M.TECH PROJECT REPORT
Towards Automated Generation of Image Descriptions for Visually Impaired

Thesis submitted by

Gopi Swamy Veerendra
2013CS50285

under the guidance of

Prof. M. Balakrishnan, Indian Institute of Technology Delhi
Prof. Volker Sorge, University of Birmingham

in partial fulfilment of the requirements
for the award of the degree of

Bachelor and Master of Technology

Department Of Computer Science
INDIAN INSTITUTE OF TECHNOLOGY DELHI

April 2022

THESIS CERTIFICATE

This is to certify that the thesis titled **Automated Generation of Image Descriptions**, submitted by **Gopi Swamy Veerendra (2013CS50285)**, to the Indian Institute of Technology, Delhi, for the award of the degree of **Bachelor and Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. M. Balakrishnan
Professor
Dept. of Computer Science
IIT-Delhi, 110016

Place: New Delhi

Date: April 2022

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor, **Prof. M. Balakrishnan** for his support, exemplary guidance, and constant encouragement throughout the course of this project.

A very special thanks to **Prof. Volker Sorge**, and **Neha Jadhav** for their guidance and help throughout the project. I also would like to thank **Prof. Prem Kalra** for his help and encouragement. I would also like to thanks my parents and brothers for their moral support and continuous encouragement.

ABSTRACT

One of the biggest impediments for the visually impaired to read and understand STEM material is their inability to see figures which describe experiments or provide illustrations. As part of the RAVI (Reading Accessibility for Visually Impaired) project, prior work by Chahal et al., 2018 has shown that providing alternate (handwritten) text description for images greatly helps understanding for visually impaired, and they have designed an effective template to write such descriptions.

In this thesis, we initiate the process of automatically filling such template using latest NLP and Computer Vision tools. We extract most of the relevant information required to fill the template automatically including image captions, image labels and their positions, and relevant text to the image from nearby text. We also generate a short summary of the relevant text, caption and labels using latest NLP tools. In future work, one can combine this information to automatically write text descriptions of images following the template.

Contents

Chapter 1

Introduction

This project makes an effort to solve one of the biggest challenges encountered by the visually impaired to comprehend figures while reading and understanding Science, Technology, Engineering, and Mathematics (STEM) material. When a visually impaired person reads the material, understanding only text is insufficient to comprehend the entire content. Analysis of all the graphical content is equally essential for STEM documents understanding. We took the first step in automating the generation of text descriptions of figures for the visually impaired.

As part of the RAVI (Reading Accessibility for Visually Impaired) project, Chahal et al., 2018 paper has shown augmenting image description with template-based information helps in the effective generation of alternate text for the images. A template is designed to capture the data shown in Table 1. A user study with manually filling the template of relevant information enhances the visualisation of diagrams, making them more understandable and accessible to visually impaired users.

Table 1. Template sections and sources

	Template Section	**Source of Content**
1	Overview	Image caption
2	Number of Objects	From Image Processing
3	What are the Objects?	From the labels (e.g. Beaker, Burner, Test Tube)
4	Any specific placement?	From label positioning (Left to right, top to bottom, or none)
5	How are the Objects placed?	Relative positioning of the objects (e.g. x is at the top, y is on the left of x)
6	How are the Objects interacting?	This has to be extracted from the text surrounding the image. (e.g. beaker heated using the burner)

But to deploy this on a large scale, a significant challenge is to automate finding the relevant information for the template section mentioned above, and template contents should be summarised into a paragraph using Natural Language Processing (NLP) technique. This technique will consider the text surrounding the diagram and the text present inside the diagram.

1.1 Motivation

In the past few years, there has been a lot of advances in the area of Machine Learning (ML), especially in NLP. In 2017, Model of Transformers in Vaswani et al., 2017 was a breakthrough in NLP, replacing the previous best models, Recurrent Neural Networks (RNN) & Long Short-Term Memory (LSTM). Unlike RNNs, transformers do not necessarily process the data in order. For example, if the input data is a natural language sentence, the transformer does not need to process the beginning of the sentence before the end. Rather, it identifies the context that confers meaning to each sentence. This feature allows for more parallelisation than RNNs and therefore reduces training times.

The additional training parallelisation allows training on larger datasets than was once possible. This led to the development of pre-trained systems such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), trained with large language datasets, such as the Wikipedia Corpus and Common Crawl, and can be fine-tuned for specific tasks. This makes Sate-Of-The-Art (SOTA) open-source models trained on large datasets available, which otherwise takes a lot of computational power and time to build.

In Chahal et al., 2018, they tested the latest image captioning tools and image descriptions tools, which are trained on general images (primarily photographs), didn't do well on STEM images where there are labels, graphs, tables, graphics and Hand drawn figures. Hence, we started the project to use Sate-Of-The-Art NLP models to solve the image description using text in and nearby the image.

1.2 Objective

Automatically extract key information to fill the template such as:

- Caption of the diagram
- Labels & their positions in the diagram
- Relevant text to the diagram from nearby text using latest NLP tools
- Summarize the relevant text obtained into a short paragraph

One can combine this information to get a complete text description according to the template in future work.

1.2.1 Categories of data

Since our project mainly deals with figures, our dataset contains NCERT science textbooks from class 6^{th} to class 12^{th} and technical research papers.

1.2.2 Choice of data format

All the datasets are available in pdf, Microsoft word and epub file formats. In the case of pdf files, images are of low resolution, making them unsuitable for using computer vision techniques like optical character recognition (OCR). In the case of word files, there are no indentations and structure to the data, making it challenging to parse the text. While epub files overcome both issues of pdf and word files, epub file has no pages or page numbers which is trivial at this point.

We are using epub files in this project because of structured data and clear images. All images will be present in the Images folder of an unzipped epub file. The text folder will have all HTML or XHTML files for each chapter, and this folder may contain one chapter or multiple chapters depending on the design of the textbook or technical paper. Figure 1.1 shows the directory folders we generally expect once we unzip an epub file.

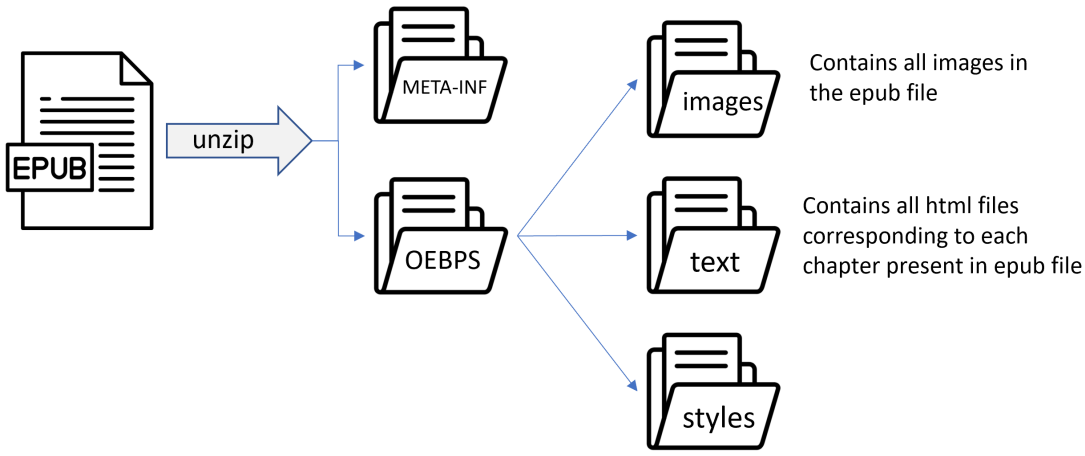


Figure 1.1: Folders directory when epub is unzipped to html or xhtml files

Chapter 2

Image and caption extraction

Generally, the caption under an image will give an idea about the diagram, so it is a good option to incorporate the caption in the description of the diagram. Figure 2.1 shows an example of a diagram and corresponding image that has to be extracted from HTML files.

Chapter 2

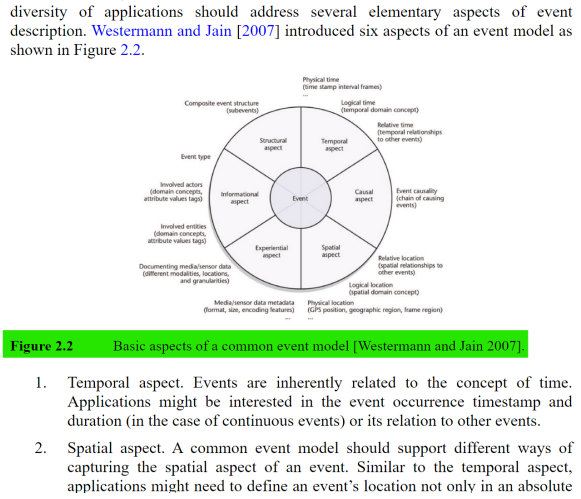
Is Matter Around Us Pure?

How do we judge whether milk, ghee, butter, salt, spices, mineral water or juice that we buy from the market are pure?

Fig. 2.1: Some consumable items

Have you ever noticed the word 'pure' written on the packs of these consumables? For a common person pure means having no adulteration. But, for a scientist all these things are actually mixtures of different substances and hence not pure. For example, milk is actually a mixture of water, fat, proteins etc. When a scientist says that something is pure, it means that

(a) Class 9th NCERT textbook



diversity of applications should address several elementary aspects of event description. Westermann and Jain [2007] introduced six aspects of an event model as shown in Figure 2.2.

Figure 2.2 Basic aspects of a common event model [Westermann and Jain 2007].

1. Temporal aspect. Events are inherently related to the concept of time. Applications might be interested in the event occurrence timestamp and duration (in the case of continuous events) or its relation to other events.
2. Spatial aspect. A common event model should support different ways of capturing the spatial aspect of an event. Similar to the temporal aspect, applications might need to define an event's location not only in an absolute

(b) Technical paper

Figure 2.1: Example of captions and corresponding image in a HTML file

So, from the HTML file, we have to extract the caption provided under the diagram and the corresponding diagram to the caption refers. For that, we use beautiful soup BS4, version 4.10.0 library in python to parse HTML or XHTML.

2.1 Caption extraction

Ideally, We expect captions under a diagram will start with "Fig" because it may be "Fig 1.2: xxxxxxx" or "Figure 1.2: xxxxxxx". We use regular expression "^Fig" to find all sentences that start with "Fig". Major problem with the HTML file is the text of one sentence can be split into multiple elements and still can be seen as single sentence while reading. To handle above problem we look not only the element that contains sentence starting with "Fig" we also look into text of whole parent element and text of whole grand parent element. We take element among element, parent element and grandparent element that just crossed the minimum number of words and less than maximum number of words.

we estimated best minimum and maximum number of words that a caption can contain as 3 and 60 respectively.

