

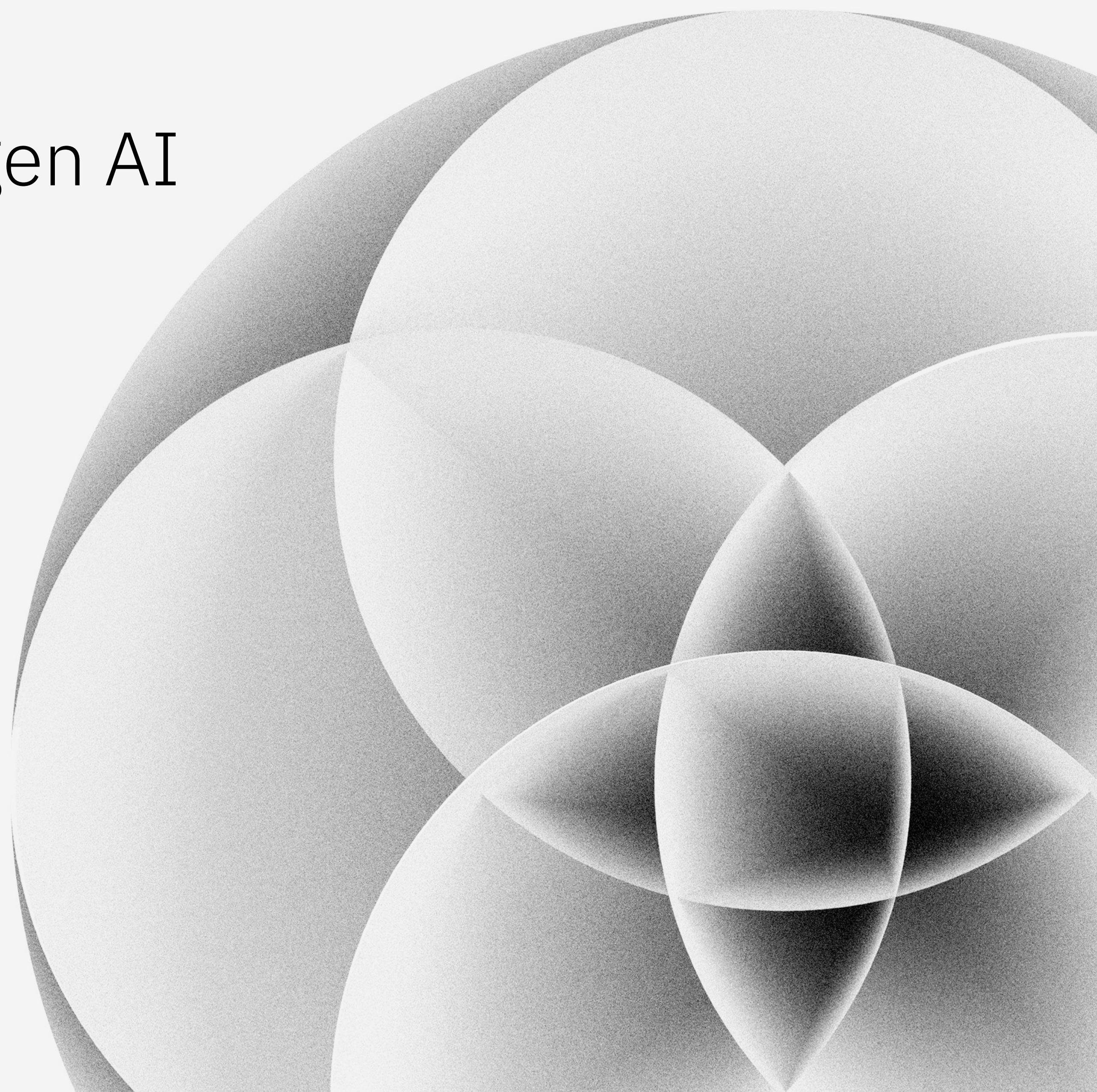
Docling

Get your documents ready for gen AI

—

Panos Vagenas
Software Engineer | AI for Knowledge
IBM Research – Europe

Dr. Michele Dolfi
Senior Technical Staff Member | AI for Knowledge
IBM Research – Europe



IBM Research

3000

Researchers

79

Years



6

Nobel Laureates



10

Medals of Technology



5

National Medals of Science



6

Turing Awards





Docling Workshop

- Learn about Docling
- Install and use Docling
- Understand the configuration options and customizations
- Run Docling with Llama Stack
- Run Docling and Llama Stack on OpenShift



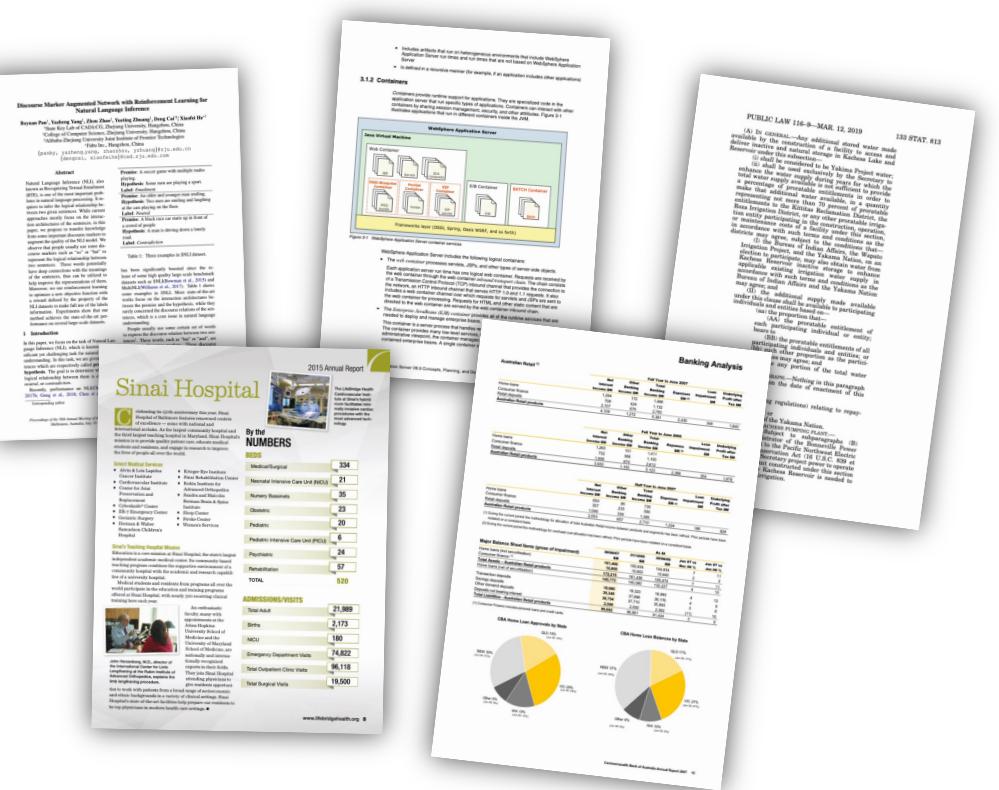
9:00-12:00	Intro to Docling <i>Presentation:</i> Introduction to Docling <i>Hands On:</i> Converting documents with Docling including advanced features
12:00-13:00	Lunch and discussions
13:00-16:00	Docling + Llama Stack + Agents <i>Hands On:</i> Build your DocumentAI use-case with Llama Stack Use Docling within your Llama Stack agents



IBM

Introducing Docling

-  Parsing of multiple document formats incl. PDF, DOCX, XLSX, HTML, images, and more
-  Advanced PDF understanding incl. page layout, reading order, table structure, code, formulas, image classification, ...
-  Unified, expressive DoclingDocument representation format
-  Various export formats (Markdown, HTML, JSON)
-  Local execution for sensitive data and air-gapped environments
-  Many plug-and-play ecosystem integrations
-  Extensive OCR support for scanned PDFs and images
-  Support of Visual Language Models
-  Simple and convenient CLI



```
pip install docling

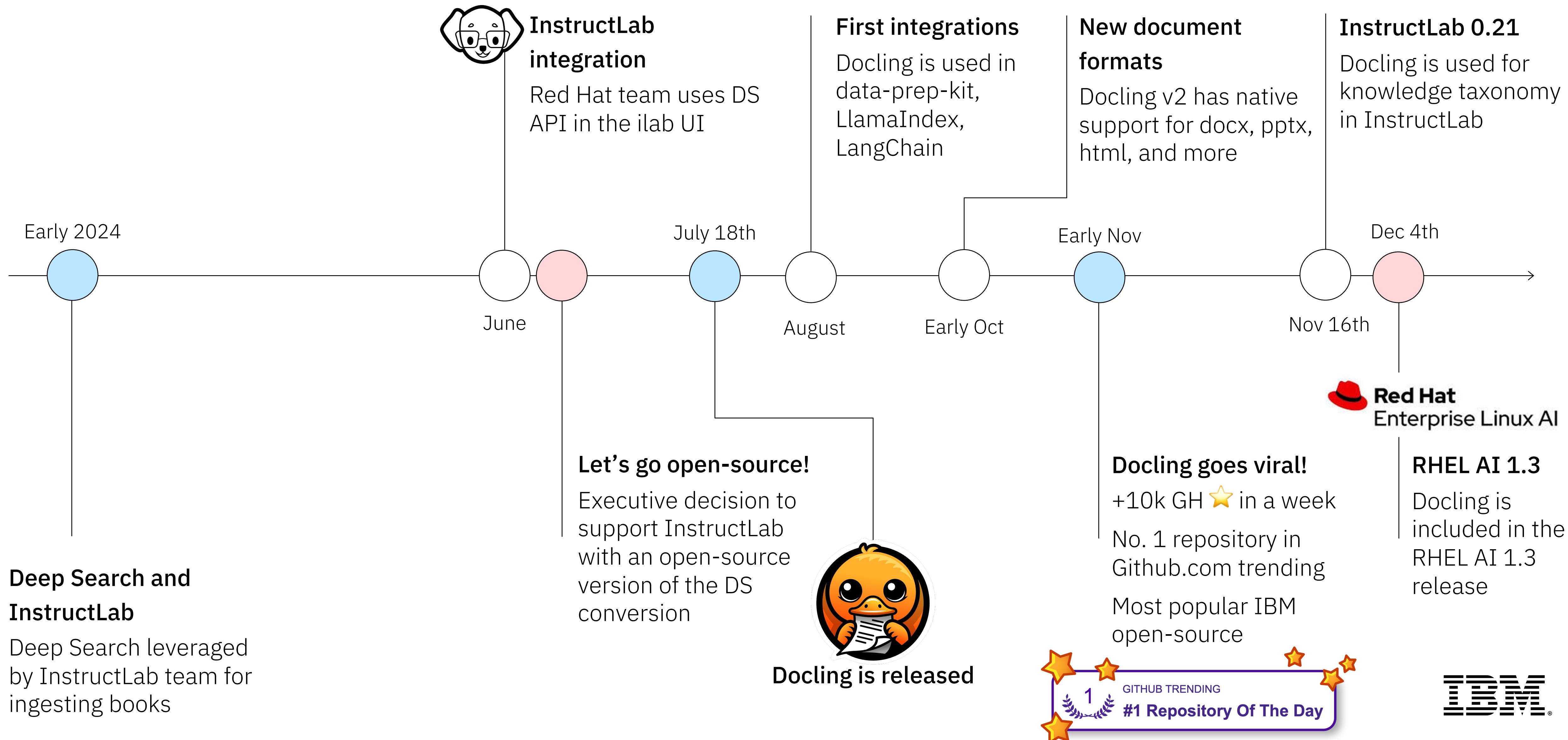
# a single document to markdown
docling https://arxiv.org/pdf/2408.09869.pdf

# a folder of documents to markdown and json
docling --to json --to md ./inputs/
```



IBM.

The story behind Docling



Docling in 1Q25



Going viral!

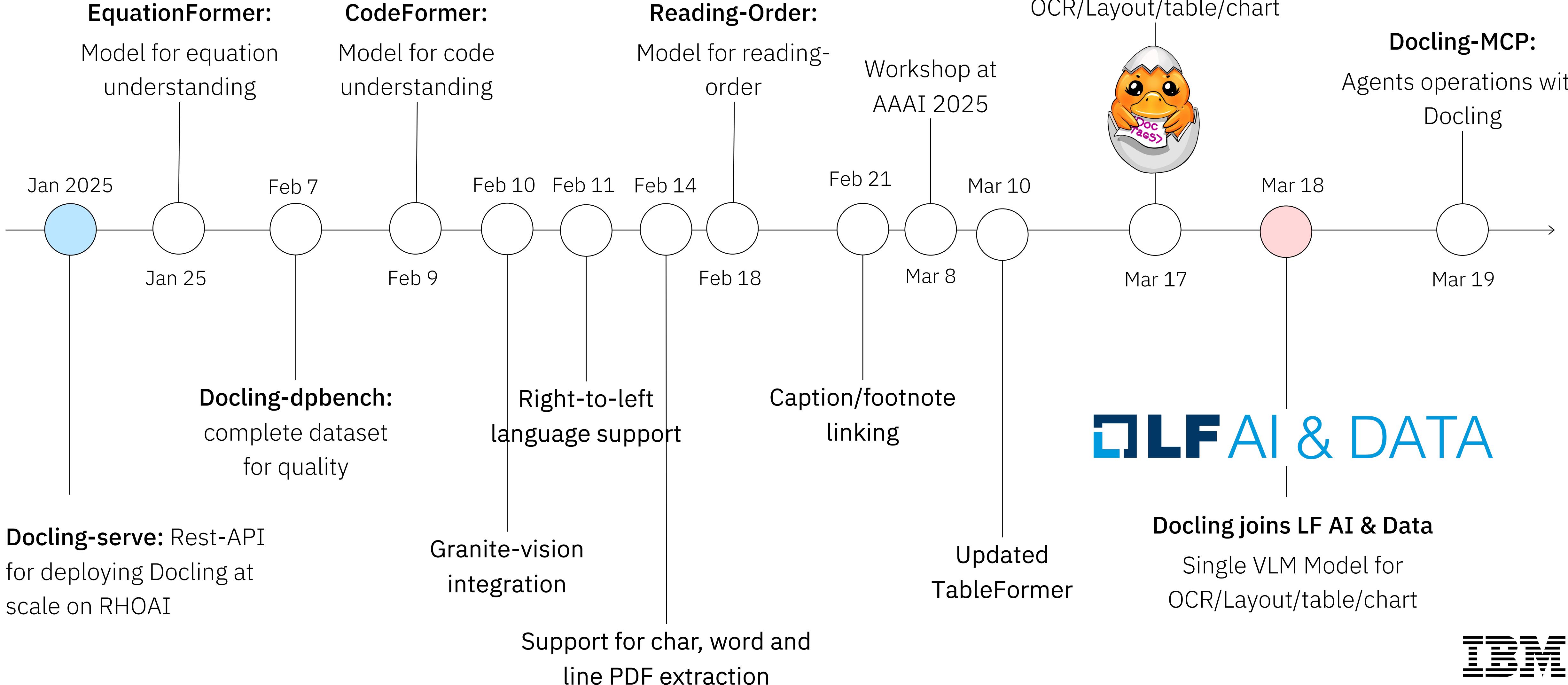
SmolDocling:

