AdaParse: An Adaptive Parallel PDF Parsing and Resource Scaling Engine

Carlo Siebenschuh[1,2], Kyle Hippe[1,2], Ozan Gokdemir[1,2], Alexander Brace[1,2], Arham Khan[1], Khalid Hossain[2], Yadu Babuji[2], Nicholas Chia[2], Venkatram Vishwanath[2], Rick Stevens[1,2], Arvind Ramanathan[1,2], Ian Foster[1,2], Robert Underwood[2]

[1]University of Chicago [2]Argonne National Laboratory

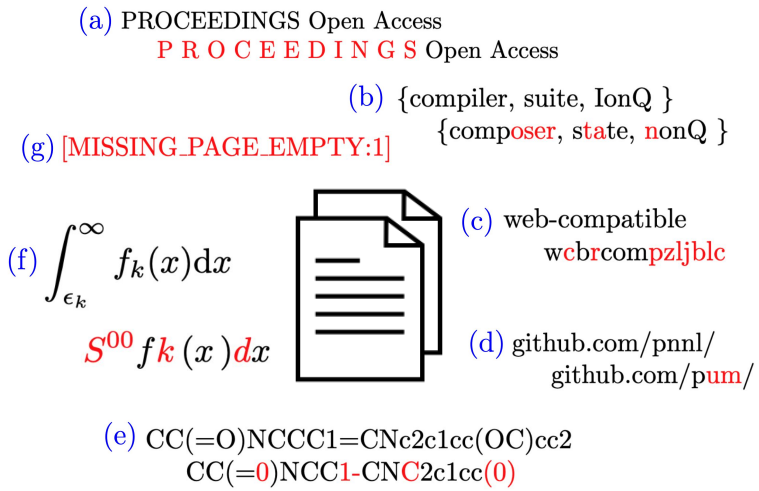
siebenschuh@uchicago.edu

MLSys 2025

May 13th, Santa Clara (CA)

Motivation

- scientific content for LLMs
- PDFs are ubiquitous → efficiency
  - S2ORC: 81 M documents
  - Google Scholar: ∼390 M documents
- parsing quality is critical → accuracy
- many parsers available
  - extraction: PyMuPDF, pypdf
  - OCR: Tesseract, GROBID
  - Transformer-based: Nougat, Marker
- all parsers produce (partially) erroneous text