E - Element that containing the sentence starting with Fig

PE - Parent element of E

GE - parent element of PE

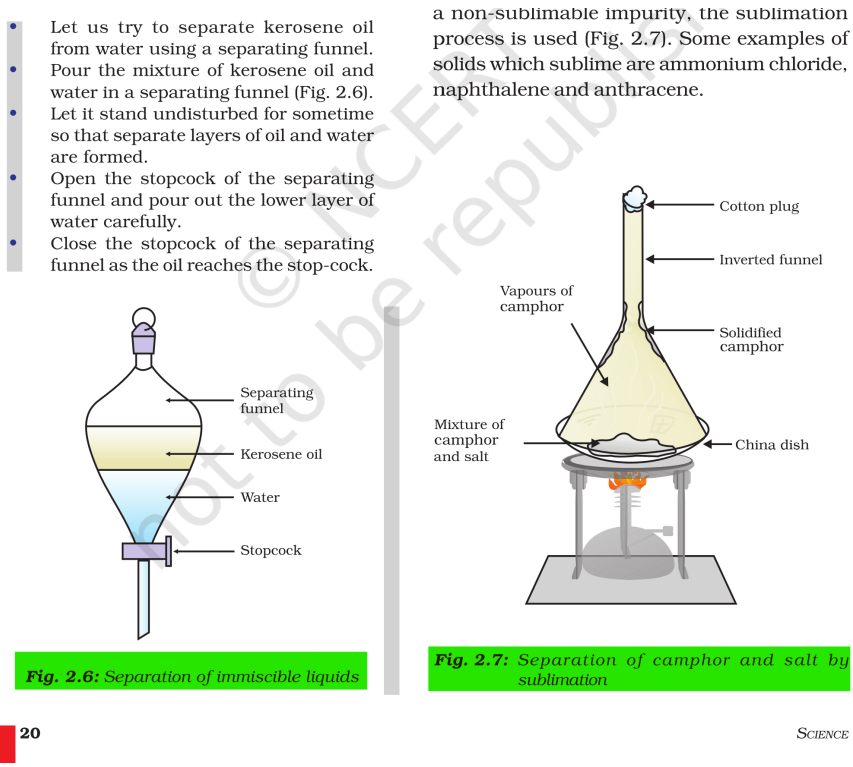
Max_words - Maximum number of words that a caption can have

Min_words - Minimum number of words that a caption should have

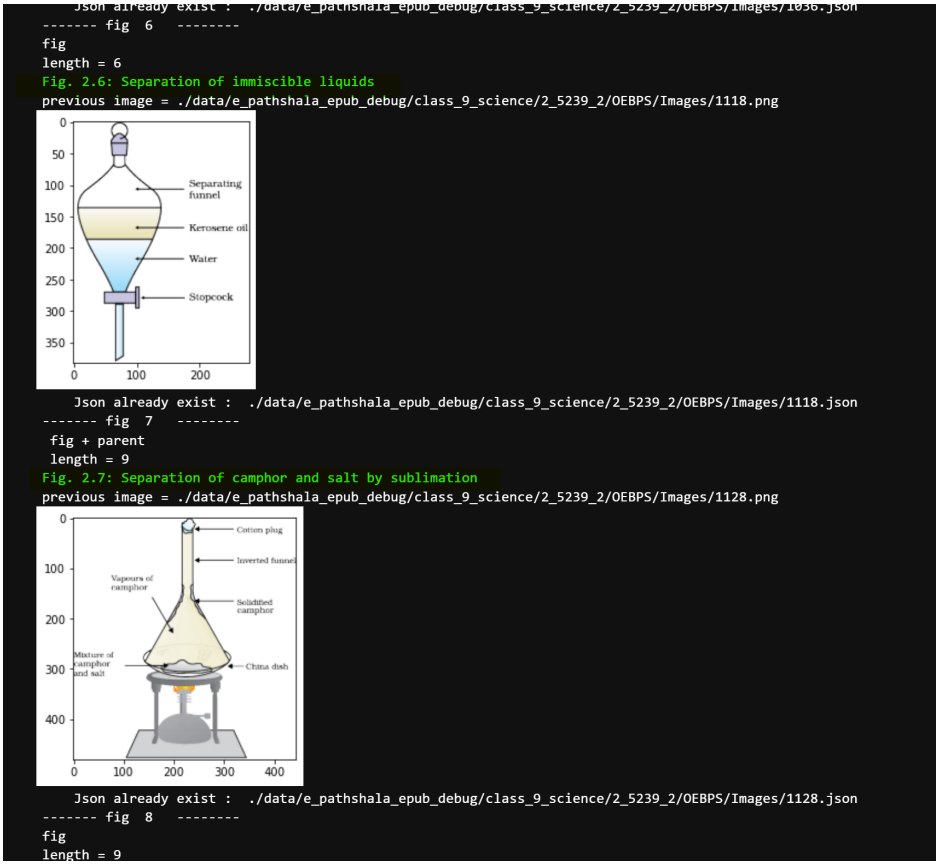
Algorithm:

```
if (E has more than Max_words):
        Skip that figure
    else if (E has more than Min_words):
        Take E as caption
    else if (PE has more than Max_words):
        Take E as caption
    else if (GE has more than Max_words):
        Take PE as caption
    else:
        Take GE as caption
```

Figure 2.2(a) shows example ground truth from a 9th class NCERT textbook and caption is highlighted with green and Figure 2.2(b) shows the output we get out of the algorithm. Figure 2.3(a) shows example ground truth from a technical paper and caption is highlighted with green and Figure 2.3(b) shows the output we get out of the algorithm.



(a) Textbook ground truth



(b) Output

Figure 2.2: Result of captions and corresponding images in a textbook HTML file

problem, the authors defined three main tasks: (1) Frequent pattern mining and association rules for chronic diseases and related treatments. (2) Frequent sequence mining to analyze compl
Periodic event mining for finding periodical sequences of certain diseases.

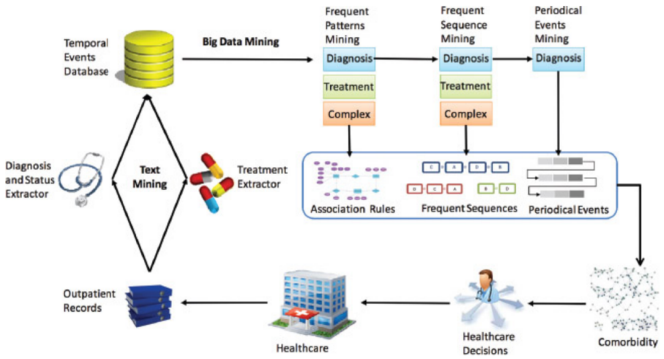
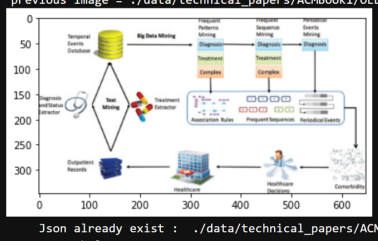


Figure 5.1 High-level architecture of a pattern mining system for understanding comorbidities of multiple chronic diseases. Reproduced from Boytcheva et al. [2019].

Although the authors did not provide any specific algorithm for each task, they applied their system to find associations among schizophrenia, diabetes mellitus type 2, and hyperprolacti
Many researchers have developed algorithms for exploring temporal events in the healthcare domain with a specific focus on visualization. Early works focused mostly on visualizing

(a) Paper ground truth

```
++++++++++++++++++++++++++   START   ++++++++++++++++++++
./data/technical_papers/ACMBook1/OEBPS/xhtml/b_9781450384827-chapter-005.xhtml
@@@@@@@@@@@@   XML parser @@@@@@@@@@@@
------- fig  1   --------
fig + grandparent
length = 21
Figure 5.1High-level architecture of a pattern mining system for understanding comorbidities of multiple chronic diseases. Reproduced from Boytcheva et al. [2019].
previous image = ./data/technical_papers/ACMBook1/OEBPS/images/b_9781450384827-005_fig_001.jpg
```




```
Json already exist :  ./data/technical_papers/ACMBook1/OEBPS/images/b_9781450384827-005_fig_001.json
```

(b) Output

Figure 2.3: Result of caption and corresponding image in a technical paper HTML file

Caption extraction limitations

- If the caption has more than Max_words (here 60 words) cant be captured as caption
- If the caption is split into two different parents and not under common grand parent then caption wont be captured fully
- If the grand parent or parent element is selected but there are multiple caption under same grand parent or parent element
- If a sentence start with a "Fig" then it assumes that the sentence is a caption and will look for corresponding image which is not expected outcome

2.2 Corresponding image extraction

Ideally, we expect caption lies just below the image or diagram it is referring to. One of the things we need to take into consideration is that image may be in same element or in parent element. Example of the results are shown in Figure 2.2 and Figure 2.3.

Algorithm:

```
if image in same element:
        get image url from image element 'src' tag
    else if image in parent's element:
        get image url from image element 'src' tag
```

Image extraction limitations

- If same element has multiple images then it only considers last image of the element because there is no way we can identify the image the caption is referring to

Chapter 3

Extracting labels and their position

In the previous chapter we get to know how we got all captions and their corresponding images from a html file. Now, lets try to get text information and positions of each text in the image so to use that information in filling the template sections shown in Table 1.

3.1 Optical Character Recognition(OCR) tools

Optical character recognition or optical character reader (OCR) is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo. Early we needed to be trained with images of each character, and worked on one font at a time but now Advanced systems capable of producing a high degree of recognition accuracy for most fonts are now common, and with support for a variety of digital image file format inputs.

To extract text as labels and their positions we tested two solutions Pytesseract python library and Google cloud vision API.

3.2 Pytesseract python library

We used Pytesseract, version 0.3.9, python library to extract labels and their positions in the image. Pytesseract is a free open source software and there is no fee for each image it analyse to detect labels and their positions. Pytesseract can't recognize transparent background as they are provided from NCERT textbooks so, we converted transparent background to white background. We can see, in Figure 3.1, that Pytesseract left multiple labels undetected and even two lines in one label is detected as two different labels.

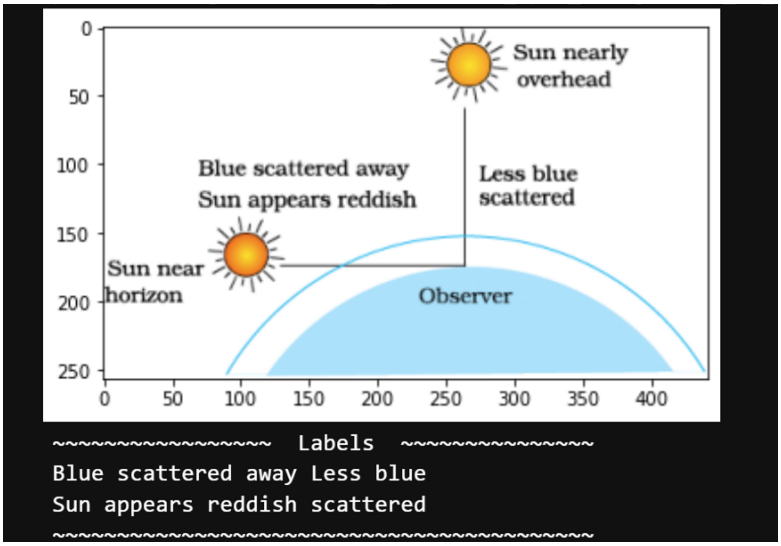


Figure 3.1: Image and results from Pytesseract python library

Limitations of Pytesseract

- Bad performance
- Can't recognize images with transparent background
- Not organizing text of different labels into separate labels
- Not providing bounding box and positions of the text it detect.

3.3 Google Cloud Vision API

The Google Cloud Vision API can detect and extract text from images. There are two annotation features that support OCR:

TEXT_DETECTION: detects and extracts text from any image. For example, a photograph might contain a street sign or traffic sign. The JSON includes the entire extracted string, as well as individual words, and their bounding boxes.

DOCUMENT_TEXT_DETECTION: also extracts text from an image, but the response is optimized for dense text and documents. The JSON includes page, block, paragraph, word, and break information.

In our testing, we saw that **DOCUMENT_TEXT_DETECTION** is doing a lot better compared to TEXT_DETECTION. Hence, as example shown in figure 3.2, we used DOCUMENT_TEXT_DETECTION to extract labels & their positions of images we extracted earlier. As Google Cloud Vision API is paid service based on number images that we use it for, we optimized the it by storing the JSON object we get from google API call

and also stored as a python dictionary which is more easy to load in our code. So, for every image it searches for JSON file and only if it is not found code will query the API and store it as mentioned earlier.