Single VLM Model for
OCR/Layout/table/chart



Docling-MCP:
Agents operations with
Docling

Mar 19

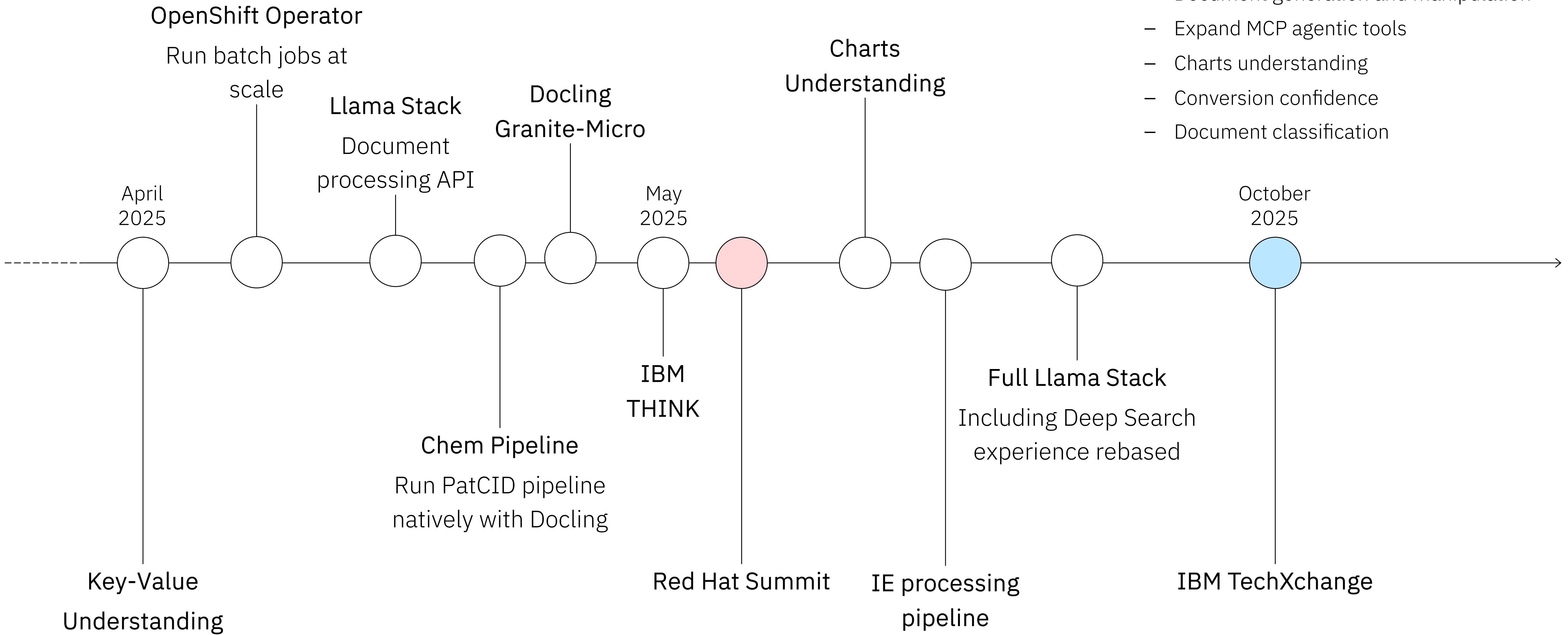


LF AI & DATA

Docling joins LF AI & Data
Single VLM Model for
OCR/Layout/table/chart

IBM

Docling: Roadmap

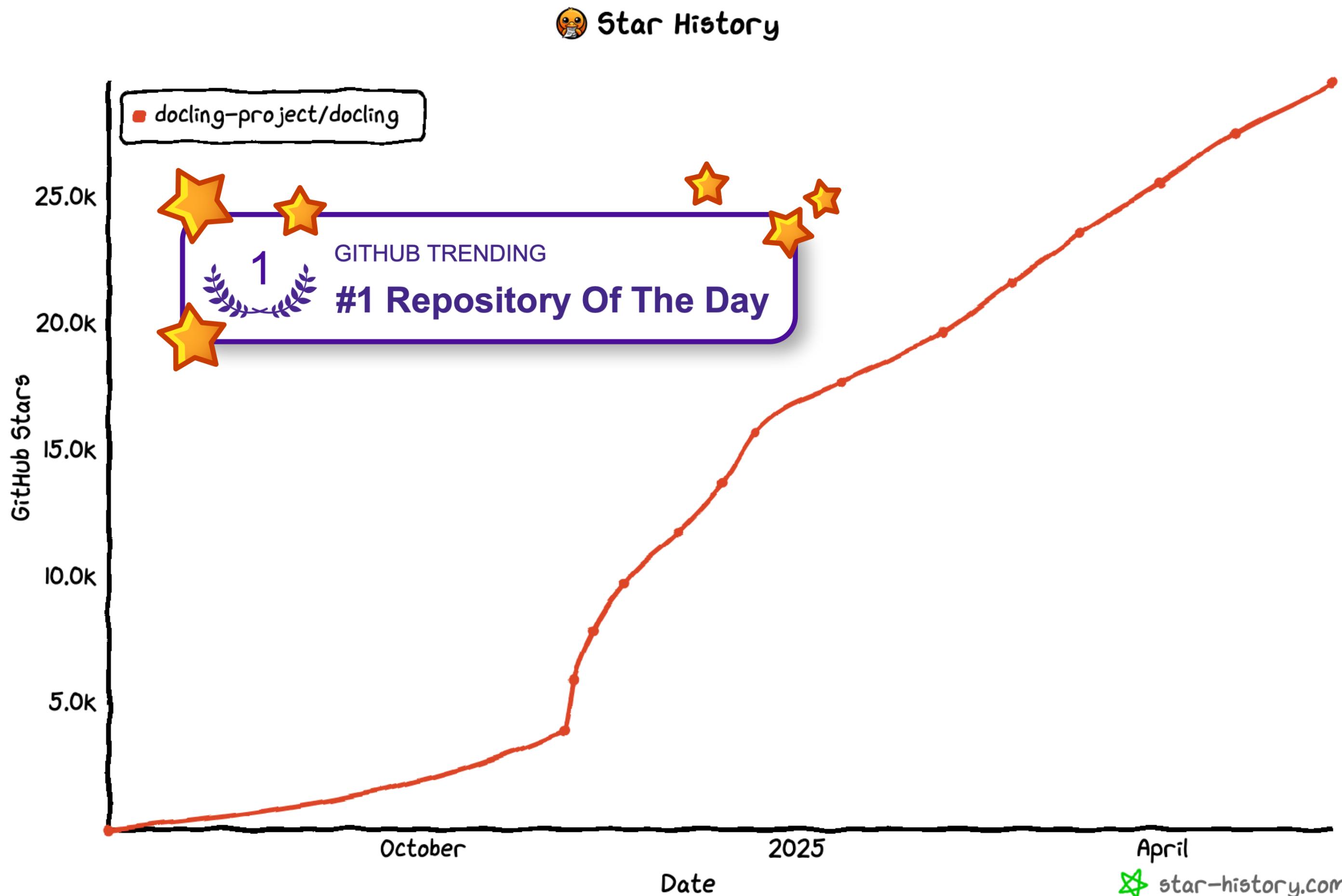


Open points

- Schema-based extraction via SmolDocling
- Expanding Markush understanding
- Document generation and manipulation
- Expand MCP agentic tools
- Charts understanding
- Conversion confidence
- Document classification

Community Adoption

A rising star in the open-source space



⭐ 29k+ GitHub stars

10M+ Social impressions

🐍 410k+ downloads/month from [PyPi](#)

30+ YouTube videos

Countless Articles (Blogs, Medium...)

Ecosystem

RAG Frameworks

Agentic systems

Dataset generation

Vector databases

Client solutions



The AI Alliance



U.S. General Services
Administration



spaCy



ag2ai



Bee Agent Framework



Docling



InstructLab



Open WebUI



Anything LLM



Open WebUI

DocLayNet: A Large Human-Annotated Dataset for Document-Layout Analysis

Birgit Pfitzmann
IBM Research
Rueschlikon, Switzerland
bpf@zurich.ibm.com

Christoph Auer
IBM Research
Rueschlikon, Switzerland
cau@zurich.ibm.com

Michele Dolfi
IBM Research
Rueschlikon, Switzerland
dol@zurich.ibm.com

Ahmed S. Nassar
IBM Research
Rueschlikon, Switzerland
ahn@zurich.ibm.com

Peter Staar
IBM Research
Rueschlikon, Switzerland
taa@zurich.ibm.com

ABSTRACT

Accurate document layout analysis is a key requirement for high-quality PDF document conversion. With the recent availability of public, large ground-truth datasets such as PubLayNet and DocBank, deep-learning models have proven to be very effective at layout detection and segmentation. While these datasets are of adequate size to train such models, they severely lack in layout variability since they are sourced from scientific article repositories such as PubMed and arXiv only. Consequently, the accuracy of the layout segmentation drops significantly when these models are applied on more challenging and diverse layouts. In this paper, we present *DocLayNet*, a new, publicly available, document-layout annotation dataset in COCO format. It contains 80863 manually annotated pages from diverse data sources to represent a wide variability in layouts. For each PDF page, the layout annotations provide labelled bounding-boxes with a choice of 11 distinct classes. *DocLayNet* also provides a subset of double- and triple-annotated pages to determine the inter-annotator agreement. In multiple experiments, we provide baseline accuracy scores (in mAP) for a set of popular object detection models. We also demonstrate that these models fall approximately 10% behind the inter-annotator agreement. Furthermore, we provide evidence that *DocLayNet* is of sufficient size. Lastly, we compare models trained on PubLayNet, DocBank and *DocLayNet*, showing that layout predictions of the *DocLayNet*-trained models are more robust and thus the preferred choice for general-purpose document-layout analysis.

CCS CONCEPTS

- Information systems → Document structure; • Applied computing → Document analysis; • Computing methodologies → Machine learning; Computer vision; Object detection;

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

KDD '22, August 14–18, 2022, Washington, DC, USA

© 2022 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-9385-0/22/08.

<https://doi.org/10.1145/3534678.3539043>

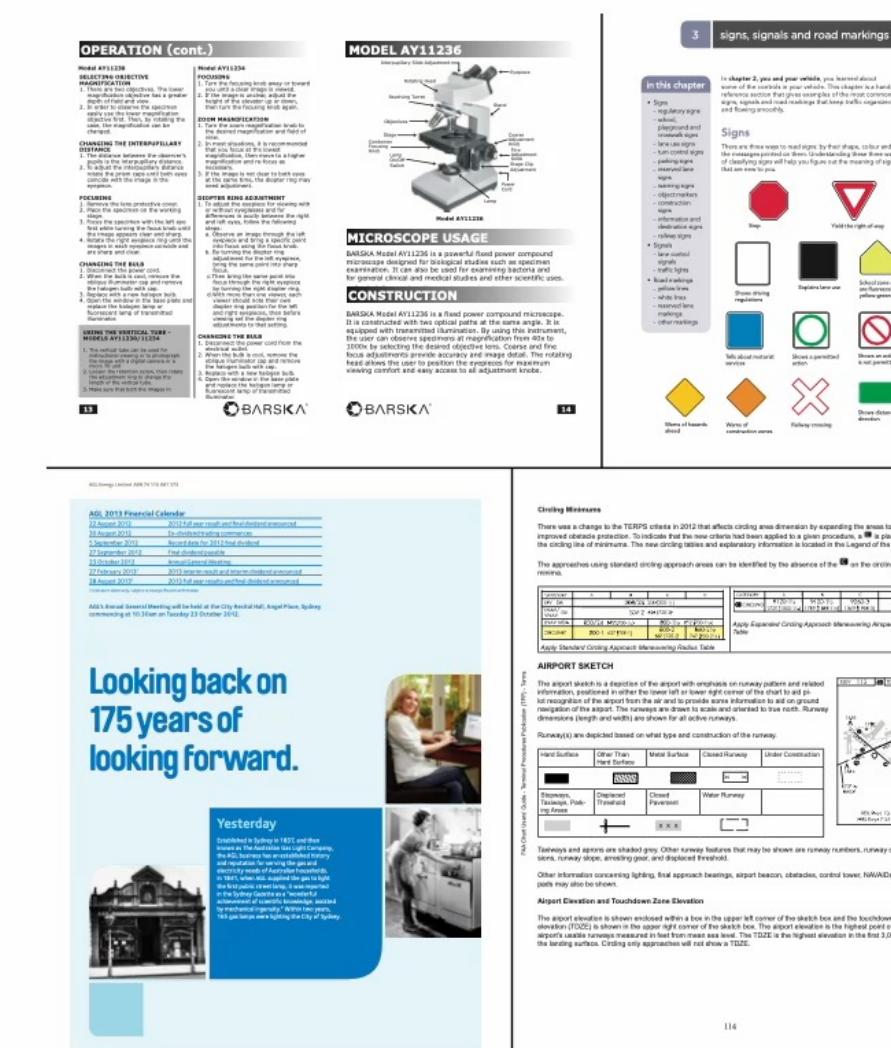


Figure 1: Four examples of complex page layouts across different document categories

KEYWORDS

PDF document conversion, layout segmentation, object-detection, data set, Machine Learning

ACM Reference Format:

Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S. Nassar, and Peter Staar. 2022. DocLayNet: A Large Human-Annotated Dataset for Document-Layout Analysis. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA*. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3534678.3539043>

Without Docling...

...it can go bad.



gurovdigital 15 h

lol, over 20 scientific papers now feature the

were incubated with an extract from spores' as integrated at pH 7.0. Peptide was released which established that the coats contained substrate for the lytic enzyme present in spores. Peptide was also released from spore coats of *B. megaterium* by the action of the enzyme from *B. cereus* spores. The lytic enzyme did not attack intact resting spores.

The spore develops in the vegetative cell, which thus becomes a sporangium. It is by no means certain what happens to the vegetative cell wall when the spore is released. In *Clostridium* species it appears that at least part of this structure is retained as an outer membrane around the spore. It is the opinion of some workers that the wall of the sporulating cell forms the exosporium which exists as an outer

...

acteristic type. It was concluded that at least part of the sporangial wall was dissolved away to allow release of the spore. It appears likely that the exosporium of *B. cereus* does not have a composition similar to that of the vegetative cell wall, from the results obtained by Dr. J. R.

Date syrup (as one of the agricultural wastes) was used to produce bacterial cellulose using *Gluconastobacter xylinus*. Fourier transform infrared spectroscopy (FTIR), vegetative electron microscopy, and X-ray diffraction were used to determine the structure of bacterial cellulose, cellulose fibers, and crystallinity of the samples (Moosavi and

Silver and gold nanoparticles for

[HTML] m



The spore develops in the vegetative cell, which thus becomes a sporangium. It is by no means certain what happens to the vegetative cell wall when the spore is released. In *Clostridium* species it appears that at least part of this structure is retained as an outer membrane around the spore. It is the opinion of some workers that the wall of the sporulating cell forms the exosporium which exists as an outer coat around spores of several *Bacillus* species. Spores of several varieties of *B. cereus* had exospria whereas these structures appeared to be absent from spores of *B. megaterium* and *B. subtilis*. It seems, however, that in *Bacillus* species at least, the greater part of the vegetative cell wall is dissolved away before the developed spore is released. If this is true, then soluble components containing the characteristic constituents should appear in the medium during spore release. Culture filtrates from *B. cereus* organisms at various stages of growth and sporulation were hydrolyzed and the hydrolyzates analyzed for amino sugars and diaminopimelic acid (28). Results showed that a large increase in the concentration of these substances in the culture filtrate occurred during spore release (table 2); they were found to be present in a nondialyzable peptide of the characteristic type. It was concluded that at least part of the sporangial wall was dissolved away to allow release of the spore. It appears likely that the exosporium of *B. cereus* does not have a composition similar to that of the vegetative cell wall, from the results obtained by Dr. J. R. Norris of Leeds University (personal communication). He treated spores with a highly active preparation of lytic enzyme from *B. cereus* spores and examined the effect by means of electron microscopy. No evidence of lysis of the exosporium was obtained.

Heart 692

Q 12

43



BRONZE PIECES FROM JEYRAN TEPE, UZBAKI

B SODAEI, H RAHNEMA - researchgate.net

This study is a report of the results of metallographic study of 5 bronze pieces found in Jeyrān Tepe dating back to the Iron

Recovering structured content from PDF with low-level PDF parsers

! undesired
page headers

KDD '22, August 14–18, 2022, Washington, DC, USA Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S. Nassar, and Peter Staar

Table 1: DocLayNet dataset overview. Along with the frequency of each class label, we present the relative occurrence (as % of row “Total”) in the train, test and validation sets. The inter-annotator agreement is computed as the mAP@0.5-0.95 metric between pairwise annotations from the triple-annotated pages, from which we obtain accuracy ranges.

class label	Count	% of Total		triple inter-annotator mAP @ 0.5-0.95 (%)								
		Train	Test	All	Fin	Man	Sci	Law	Pat	Ten		
Caption	22524	2.04	1.77	2.32	84-89	40-61	86-92	94-99	95-99	69-78	n/a	
Footnote	6318	0.60	0.31	0.58	83-91	n/a	100	62-88	85-94	n/a	82-97	
Formula	25027	2.25	1.90	2.96	83-85	n/a	n/a	84-87	86-96	n/a	n/a	
List-item	185660	17.19	13.34	15.82	87-88	74-83	90-92	97-97	81-85	75-88	93-95	
Page-footer	70878	6.51	5.58	6.00	93-94	88-90	95-96	100	92-97	100	96-98	
Page-header	58022	5.10	6.70	5.06	85-89	66-76	90-94	98-100	91-92	97-99	81-86	
Picture	45976	4.21	2.78	5.31	69-71	56-59	82-86	69-82	80-95	66-71	59-76	
Section-header	142884	12.60	15.77	12.85	83-84	76-81	90-92	94-95	87-94	69-73	78-86	
Table	34733	3.20	2.27	3.60	77-81	75-80	83-86	98-99	58-80	79-84	70-85	
Text	510377	45.82	49.28	45.00	84-86	81-86	88-93	89-93	87-92	71-79	87-95	
Title	5071	0.47	0.30	0.50	60-72	24-63	50-63	94-100	82-96	68-79	24-56	
Total	1107470	941123	99816	66531	82-83	71-74	79-81	89-94	86-91	71-76	68-85	

Figure 3: Corpus Conversion Service annotation user interface. The PDF page is shown in the background, with overlaid text-cells (in darker shades). The annotation boxes can be drawn by dragging a rectangle over each segment with the respective label from the palette on the right.

we distributed the annotation workload and performed continuous quality controls. Phase one and two required a small team of experts only. For phases three and four, a group of 40 dedicated annotators were assembled and supervised.

Phase 1: Data selection and preparation. Our inclusion criteria for documents were described in Section 3. A large effort went into ensuring that all documents are free to use. The data sources

³<https://arxiv.org/>

KDD '22, August 14–18, 2022, Washington, DC, USA Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S. Nassar, and Peter Staar

Table 1: DocLayNet dataset overview. Along with the frequency of each class label, we present the relative occurrence (as % of row “Total”) in the train, test and validation sets. The inter-annotator agreement is computed as the mAP@0.5-0.95 metric between pairwise annotations from the triple-annotated pages, from which we obtain accuracy ranges.

% of Total

triple inter-annotator mAP @ 0.5-0.95 (%)

[...]

Count

22524
6318
25027
185660
70878
58022
45976
142884
34733
510377
5071
1107470

[...]

include publication repositories such as arXiv³, government offices, company websites as well as data directory services for financial reports and patents. Scanned documents were excluded wherever possible because they can be rotated or skewed. This would not allow us to perform annotation with rectangular bounding-boxes and therefore complicate the annotation process.

Preparation work included uploading and parsing the sourced PDF documents in the Corpus Conversion Service (CCS) [22], a cloud-native platform which provides a visual annotation interface and allows for dataset inspection and analysis. The annotation interface of CCS is shown in Figure 3. The desired balance of pages between the different document categories was achieved by selective subsampling of pages with certain desired properties. For example, we made sure to include the title page of each document and bias the remaining page selection to those with figures or tables. The latter was achieved by leveraging pre-trained object detection models from PubLayNet, which helped us estimate how many figures and tables a given page contains.

