Question 1: Vector Database Selection:

♦ Choose either FAISS or ChromaDB as the vector database.

♦ Justify your choice by comparing the features, scalability, and performance characteristics of both databases in the context of this project.

Answer 1:

Based on the project requirements and comparison criteria , FAISS is recommended as

- It provides Superior handling of large-scale datasets hence **scalable** .
- It's optimized for high speed and efficient similarity search hence has **great performance**.
- It also have extensive options for tuning and optimization

Question 2 : Layout Analysis Model:

♦ Identify and implement a layout analysis model that can effectively segment documents into logical chunks (e.g., paragraphs, sections, tables).

♦ Explain the chosen model's capabilities and limitations, and how it will be used to determine chunk boundaries.

Answer 2 :

There are many models but I have worked on 3 models

- **Detectron2LayoutModel with Faster R-CNN R-50 FPN on PubLayNet**
  - Uses the Detectron2 framework with Faster R-CNN R-50 FPN for document layout analysis, leveraging the PubLayNet dataset for training
  - Combines ResNet-50 for deep feature extraction and Feature Pyramid Network for multi-scale object detection.
  - Trained on PubLayNet, which enhances the model's ability to detect and classify various document layout components such as text blocks, tables, figures, and lists.
  - Effective in large-scale document analysis tasks, such as digitizing paper documents, automated form processing, and enhancing the accessibility of scanned documents. Suitable for robust and efficient document processing tasks due to the powerful combination of Detectron2 and Faster R-CNN architecture.
- **LayoutLLMV3(and it's older version)**
  - LayoutLM and its advanced version, LayoutLMv3, are designed to integrate visual and textual information for document understanding tasks. LayoutLM is effective in processing scanned documents, PDFs, and forms, leveraging its capability to understand both text and layout for applications like automated document processing and information retrieval systems. LayoutLMv3 builds upon these capabilities with enhanced attention mechanisms and larger pre-training datasets, resulting in superior performance in form understanding, receipt recognition, and document classification. Both versions excel in scenarios where the

spatial arrangement of text is crucial for accurate information extraction, making them ideal for comprehensive document understanding tasks.

- **LiLT model**
    - The LiLT (Lightweight Layout Transformer) model is tailored for computational efficiency and scalability in document understanding tasks. It employs a transformer-based architecture optimized for handling the spatial structure of documents while minimizing computational overhead. Despite its lightweight design, LiLT delivers competitive results in document layout analysis and text recognition, making it suitable for environments with limited computational resources, such as mobile devices or edge computing. LiLT is particularly valuable for efficient document processing in resource-constrained scenarios, balancing performance with resource efficiency.

- **LiLT model and LayoutLLMV3** are also good but requires to hand mark the documents using Label Studio and then have to make it suitable for the consumption of each model. Have Tried a bit for **LayoutLLMV3** .

Question 3 : Vector Embeddings:

♦ Generate vector embeddings for each document chunk using a suitable embedding model (e.g., Sentence Transformers, OpenAI Embeddings).

♦ Briefly describe the embedding model and its parameters.

Answer 3

- Have used paraphrase-MiniLM-L6-v2 as it was free to use
- Paraphrase-MiniLM-L6-v2 is a transformer-based model specifically designed for generating high-quality sentence embeddings. It is part of the Sentence-BERT family and utilizes a smaller, more efficient architecture compared to its predecessors, making it ideal for tasks requiring fast and accurate text embeddings.

Question 4 : Metadata:

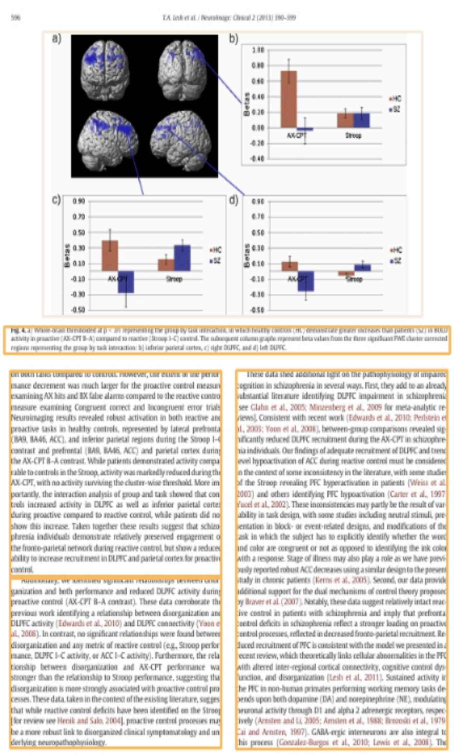
♦ Define the metadata fields that will be associated with each document chunk (e.g., source document ID, chunk number, page number, section title).