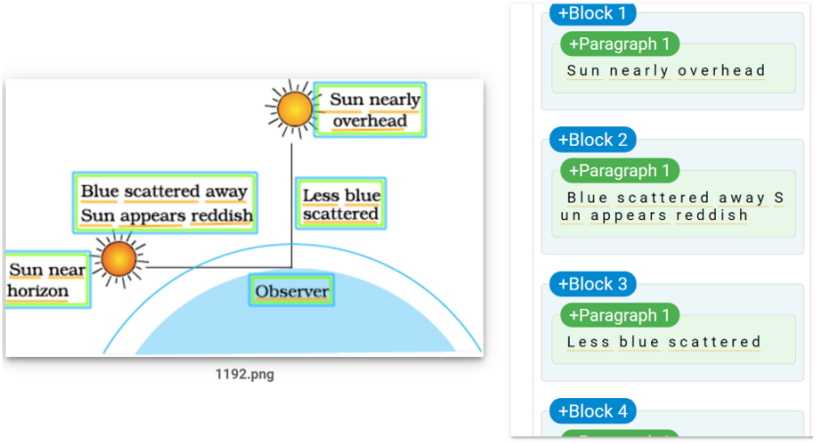
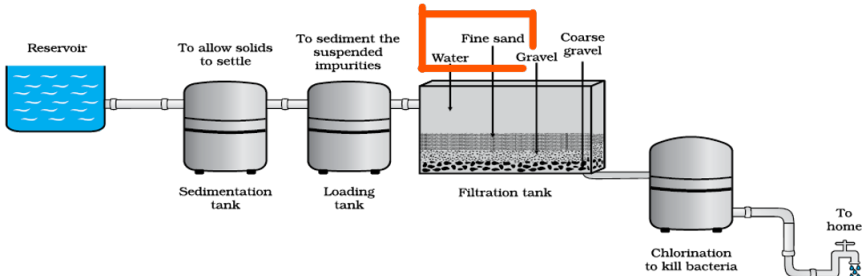


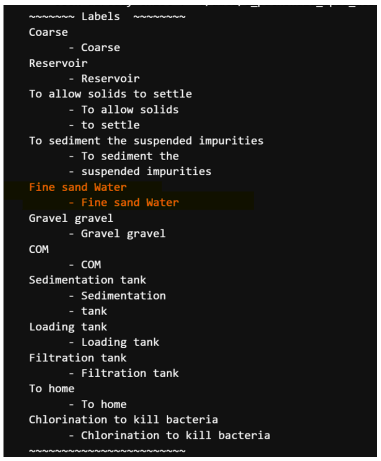
Figure 3.2: Visual depection of labels and their positions from Google Cloud Vision API

Limitations of Google cloud vision API

Even Google vision API is not perfect, in figure 3.3 it combined "water" and "fine sand" as single label which are in fact seperate labels



(a) image uploaded to Google API



(b) Labels from Google API

Figure 3.3: Image and results from Google Cloud Vision API

Chapter 4

Extracting relevant text from nearby text

In previous chapter we extracted caption, corresponding image, labels and their positions in that image. Now in order get what image is describing or referring to, we need to find relevant text from the nearby text around the image. The main problem of finding relevant text from the nearby text is that how to find relevance between the image and the text surrounding it? Since we had information of caption and labels of the image, we use them to find the relevance between the image and text surrounding it. We append labels to caption to make a sentence i.e., (caption + "This figure contains " + labels separated by commas). Finally the objective bowls down to finding relevance between the labels appended caption and sentences surrounding the image.

4.1 Sentence Transformers

The problem of finding relevance between sentences falls under NLP, specifically under Semantic textual similarity. In 2017, a paper Vaswani et al., 2017 titled Attention Is All You Need was published. According to Briggs, 2022 this marked a turning point in NLP. The authors demonstrated that we could remove the RNN networks and get superior performance using just the attention mechanism — with a few changes. This new attention-based model was named a 'transformer'. Since then, the NLP ecosystem has entirely shifted from RNNs to transformers thanks to their vastly superior performance and incredible capability for generalization.The first transformer removed the need for RNNs through the use of three key components:

- **Positional encoding:** replaced the key advantage of RNNs in NLP — the ability to consider the order of a sequence (they were recurrent). It worked by adding a set of varying sine wave activations to each input embedding based on position.
- **Self-attention:** is where the attention mechanism is applied between a word and all of the other words in its own context (sentence/paragraph). This is different from vanilla attention which specifically focused on attention between encoders and decoders.
- **Multi-head attention:** can be seen as several parallel attention mechanisms working together. Using several attention heads allowed the representation of several sets of relationships (rather than a single set).

The new transformer models generalized much better than previous RNNs, which were often built specifically for each use-case. With transformer models, it is possible to use the

same 'core' of a model and simply swap the last few layers for different use cases (without retraining the core).This new property resulted in the rise of pretrained models for NLP. Pretrained transformer models are trained on vast amounts of training data — often at high costs by the likes of Google or OpenAI, then released for the public to use for free. Similarly HuggingFace is a open source AI community that offers plenty of free pretrained models and datasets.

Sentence Embeddings

Sentence Embeddings is the collective name for a set of techniques in NLP where sentences are mapped to d-dimensional vectors of real numbers i.e. sentence $S \rightarrow F(S) \in \mathbb{R}^d$. Sentence Transformers is a Python framework for state-of-the-art sentence, text and image embeddings. The initial work is described in Sentence-BERT: Sentence Embeddings using Siamese BERT-Network by Reimers and Gurevych, 2019. These embeddings can then be compared e.g. with cosine-similarity to find sentences with a similar meaning. This can be useful for semantic textual similar, semantic search, or paraphrase mining. Here, we convert our sentences into sentence embeddings using various models of Sentence Transformers to find relevant sentences between the labels appended caption and sentences surrounding the image. If S_1 and S_2 are similar then $F(S_1).F(S_2)$ is large or Cosine similarity $(cos\theta)$ between $F(S_1)$ and $F(S_2)$ is close to 1. Figure 4.1 shows an example of how sentence embeddings dot product values look like.

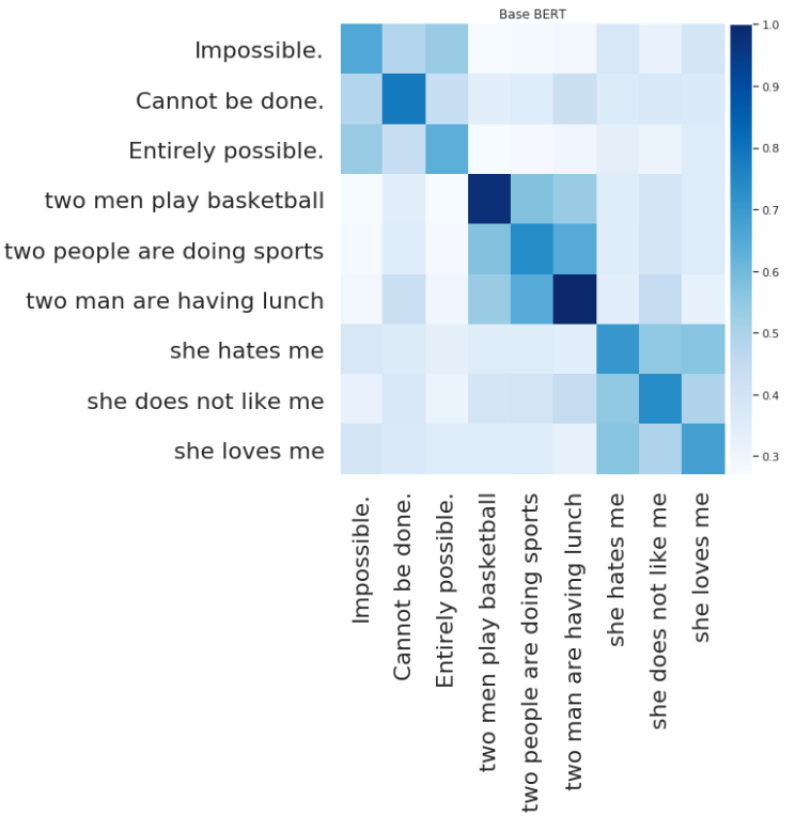


Figure 4.1: Similarity between sentence pairs encoded by sentence transformer model

We Tried various models from Hugging Face pre-trained models like

- BERT (bert-base-nli-mean-tokens)
- all-MiniLM-L6-v2
- paraphrase-multilingual-MiniLM-L12-v2
- multi-qa-MiniLM-L6-cos-v1
- MPnet (all-mpnet-base-v2)

4.2 Generating test set

To test the models, we have to have ground truth of sentences that are actually relevant to the labels appended caption sentence. An assumption, that figures are present inside the section with the relevant text, was made for NCERT textbooks to generate ground truth for each image in a textbook. So, we consider all sentences of the section that contains figure as relevant sentences.

While for technical papers, the figure may be in another section which is not relevant to the figure. So, we manually find the relevant sentences for each image in that paper.

4.3 Testing methodology

In our case, **False Negatives** are the sentences that are not predicted by the model as relevant sentences but are relevant sentences as per ground truth. **False positives** are the sentences that are predicted by the model as relevant sentences but are not relevant sentences as per ground truth. Figure 4.2 visualize a general scenario of False Positives and False Negatives.

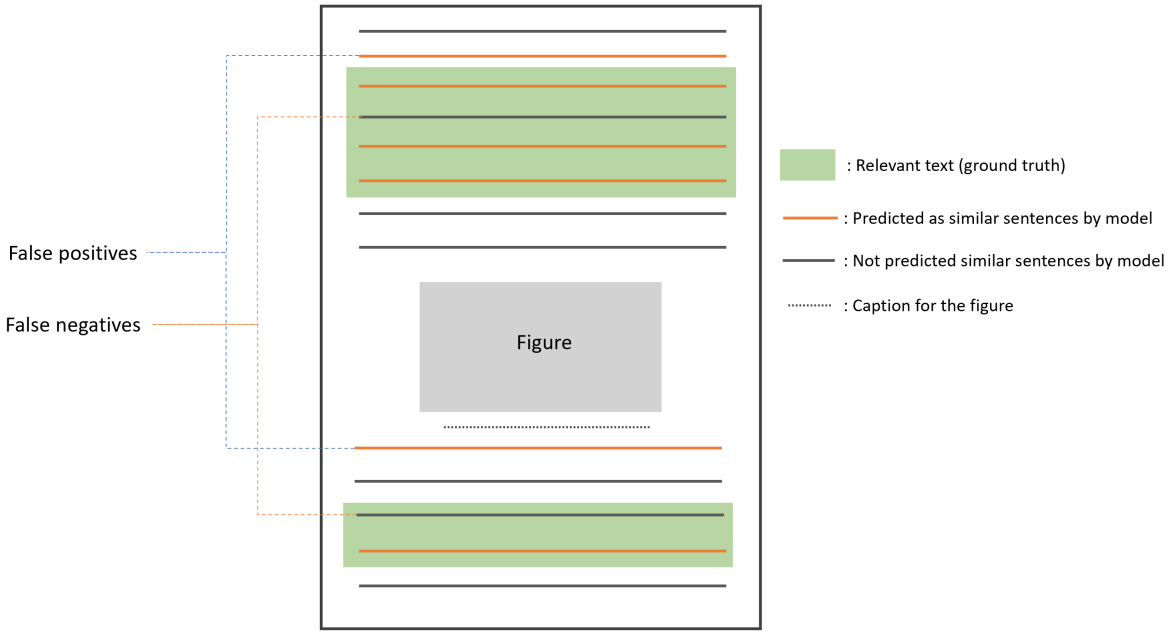


Figure 4.2: Visualizing false positives and false negatives in our scenario

Now we define False Negative Rate (FNR) and False Positive Rate (FPR) as follows:

$$False\ Negative\ Rate(FNR) = \frac{Number\ of\ False\ Negatives}{Number\ of\ relevant\ sentences\ as\ per\ ground\ truth}$$

$$False\ Positive\ Rate(FPR) = \frac{Number\ of\ False\ Positives}{Total\ number\ of\ sentences\ predicted\ as\ relevant\ by\ model}$$

We fix some threshold ϕ and all sentences whose cosine similarity to labels appended caption sentence crosses ϕ is predicted as relevant by the model. Then we calculate average FPR and FNR for varying threshold ϕ and plot a scatter plot for each model.