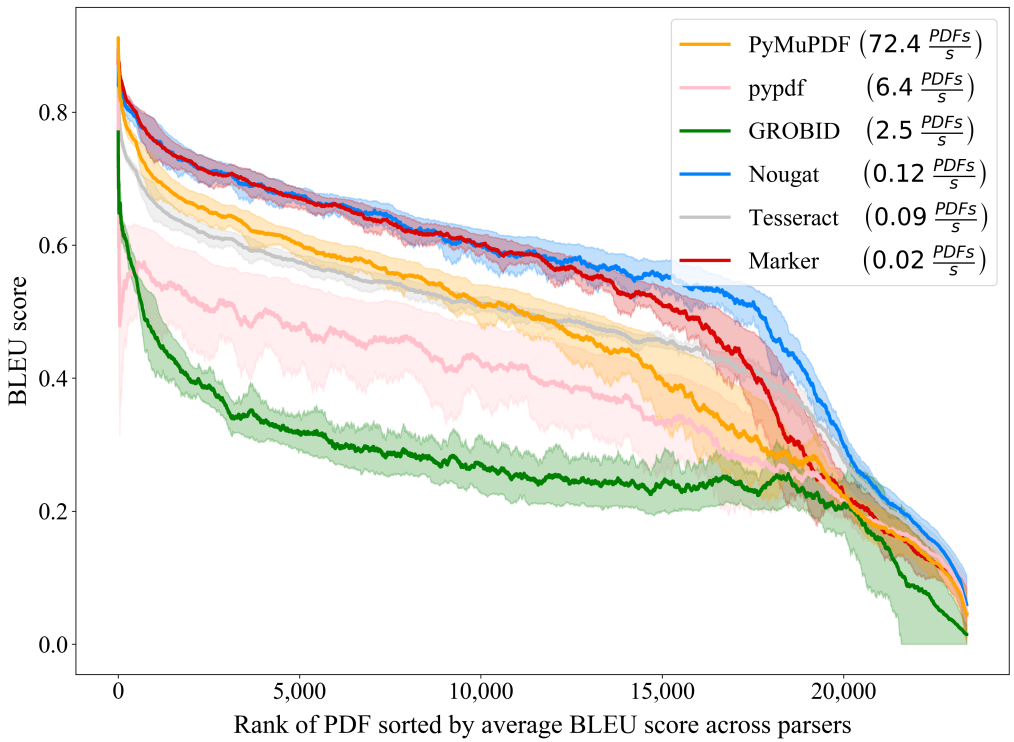


Common failure modes

(a) whitespace injection
(b) word substitution
(c) character scrambling
(d) character substitution
(e) corrupted SMILES
(f) LaTeX to plaintext
(g) page dropped

Stylized Facts

- 23 K documents, 8 domains, 6 publishers
- quality of parsed text output (BLEU
- PDFs sorted by parsing "difficulty" (avg. BLEU)
- quality varies significantly (but no predictive heuristic)
- throughput varies dramatically acr parsers
- accuracy leverages this diversity b choosing a suitable parser for each PDF



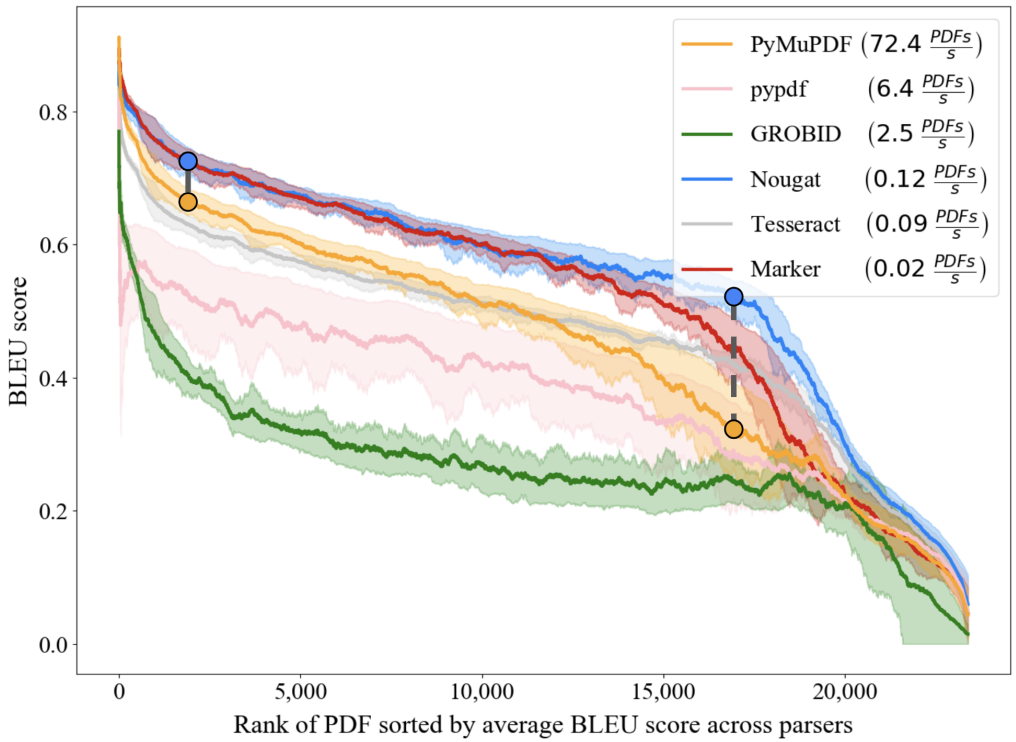
Parser performance across documents

Optimization Problem

$$\max_{\mathbf{j}} \left\{ \sum_{i=1}^{n} \mathbb{E}\left[\mathcal{A}(\phi_{j_i}, \psi_i) \mid \phi_1^1(d_i) \right] \right\}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \mathcal{T}(\phi_{j_i}, d_i) \leq \overline{\mathcal{T}}$$

- $\mathbf{max}$! predicted accuracy $\mathbb{E}[\mathcal{A}]$ over documents d_i, $i \in [n]$, under budget $\bar{\mathcal{T}}$
- PDFs sorted by parsing "difficulty" (avg. BLEU)
- human-aligned accuracy $\mathcal{A}$: BLEU & preferences (DPO)
- predictive signal $\mathcal{F}_i^0$: page 1 text
- tractable model $\mathbb{E}[\cdot \mid \mathcal{F}_i^0]$