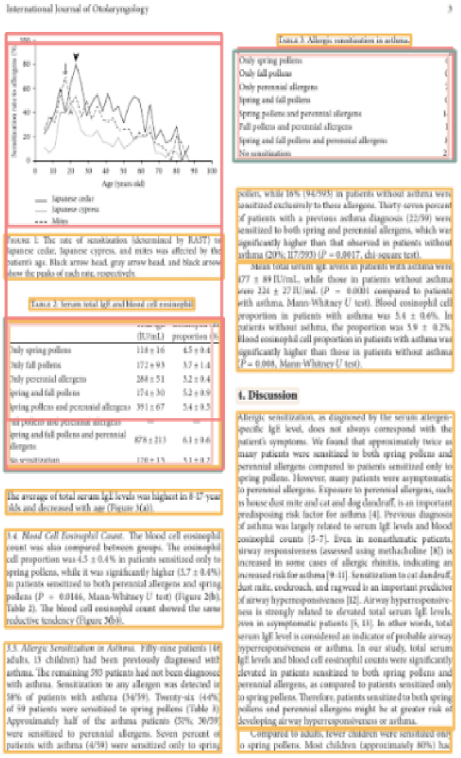
- **Document_id :** Is the name of the image under consideration , As if we consider it .It's unique for a page . Though Wanted a number but the last bits are repeating in the name .
- **Chunk_number :** It's a subpart of a text paragraph extracted from the image. It's made when the token size is exceeded
- **Text_chunk :** It's the chunk of text in a box recognised via the model
- **Parahnum:** It' s the number given to a Text_chunk

Documentation 1

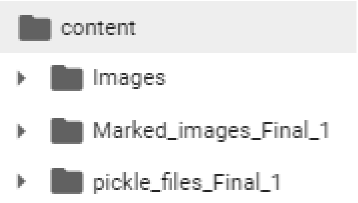
- It contains 2 notebooks

- In **Notebook 1A** we have labeled the un marked images and have generated a pickle file for each image which will later be used by Notebook 1B
  - These are some of the auto marked images