Figure 4.3 shows an example FNR-FPR scatter plot with varying cosine thresholds. Ideally, we expect FNR and FPR to be small. We will choose ϕ from the plots such that

both FNR and FPR are minimized.

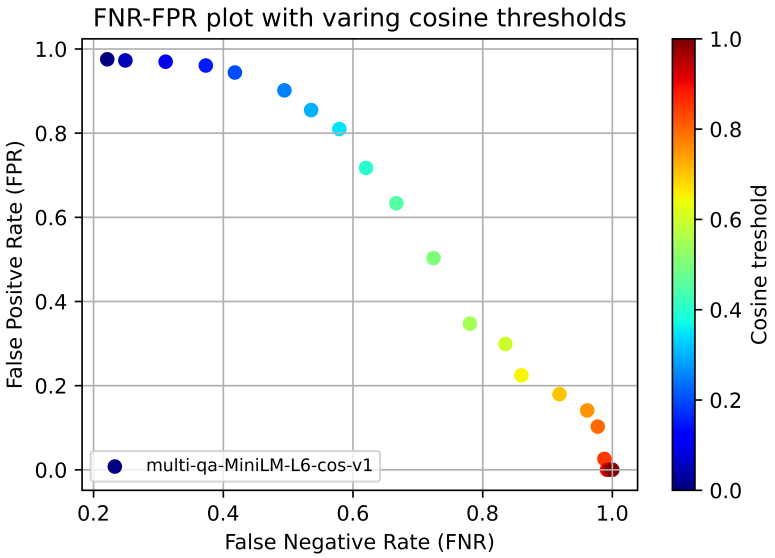


Figure 4.3: Example FNR-FPR scatter plot for a single model

To compare models, we plot above scatter plots for each model into one plot as shown in figure 4.4. We can see in figure 4.4 that **all-mpnet-base-v2, all-MiniLM-L6-v2,0 and multi-qa-MiniLM-L6-cos-v1** models are doing better compared to remaining models.

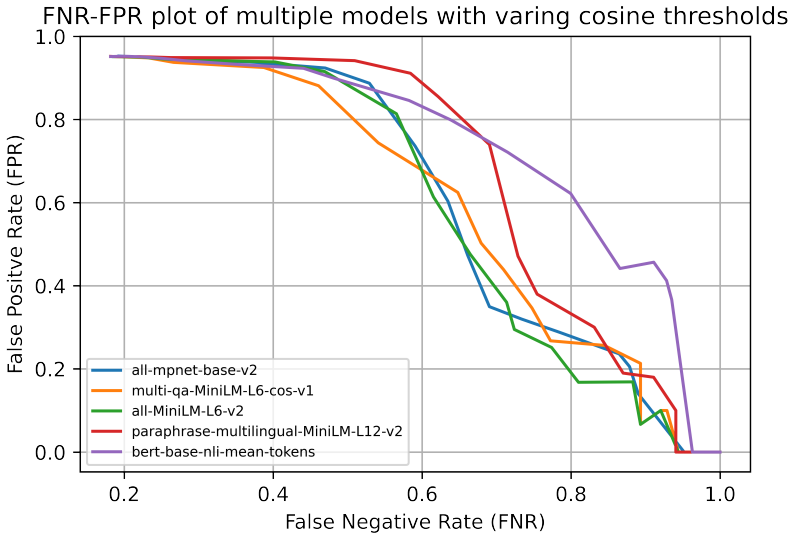


Figure 4.4: Example FNR-FPR plots for all models

4.4 Heuristics for improving performance

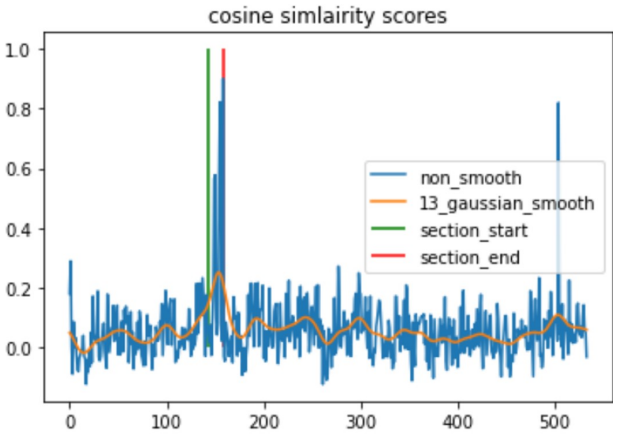
Let's introduce various heuristics we used to improve FNR-FPR plots i.e. to bring down FNR and FPR values.

4.4.1 Smoothing cosine similarity scores

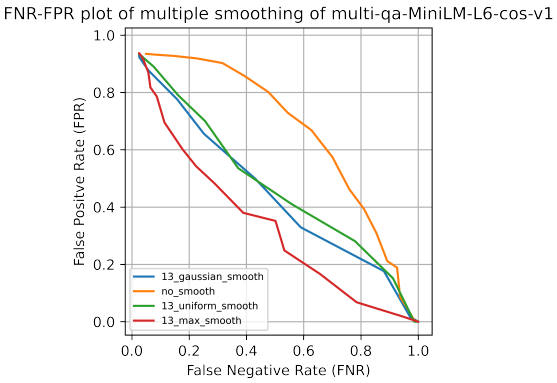
Ideally, we expect relevant sentences are usually surrounded by other relevant sentences because all text in a generally area discuss about one topic. Hence, by locally smoothing cosine similarity scores i.e. convolution of cosine values with a smoothing curve, we can expect to reduce FPR-FNR rates. But there are various smoothing curves to locally smooth a curve like:

- Convolution with Uniform curve with various window size
- Convolution with Gaussian curve with various window size
- Rolling max with various window size

First let's see in Figure 4.5(b) whether smoothing improve our predictions.



(a) example of a smoothed cosine scores



(b) Comparison of various smoothings for a single model

Figure 4.5: Smoothing cosine scores

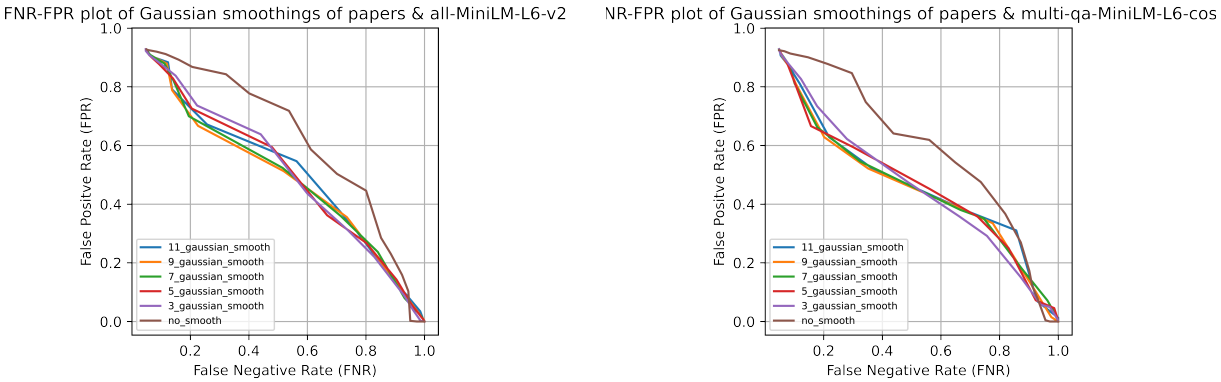
Note: Number before smoothing curve labels in figure 4.5(b) or later is window size for convolution

We can see that any locally smoothing of cosine similarity scores improve predictions. This trend is same for all other models.

Why smoothing is improving the prediction

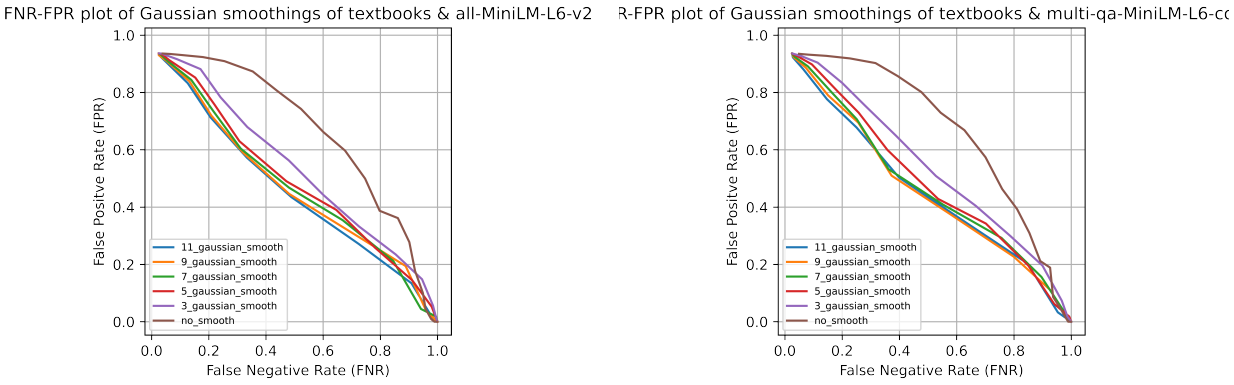
Ideally, we expect relevant sentences are usually surrounded by other relevant sentences but there are many cases where few sentences talk about similar thing in a different scenario for example figure 4.5(a) shows there is a peak of cosine score outside the relevant section but after smoothing the peak outside the section got smoothed by surrounding low cosine scores.

Let's compare various window sizes for Gaussian smoothing in both technical papers and textbooks. We can see for papers and textbooks in both models figure 4.6 and figure 4.7 further increasing window size for locally smoothing is no longer improving the prediction and we found that window size of 21 sentences working the best. so, we consider **7_gaussian_smooth** is the best gaussian smoothing with minimum window size.



(a) all-MiniLM-L6-v2 model (b) multi-qa-MiniLM-L6-cos-v1 model

Figure 4.6: Comparison of various window size for Gaussian smoothing for technical papers

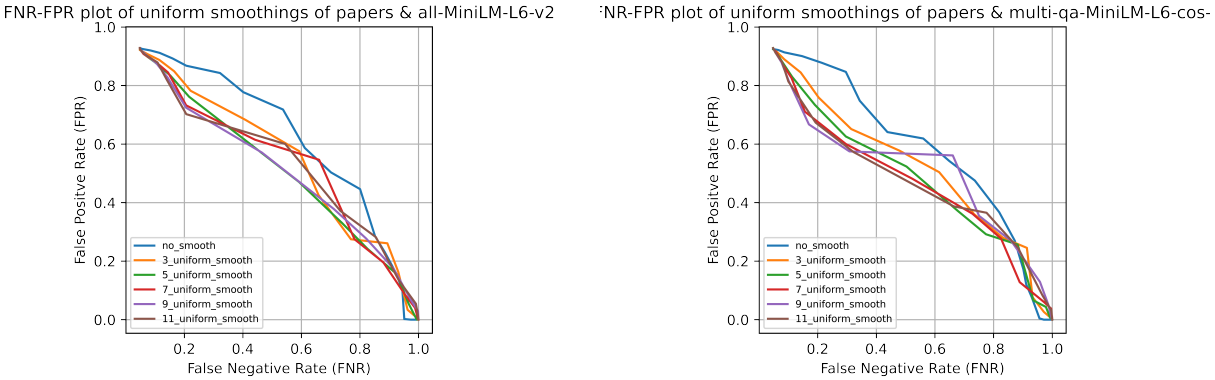


(a) all-MiniLM-L6-v2 model (b) multi-qa-MiniLM-L6-cos-v1 model

Figure 4.7: Comparison of various window size for Gaussian smoothing for textbooks

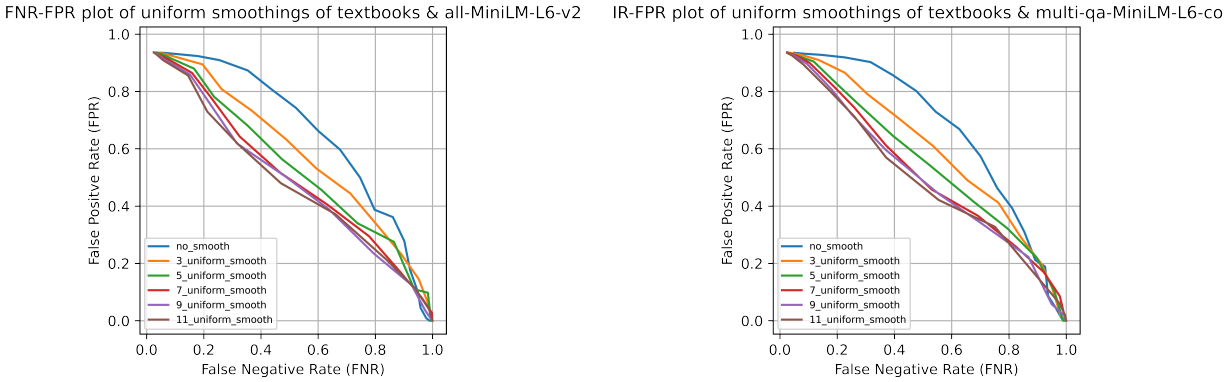
Let's compare various window sizes for uniform smoothing in both technical papers and textbooks. We can see for papers and textbooks in both models figure 4.8 and figure 4.9

further increasing window size for locally smoothing is no longer improving the prediction and we found that window size of 9 sentences working the best. so, we consider **9_uniform_smooth** is the best uniform smoothing with minimum window size.



