

[18] Cao, Y. and Fang, H. (2020). Pdf information extraction beyond text: a survey. In *Proceedings of the International Conference on Document Analysis and Recognition*.

[19] Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. In *ACL*, pages 310–318.

[20] Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al. (2021). Rethinking attention with performers. In *ICLR*.

[21] Clark, A. S. (2018). Mupdf and pymupdf: lightweight tools for pdf rendering and text extraction. *Artifex Software Documentation*.

[22] Cloud, G. (2025). Gemini 2.5 flash — vertex ai model limits. Accessed 2025-08-26.

[23] Contributors, C. (2021). pdfminer.six documentation. https://pdfminersix.readthedocs.io/.

[24] Contributors, P. (2022). Pypdf2 documentation. https://pypdf2.readthedocs.io/.

[25] CORE project (2023). Core + grobid: Structured text from 34 million scientific documents (and counting). CORE Blog. Accessed 2025-08-26.

[26] Councill, I. G., Giles, C. L., and Kan, M.-Y. (2008). Parscit: An open-source crf reference string parsing package. In *Proceedings of LREC*.

[27] Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., and Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*.

[28] DeepMind, G. (2025). Google gemini 2.5 flash-lite documentation. Accessed: 2025-08-26.

[29] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

[30] Denk, T., Reisswig, C., et al. (2023). Docformer: End-to-end transformer for document understanding. *Pattern Recognition*.

[31] Denton, E. et al. (2015). Deeppdf: A deep learning approach for pdf document segmentation. *arXiv preprint arXiv:1506.01497*.

[32] Developers, P. (2022). pypdfium2: Python bindings for pdfium. https://pypdfium2.readthedocs.io/.

[33] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

[34] Ding, M., Zheng, W., et al. (2022). Layoutreader: Pre-training of text and layout for reading order detection. *arXiv preprint arXiv:2203.16528*.

[35] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*.

[36] Duan, Q. (2024). Renewable energy integration and carbon neutrality green technology innovation based on intelligent microgrid control.

[37] Giles, C. L., Bollacker, K. D., and Lawrence, S. (2004). Citeseer: An automatic citation indexing system. *ACM DL*.

[38] Goldberg, Y. and Levy, O. (2014). Word representations: A simple and general method for semi-supervised learning. *arXiv preprint arXiv:1402.3722*.

[39] Google (2025). Ai studio pricing — gemini 2.5. Accessed 2025-08-26.

[40] Google Developers (2025). Gemini 2.5 flash lite: High throughput, low latency. Accessed 2025-08-26.

[41] GROBID documentation (2011). Introduction to grobid. GROBID ReadTheDocs. Accessed 2025-08-26.

[42] GROBID documentation (2023a). Benchmarking on biorxiv – grobid documentation. Read the Docs. Accessed 2025-08-26.

[43] GROBID documentation (2023b). Benchmarking on pubmed central – grobid documentation. Read the Docs. Accessed 2025-08-26.

[44] Group, A. (2025). Qwen2.5 model announcement. Accessed: 2025-08-26.

[45] Han, H., Giles, C. L., and Zha, H. (2003). Automatic metadata extraction using support vector machines. In *JCDL*, pages 37–48.

[46] He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969.

[47] Hendricks, G., Tkaczyk, D., Lin, J., and Feeney, P. (2020). Crossref: The sustainable source of community-owned scholarly metadata. *Quantitative Science Studies*, 1(1):414–427.

[48] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

[49] Huang, Z., Xu, W., and Yu, K. (2015). Bidirectional lstm-crf models for sequence tagging. In *Proceedings of the 29th Conference on Neural Information Processing Systems (NIPS)*.

[50] International Organization for Standardization (2008). Iso 32000-1:2008 document management—portable document format—part 1: Pdf 1.7.

[51] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y., Madotto, A., and Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*.

[52] Jurafsky, D. and Martin, J. H. (2009). *Speech and Language Processing*. Prentice Hall.

[53] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*.

[54] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

[55] Li, Y., Xu, Y., Cui, L., Huang, S., Wei, F., and Zhou, M. (2020). Docbank: A benchmark dataset for document layout analysis. *arXiv preprint arXiv:2006.01038*.

[56] Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81. ACL.

[57] Liu, X., Gao, F., et al. (2019). A neural network approach for joint document layout analysis and metadata extraction. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*.

[58] Lopez, P. (2009). Grobid: Combining automatic bibliographic data recognition and term extraction for scholarship publications. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries (JCDL)*.

[59] Lopez, P. (2010). Grobid: Generation of bibliographic data. https://github.com/kermitt2/grobid.

[60] Lopez, P. and contributors (2025). Grobid: A machine learning software for extracting information from scholarly documents (readme). GitHub repository. Accessed 2025-08-26.

[61] Lu, Y. and Tan, C. L. (1996). Document layout analysis: a review. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*.

[62] Luo, X., Peng, N., and Gildea, D. (2017). Segment attention and hierarchical networks for citation field extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[63] Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

[64] Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

[65] Meta (2024). meta-llama/llama-3.2-3b-instruct — model card. Accessed 2025-08-26.

[66] Meuschke, N., Jagdale, A., Spinde, T., Mitrović, J., and Gipp, B. (2023). A benchmark of pdf information extraction tools using a multi-task and multi-domain evaluation framework for academic documents. *arXiv preprint arXiv:2303.09957*.

[67] Microsoft (2025a). Empowering innovation: The next generation of the phi family. Accessed 2025-08-26.