Phase 2: Label selection and guideline. We reviewed the collected documents and identified the most common structural features they exhibit. This was achieved by identifying recurrent layout elements and lead us to the definition of 11 distinct class labels. These 11 class labels are *Caption*, *Footnote*, *Formula*, *List-item*, *Page-footer*, *Page-header*, *Picture*, *Section-header*, *Table*, *Text*, and *Title*. Critical factors that were considered for the choice of these class labels were (1) the overall occurrence of the label, (2) the specificity of the label, (3) recognisability on a single page (i.e. no need for context from previous or next page) and (4) overall coverage of the page. Specificity ensures that the choice of label is not ambiguous, while coverage ensures that all meaningful items on a page can be annotated. We refrained from class labels that are very specific to a document category, such as *Abstract* in the *Scientific Articles* category. We also avoided class labels that are tightly linked to the semantics of the text. Labels such as *Author* and *Affiliation*, as seen in DocBank, are often only distinguishable by discriminating on

! Tables not understood

! Image content missing

! Line wraps not understood

! Multi-column often breaks order

- ✓ Very fast and cheap
- ✗ Incomplete
- ✗ Loss of structure
- ✗ Noisy
- ➔ Unfit for most use cases

Recovering structured content from PDF with Docling

KDD '22, August 14–18, 2022, Washington, DC, USA Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S. Nassar, and Peter Staar

Table 1: DocLayNet dataset overview. Along with the frequency of each class label, we present the relative occurrence (as % of row ‘Total’) in the train, test and validation sets. The inter-annotator agreement is computed as the mAP@0.5-0.95 metric between pairwise annotations from the triple-annotated pages, from which we obtain accuracy ranges.

class label	Count	% of Total		triple inter-annotator mAP @ 0.5-0.95 (%)							
		Train	Test	All	Fin	Man	Sci	Law	Pat	Ten	
Caption	22524	2.04	1.77	2.32	84-89	40-61	86-92	94-99	95-99	69-78	n/a
Footnote	6318	0.60	0.31	0.58	83-91	n/a	100	62-88	85-94	n/a	82-97
Formula	25027	2.25	1.90	2.96	83-85	n/a	n/a	84-87	86-96	n/a	n/a
List-item	185660	17.19	13.34	15.82	87-88	74-83	90-92	97-97	81-85	75-88	93-95
Page-footer	70878	6.51	5.58	6.00	93-94	88-90	95-96	100	92-97	100	96-98
Page-header	58022	5.10	6.70	5.06	85-89	66-76	90-94	98-100	91-92	97-99	81-86
Picture	45976	4.21	2.78	5.31	69-71	56-59	82-86	69-82	80-95	66-71	59-76
Section-header	142884	12.60	15.77	12.85	83-84	76-81	90-92	94-95	87-94	69-73	78-86
Table	34733	3.20	2.27	3.60	77-81	75-80	83-86	98-99	58-80	79-84	70-85
Text	510377	45.82	49.28	45.00	84-86	81-86	88-93	89-93	87-92	71-79	87-95
Title	5071	0.47	0.30	0.50	60-72	24-63	50-63	94-100	82-96	68-79	24-56
Total	1107470	941123	99816	66531	82-83	71-74	79-81	89-94	86-91	71-76	68-85

Figure 3: Corpus Conversion Service annotation user interface. The PDF page is shown in the background, with overlaid text-cells (in darker shades). The annotation boxes can be drawn by dragging a rectangle over each segment with the respective label from the palette on the right.

we distributed the annotation workload and performed continuous quality controls. Phase one and two required a small team of experts only. For phases three and four, a group of 40 dedicated annotators were assembled and supervised.

Phase 1: Data selection and preparation. Our inclusion criteria for documents were described in Section 3. A large effort went into ensuring that all documents are free to use. The data sources

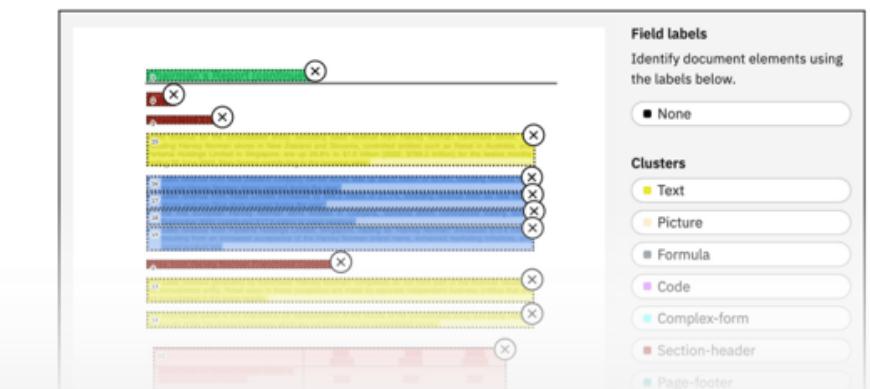
³<https://arxiv.org/>



Table 1: DocLayNet dataset overview. Along with the frequency of each class label, we present the relative occurrence (as % of row ‘Total’) in the train, test and validation sets. The inter-annotator agreement is computed as the mAP@0.5-0.95 metric between pairwise annotations from the triple-annotated pages, from which we obtain accuracy ranges.

class label	Count	% of Total					triple inter-annotator mAP @ 0.5-0.95 (%)				
		Train	Test	Val	All	Fin	Man	Sci	Law	Pat	Ten
Caption	22524	2.04	1.77	2.32	84-89	40-61	86-92	94-99	95-99	69-78	n/a
Footnote	6318	0.60	0.31	0.58	83-91	n/a	100	62-88	85-94	n/a	82-97
Formula	25027	2.25	1.90	2.96	83-85	n/a	n/a	84-87	86-96	n/a	n/a
List-item	185660	17.19	13.34	15.82	87-88	74-83	90-92	97-97	81-85	75-88	93-95
Page-footer	70878	6.51	5.58	6.00	93-94	88-90	95-96	100	92-97	100	96-98
Page-header	58022	5.10	6.70	5.06	85-89	66-76	90-94	98-100	91-92	97-99	81-86
Picture	45976	4.21	2.78	5.31	69-71	56-59	82-86	69-82	80-95	66-71	59-76
Section-header	142884	12.60	15.77	12.85	83-84	76-81	90-92	94-95	87-94	69-73	78-86
Table	34733	3.20	2.27	3.60	77-81	75-80	83-86	98-99	58-80	79-84	70-85
Text	510377	45.82	49.28	45.00	84-86	81-86	88-93	89-93	87-92	71-79	87-95
Title	5071	0.47	0.30	0.50	60-72	24-63	50-63	94-100	82-96	68-79	24-56
Total	1107470	941123	99816	66531	82-83	71-74	79-81	89-94	86-91	71-76	68-85

Figure 3: Corpus Conversion Service annotation user interface. The PDF page is shown in the background, with overlaid text-cells (in darker shades). The annotation boxes can be drawn by dragging a rectangle over each segment with the respective label from the palette on the right.



*results rendered as HTML for visualization purposes

- ✓ Good quality
 - ✓ Fast and cheap
 - ✓ Fully local operation
 - ✓ Structured format output
-
- Cost-effective at scale, with consistent representation and high quality

A full-stack approach for the open hybrid cloud to scale Document processing across the enterprise



Docling

Forever opensource

[LLM/VLM & Agentic Ready](#)

- Open source python package for PDF document conversion
- Understand Layout, Reading order, tables and extract meta datas
- Includes a simple CLI



Red Hat

Docling on RedHat

Scale Docling to Enterprise

[Deploy and scale conversion anywhere](#)

- Easy integration to llamaindex and langchain
- Agents integration as a tool to process PDFs in frameworks like crew.ai and LangGraph
- Leverage SOTA LLM/VLM

[Resource optimization and management](#)

- Delivers consistency, cloud-to-edge production deployment, and monitoring capabilities for document conversion
- Fully managed, and self-managed via OpenShift Enterprise Linux footprint

[watsonx Discovery](#)

- Search millions of preloaded documents: Patents, Open-Access Publications, Public Repositories and News articles
- Search made for enterprise in multimodal

[watsonx Orchestrate](#)

- Orchestrate AI Agents to discover knowledge in documents and mix them with other datasources to create knowledge graphs and datasets.
- Schema based structured data extraction from unstructured documents.

IBM Consulting

[Transforms your organization with advanced technologies](#)

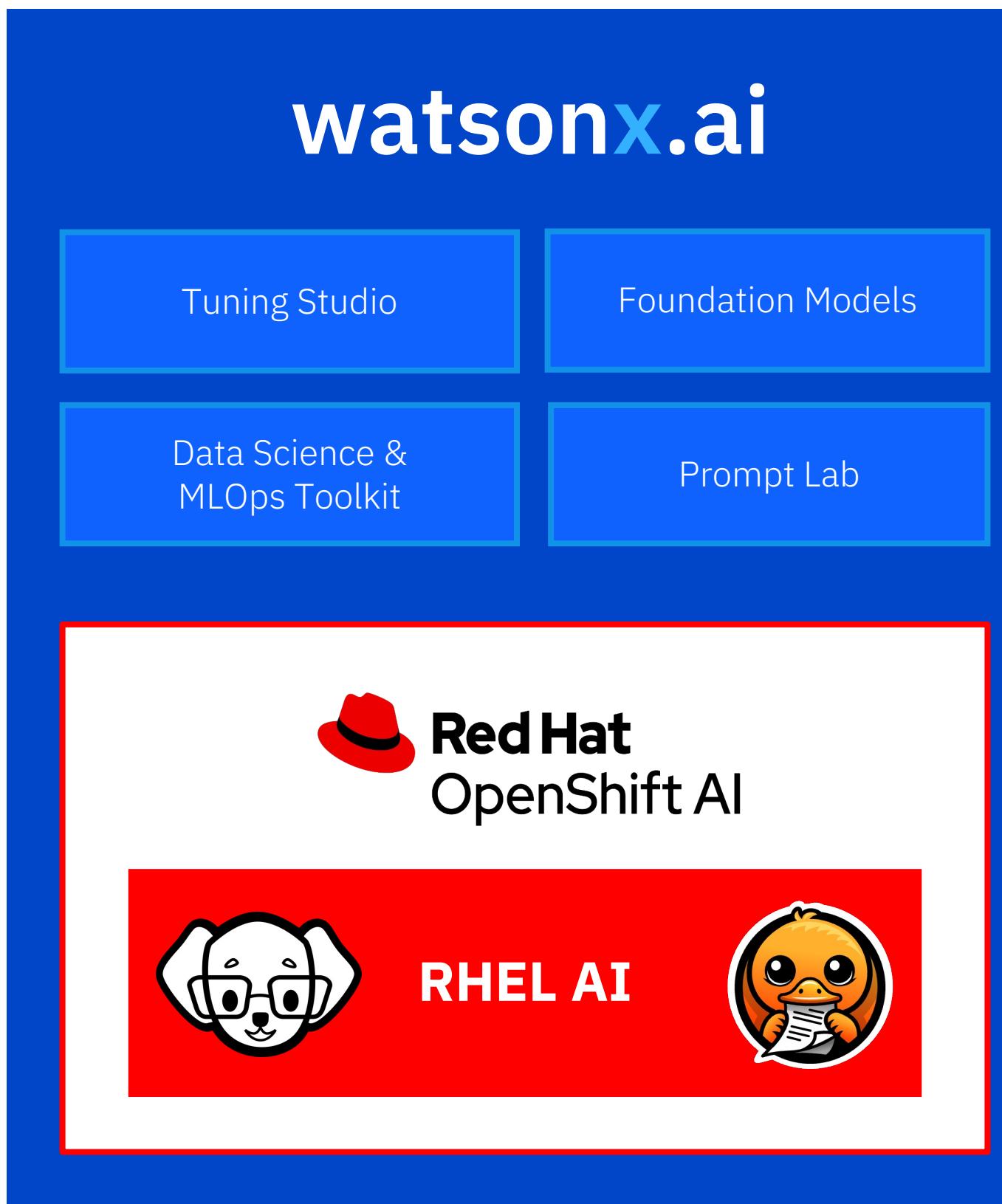
- Ensure design, integration, deployment and adoption of novel methods and tools for processing documents
- Create novel components with IBM Research to make your science unique and disruptive

watsonx.ai™

IBM

Build advanced applications

watsonx.ai leverages the power of community that Red Hat OpenShift AI and RHEL AI offer



watsonx.ai

Multi-model flexibility

Access a variety of fit-for-purpose foundation models

Enterprise AI/ML lifecycle

Address LOB multi-user and organizational needs for scale and lifecycle governance

Red Hat
OpenShift AI

Model serving & monitoring

Deploy models anywhere with consistent cloud-to-edge production deployment, and monitoring capabilities

Resource optimization and management

Create model pipelines for tuning and alignment, validation and delivery; Share and optimize resources across teams

RHEL AI

Foundation model runtime engine

Develop, deploy, and test open Granite language/code models, supported & indemnified from Red Hat and IBM

Developer-friendly model toolkit

Bring rapid LLM experimentation to developers via supported InstructLab CLI

Runs on any infrastructure



Physical



Virtual

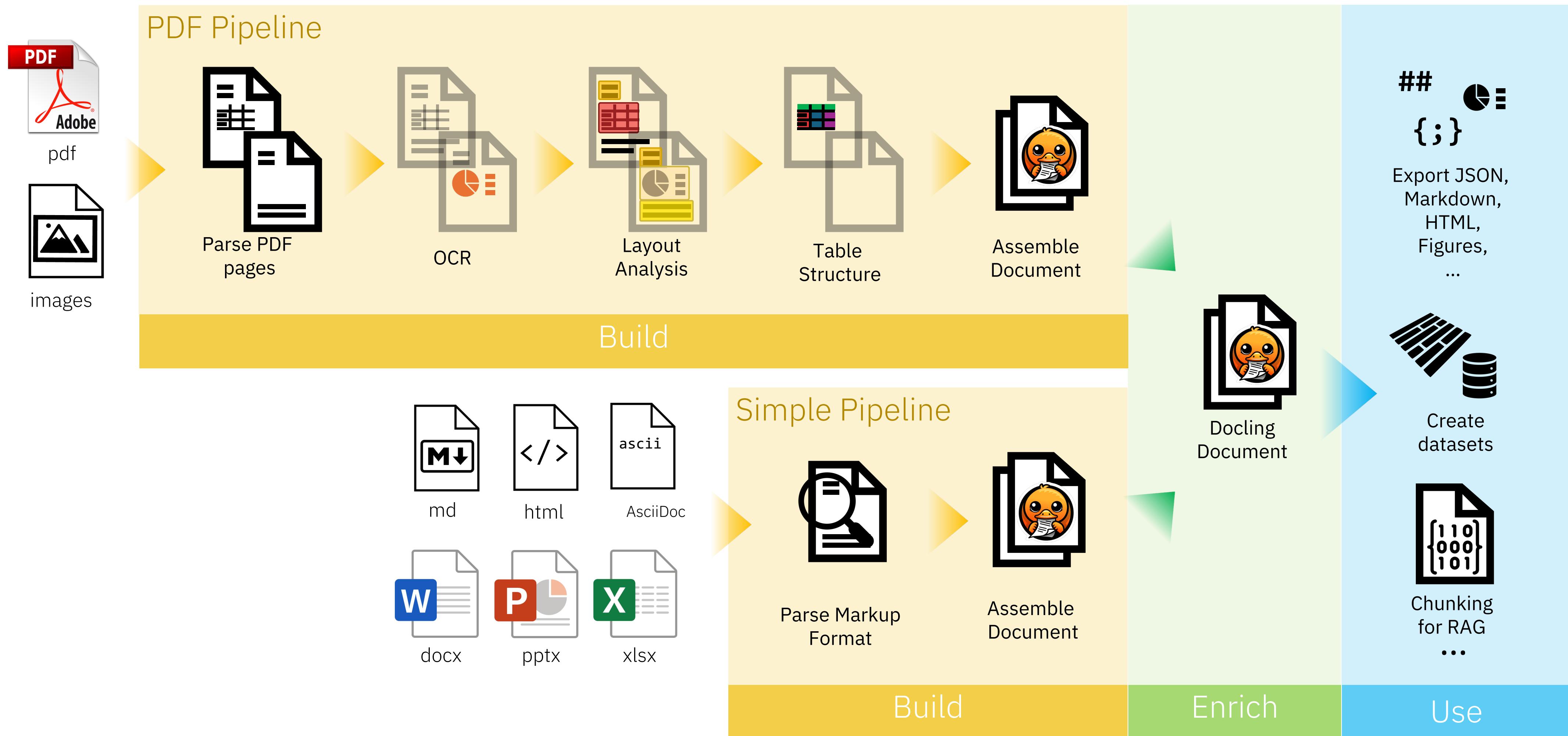


Private



Public

Design and Architecture



Enrichment models

Code and Formula model

A.1.4. Direct Preference Optimization (DPO)

The objective of DPO is:

$$J_{DPO}(\theta) = E[q \sim P_{sft}(Q), o^+, o^- \sim \pi_{sft}(O \mid q)] \log \left(\beta \frac{1}{|o^+|} \sum_{t=1}^{|o^+|} \log \frac{\pi_\theta(o_t^+ \mid q, o_{<t}^+) \pi_\theta(o_t^- \mid q, o_{<t}^-)}{\pi_{\text{SFT}}(o_t^+ \mid q, o_{<t}^+) \pi_{\text{SFT}}(o_t^- \mid q, o_{<t}^-)} - \beta \frac{1}{|o^-|} \sum_{t=1}^{|o^-|} \log \frac{\pi_\theta(o_t^- \mid q, o_{<t}^-) \pi_\theta(o_t^+ \mid q, o_{<t}^+)}{\pi_{\text{SFT}}(o_t^- \mid q, o_{<t}^-) \pi_{\text{SFT}}(o_t^+ \mid q, o_{<t}^+)} \right) \quad (12)$$

The gradient of $J_{DPO}(\theta)$ is:

$$\nabla_\theta J_{DPO}(\theta) = E[q \sim P_{sft}(Q), \theta^+, \theta^- \sim \pi_{sft}(O \mid q)] \left(\frac{1}{|\theta^+|} \sum_{t=1}^{|o^+|} GC_{DPO}(q, \theta, t) \nabla_\theta \log \pi_\theta(o_t^+ \mid q, \theta_{<t}^+) - \frac{1}{|\theta^-|} \sum_{t=1}^{|o^-|} GC_{DPO}(q, \theta, t) \nabla_\theta \log \pi_\theta(o_t^- \mid q, \theta_{<t}^-) \right)$$