(a) all-MiniLM-L6-v2 model (b) multi-qa-MiniLM-L6-cos-v1 model

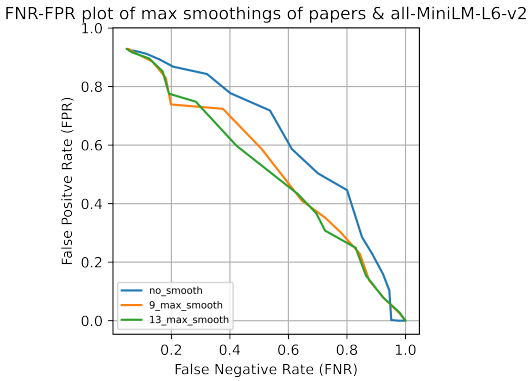
Figure 4.8: Comparison of various window size for uniform smoothing for technical papers



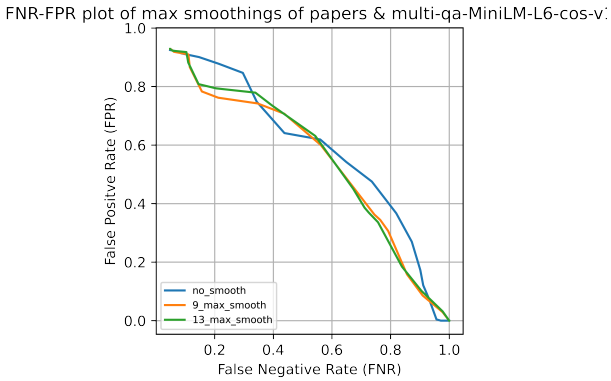
(a) all-MiniLM-L6-v2 model (b) multi-qa-MiniLM-L6-cos-v1 model

Figure 4.9: Comparison of various window size for uniform smoothing for textbooks

Let's compare various window sizes for rolling max smoothing in both technical papers and textbooks. We can see for papers and textbooks in both models figure 4.10 and figure 4.11 further increasing window size for locally smoothing is no longer improving the prediction and we found that window size of 9 sentences working the best. so, we consider **9_max_smooth** is the best uniform smoothing with minimum window size.

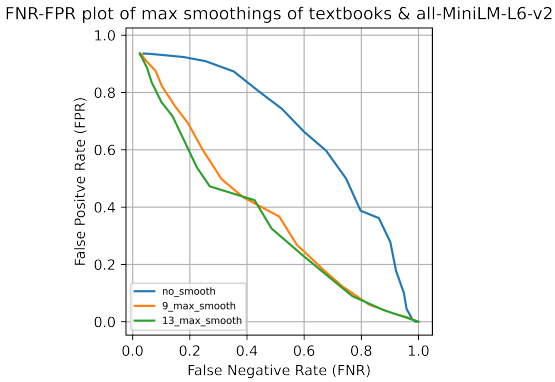


(a) all-MiniLM-L6-v2 model

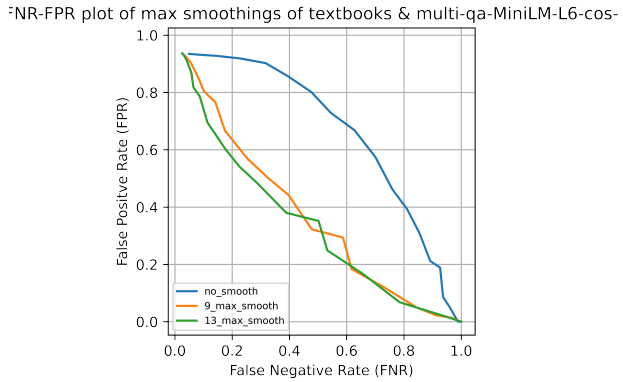


(b) multi-qa-MiniLM-L6-cos-v1 model

Figure 4.10: Comparison of various window size for rolling max smoothing for technical papers



(a) all-MiniLM-L6-v2 model



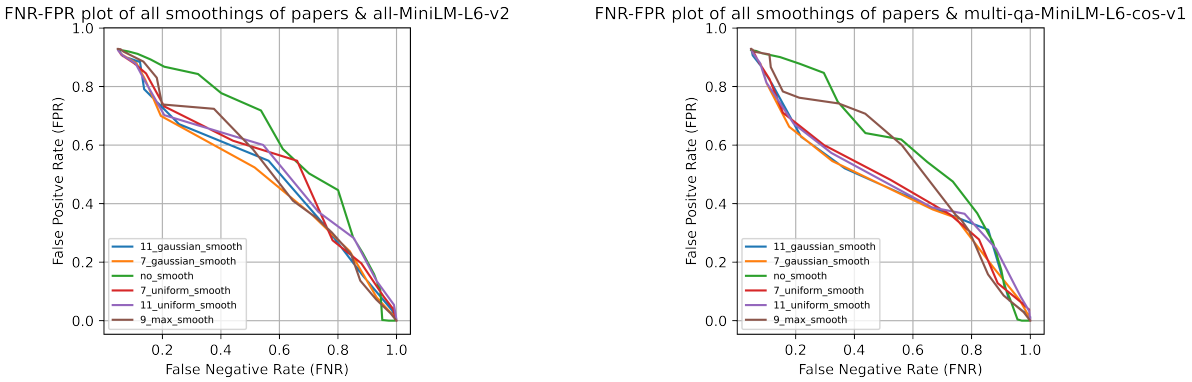
(b) multi-qa-MiniLM-L6-cos-v1 model

Figure 4.11: Comparison of various window size for rolling max smoothing for textbooks

Now, the question is which smoothing works best?

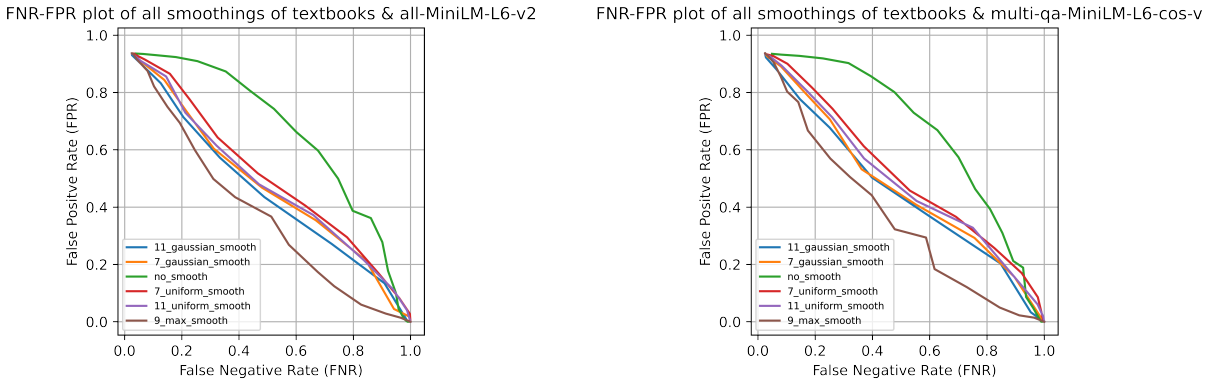
Let's compare all smoothing curves for both papers and textbooks in figure 4.12 and figure 4.13. We can see that 9_max_smooth is best working for textbooks while 7_gaussian_smooth is working best for technical papers. Since we assumed the ground truth for textbooks as whole section that the figure is contained so, the rolling max is working great for textbooks because rolling max will maximize the number of predictions it make in that sections. Hence, we can't accurately consider rolling max as good model for both papers and textbooks. While **7_gaussian_smooth** is working good for both technical papers and textbooks.

Concluding that **7_gaussian_smooth** is the best smoothing that improves the prediction i.e. lowering the FNR and FPR values.



(a) all-MiniLM-L6-v2 model (b) multi-qa-MiniLM-L6-cos-v1 model

Figure 4.12: Comparison of various smoothing for technical papers



(a) all-MiniLM-L6-v2 model (b) multi-qa-MiniLM-L6-cos-v1 model

Figure 4.13: Comparison of various smoothing for textbooks

4.4.2 Various heuristics for threshold

Until now, we are finding optimal threshold from the FNR-FPR plots for predicting a sentence cosine similarity crosses that absolute threshold but we are not considering the noise, i.e. here many other sentence uses same words as relevant sentences, that exist in the text. To remove the noise in the cosine similarity values we introduced six heuristics:

- H_0 : Predict above λ
- H_1 : Predict all text between the first and last sentence above λ
- H_2 : Predict above $\mu + \lambda$
- H_3 : Predict above $\mu + \lambda * \sigma$
- H_4 : Predict above $\mu^* + \lambda$
- H_5 : Predict above $\mu^* + \lambda * \sigma^*$

- H_6 : Predict above $\mu + (\lambda * \mu)$

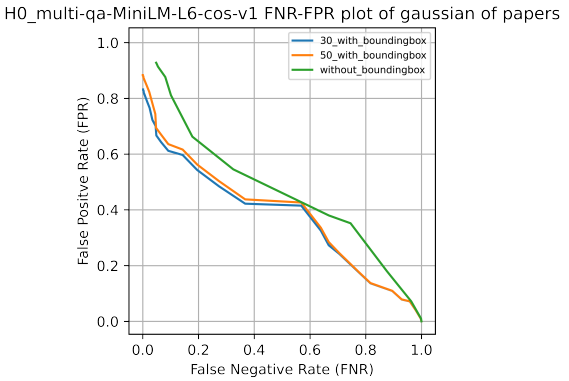
μ, σ : mean, standard deviation of all cosine scores in that file

μ^*, σ^* : mean, standard deviation of cosine scores inside the bounding box

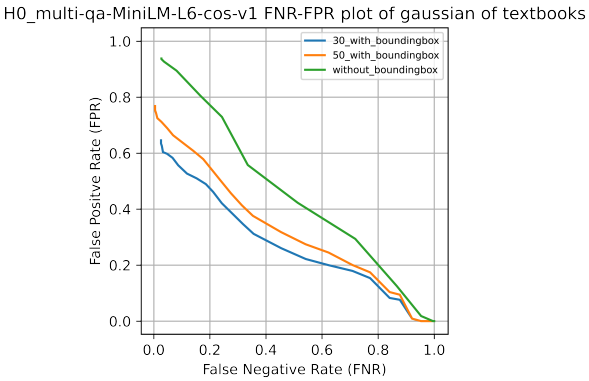
λ : optimal threshold to find

Bounding box

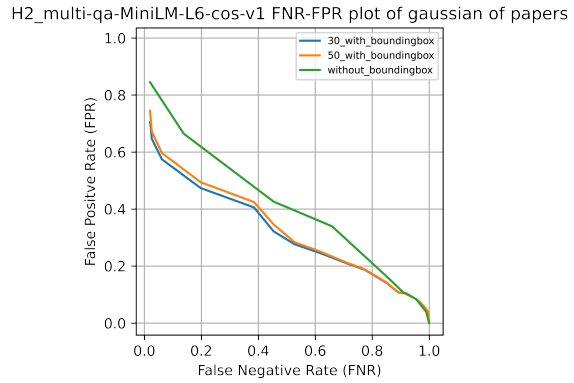
From our experience whenever we encounter a figure, we expect that the relevant text will be within 1-2 pages around the figure. To capture this experience into our code, we will keep a bounding box(BB) around the figure and only within the bounding box we will considered as predicted sentences and sentences outside the bounding box if predicted as relevant by the model are not considered as predicted. We will apply this Bounding box concept to H_0, H_2, H_3, H_4, H_5, and H_6. Now, let's observe that all plots in figure 4.14 that for H_0, H_2, and H_3 shows improvement after the BB got applied compared to no BB. The same trend is observed for other heuristics. BB with width of 60 sentences, i.e. (**30_with_boundingbox**) 30 sentences either side of the figure as bounding box, works best.