  - This is the file type



content

Images

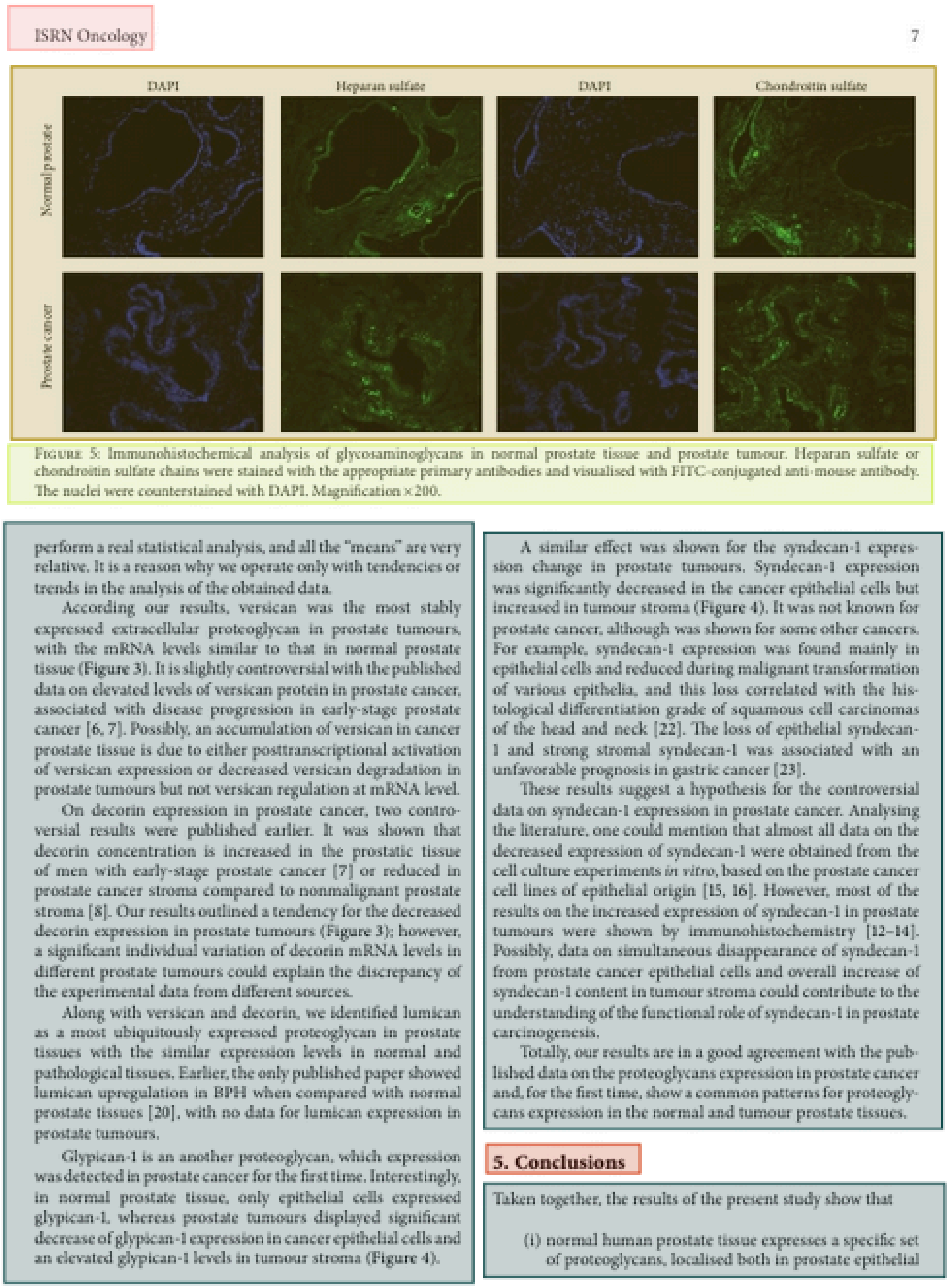
Marked_images_Final_1

pickle_files_Final_1

- In **Notebook 1B** We have used those pickle files and via using TesseractAgent have got the text inside each box marked via using R-CNN R-50 FPN model model. Then we have encoded the text in a way such that the encoder which we used there must not go out of range. If the parah from a Box goes out of range a new Chunk is formed . Then Meta data is added before encoding .
- **This is WORKING**

Documentation 2:
This is my attempt to use LayoutLLMV3

- Firstly we have to label different parts of the document using Label Studio .Though I would recommend using UBIAI as it can also do OCR and we are mainly marking this and have to do ocr to make a training dataset .



ISRN Oncology 7

Figure 5: Immunohistochemical analysis of glycosaminoglycans in normal prostate tissue and prostate tumour. Heparan sulfate or chondroitin sulfate chains were stained with the appropriate primary antibodies and visualised with FITC-conjugated anti-mouse antibody. The nuclei were counterstained with DAPI. Magnification ×200.

perform a real statistical analysis, and all the "means" are very relative. It is a reason why we operate only with tendencies or trends in the analysis of the obtained data.

According our results, versican was the most stably expressed extracellular proteoglycan in prostate tumours, with the mRNA levels similar to that in normal prostate tissue (Figure 3). It is slightly controversial with the published data on elevated levels of versican protein in prostate cancer, associated with disease progression in early-stage prostate cancer [6, 7]. Possibly, an accumulation of versican in cancer prostate tissue is due to either posttranscriptional activation of versican expression or decreased versican degradation in prostate tumours but not versican regulation at mRNA level.

On decorin expression in prostate cancer, two controversial results were published earlier. It was shown that decorin concentration is increased in the prostatic tissue of men with early-stage prostate cancer [7] or reduced in prostate cancer stroma compared to nonmalignant prostate stroma [8]. Our results outlined a tendency for the decreased decorin expression in prostate tumours (Figure 3); however, a significant individual variation of decorin mRNA levels in different prostate tumours could explain the discrepancy of the experimental data from different sources.

Along with versican and decorin, we identified lumican as a most ubiquitously expressed proteoglycan in prostate tissues with the similar expression levels in normal and pathological tissues. Earlier, the only published paper showed lumican upregulation in BPH when compared with normal prostate tissues [20], with no data for lumican expression in prostate tumours.

Glypican-1 is an another proteoglycan, which expression was detected in prostate cancer for the first time. Interestingly, in normal prostate tissue, only epithelial cells expressed glypican-1, whereas prostate tumours displayed significant decrease of glypican-1 expression in cancer epithelial cells and an elevated glypican-1 levels in tumour stroma (Figure 4).