Data Source: question in SFT dataset with outputs sampled from SFT model. Reward Function: human preference in the general domain (can be 'Rule' in mathematical tasks). Gradient Coefficient:

$$GC_{DPO}(q, \theta, t) = \sigma \left(\beta \log \frac{\pi_\theta(o_t^+ \mid q, o_{<t}^+)}{\pi_{\text{SFT}}(o_t^+ \mid q, o_{<t}^+)} - \beta \log \frac{\pi_\theta(o_t^- \mid q, o_{<t}^-)}{\pi_{\text{SFT}}(o_t^- \mid q, o_{<t}^-)} \right)$$

A.1.5. Proximal Policy Optimization (PPO)

The objective of PPO is:

$$J_{PPO}(\theta) = E[q \sim P_{sft}(Q), o \sim \pi_{sft}(O \mid q)] \frac{1}{|o|} \sum_{t=1}^{|o|} \min \left[\frac{\pi_\theta(o_t \mid q, o_{<t})}{\pi_{\text{SFT}}(o_t \mid q, o_{<t})} A_t, \text{clip} \left(\frac{\pi_\theta(o_t \mid q, o_{<t})}{\pi_{\text{SFT}}(o_t \mid q, o_{<t})}, 1 - \epsilon, 1 + \epsilon \right) A_t \right].$$

To simplify the analysis, it is assumed that the model only has a single update following each exploration stage, thereby ensuring that $\pi \neq \pi_{old}$. In this case, we can remove the min and clip operation:

$$J_{PPO}(\theta) = E[q \sim P_{sft}(Q), o \sim \pi_{sft}(O \mid q)] \frac{1}{|o|} \sum_{t=1}^{|o|} \frac{\pi_\theta(o_t \mid q, o_{<t})}{\pi_{\text{SFT}}(o_t \mid q, o_{<t})} A_t.$$

The gradient of $J_{PPO}(\theta)$ is:

$$\nabla_\theta J_{PPO}(\theta) = E[q \sim P_{sft}(Q), o \sim \pi_{sft}(O \mid q)] \frac{1}{|o|} \sum_{t=1}^{|o|} A_t \nabla_\theta \log \pi_\theta(o_t \mid q, o_{<t}).$$

Data Source: question in SFT dataset with outputs sampled from policy model. Reward Function: reward model. Gradient Coefficient:

$$GC_{PPO}(q, a, t, \pi_{old}) = A_t,$$

where A_t is the advantage, which is computed by applying Generalized Advantage Estimation (GAE) (Schulman et al., 2015), based on the rewards $\{r_t\}$ and a learned value function $V \psi$.

A.1.6. Group Relative Policy Optimization (GRPO)

The objective of GRPO is (assume $\pi_{old} = \pi$ for simplified analysis):

$$J_{GRPO}(\theta) = E[q \sim P_{sft}(Q), \{o_i\}_{i=1}^G \sim \pi_{sft}(O \mid q)] \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left[\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\text{SFT}}(o_{i,t} \mid q, o_{i,<t})} \hat{A}_{i,t} - \beta \left(\frac{\pi_{ref}(o_{i,t} \mid q, o_{i,<t})}{\pi_\theta(o_{i,t} \mid q, o_{i,<t})} - \log \frac{\pi_{ref}(o_{i,t} \mid q, o_{i,<t})}{\pi_\theta(o_{i,t} \mid q, o_{i,<t})} - 1 \right) \right].$$

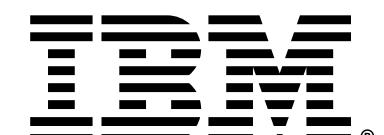
The gradient of $J_{GRPO}(\theta)$ is:

$$\nabla_\theta J_{GRPO}(\theta) = E[q \sim P_{sft}(Q), \{o_i\}_{i=1}^G \sim \pi_{sft}(O \mid q)] \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left[\hat{A}_{i,t} + \beta \left(\frac{\pi_{ref}(o_{i,t} \mid o_{i,<t})}{\pi_\theta(o_{i,t} \mid o_{i,<t})} - 1 \right) \right] \nabla_\theta \log \pi_\theta(o_{i,t} \mid q, o_{i,<t}).$$

- Specialized enrichment for Code blocks and Formula
- Detection of 50+ programming languages
- Extraction of equation formula in LaTeX



<https://huggingface.co/ds4sd/CodeFormula>



Enrichment models

Picture understanding

```
from docling.document_converter import DocumentConverter, PdfFormatOption
from docling.datamodel.pipeline_options import (
    PdfPipelineOptions, PictureDescriptionVlmOptions
)
from docling.datamodel.base_models import InputFormat

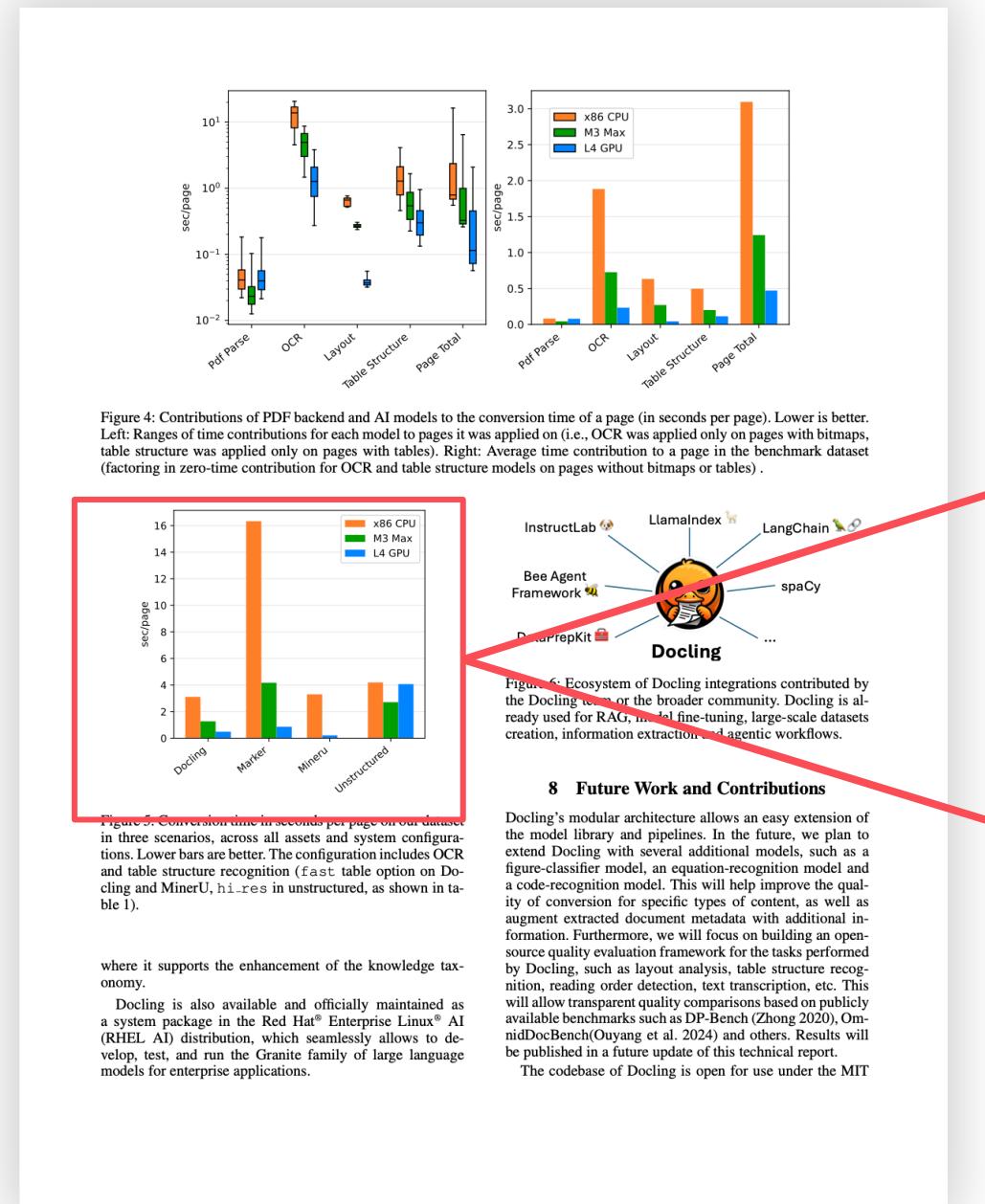
pipeline_options = PdfPipelineOptions()
pipeline_options.generate_picture_images = True
pipeline_options.images_scale = 2

# Picture classification
pipeline_options.do_picture_classification = True

# Picture description
pipeline_options.do_picture_description = True
picture_description_options = PictureDescriptionVlmOptions(
    repo_id="ibm-granite/granite-vision-3.2-2b",
    prompt="Describe the image in three sentences. Be concise and accurate.",
)

converter = DocumentConverter(format_options={
    InputFormat.PDF: PdfFormatOption(pipeline_options=pipeline_options)
})

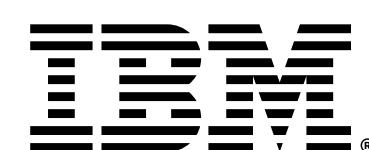
result = converter.convert("https://arxiv.org/pdf/2501.17887")
doc = result.document
```



PictureClassificationClass(
 class_name='bar_chart',
 confidence=0.9999500513076782
)

In this image we can see a bar chart. In the chart we can see the CPU, Max, GPU, and sec/page.

- Picture classifier specialized for figures found in documents, e.g. different chart types, flow diagrams, logos, signatures, etc.
- Leverage models to create a textual description of the picture



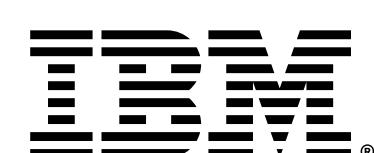
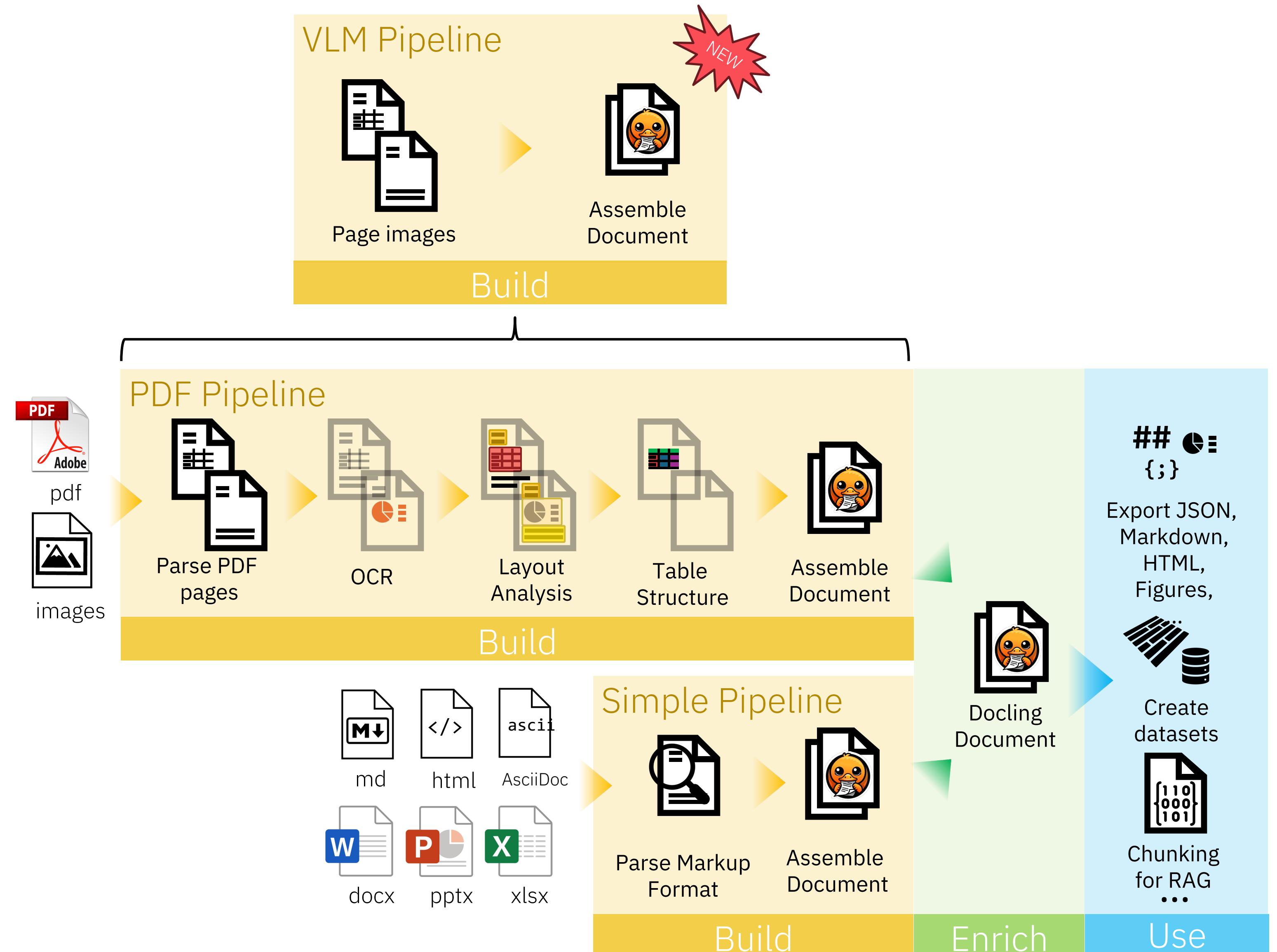
A new pipeline

Vision model pipelines

- Leverage new vision models for converting documents
- Perform all steps in one shot

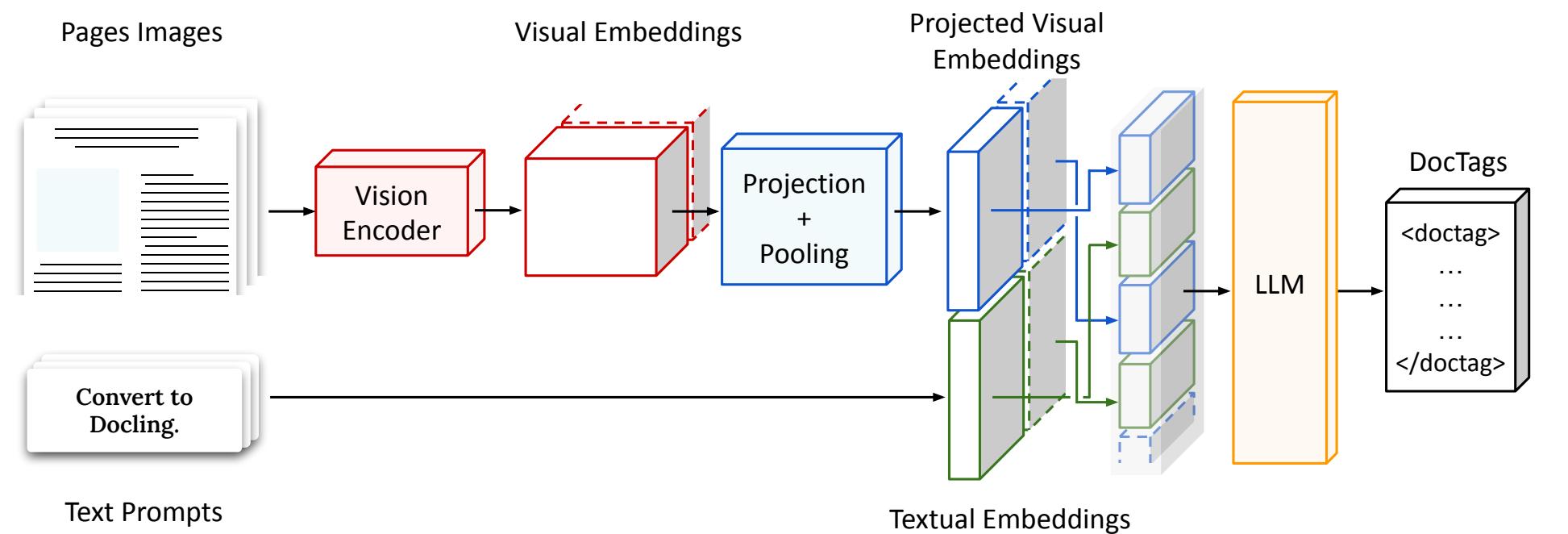
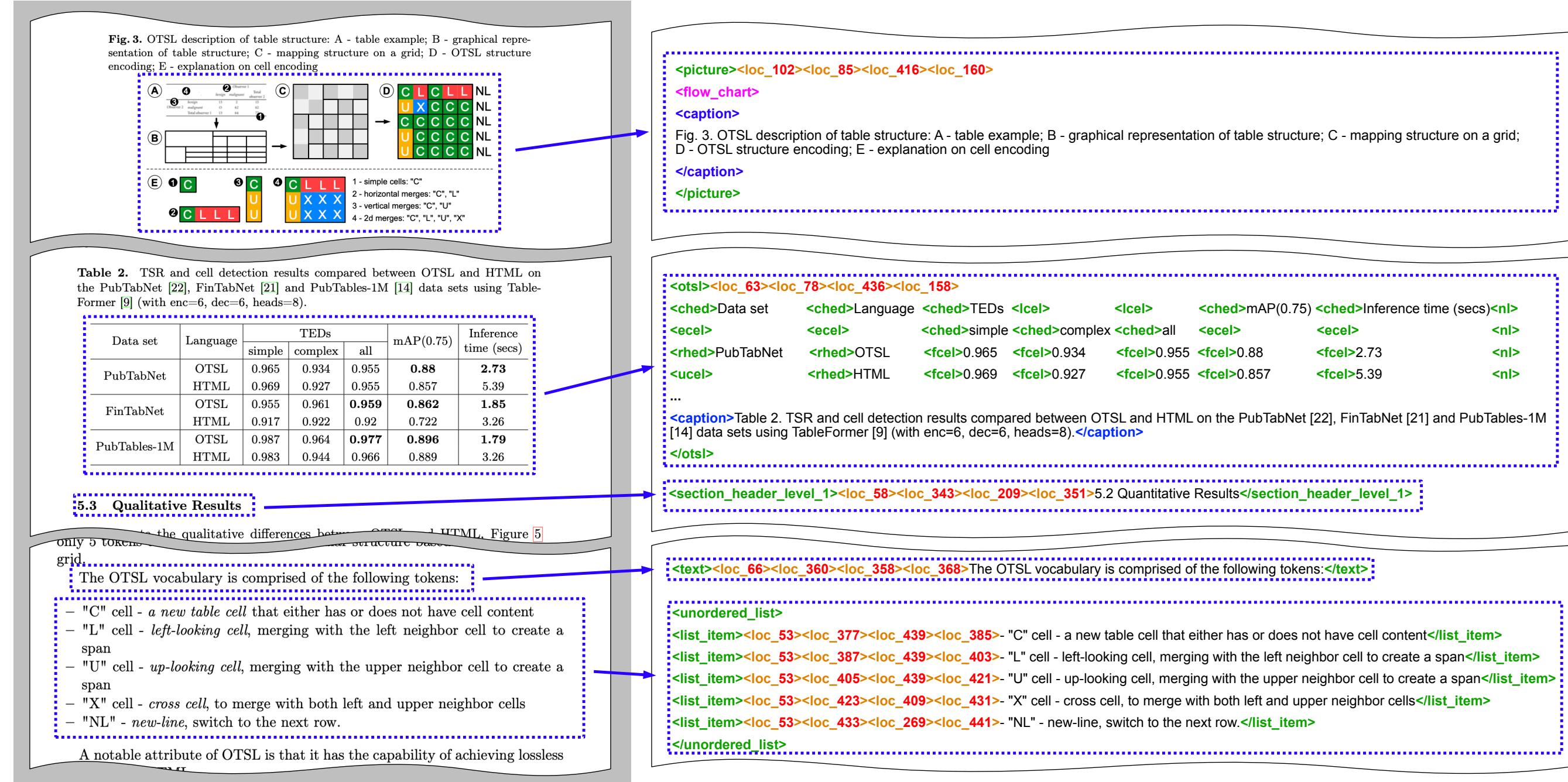
Option 1
Extract Markdown
→ DoclingDocument

