Metadata Extraction From Publications

Problem Statement

Develop an automated method that reliably extracts core metadata—**title, authors, affiliations, email IDs, DOI, publication date, publisher, keywords, and abstract**—from born-digital academic PDFs whose layouts vary widely across publisher styles (e.g., IEEE, Springer) and column structures. The solution must overcome layout heterogeneity to deliver metadata that is both complete and consistently structured for downstream use.

Data

Step	Procedure
Corpus selection	50 born-digital PDF articles drawn from seven publishers—PLOS, Elsevier, arXiv, Springer, PMLR, MDPI, and Frontiers Media—to ensure layout diversity.
Initial text extraction	Parsed each PDF with **PyMuPDF** to obtain the full plain-text content.
Metadata triangulation	Supplied the extracted text to three lightweight LLMs—`openai/o3-mini`, `google/gemini-2.5-flash-lite`, and `x-ai/grok-3-mini`—each prompted to return the nine target metadata fields.
Agreement filtering	Retained any field whose value **exactly matched across all three models** as ground truth.
Discrepancy resolution	For fields with conflicting outputs, forwarded the three candidate values to `openai/gpt-4.1`, requesting a single corrected value.
Structured storage	Consolidated the final, reconciled metadata into a **JSON** record per document for downstream evaluation.

GROBID Pipeline

- Deployed GROBID (DL+CRF models) in Docker and called the `processHeader` service to convert each of the 50 PDFs into TEI-XML header sections.
- Parsed this TEI header to extract all target metadata **except e-mail addresses** (no support).

Small Language Model Pipeline

- Extracted **only the first page** of each document using *PyMuPDF* to remain within the input token limit.
- Queried multiple small language models to retrieve the required metadata fields from the extracted text.
- Enforced strict formatting rules for a **structured output** and validated the results using *Pydantic* models.

Results

- Inference on 61 PDF files

fields	metric	grobid_dl	grobid_crf	gpt_oss	phi4_mini	qwen3b	qwen4b	llama3b
title	Levenshtein Dist	0.672131	3.04918	0.786885	4.2623	0.754098	24.1148	0.622951
doi	Levenshtein Dist	2.81967	2.81967	2.09836	6.03279	4.42623	7.45902	3.2459

fields	metric	grobid_dl	grobid_crf	gpt_oss	phi4_mini	qwen3b	qwen4b	llama3b
publication_date	Levenshtein Dist	1.70492	1.68852	1.27869	6.72131	1.31148	3.4918	1.7541
publisher	Levenshtein Dist	14.3443	14.3443	3.62295	8.44262	10.8852	6.70492	7.18033
abstract	Cosine Sim	0.973934	0.960328	0.971311	0.905738	0.874426	0.932295	0.677213
authors	F1 Score	0.956066	0.958033	0.96918	0.912131	0.965082	0.74541	0.926557
affiliations	F1 Score	0.82377	0.808033	0.91459	0.878852	0.913279	0.713443	0.892295
keywords	F1 Score	0.82541	0.788033	0.783934	0.611148	0.630656	0.636885	0.705738
email_ids	F1 Score	0	0	0.983607	0.889836	0.864918	0.699836	0.814754

	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	19,851s	-	-

- Computation metrics for **Qwen**, **Phi**, and **LLaMA** were measured using *PyTorch*.
- **GROBID** (Docker) and **gpt_oss** (Ollama) did not provide straightforward resource usage measurements, making them difficult to track.
- When the first **two pages** were used as input, the small language models exhibited unpredictable behavior and failed to produce consistent structured outputs.

Observations

- Fastest: grobid-crf (15 s) → lowest resource cost, but mediocre accuracy outside abstracts/keywords.
- Best accuracy: gpt_oss_20b → highest accuracy across most fields, but extremely slow (nearly 5.5 h) and resource-intensive.
- Midpoint: grobid-dl offers solid accuracy in specific fields at <2 min runtime.

Conclusions

- **GROBID** excels in semi-structured fields such as *abstract*, *authors*, and *title*, but its performance drops on highly variable, multi-format entity fields like *affiliations* and *publisher*.
- **Language models** outperform GROBID on these complex fields, though with lower computational efficiency.
- Expanding the scope of extraction with language models is straightforward via prompt adjustments, whereas GROBID would likely require full model retraining.

Future Scope

- Design an **ideal hybrid pipeline** that uses GROBID for semi-structured fields and language models for highly variable fields.
- Implement a **fallback mechanism** to process the second page of PDFs with small language models when needed.