Parser performance across documents

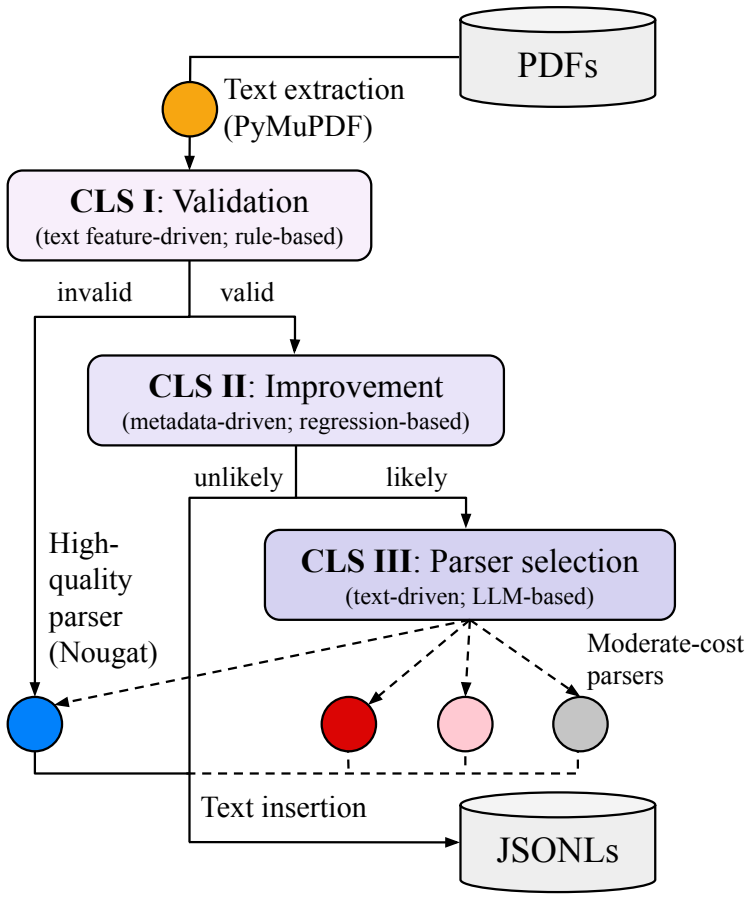
Algorithm

Approach

- input: PyMuPDF-extracted text of page 1 → page 1 text
- model: SciBERT (multiv. regression) to predict parsers' BLEU scores for any given PDF → BLEU prediction
- fine-tuning: BLEU score prediction (page/doc) & DPO → preference-aligned

Advantages

- high-quality parsers (e.g. Nougat) require PDF content loaded → document text in memory
- different hardware utilization → GPU vs. CPU
- user preferences highly consistent → 82% agreement
- 1st page's tokens predictive → $R^2 \approx 46\%$



Algorithmic template

Direct Preference Optimization (DPO)

Fine-tune $\pi_\theta : x^1 \mapsto y$ via

ℓ_2 loss

$$\mathcal{L}_{\text{REG}}(\theta) = \mathbb{E}_{\mathcal{D}} \left[\|\pi_\theta(x^1) - y\|_2^2 \right],$$

and subsequently with **DPO loss**

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{\mathcal{D}_{\text{pref}}} \left[\log \sigma \left(\beta \log \frac{g_\varphi(x^+)}{g_\varphi^{\text{ref}}(x^+)} - \beta \log \frac{g_\varphi(x^-)}{g_\varphi^{\text{ref}}(x^-)} \right) \right]$$

to

- re-weight importance of text modalities (e.g., equations)
- increase sensitivity of subtle differences (*pH*/*Ph*, *hyper*/*hypo*, $X_\theta / \mathcal{X}_G$)

Accuracy

Parser	Coverage	BLEU	ROUGE	CAR	WR	AT
Marker	**96.7**	47.5	64.2	59.6	26.6	73.3
Nougat	93.0	48.1	66.5	65.8	**27.9**	69.8
PyMuPDF	91.3	51.9	67.3	67.0	24.4	76.7
pypdf	92.0	43.6	58.7	32.3	2.4	72.4
GROBID	81.0	26.5	52.4	54.8	–	20.6
Tesseract	91.3	48.8	64.2	**67.8**	18.7	72.5
AdaParse	91.5	**52.1**	**67.6**	67.1	25.5	**76.9**

Born-digital PDFs (N=1,000): Document (coverage), word (BLEU, ROUGE), and character (CAR) accuracies; Win rate (WR) and accepted tokens (AT). In %.