Option 2:
Specialized token for layout prediction
→ DoclingDocument



SmolDocling

ultra-compact vision-language model for end-to-end document conversion



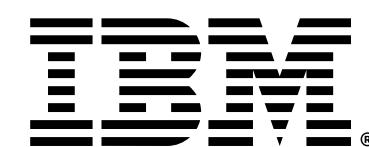
<https://huggingface.co/ds4sd/SmolDocling-256M-preview>

Ultra-compact size (256M parameters) with performance matching much larger models

Introduction of DocTags – a universal markup format preserving content with location

Comprehensive handling of tables, code, equations, and charts

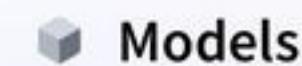
Lightning fast: Processes a single page in just 0.3 seconds on average



SmolDocling

ultra-compact vision-language model for end-to-end document conversion

Trending on 😊 this week



ds4sd/SmolDocling-256M-preview

Updated 1 day ago • ⚡ 27.9k • ❤ 847

mistralai/Mistral-Small-3.1-24B-Ins...

Updated 2 days ago • ⚡ 60.4k • ❤ 924

manycore-research/SpatialLM-Llama-1B

Updated 4 days ago • ⚡ 2.38k • ❤ 588

sesame/csm-1b

Updated 8 days ago • ⚡ 32k • ❤ 1.58k

deepseek-ai/DeepSeek-V3-0324

Updated about 6 hours ago • ❤ 471

Browse 1M+ models

Day 1

SmolDocling is trending #8 across all models in the hub!! #2 for vision models behind Gemma-3-27b 🍩

Day 2

Almost 8k downloads.

SmolDocling is #4 global trending (above qwen!). still #2 for vision!

...

Day 6

Almost 28k downloads.

SmolDocling is #1 global trending (above all!)

🎯 Ultra-compact size (256M parameters) with performance matching much larger models

💡 Introduction of DocTags – a universal markup format preserving content with location

📊 Comprehensive handling of tables, code, equations, and charts

⚡ Lightning fast: Processes a single page in just 0.3 seconds on average



<https://huggingface.co/ds4sd/SmolDocling-256M-preview>

IBM

SmolDocling

ultra-compact vision-language model for end-to-end document conversion

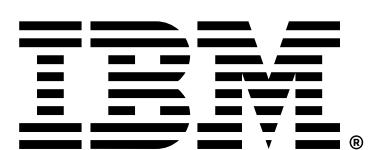


Blazing fast
SmolDocling conversion
on **Apple Silicon** via **MLX**



```
docling --pipeline vlm --vlm-model smoldocling my.pdf
```

<https://huggingface.co/ds4sd/SmolDocling-256M-preview>



Docling is for developers!

Examples

Conversion

Simple conversion

Custom conversion

Batch conversion

Multi-format conversion

Figure export

Figure enrichment

Table export

Multimodal export

Force full page OCR

Automatic OCR language detection with tesseract

RapidOCR with custom OCR models

Accelerator options

Simple translation

Conversion of custom XML

Chunking

Hybrid chunking

RAG with AI dev frameworks

RAG with Haystack

RAG with LangChain

RAG with LlamaIndex

More examples

```
● ● ●

files = ["test/data/2206.01062.pdf", ...]

doc_converter = DocumentConverter()
results = doc_converter.convert_all(files)

for conv_result in results:
    if conv_result.status == ConversionStatus.SUCCESS:
        doc = conv_result.document

        # save results
        doc.save_as_json("...")

        # inspect tables to pandas dataframe...
        for table in doc.tables:
            df_table = table.export_to_dataframe()
            ...

        # inspect the images
        for pic in doc.pictures:
            pil_img = pic.get_image()
            pil_img.show()
            ...
```

The screenshot shows a web-based API reference for the DoclingDocument type. At the top, there's a navigation bar with links to Home, Concepts, Examples, Integrations, and API reference. Below the navigation, the title "Docling Document" is displayed. A sidebar on the left contains links to API reference, Document Converter, Pipeline options, and Docling Document. The main content area starts with a section titled "mod doc" which describes it as a package for models defined by the Document type. It then lists "Classes:" with a bullet-pointed list of classes: DoclingDocument, DocumentOrigin, DocItem, DocItemLabel, ProvenanceItem, GroupItem, GroupLabel, NodeItem, PageItem, and FloatingItem. To the right of the class list is a "Table of contents" sidebar with detailed descriptions for each class, including their attributes and methods.

- Full-fledged SDK to inspect document content
- APIs to transform content to various formats and representations
- Tons of usage examples and notebooks
- Carefully designed `DoclingDocument` data model

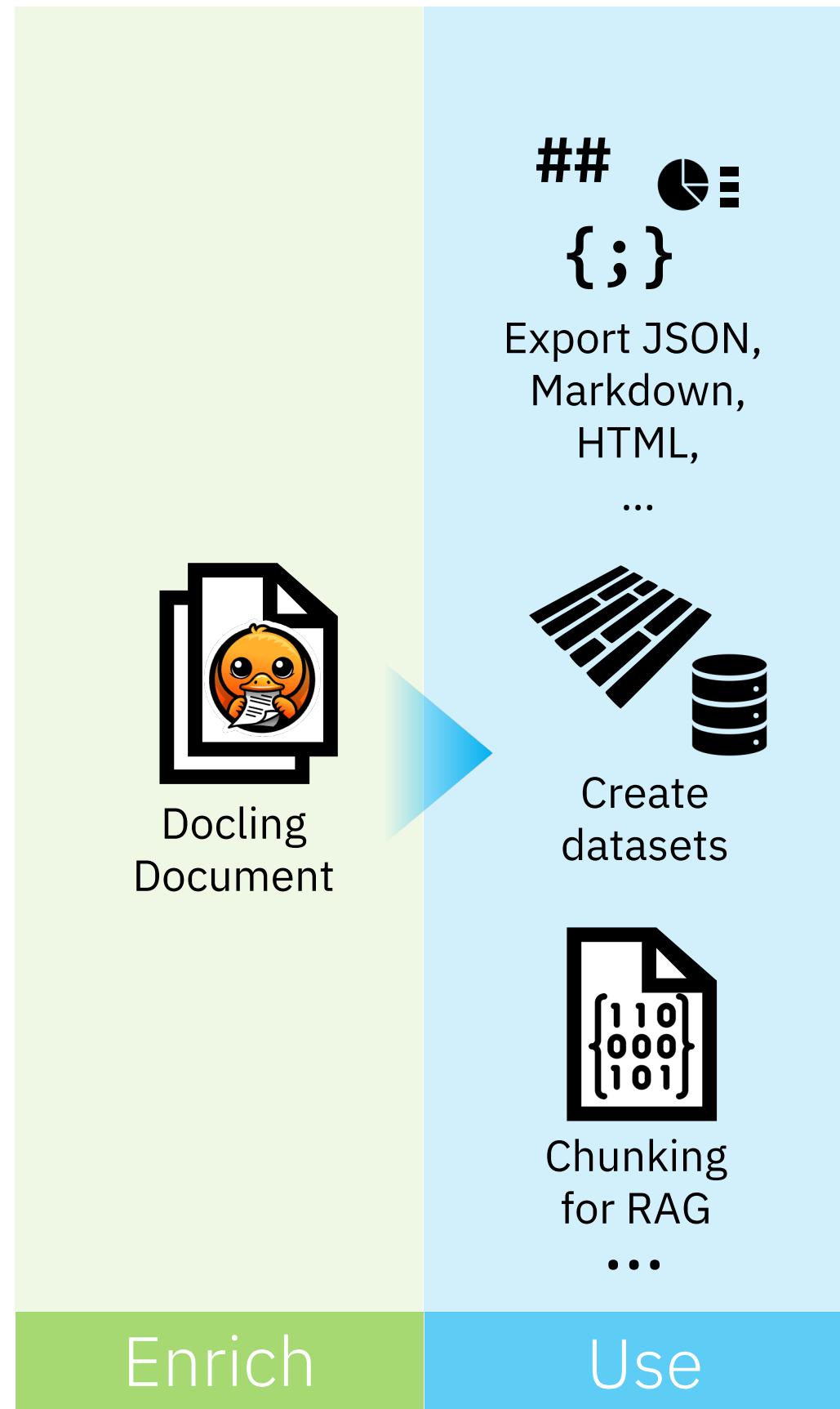
Live Demo

- Let's start from the minimal example
- How do we work with the output?
 - Look at tables
 - Look at pictures
- How do tune the options?
- How to navigate the docs!

<https://docling-project.github.io/docling/>



DoclingDocument data model



```
1 version: 1.0.0
2 schema_name: DoclingDocument
3
4 ▼ body: # The root node of the document content (excluding headers, footers, ...)
5   children:
6     - $ref: '#/texts/0' # text: Summer activities
7     - $ref: '#/texts/1' # title: Swimming in the lake
8   label: unspecified
9   name: _root_
10  self_ref: '#/body'
11
12  texts: # The plain text items in this document.
13  ▼ - self_ref: '#/texts/0'
14    orig: Summer activities
15    text: Summer activities
16    label: paragraph # The semantics of a text element are represented by the label
17    children: []
18    parent:
19      | $ref: '#/body'
20    prov: []
21  ▼ - self_ref: '#/texts/1'
22    orig: Swimming in the lake
23    text: Swimming in the lake
24    label: title
25    children: # Any item can have children to reflect section hierarchy
26      - $ref: '#/texts/2' # text: Duck
27      - $ref: '#/texts/3' # text: (empty text)
28      - $ref: '#/texts/4' # text: Figure 1: This is a cute duckling
29      - $ref: '#/texts/5' # section_header: Let's swim!
30    # ...
31    parent:
32      | $ref: '#/body'
33    prov: []
34  # ...
```

Summer activities

Swimming in the lake

Duck

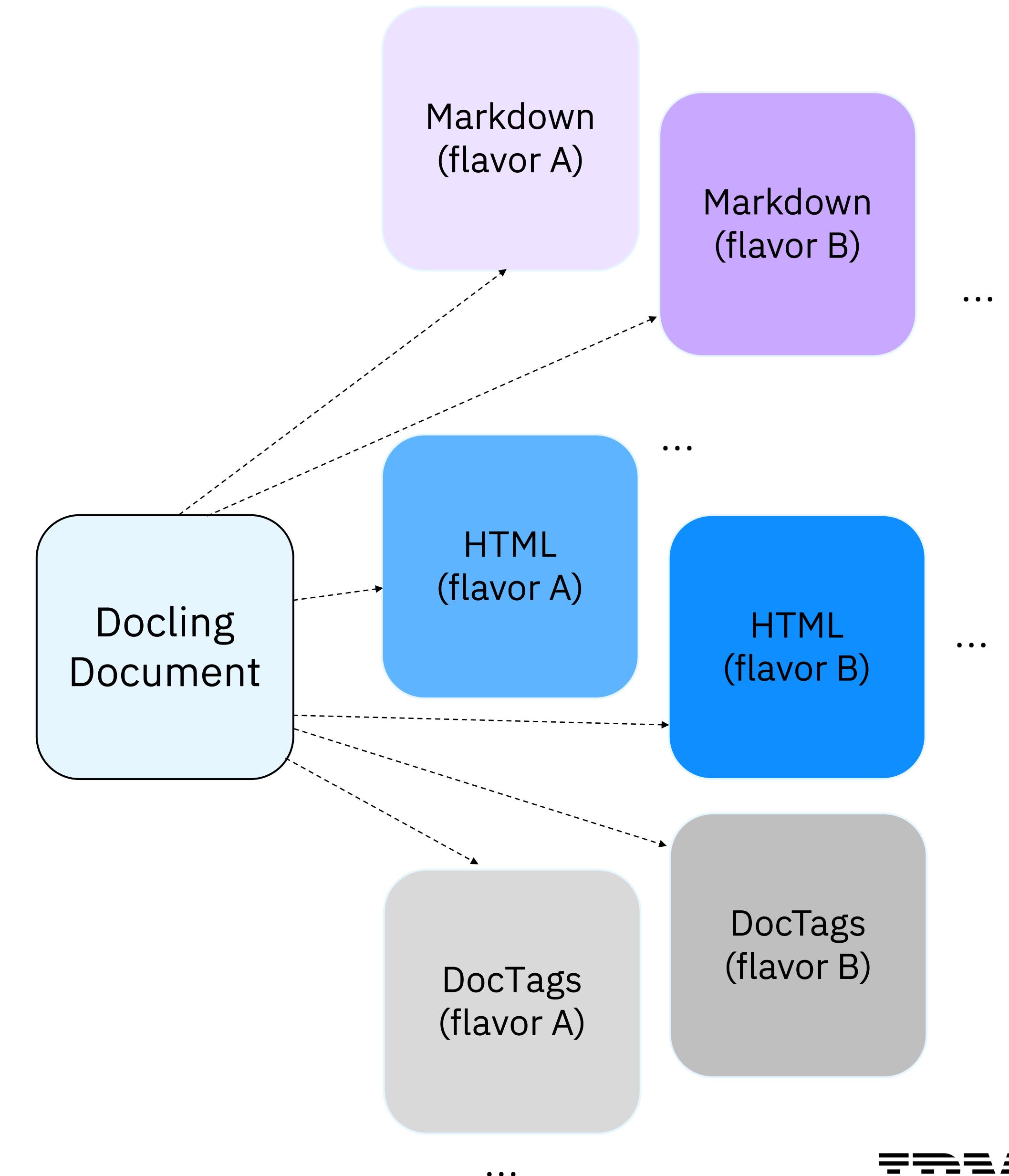


Figure 1: This is a cute duckling

Let's swim!