A similar effect was shown for the syndecan-1 expression change in prostate tumours. Syndecan-1 expression was significantly decreased in the cancer epithelial cells but increased in tumour stroma (Figure 4). It was not known for prostate cancer, although was shown for some other cancers. For example, syndecan-1 expression was found mainly in epithelial cells and reduced during malignant transformation of various epithelia, and this loss correlated with the histological differentiation grade of squamous cell carcinomas of the head and neck [22]. The loss of epithelial syndecan-1 and strong stromal syndecan-1 was associated with an unfavorable prognosis in gastric cancer [23].

These results suggest a hypothesis for the controversial data on syndecan-1 expression in prostate cancer. Analysing the literature, one could mention that almost all data on the decreased expression of syndecan-1 were obtained from the cell culture experiments *in vitro*, based on the prostate cancer cell lines of epithelial origin [15, 16]. However, most of the results on the increased expression of syndecan-1 in prostate tumours were shown by immunohistochemistry [12–14]. Possibly, data on simultaneous disappearance of syndecan-1 from prostate cancer epithelial cells and overall increase of syndecan-1 content in tumour stroma could contribute to the understanding of the functional role of syndecan-1 in prostate carcinogenesis.

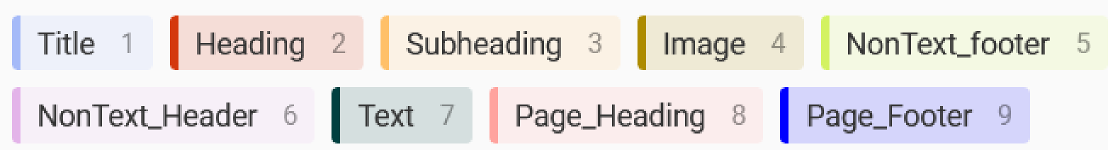
Totally, our results are in a good agreement with the published data on the proteoglycans expression in prostate cancer and, for the first time, show a common patterns for proteoglycans expression in the normal and tumour prostate tissues.

5. Conclusions

Taken together, the results of the present study show that

(i) normal human prostate tissue expresses a specific set of proteoglycans, localised both in prostate epithelial

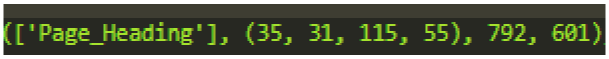
- The below are the labels which I was using

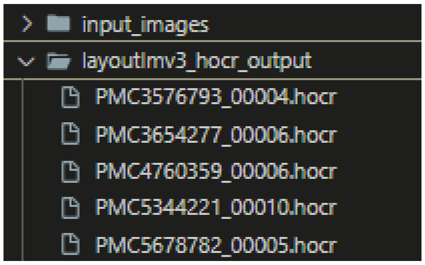


Title 1
Heading 2
Subheading 3
Image 4
NonText_footer 5
NonText_Header 6
Text 7
Page_Heading 8
Page_Footer 9

- After marking this on multiple images we should extract the json file .It's in the folder of the Notebook 2 named project-2-at-2024-07-11-20-12-5822c964.json. In the Json we should change the file names as it adds random values in front of file names.

- The **class_list.txt** contains different labels used: Title, Heading, Subheading, Image, NonText_footer, NonText_Header, Text, Page_Heading, Page_Footer
- **GettingData.py**
  - We have also made a custom dataframe by the name of data.csv which contains bbox coordinates for the opposite corners which would later be used to make rectangles and extract text .also have labels for that rectangle and also have size of the image .

    ```
    (['Page_Heading'], (35, 31, 115, 55), 792, 601)
    ```
  - Use Pyteseract to extract the data from it
    - 

      ```
      > input_images
      v layoutlmv3_hocr_output
          PMC3576793_00004.hocr
          PMC3654277_00006.hocr
          PMC4760359_00006.hocr
          PMC5344221_00010.hocr
          PMC5678782_00005.hocr
      ```
  - After that use tesseract and label-studio labels to create training files : **GettingData.py**
    - Results in formation of
      - test.txt
      - train.txt

- **layoutlmv3.py**
  - Contains the code to get the data in the order as required by the LayoutLMV3 model
- **Invoice_dataset_loading.py**
  - Contains code to form the dataset via using layoutlmv3

Work left
Have to train the model and test it and evaluate it

We can measure it via different classification performance measuring metrics like when we to see in terms of it getting accurate OCR and bounding box
The False Negative being : It left something unmarked
The False Positive being : It marked the box when it is not required like in the middle of the page

- Accuracy: It's one of the best measures out here .

- Precision
- Recall