- AdaParse's high-quality parser frequency limited to $\alpha = 5\%$
- AdaParse superior on BLEU and AT, competitive on CAR
- Marker/AdaParse trade off BLEU vs. CAR
- Tesseract excels in character accuracy (CAR)
- Coverage varies by parser—GROBID lowest at 81%

Generalization to corrupted PDFs

Parser	Coverage	BLEU	ROUGE	CAR	WR	AT
Marker	**96.5**	46.6	62.9	60.5	**28.0**	70.1
Nougat	91.9	45.1	63.1	63.4	27.2	63.5
Tesseract	90.0	44.0	58.2	65.2	12.8	59.0
AdaParse	92.8	**52.0**	**67.5**	**67.0**	18.4	**77.0**

Simulated scanned PDFs. In %.

Parser	Coverage	BLEU	ROUGE	CAR	WR	AT
PyMuPDF	90.8	42.0	55.6	56.5	**13.1**	58.8
pypdf	**91.2**	35.6	48.9	29.8	1.2	56.9
AdaParse	**91.2**	**42.4**	**55.9**	**56.7**	12.0	**59.5**

Text-layer corrupted PDFs. In %.

- AdaParse hybdrid strategy of adaptive OCR and extraction improves its robustness

The Impact of DPO

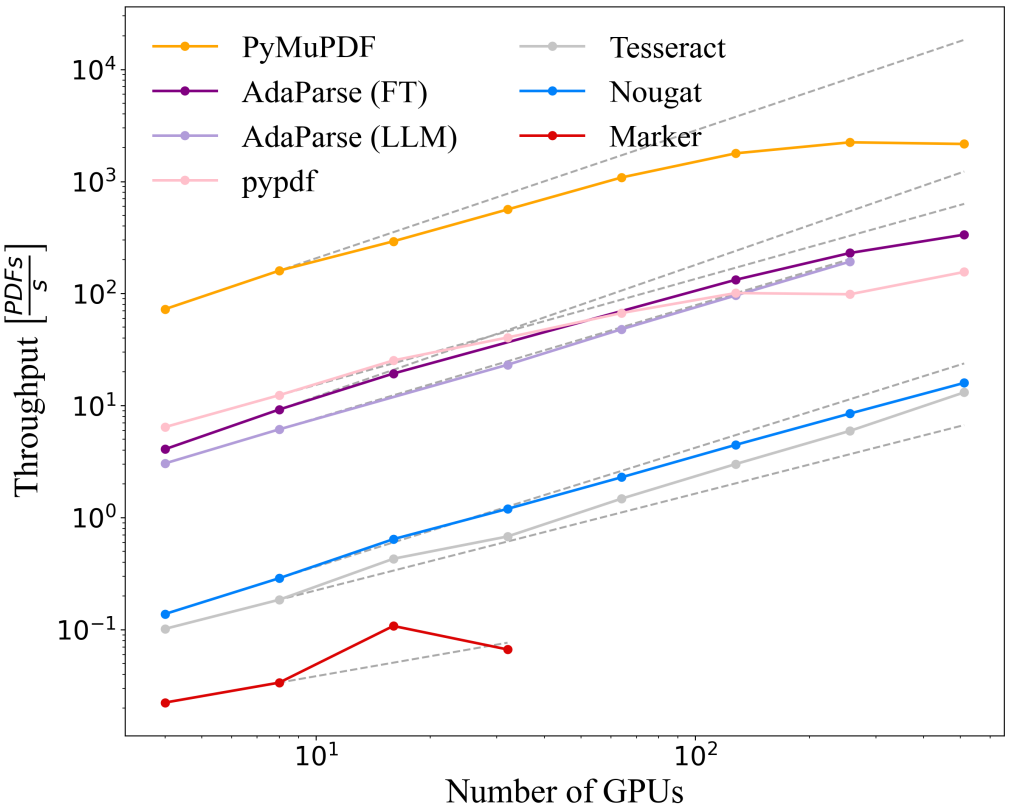
Features (Model)	BLEU	ROUGE	CAR	WR	ACC
CLS III: Document Text					
Text (SciBERT+DPO) [AdaParse]	**52.7**	69.4	**68.0**	**31.4**	36.7
Text (SciBERT)	51.6	**69.5**	66.9	25.0	**48.3**
Text (BERT)	49.7	66.0	63.4	24.8	40.0
CLS II: Metadata and Title Text					
Title + Metadata (SPECTER)	47.9	64.5	62.9	25.2	18.1
Title (SPECTER)	46.4	63.3	61.8	26.2	15.2
CLS I: Metadata					
Year + Producer (SVC)	47.3	63.7	60.1	28.8	14.8
(Sub-)category (SVC)	43.6	63.5	62.5	24.9	12.9
Key Reference					
BLEU-maximal selection	56.8	72.3	70.4	26.5	100.0
Random selection	44.0	61.7	57.4	20.5	16.7