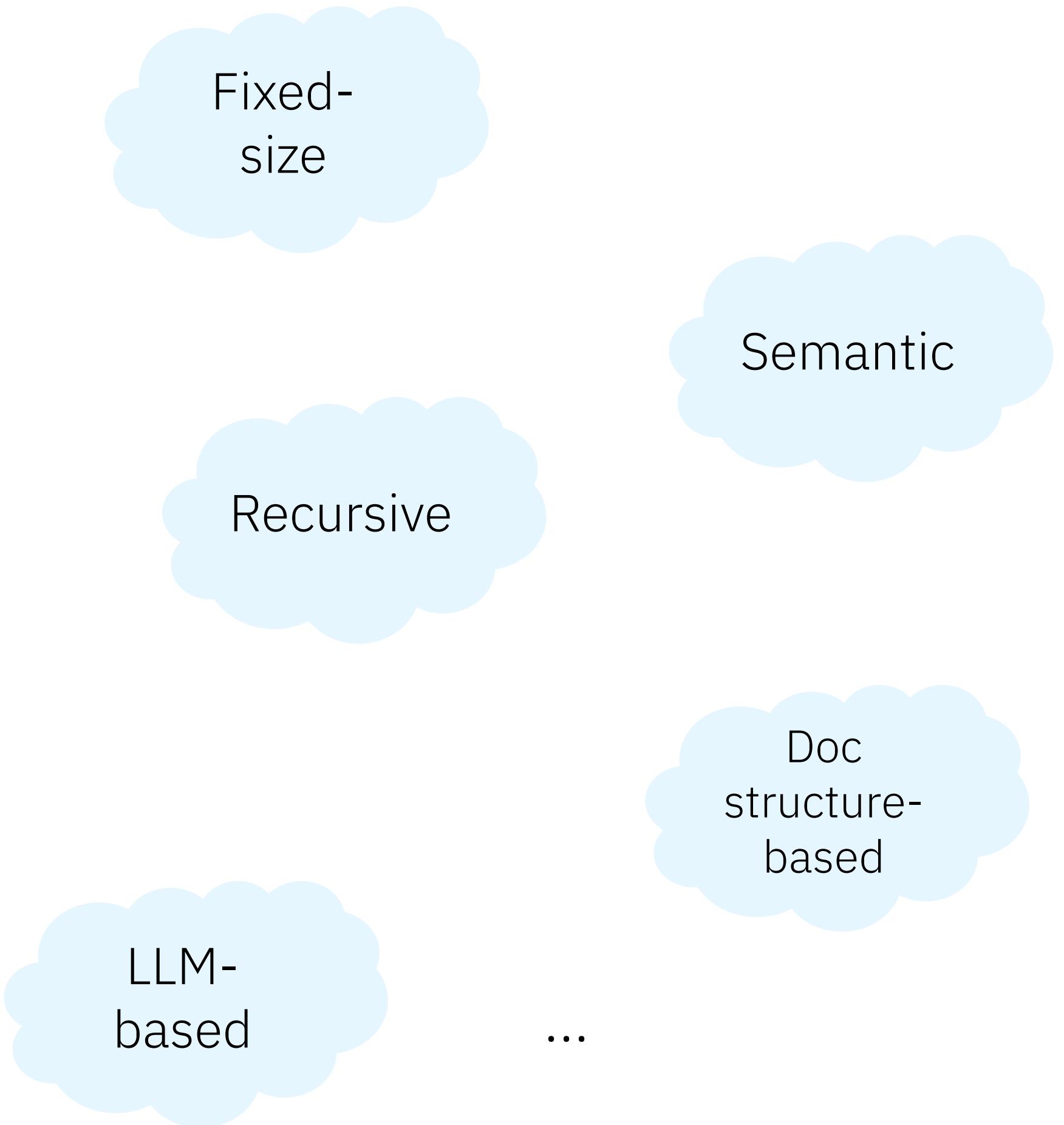
Serialization

- Derive any type of text export from a `DoclingDocument`
- Built-in serializers for Markdown, HTML, and more
- Easily customize – or plug in user-provided serializers
- Multimodal capabilities tailored to user needs
 - E.g. picture serialization
- Docs: <https://docling-project.github.io/docling/concepts/serialization/>
- Hands-on examples: <https://docling-project.github.io/docling/examples/serialization/>



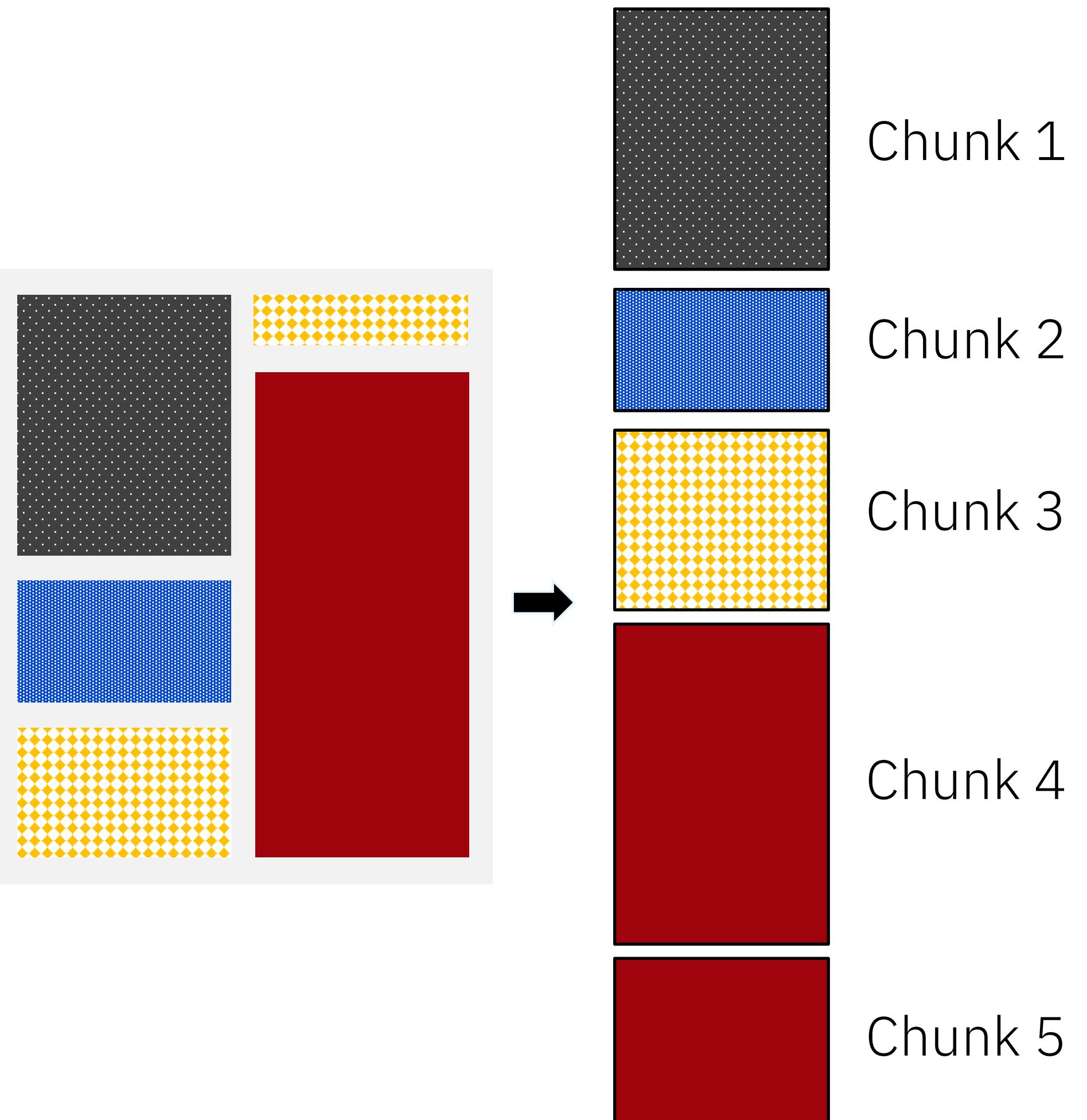
Chunking

- Splitting documents in meaningful parts
- Significant part of RAG as directly affects retrieval
- Various strategies one can consider



Chunking

- Docling supports different strategies via built-in chunkers & extensible design that allows plugging in user-defined ones
 - All native on rich **DoclingDocument** level, not just text
- Hierarchical: based on the doc layout extracted by Docling
- Hybrid: combines doc layout and tokenization awareness
- Any built-in or user-defined serializer can be used
- Docs: <https://docling-project.github.io/docling/concepts/chunking/>
- Examples: https://docling-project.github.io/docling/examples/hybrid_chunking/



Hybrid chunking captures both document layout and token limitations

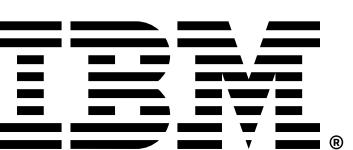


Hands On

Now it is your turn

1. Install Docling
2. Convert some of your documents
3. Inspect some of the tables
4. Inspect the figures
5. Enable picture classification and description
6. Chunk your document
7. Customize the output

<https://docling-project.github.io/docling/>



Before lunch

Wrapping up

1. ✓ Install Docling
2. ✓ Convert some of your documents
3. ✓ Inspect some of the tables
4. ✓ Inspect the figures
5. ✓ Enable picture classification and description
6. ✓ Chunk your document
7. ✓ Customize the output



<https://docling-project.github.io/docling/>

Docling Workshop

- Learn about Docling
- Install and use Docling
- Understand the configuration options and customizations
- Run Docling with Llama Stack
- Run Docling and Llama Stack on OpenShift



9:00-12:00	Intro to Docling <i>Presentation:</i> Introduction to Docling <i>Hands On:</i> Converting documents with Docling including advanced features
12:00-13:00	Lunch and discussions
13:00-16:00	Docling + Llama Stack + Agents <i>Hands On:</i> Build your DocumentAI use-case with Llama Stack Use Docling within your Llama Stack agents



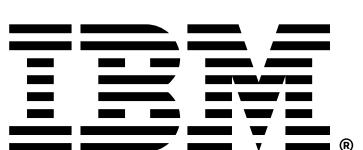
IBM

Llama Stack RAG with Docling

Enhance the Llama Stack RAG with:

- Amazing document conversion (it does a difference!)
- RAG on complex document components
- Multi-modal RAG
- Visual grounding

Live demo



Hands On

Now it is your turn

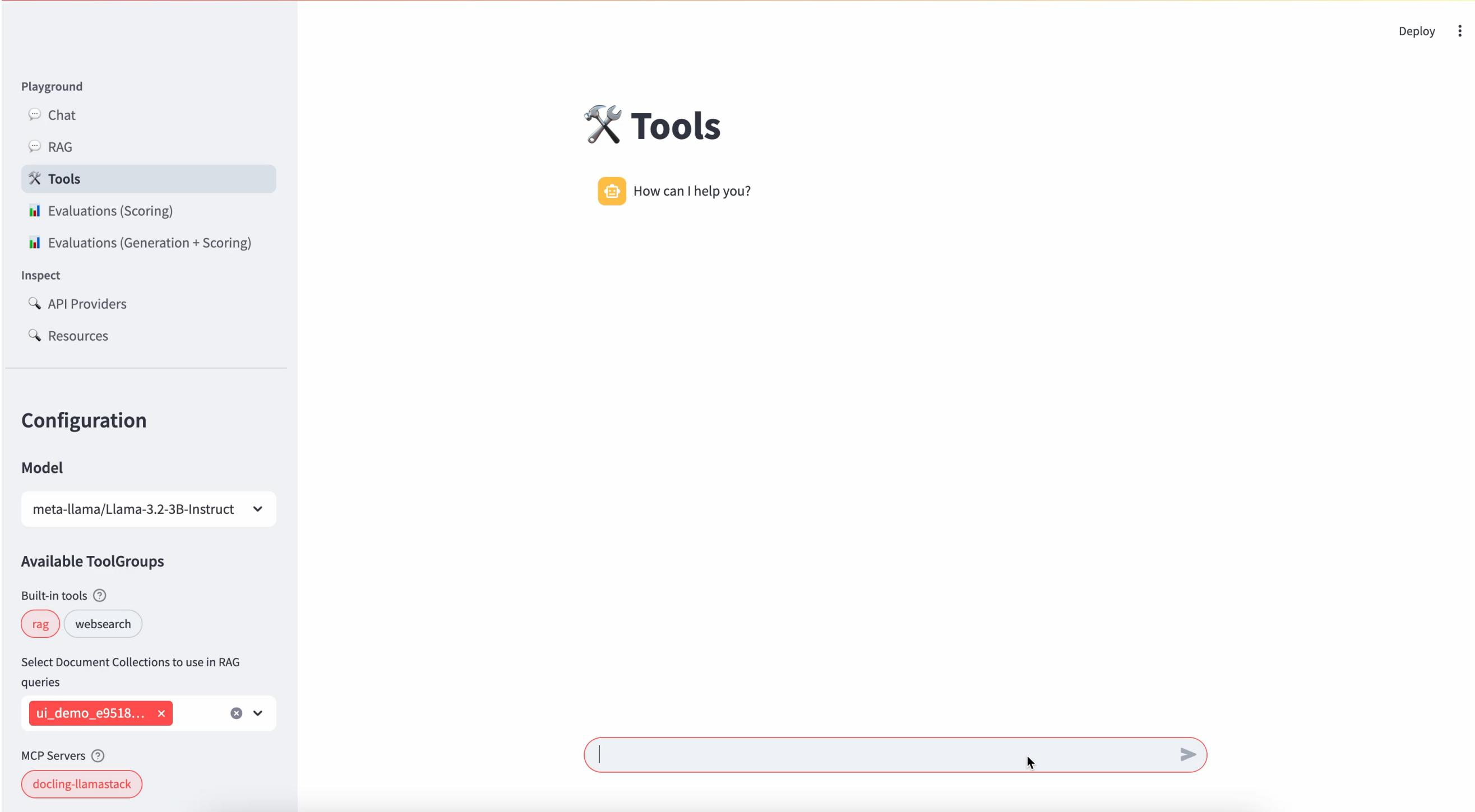
1. Try enhancing the Llama Stack RAG demo with Docling
2. Docling conversion
3. Multi-modal
4. Visual grounding

<https://docling-project.github.io/docling/>



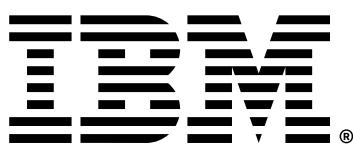
Docling with Agents

- Augment Llama Stack with a Docling tool
- For example, allow
 - Convert and ingest a document with Docling conversion and chunking
 - Manipulate the document (save in different formats, add/edit/delete parts, etc)



The screenshot shows the Docling configuration interface. On the left, a sidebar menu includes 'Playground' (Chat, RAG, Tools), 'Inspect' (API Providers, Resources), and 'Configuration'. Under Configuration, the 'Model' dropdown is set to 'meta-llama/Llama-3.2-3B-Instruct'. The 'Available ToolGroups' section shows 'Built-in tools' with 'rag' and 'websearch' selected. Below that, 'Select Document Collections to use in RAG queries' lists 'ui_demo_e9518...' and 'docling-llamastack'. A large input field at the bottom is empty.

Live demo



Hands On

Now it is your turn

1. Build an agent and connect it to Llama Stack
2. Deploy your MCP tool in OpenShift
3. Run all-in-one with the Llama Stack Playground UI

<https://docling-project.github.io/docling/>



Docling



Try it out!

🔗 GitHub: github.com/docling-project

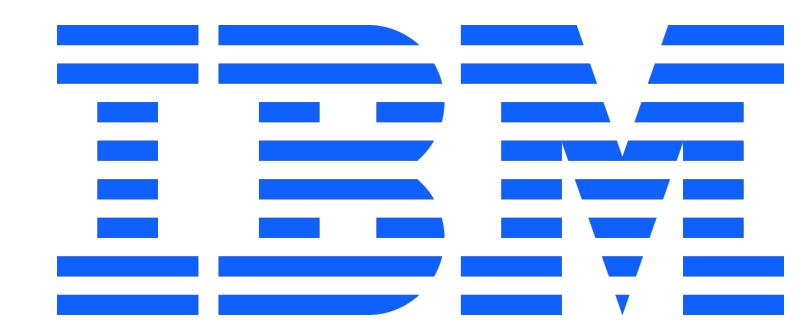
📦 Install: pip install docling

📚 Tech paper: arxiv.org/abs/2501.17887

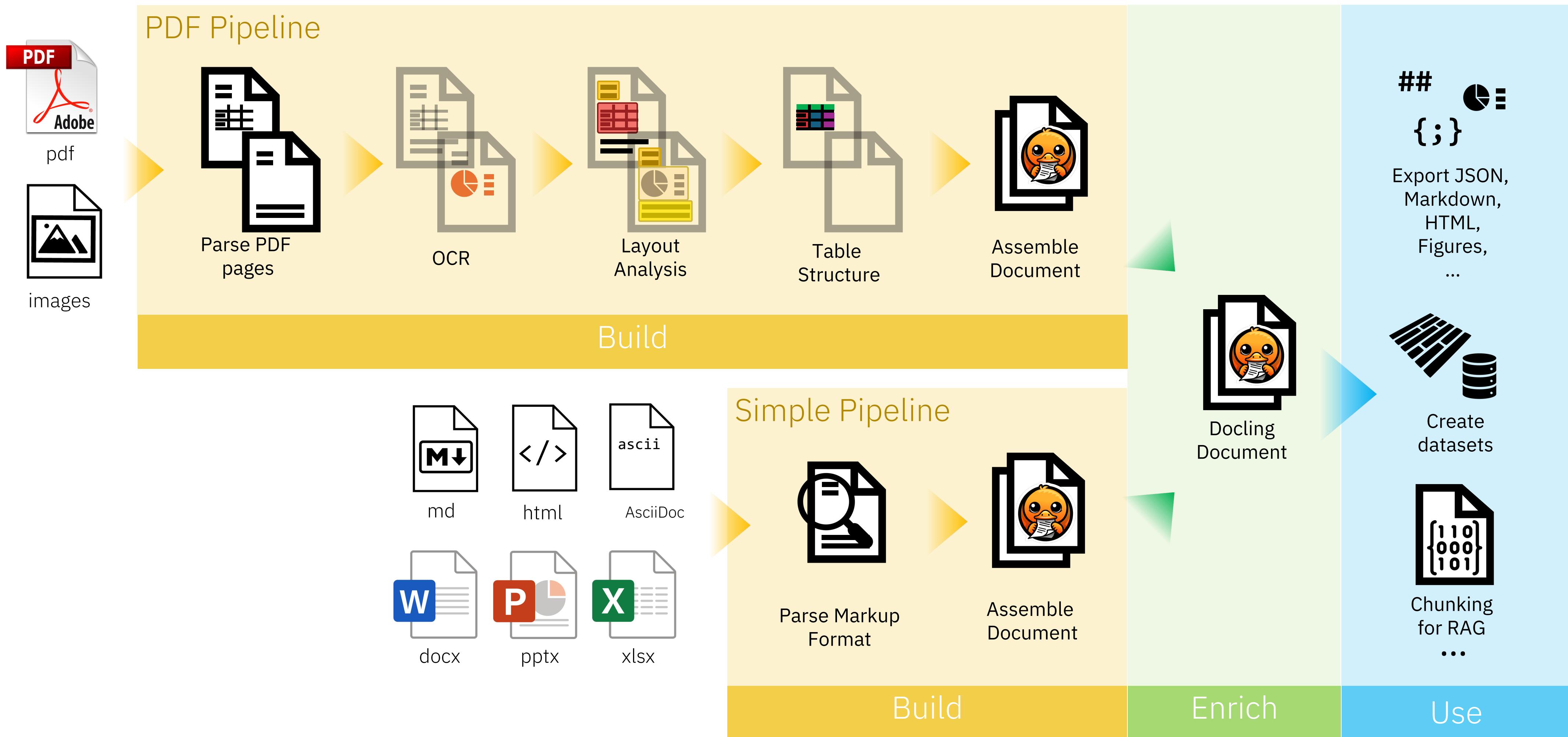


💻 Use it, ⭐ star it,
🚀 spread the word about Docling!

