Unstructured libaray has 3 types, serverless, api, and local, api and serverless are more powerful.

So much testing to do, use api, use html parser, use detection model…etc

**First let see things for html file.**

Very good results, Able to detect table, metadata, eliminate all the html metadata and get the metadata that is necessary and provide things in nice Json format (tested with labtestonline scrapped website) but not 100 percent, didn’t get image tags but able to get all the textual data without much effort.

Next parsing pptx:  
needs data for this  
  
  
my thought. To include metadata on rag, a user can first send a query and then instead of similarity search we give it to llm, and ask if there is metadata like date to be extracted and then we do similarty search on the basis of query and metadata filtering (\*\*)

L3: there is a metadata filtering for rag which is interesting: as unstructured gives a json with element ids. These ids are mapped to embedded objects as parent ids so you can track e.g chapter : Malnutrition would be a title and its elm id would be a parent id inside the sub headings of this chapter. .. (in any case get all the chapter ids, match it with elem id and you get all the chapters..then things proceed further…)

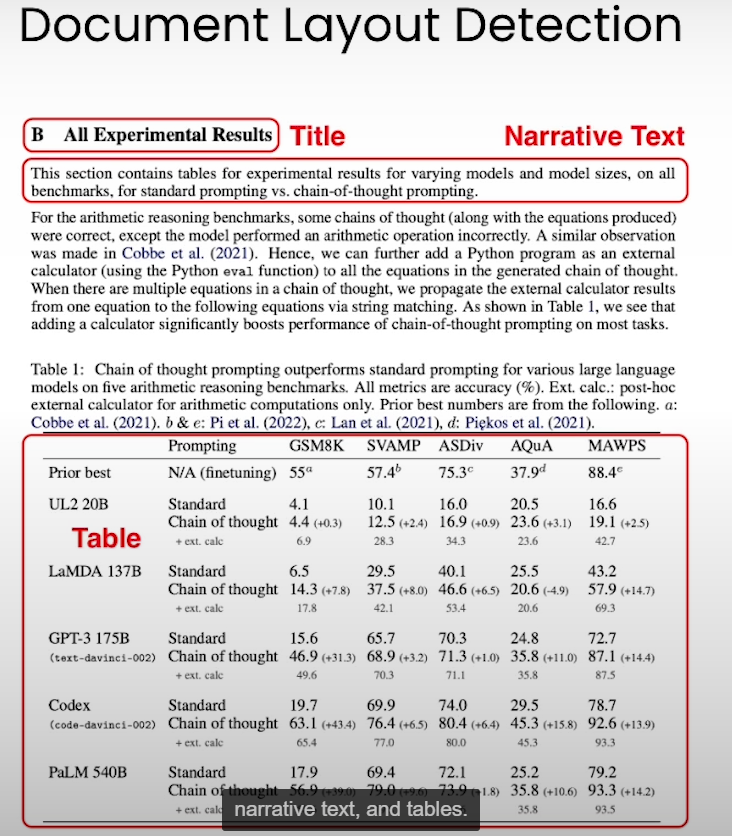
(little issue of how they set text to vector db, as there could be small texts and each time….)

**New chunking could be using unstructured like create a chunk each titile and its related elements….  
also apply condition that if chunk is small then add next titile as well and its releated elements texts.  
in summary creates chunks using elemnets.**

**Preprocessing documents: rule based where there is some kind of structured info  
but for some documents like pdf, you need visial based parsing system.**

**Document Image analysis**

**Document Layout Detection-> uses object detection model-> bounding boxes gets labeled-> and elements gets extracted using ocr…e.g yolox model**

****

**Vision Transformer detection-> takes an image of document as an input and produce text.**

**Do processing in a single step. Convert image into json string. E.g DONUT model**

**Worked with docx file for survey:  
able to extract table on multiple pages.**

The library supports a wide range of document types, including:

* Text files (.txt, .md)
* Word documents (.docx, .doc, .odt)
* Presentations (.pptx, .ppt)
* Spreadsheets (.xlsx, .csv, .tsv)
* Emails (.eml, .msg)
* Rich Text Format (.rtf)
* E-books (.epub)
* Web pages (.html, .xml)
* Images (.png, .jpg, .heic)

**Unstructured Library**

* Has 3 types: serverless, API, and local
* API and serverless are more powerful, but images weren’t get drawn from it.

**Testing and HTML Parsing**

* Used API, HTML parser, and detection model for testing
* Successfully detected table, metadata, and textual data from HTML file
* Didn't get image tags, but results were good

**Parsing docx**

* Able to test on that big table.

**Metadata Extraction and RAG**

* Proposed method to include metadata in RAG using LLM
* Metadata filtering using element IDs and parent IDs (I can make it better.)

**Chunking and Preprocessing**

* New chunking method using unstructured library
* Create chunks using elements and apply conditions for small chunks
* Preprocessing documents using rule-based and visual-based parsing systems

**Document Image Analysis theory**

* Document layout detection using object detection model
* Vision transformer detection to produce text from document image

**Synthetic Data and Evaluation**

* Create synthetic data to evaluate RAG pipeline (not tested)
* Shared Colab link for reference

Create synthetic data and use to evaluate Rag pipiline

* <https://colab.research.google.com/drive/1VvOauC46xXeZrhh8nlTyv77yvoroMQjr?usp=sharing>
* https://docs.ragas.io/en/latest/getstarted/evaluation.html

Kai notes:

Docx file is a format of xml, that’s why they are easy to parse….