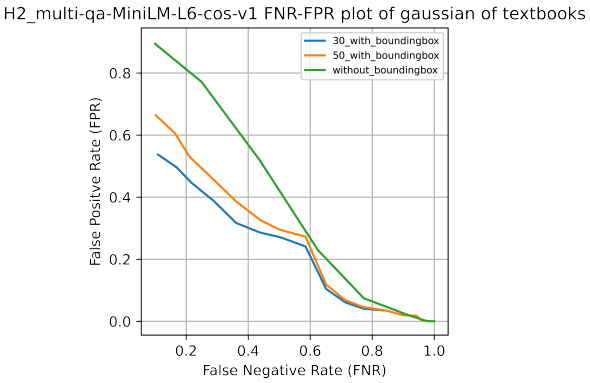
(a) H_0 heuristic for papers



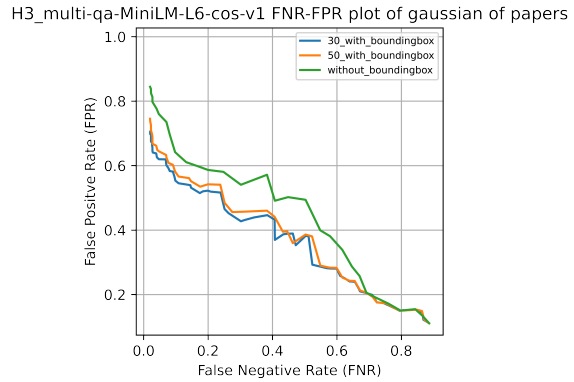
(b) H_0 heuristic for textbooks



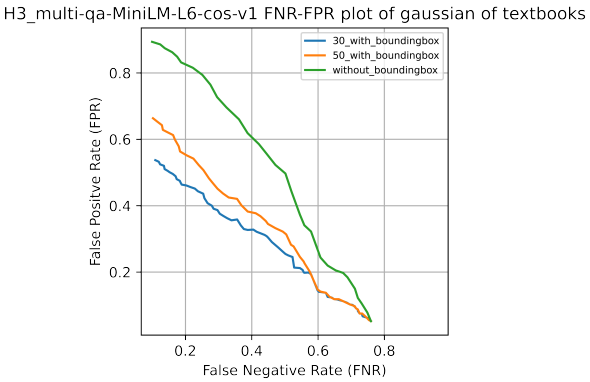
(c) H_2 heuristic for papers



(d) H_2 heuristic for textbooks



(e) H_3 heuristic for papers

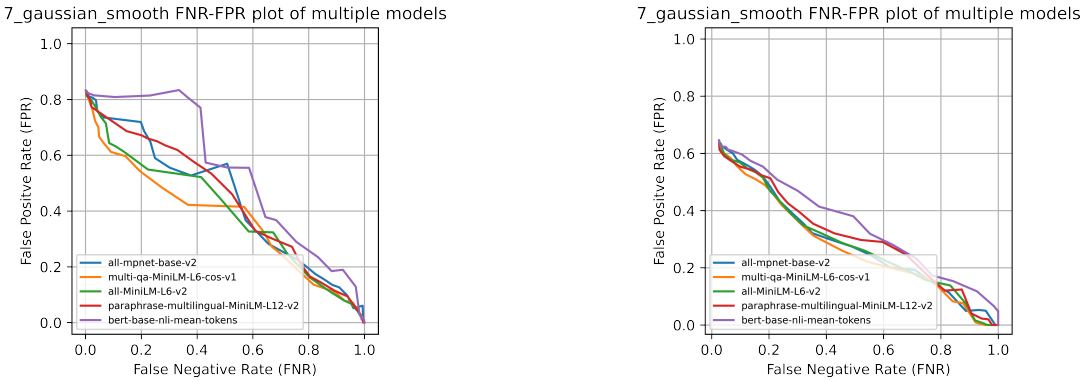


(f) H_3 heuristic for textbooks

Figure 4.14: Comparison of different sized bounding boxes for various heuristics

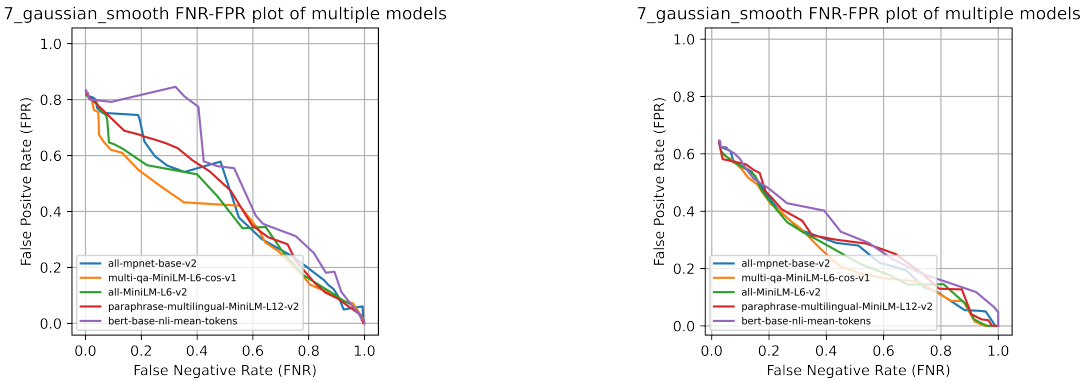
Finding best model

From observing the figures 4.15 to 4.21, we can identify **multi-qa-MiniLM-L6-cos-v1** is doing best for all heuristics.