IBM



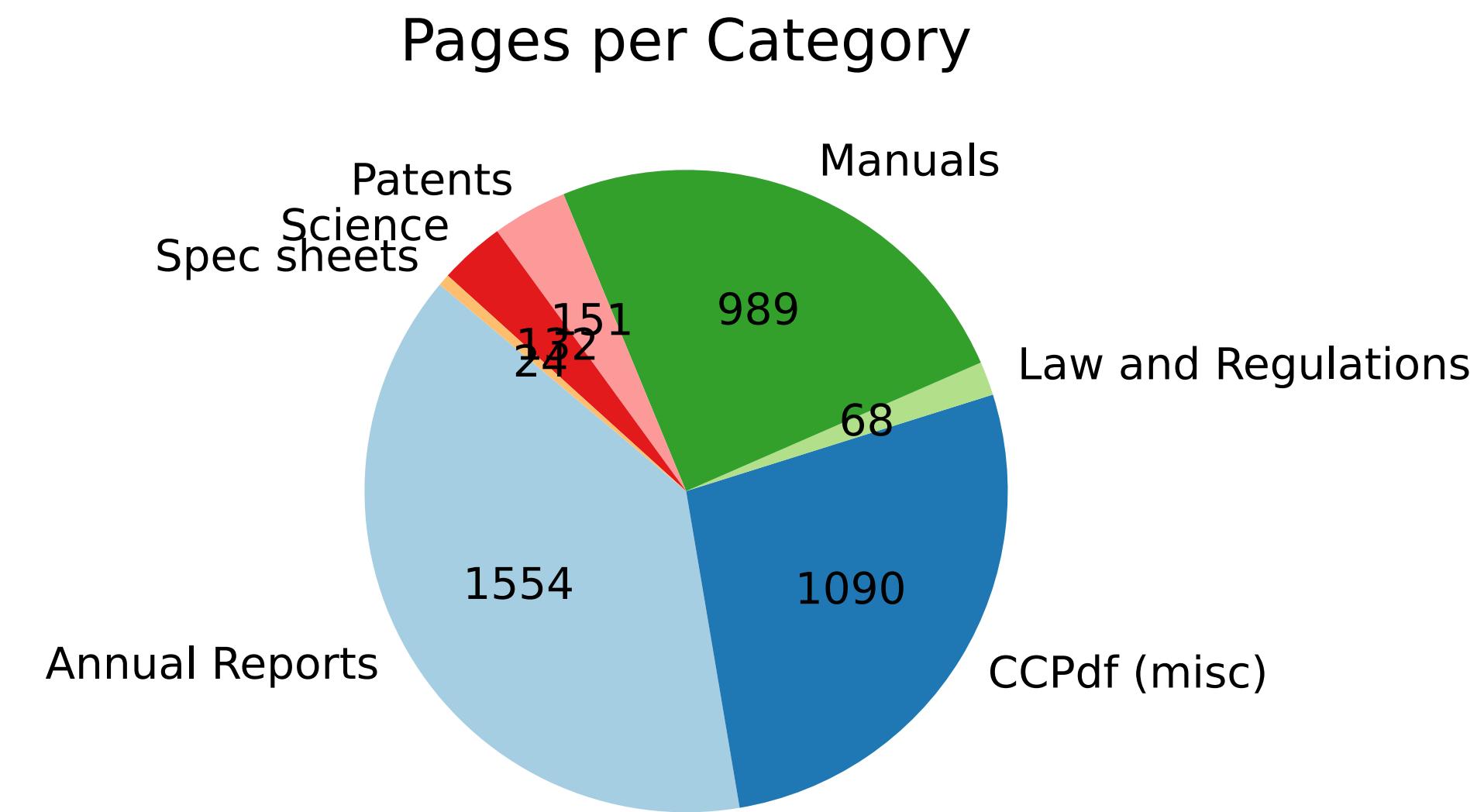
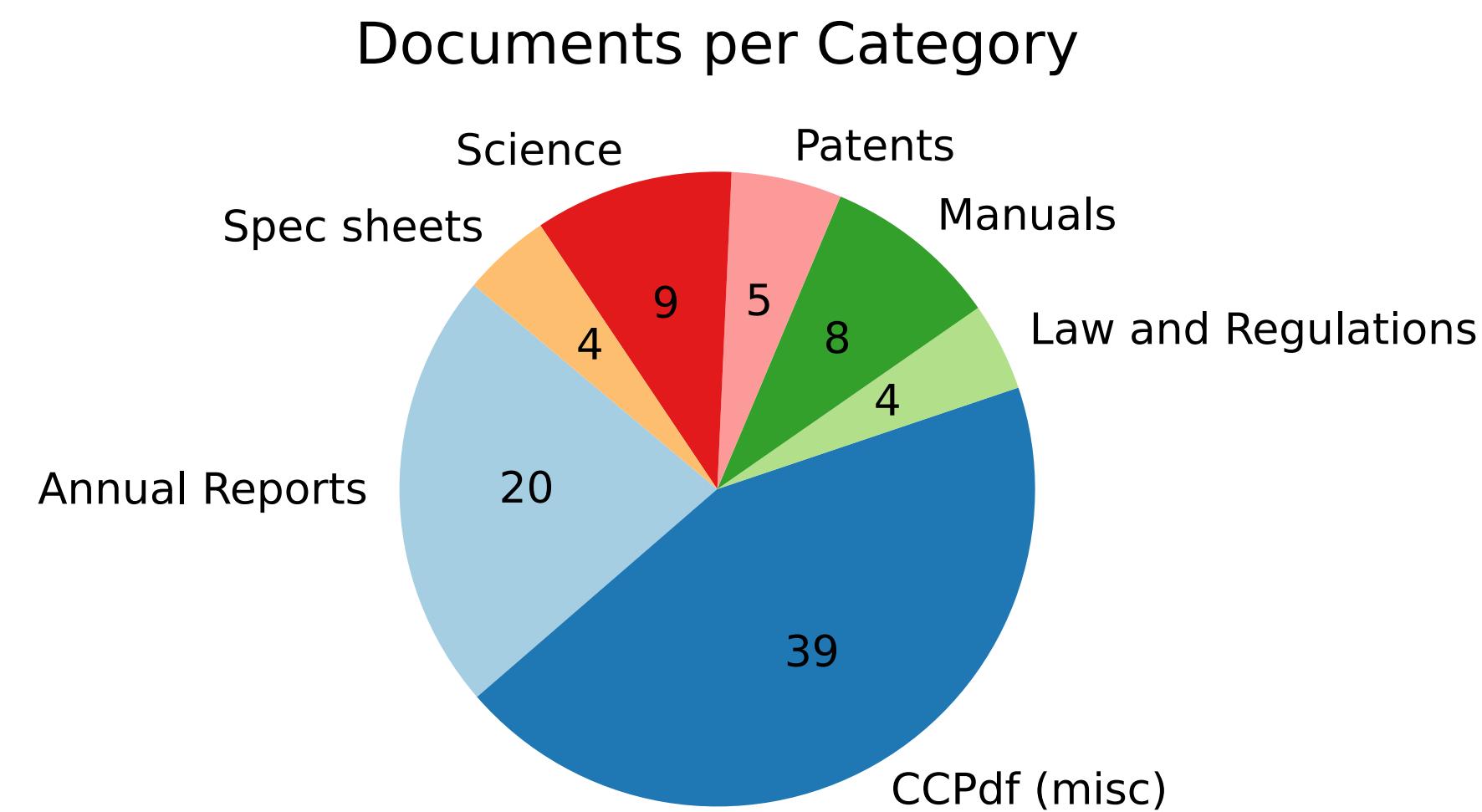
Design and Architecture



PDF Pipeline analysis

Benchmark dataset

- 4008 pages
- 56 246 text items
- 1842 tables
- 4676 pictures

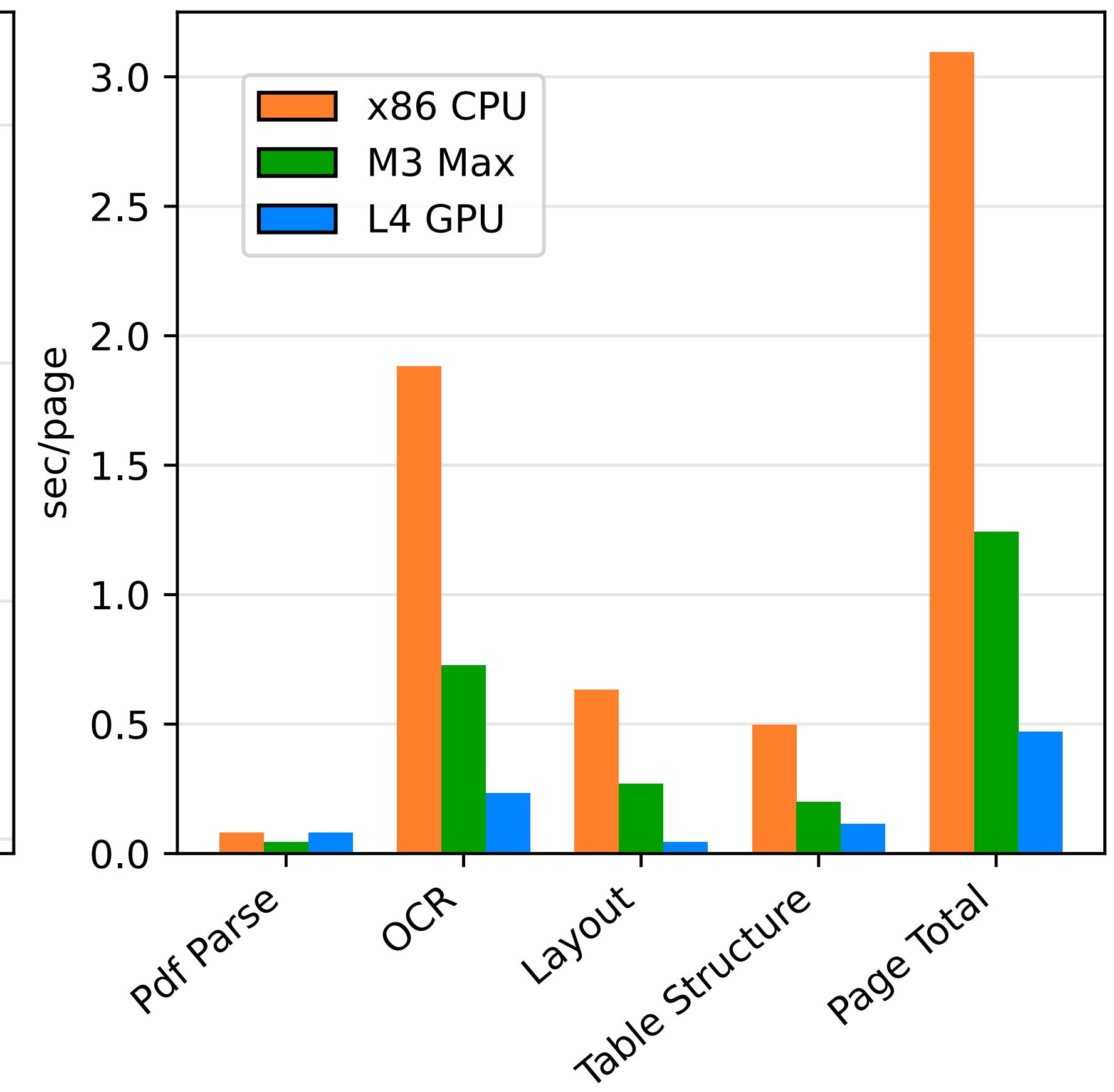
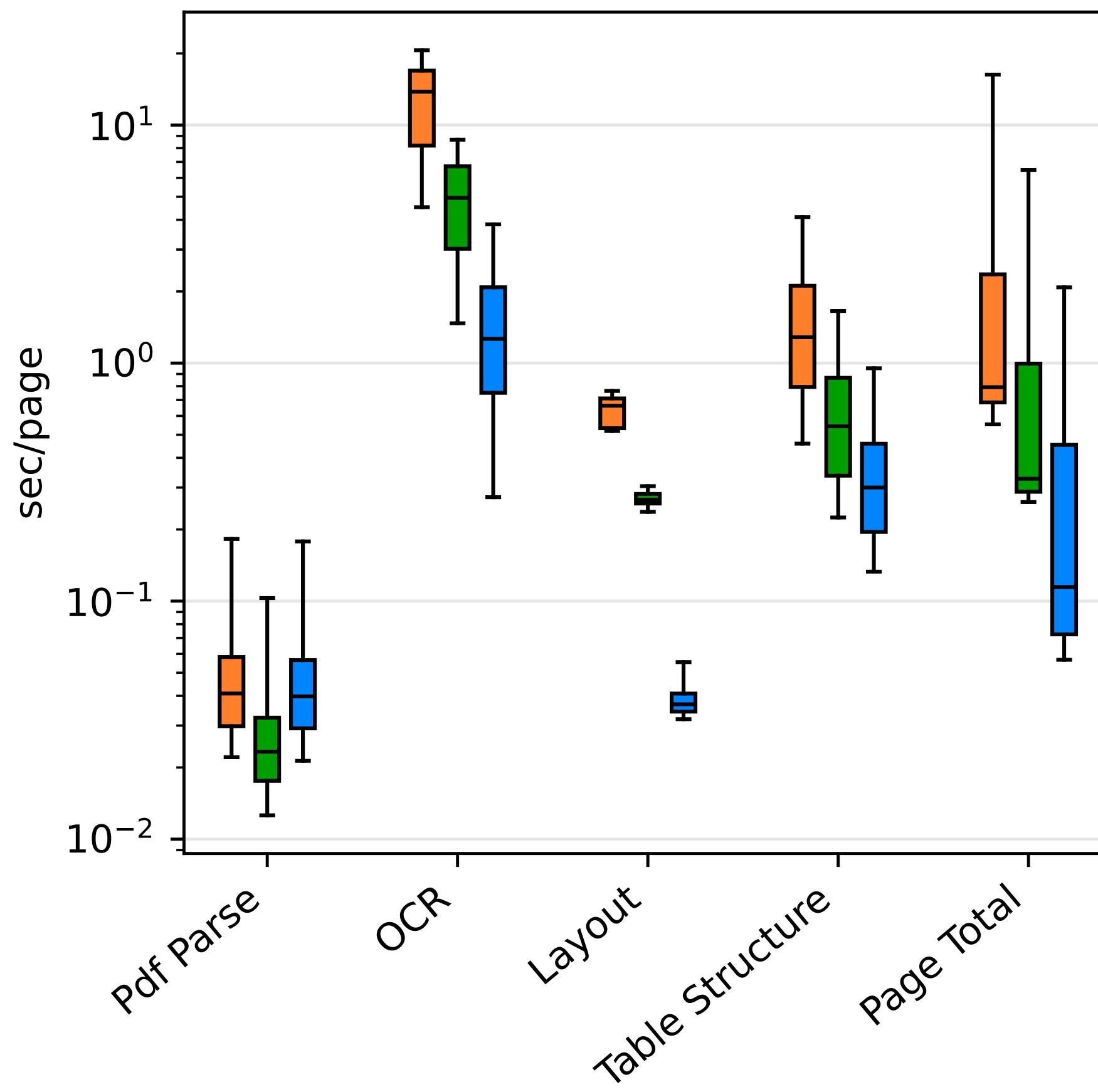


Categories	
Annual Reports	Patents
CCPpdf (misc)	Science
Law and Regulations	Spec sheets
Manuals	