Word (BLEU, ROUGE), character (CAR), and classification accuracy (ACC). Win rate (WR) as a preference-based metric.

Throughput

- tools vary in throughput: 0.1 to 100 PDFs per node second
- AdaParse (LLM) vs. AdaParse (FT) with **`fasttext`**-based CLS
- extraction the fastest paradigm (e.g., PyMuPDF)
- AdaParse's throughput $17\times$ that of Nougat's
- comprehensive pipelines (Marker) and RNN-based tools (Tesseract) have lower throughput

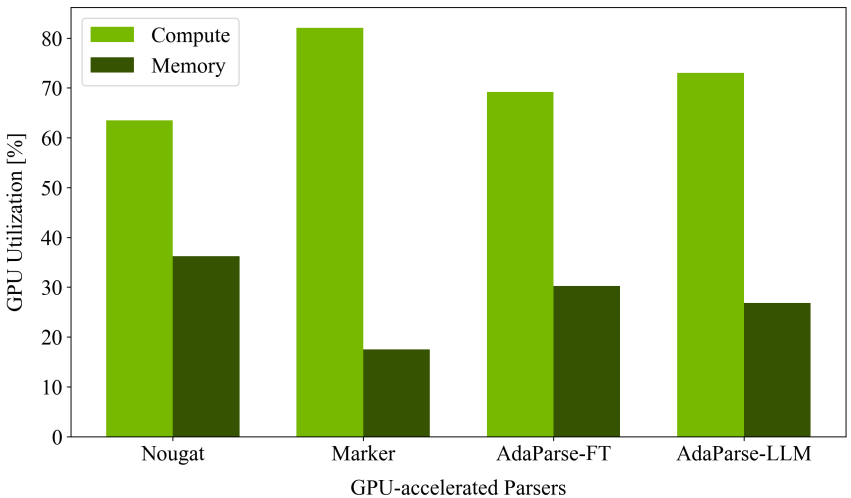
Polaris @ ALCF



Throughput across parsers

GPU Utilization

- resource utilization weakly associated with throughput
- Marker dominates GPU FLOP utilization
- Text decoders (not visual encoders) dominate GPU utilization for Transformer-based parsers
- pre-/post-processing negligible for throughput



Per-node, measured with the NVIDIA Nsight Systems profiler (Nsys)

Summary

- A comprehensive **Benchmark** on parsing quality and domain expert preferences
  - diverse set of scientific documents
  - domain expert's feedback
- **AdaParse**, a parsing strategy that assigns the predictably best parser to each document
  - prediction model (DPO)
  - per-node optimization
  - embarrassingly parallel
- Empirical investigation
  - steer-ability of HPC workloads through few preferences
  - predictability of long-form text quality
  - regularization of DPO

The Team













THE UNIVERSITY OF CHICAGO

Argonne NATIONAL LABORATORY

AI FOR SCIENCE

Thank you!

Questions?

`github.com/7shoe/AdaParse/`

Carlo Siebenschuh

siebenschuh@uchicago.edu

References

- **MuPDF** (2024): Artifex Software
- **Marker – Data Lab** (2024): Data Lab
- **The pypdf library** (2024): Fenniak et al.
- **Nougat: Neural optical understanding for academic documents** (2023): Blecher et al.
- **Specter: Document-level representation learning using citation-informed transformers** (2020): Cohan et al.
- **SciBERT: A pretrained LM for scientific text** (2019): Beltagy et al.
- **S2ORC: The semantic scholar open research corpus** (2019): Lo et al.
- **GROBID: Combining automatic bibliographic data recognition and term extraction for scholarship publications** (2009): Lopez et al.