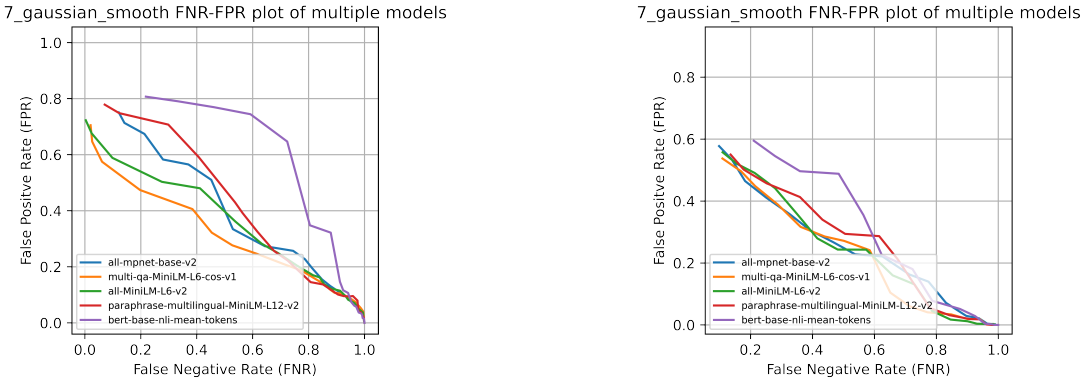
(a) Technical papers (b) Textbooks

Figure 4.15: Comparison of H_0 heuristic for all models with 30 BB



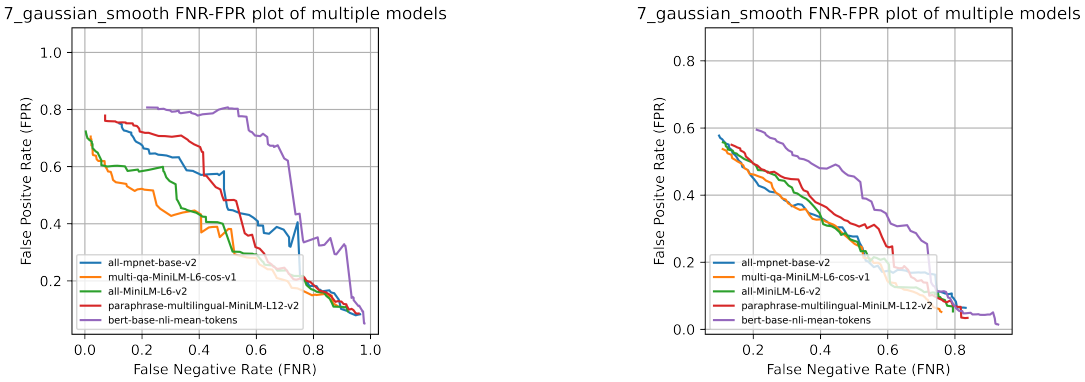
(a) Technical papers (b) Textbooks

Figure 4.16: Comparison of H_1 heuristic for all models with 30 BB



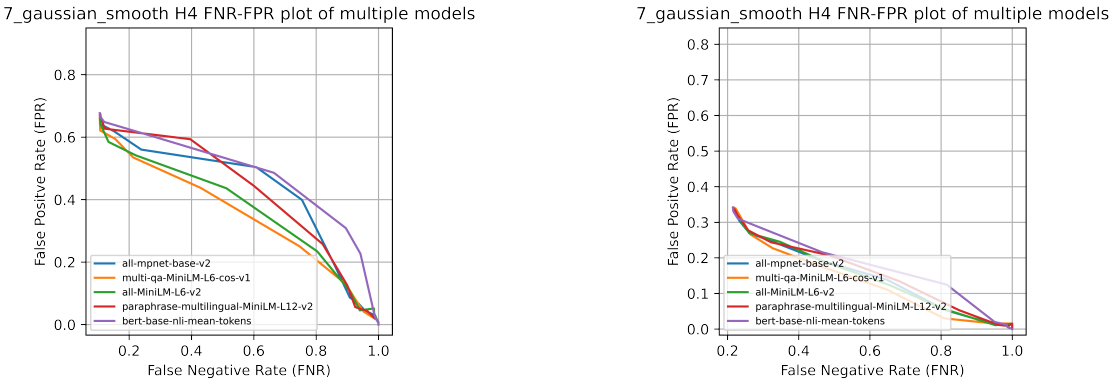
(a) Technical papers (b) Textbooks

Figure 4.17: Comparison of H_2 heuristic for all models with 30 BB



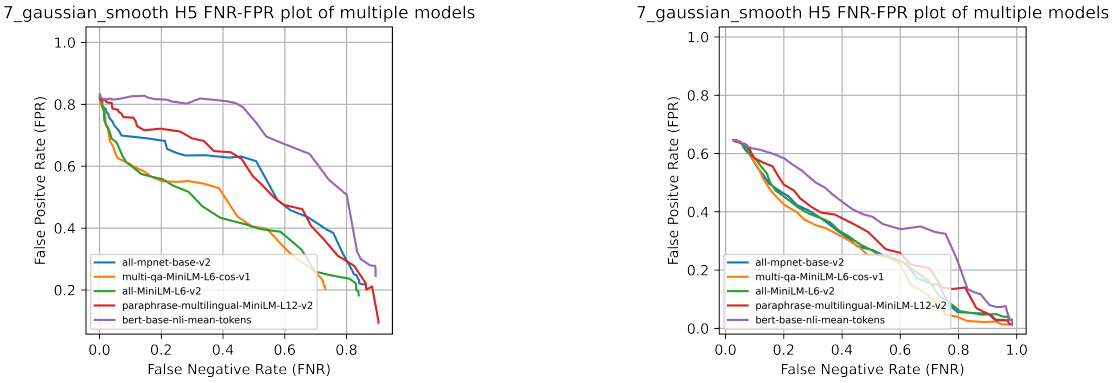
(a) Technical papers (b) Textbooks

Figure 4.18: Comparison of H_3 heuristic for all models with 30 BB



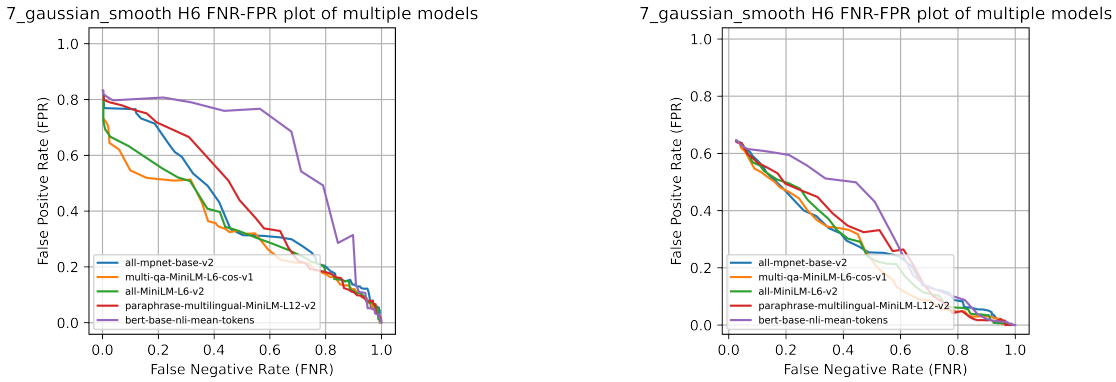
(a) Technical papers (b) Textbooks

Figure 4.19: Comparison of H_4 heuristic for all models with 30 BB



(a) Technical papers (b) Textbooks

Figure 4.20: Comparison of H_5 heuristic for all models with 30 BB

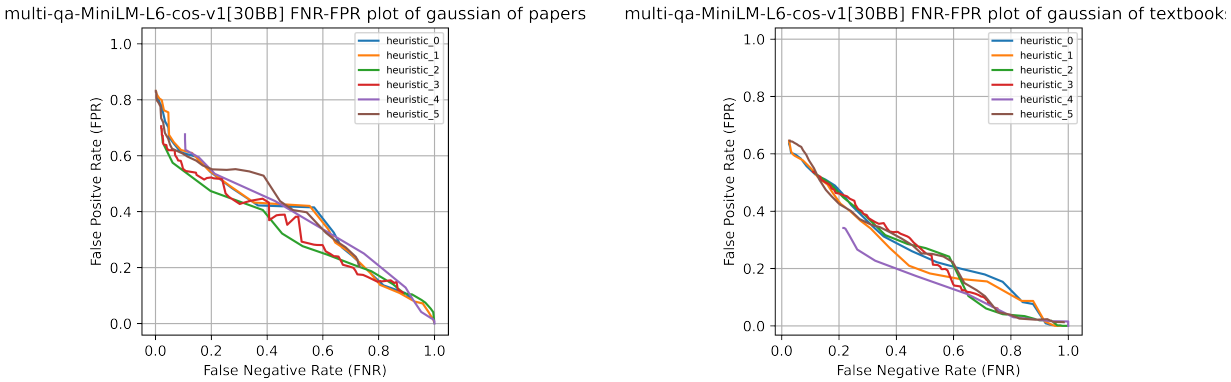


(a) Technical papers (b) Textbooks

Figure 4.21: Comparison of H_6 heuristic for all models with 30 BB

Finding optimal heuristic

Since H_6 is just a variation of the H_2, let's compare H_0, H_1, H_2, H_3, H_4, and H_5 in figure . We can see that H_2 is doing good in both technical papers and textbooks.



(a) Technical papers (b) Textbooks

Figure 4.22: Comparison of various heuristic for multi-qa-MiniLM-L6-cos-v1 models with 30 BB

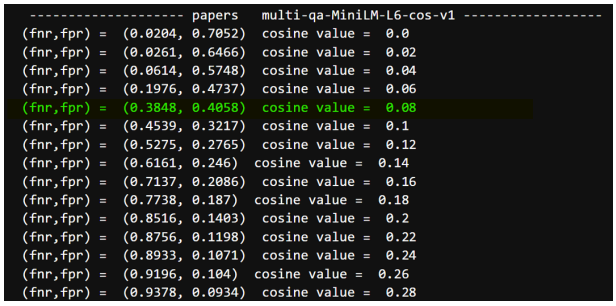
4.4.3 Conclusions

Let's recap the generalized results for both papers and textbooks we got until now:

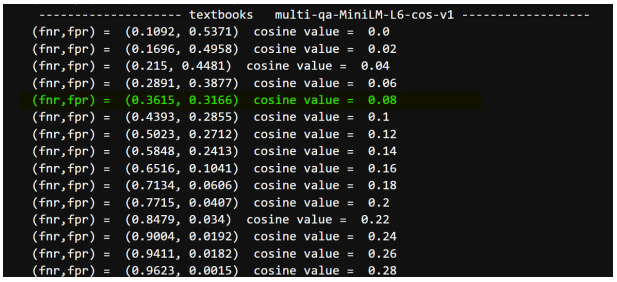
- Best smoothing is 7_gaussian_smooth
- 30 sentences on either side of figure as BB works best i.e. 30_with_boundingbox
- Best model for all heuristics is multi-qa-MiniLM-L6-cos-v1
- Best Heuristics are H_2 and H_6

Optimal threshold

Optimal threshold for H_2 i.e $\lambda = 0.08$ from figure 4.23 hence, we predict sentences as relevant sentences if the cosine score crosses $\mu + 0.08$ where μ is mean of cosine scores of sentences in whole file.



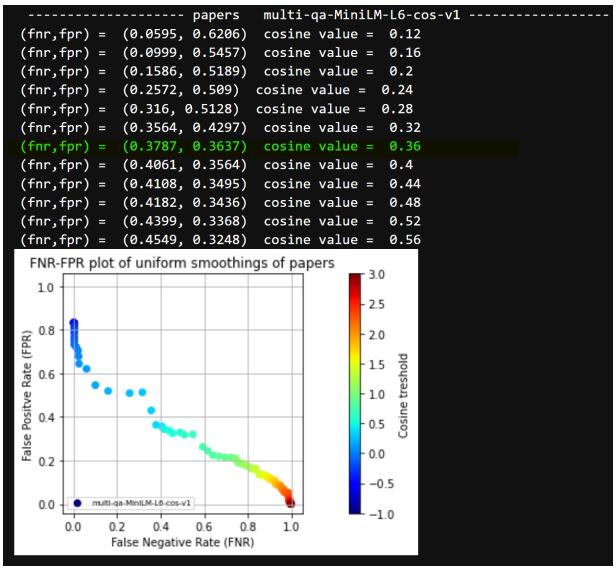
(a) Technical papers



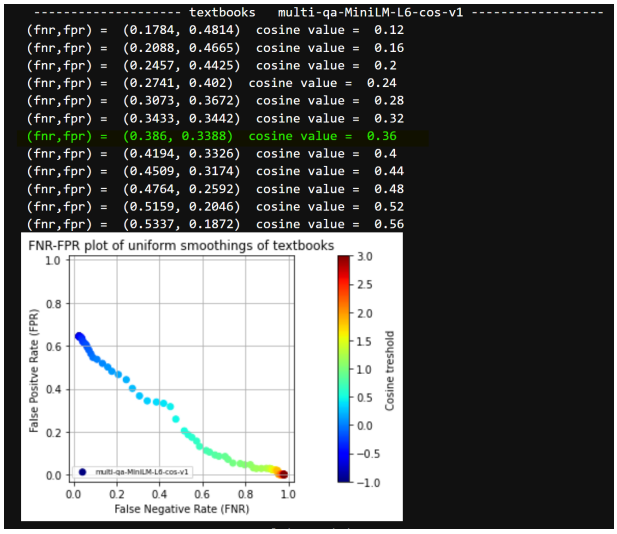
(b) Textbooks

Figure 4.23: FNR-FPR values for H_2 heuristic at each thresholds

Optimal threshold for H_6 i.e $\lambda = 0.36$ from figure 4.24 hence, we predict sentences as relevant sentences if the cosine score crosses $\mu + 0.36 * \mu$ where μ is mean of cosine scores of sentences in whole file.



(a) Technical papers



(b) Textbooks

Figure 4.24: FNR-FPR values for H_6 at each thresholds

Chapter 5

Generating summary from relevant text

In previous chapter we extracted relevant text from nearby text so, currently we have caption, corresponding figure, labels & their positions in the figure, and relevant sentences to the labels appended caption sentence. Now, let's try to get a description of the image from the information available to us.

5.1 Summarization

Summarization is the task of producing a shorter version of a document while preserving its important information. Some models can extract text from the original input, while other models can generate entirely new text. Transformer models can be used to condense long documents into summaries, a task known as text summarization. This is one of the most challenging NLP tasks as it requires a range of abilities, such as understanding long passages and generating coherent text that captures the main topics in a document. However, when done well, text summarization is a powerful tool that can speed up various business processes by relieving the burden of domain experts to read long documents in detail. Although there are multiple pre-trained summarization models, we tried particularly one model sshleifer/distilbart-cnn 12-6

We used this summarization tool to summarize the extracted information i.e caption + labels in the figure + extracted relevant sentences and caption + extracted relevant sentences.

5.2 Few results

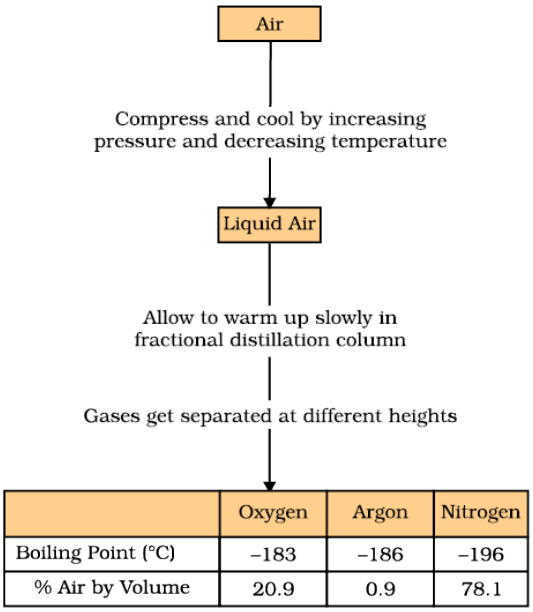
All results of summarizing are written into html files, these are few examples of the summarization tool.

5.2.1 Example 1

Input

figure 2.11: flow diagram shows the process of obtaining gases from air. this figure contains air, compress and cool by increasing pressure and decreasing temperature, liquid air, allow

to warm up slowly in fractional distillation column, gases get separated at different heights, oxygen, argon, nitrogen, 183, 186, 196, boiling point (c) % air by volume, 20.9, 0.9, 78.1.