PDF Pipeline analysis

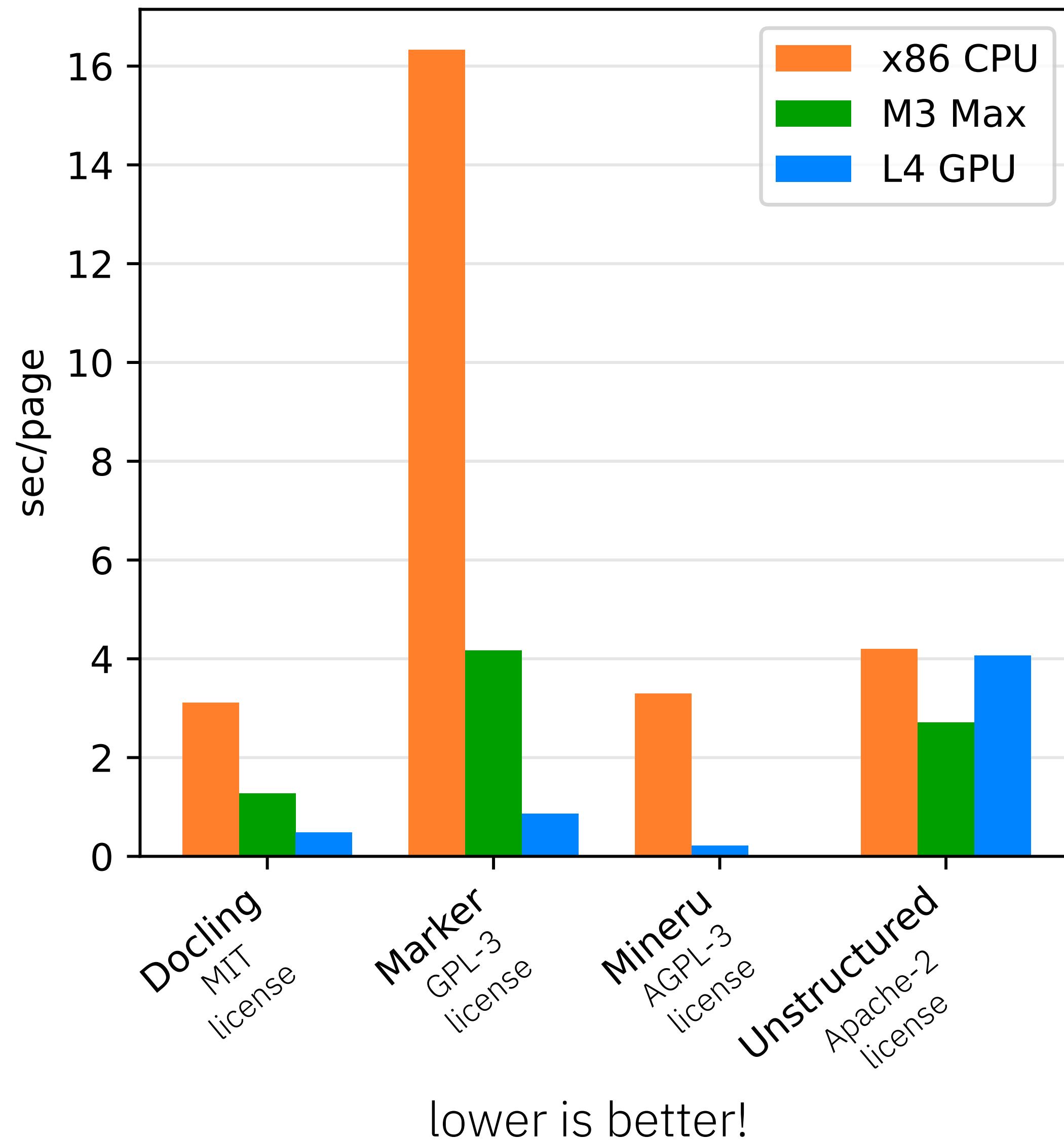
Benchmark dataset

- 4008 pages
- 56 246 text items
- 1842 tables
- 4676 pictures



Performance

- Docling is fastest among all open-source assets on CPU!
- Processing an average page takes...
1.3 sec on MacBook M3 
- 0.5 sec on an NVIDIA L4 GPU 
- 200k pages in a day on a standard GPU!
- ~2.5x speedup without OCR

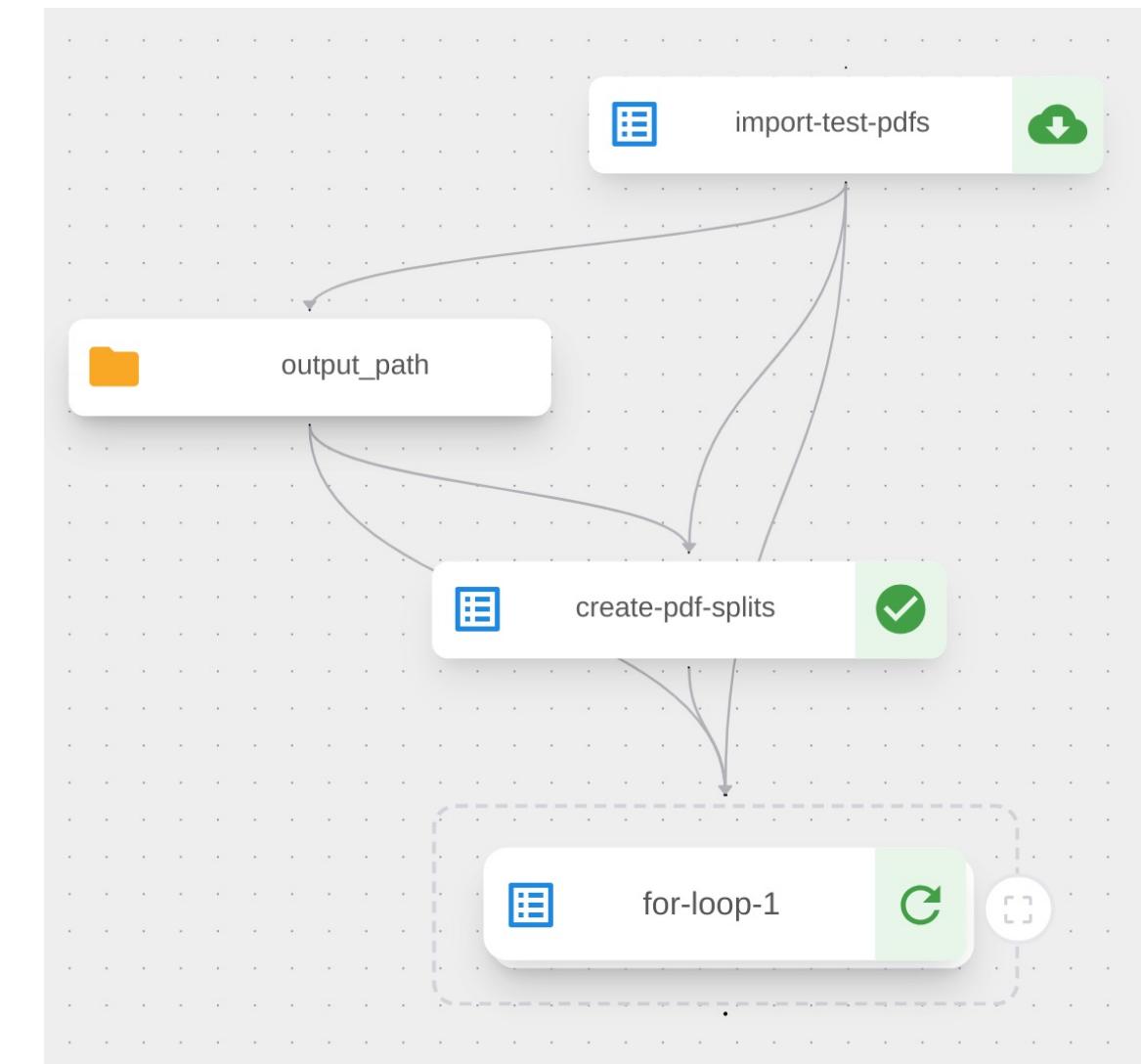
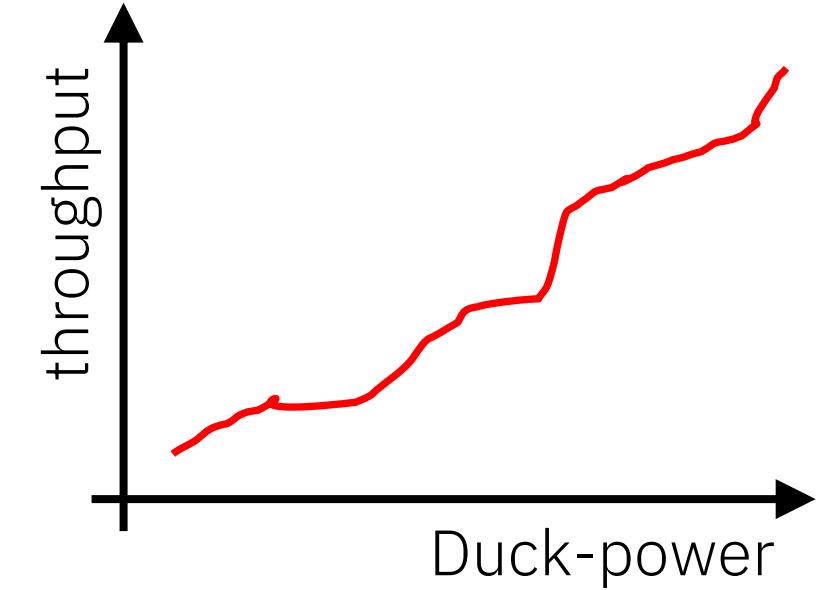
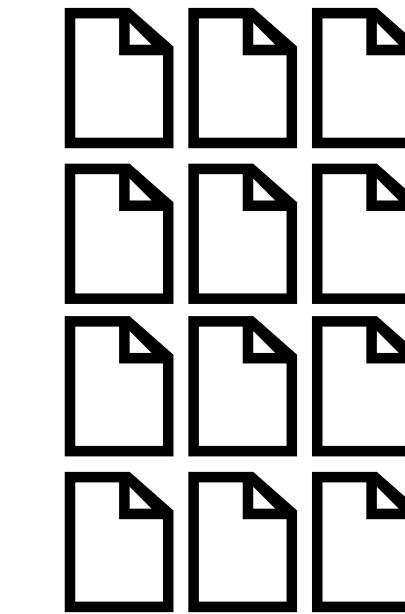
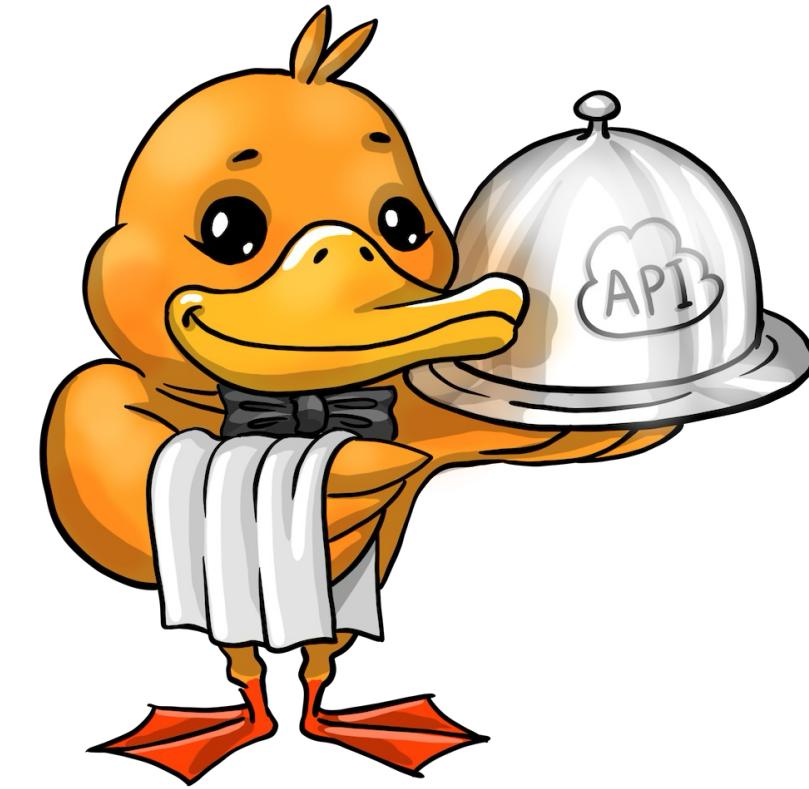


What is next?

Docling large scale jobs

Scale, scale, scale!

- Scale out to thousands of cores
- Support for multiple gpus
- Orchestrate conversion and enrichment steps (e.g. industry verticals)
- Telemetry and seamless monitoring of large jobs
- Built on OpenShift AI tools



Kubeflow

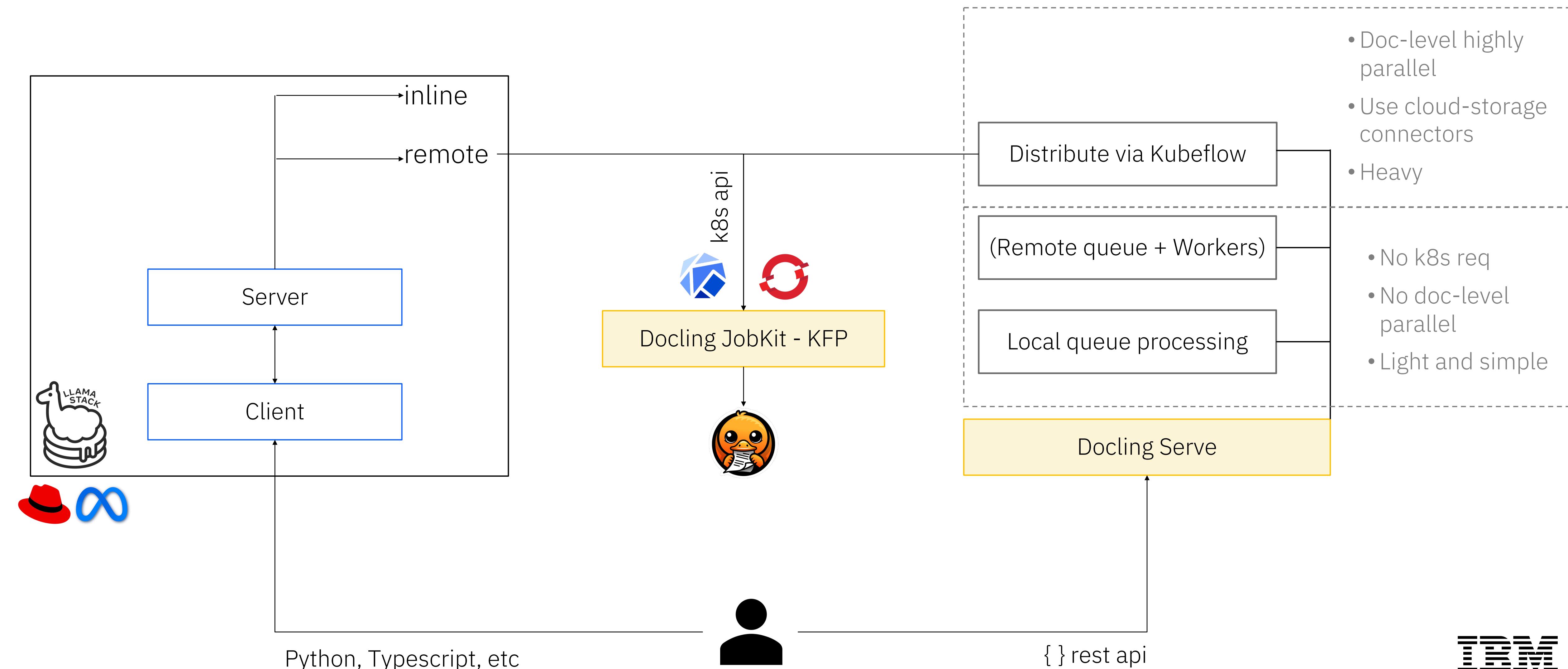
<https://github.com/docling-project/docling-jobkit>

IBM.

Llama Stack and Docling

Docling large scale jobs and APIs

A deep collaboration with  Red Hat



Docling-MCP for Agents

Make Docling agentic

[README](#) [MIT license](#) [Security](#)

Docling MCP: making docling agentic

pypi v0.3.0 python 3 code style black imports isort Pydantic v2 pre-commit enabled license MIT downloads/month 2k

A document processing service using the Docling-MCP library and MCP (Message Control Protocol) for tool integration.

Note
This is an unstable draft implementation which will quickly evolve.

Overview

Docling MCP is a service that provides tools for document conversion, processing and generation. It uses the Docling library to convert PDF documents into structured formats and provides a caching mechanism to improve performance. The service exposes functionality through a set of tools that can be called by client applications.

Features

- conversion tools:
 - PDF document conversion to structured JSON format (DoclingDocument)
- generation tools:
 - Document generation in DoclingDocument, which can be exported to multiple formats
- Local document caching for improved performance
- Support for local files and URLs as document sources
- Memory management for handling large documents
- Logging system for debugging and monitoring

Claude

[New chat](#)

Projects

Chats

Recents

Polymer and Water Vapor Trans...

Parsing HTML Tables with Beautif...

Untitled

Untitled

Polymer and Water Vapor Trans... [...](#)

Polymer and Water Vapor Trans...

Add Title to Docling Document

Preventing Ruff from Removing S...

Creating a New Docling Document

Tools for Collaboration and Analysis

Investigating Tokenizer Impact on ...

Fixing "Too many positional argum..."

Investigating Tokenizer Impact on ...

Untitled

Impact of Tokenizers on LLM Quali...

Impact of Tokenizers on LLM Quali...

Impact of Tokenizers on LLM Quali...

Untitled

Impact of Tokenizers on LLM Quali...

Peter Professional plan

Polymers and Water Vapor Transmission in Food Packaging [Share](#)

P I want you to write a Docling document. To do this, you will create a document first by invoking `create_new_docling_document`. Next you can add a title (by invoking `add_title_to_docling_document`) and then iteratively add new section-headings and paragraphs. If you want to insert lists (or nested lists), you will first open a list (by invoking `open_list_in_docling_document`), next add the list_items (by invoking `add_listitem_to_list_in_docling_document`). After adding list-items, you must close the list (by invoking `close_list_in_docling_document`). Nested lists can be created in the same way, by opening and closing additional lists.

During the writing process, you can check what has been written already by calling the `export_docling_document_to_markdown` tool, which will return the currently written document. At the end of the writing, you must save the document and return me the filepath of the saved document.

The document should investigate the polymers and their relation to water vapor transmission rate in food packaging. Please make a document with a title and add only a single table with polymers versus thickness.

I'll create a Docling document about polymers and water vapor transmission rates in food packaging, including a table of polymers versus thickness. Let me work on this step by step.

[View result from `create_new_docling_document` from `docling` \(local\) >](#)

Now I'll add a title to the document:

[View result from `add_title_to_docling_document` from `docling` \(local\) >](#)

Next, I'll add a section heading and introduce the topic:

[View result from `add_section_heading_to_docling_document` from `docling` \(local\) >](#)

[View result from `add_paragraph_to_docling_document` from `docling` \(local\) >](#)

Now I'll add a section for the polymer and thickness data table:

[View result from `add_section_heading_to_docling_document` from `docling` \(local\) >](#)

[View result from `add_paragraph_to_docling_document` from `docling` \(local\) >](#)

Now I'll add the table of polymers versus thickness:

[View result from `add_table_in_html_format_to_docling_document` from `docling` \(local\) >](#)

Reply to Claude...

+ ⌂ ⌂ 15 ⌂

Claude 3.7 Sonnet ▲