[68] Microsoft (2025b). microsoft/phi-4-mini-instruct — model card. Accessed 2025-08-26.

[69] Microsoft (2025c). Phi-4-mini-instruct — azure ai foundry model catalog. Accessed 2025-08-26.

[70] Mikolov, T., Chen, K., Corrado, G. S., and Dean, J. (2013). Efficient estimation of word representations in vector space. In *International Conference on Learning Representations*.

[71] Morris, A., Maier, V., and Green, P. (2004). From wer and ril to mer and wil: improved evaluation measures for connected speech recognition. *Interspeech*.

[72] Okamoto, K., Seki, K., and Uehara, K. (2001). Robust rule-based information extraction methodology and its application. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*.

[73] OpenAI (2023). Gpt-4 technical report.

[74] OpenAI (2025a). Gpt-4.1. Accessed 2025-08-26.

[75] OpenAI (2025b). gpt-oss-120b & gpt-oss-20b model card. Accessed 2025-08-26.

[76] OpenAI (2025c). gpt-oss-20b — openai platform model docs. Accessed 2025-08-26.

[77] OpenAI (2025d). Introducing gpt-oss. Accessed 2025-08-26.

[78] OpenAI (2025e). Introducing openai o3 and o4-mini. Accessed 2025-08-26.

[79] OpenAI (2025f). Openai gpt-4.1. Accessed: 2025-08-26.

[80] OpenAI (2025g). Openai gpt_oss_20b. Accessed: 2025-08-26.

[81] OpenAI (2025h). Openai o3-mini. Accessed 2025-08-26.

[82] OpenAI (2025i). Openai o3-mini system card. Accessed 2025-08-26.

[83] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318. ACL.

[84] Peng, C. et al. (2023). Knowledge graphs: Opportunities and challenges. *Artificial Intelligence Review*, 56(7):6271–6315.

[85] Peng, F. and McCallum, A. (2006). Accurate information extraction from research papers using conditional random fields. In *HLT-NAACL*.

[86] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

[87] Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *NAACL*.

[88] Pfitzmann, B. et al. (2022a). Doclaynet: A large human-annotated dataset for document-layout analysis. *arXiv preprint arXiv:2206.01062*.

[89] Pfitzmann, B. et al. (2022b). Doclaynet: A large human-annotated dataset for document-layout analysis. In *NeurIPS Datasets and Benchmarks Track*.

[90] Qwen (2024). Qwen/qwen2.5-3b-instruct — model card. Accessed 2025-08-26.

[91] Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

[92] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. https://openai.com/research/language-unsupervised.

[93] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. In *JMLR*.

[94] Research, M. (2025). Phi-4-mini-instruct model card. Accessed: 2025-08-26.

[95] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241.

[96] Shilman, M., Liang, J., and Wu, Y. (2005). Document image text extraction: heuristic and learning approaches. In *ICDAR*.

[97] Shreeves, S. L., Knutson, E. M., Stvilia, B., Palmer, C. L., Twidale, M. B., and Cole, T. W. (2005). Is 'quality' metadata 'shareable' metadata? the implications of local metadata practices for federated collections. In *Proceedings of the 12th National Conference of the Association of College and Research Libraries*, pages 223–237.

[98] Singh, A., Zhang, W., and Goyal, N. (2023). Mole: Metadata extraction from scientific literature using large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[99] SplxAI (2025). The missing gpt-4.1 safety report. Accessed 2025-08-26.

[100] Strotgen, J., Boukhers, Z., Habibi, M., and Gipp, B. (2020). Lame: Layout-aware metadata extraction for scientific publications. In *Proceedings of the 12th International Conference on Knowledge Capture (K-CAP)*.

[101] Team, Q. (2024). Qwen2.5: A party of foundation models! Accessed 2025-08-26.

[102] Tkaczyk, D., Szostek, P., Fedoryszak, M., Dendek, P. J., and Bolikowski, (2015). Cermine: Automatic extraction of structured metadata from scientific literature. *International Journal on Document Analysis and Recognition*, 18(4):317–335.

[103] Ultralytics (2023). Ultralytics yolov8. https://github.com/ultralytics/ultralytics.

[104] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *NeurIPS*.

[105] Wang, S., Li, B., Khabsa, M., Fang, H., and Ma, H. (2020). Linformer: Self-attention with linear complexity. In *ACL*.

[106] Wang, Z., Zhang, Y., et al. (2023). A survey on large language models for information extraction. *arXiv preprint arXiv:2309.01469*.

[107] Wiggers, K. (2025). Openai's gpt-4.1 may be less aligned than the company's previous ai models. Accessed 2025-08-26.

[108] xAI (2025a). Chat with reasoning (grok 3/4). Accessed 2025-08-26.

[109] xAI (2025b). Models and pricing. Accessed 2025-08-26.

[110] xAI (2025c). xai grok-3-mini model card. Accessed: 2025-08-26.

[111] 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. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

[112] Zhang, L., Liu, Z., et al. (2024). Instruction-tuned language models for structured data extraction. *Transactions on Machine Learning Research*.

[113] Zhong, X., Tang, J., and Yepes, A. J. (2019). Publaynet: largest dataset ever for document layout analysis. In *ICDAR*.

[114] Zhou, W., Xu, Q., and Pratama, M. (2019). Evaluating text generation with bleu and rouge. *Journal of Artificial Intelligence Research*.