	Oxygen	Argon	Nitrogen
Boiling Point (°C)	−183	−186	−196
% Air by Volume	20.9	0.9	78.1

Figure 5.1: figure for example 1

Output with labels included

The method is called distillation . It is used for the separation of components of a mixture containing two miscible liquids . The two components separate . At what temperature does the thermometer reading become constant for some time .

Output without labels

The method is called distillation . It is used for the separation of components of a mixture containing two miscible liquids that boil without decomposition and have sufficient difference in their boiling points . Now answer: What do you observe as you start heating the mixture .

5.2.2 Example 2

Input

figure 2.5: evaporation. this figure contains vapours ink, watch glass, beaker water.

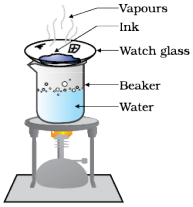


Figure 5.2: figure for example 2

Output with labels included

Figure 2.5: Evaporation. this figure contains vapours ink, watch glass, beaker water . You will see that evaporation is taking place from the watch glass . What do you think has got evaporated? Is ink a single substance (pure) or is it a mixture?

Output without labels

Fill half a beaker with water . Put a watch glass on the mouth of the beaker (Figure 2.5) Put a few drops of ink on the watch glass . Now start heating the ink . You will see that evaporation is taking place . Now answer: Is ink a single substance (pure) or is it a mixture?

5.2.3 Example 3

Input

figure 4.2: scattering of -particles by a gold foil. this figure contains gold atoms, a particles.

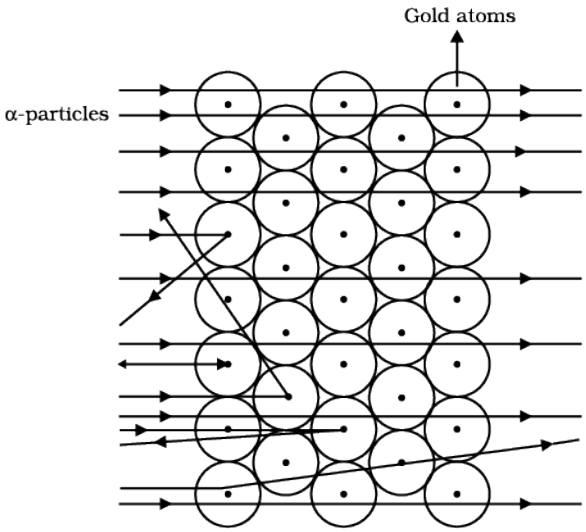


Figure 5.3: figure for example 2

Output with labels included

Ernest Rutherford was interested in knowing how the electrons are arranged within an atom . Rutherford designed an experiment for this . Fast moving alpha ( )-particles were made to fall on a thin gold foil . He selected a gold foil because he wanted as thin a layer as possible .

Output without labels

Ernest Rutherford was interested in knowing how the electrons are arranged within an atom . Rutherford designed an experiment for this . In this experiment, fast moving alpha ( )-particles were made to fall on a thin gold foil . He selected a gold foil because he wanted as thin a layer as possible .

Chapter 6

Examples

In the code, we generated colour coded HTML as output so that it would be easy to revisit the results. The output contains figure, labels appended caption, cosine scores with predicted & actual relevant sentences, summary with labels, summary without labels, and all colour coded sentences in a file. More examples are in the *relevant text extraction* github repository.

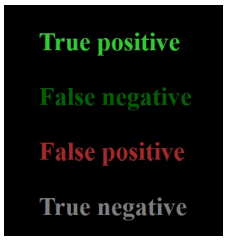


Figure 6.1: Colour code of sentences for output in HTML

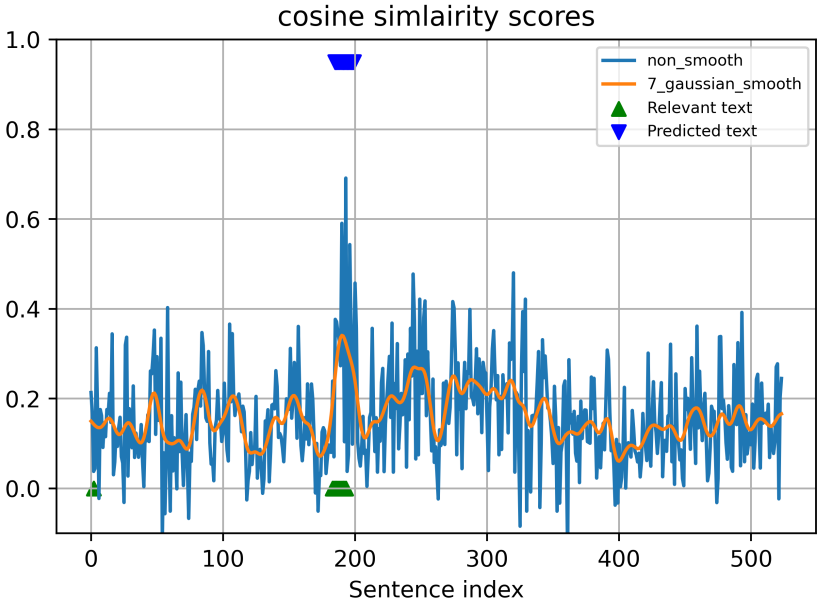


Figure 6.2: Example of a cosine scores with predicted & actual relevant sentences

6.1 Example 1

Expected output

2.3 Separating the Components of a Mixture

We have learnt that most of the natural substances are not chemically pure. Different methods of separation are used to get individual components from a mixture. Separation makes it possible to study and use the individual components of a mixture.

Heterogeneous mixtures can be separated into their respective constituents by simple physical methods like handpicking, sieving, filtration that we use in our day-to-day life. Sometimes special techniques have to be used for the separation of the components of a mixture.

2.3.1 How can we obtain coloured component (dye) from blue/black ink?

Activity 2.4

- Fill half a beaker with water.
- Put a watch glass on the mouth of the beaker (Fig. 2.5).
- Put few drops of ink on the watch glass.
- Now start heating the beaker. We do not want to heat the ink directly. You will see that evaporation is taking place from the watch glass.
- Continue heating as the evaporation goes on and stop heating when you do not see any further change on the watch glass.
- Observe carefully and record your observations.

Fig. 2.5: Evaporation

Is Matter Around Us Pure?

Now answer

- What do you think has got evaporated from the watch glass?
- Is there a residue on the watch glass?
- What is your interpretation? Is ink a single substance (pure) or is it a mixture?

Figure 6.3: Actual relevant text from the textbook

Figure

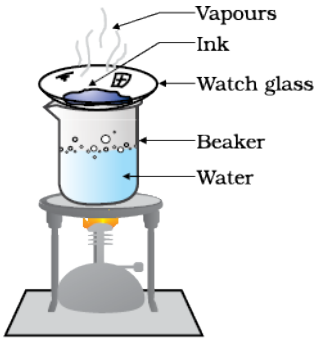


Figure 6.4: Figure extracted from HTML

Extracted caption

Figure 2.5: evaporation

Extracted labels

- Vapours ink
- watch glass
- beaker
- water

Labels appended caption

Figure 2.5: evaporation. this figure contains vapours ink, watch glass, beaker, water.

Cosine scores of labels appended caption with rest of the sentences

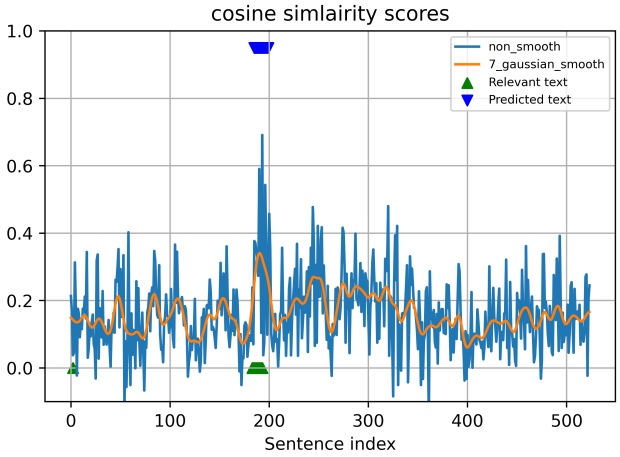


Figure 6.5: Cosine scores with predicted & actual relevant sentences

Relevant sentences

180: Separation makes it possible to study and use the individual components of a mixture.

181: Heterogeneous mixtures can be separated into their respective constituents by simple physical methods like handpicking, sieving, filtration that we use in our day-to-day life

182: Sometimes special techniques have to be used for the separation of the components of a mixture.

183: 2.3.1 How can we obtain coloured component (dye) from blue/black ink

184: Activity 2.4

185: Fill half a beaker with water.

186: Put a watch glass on the mouth of the beaker (Figure 2.5).

187: Put few drops of ink on the watch glass.

188: Now start heating the beaker

189: We do not want to heat the ink directly

190: You will see that evaporation is taking place from the watch glass.

191: Continue heating as the evaporation goes on and stop heating when you do not see any further change on the watch glass.

192: Observe carefully and record your observations.

193: Figure 2.5: Evaporation

194: Is Matter Around Us Pure

195: Now answer

196: What do you think has got evaporated from the watch glass

197: Is there a residue on the watch glass

198: What is your interpretation

199: Is ink a single substance (pure) or is it a mixture

200: We find that ink is a mixture of a dye in water

201: Thus, we can separate the volatile component (solvent) from its non-volatile solute by the method of evaporation.

Figure 6.6: Colour coded sentences

Summary with labels

Figure 2.5: Evaporation. this figure contains vapours ink, watch glass, beaker water . You will see that evaporation is taking place from the watch glass . What do you think has got evaporated? Is ink a single substance (pure) or is it a mixture?

Summary without labels

Fill half a beaker with water . Put a watch glass on the mouth of the beaker (Figure 2.5) Put a few drops of ink on the watch glass . Now start heating the ink . You will see that evaporation is taking place . Now answer: Is ink a single substance (pure) or is it a mixture?

Chapter 7

Future work

Summarization tool

One can try various other models from HuggingFace and also fine-tune to use Summarization tool in our use case and improve the figure description.

Complete template

As template described in Chahal et al., 2018, we extracted most of the information needed but we didn't use position of the labels and figure itself. So, one can use our extracted information and understanding of the figure to give a better description of the figure.

ClipCap

It is a SOTA Image Captioning tool initially introduced in Mokady et al., 2021. Image Captioning is the ability of a machine to generate a natural description of an image. Indeed, it is almost as difficult as the machine needs to understand the image and the text it generates, just like in text-to-image synthesis. It's easy to simply tag the objects you see in the image. This can be done using a classic classifier model. But it is quite another challenge to understand what's happening in a single 2-dimensional picture. Humans can do it quite easily since we can interpolate from our past experience, and we can even put ourselves in the place of the person in the picture and quickly get what's going on. This is a whole other challenge for a machine that only sees pixels. But, for our use case we can use fine-tuned version ClipCap which will expected to work very well.

Bibliography

J. Briggs. Natural language processing (nlp) for semantic search. 2022. URL https://www.pinecone.io/learn/sentence-embeddings/.

BS4. Beautiful soup. version 4.10.0. URL https://www.crummy.com/software/BeautifulSoup/bs4/doc/.

A. Chahal, M. Belani, A. Bansal, N. Jadhav, and M. Balakrishnan. Template based approach for augmenting image descriptions. In *International Conference on Computers Helping People with Special Needs*, pages 104–112. Springer, 2018.

HuggingFace. Ai community. URL https://huggingface.co/.

R. Mokady, A. Hertz, and A. H. Bermano. Clipcap: Clip prefix for image captioning. *arXiv preprint arXiv:2111.09734*, 2021.

S. pre-trained models. Ai community. URL https://www.sbert.net/docs/pretrained_models.html.

Pytesseract. Pytesseract. version 0.3.9. URL https://pypi.org/project/pytesseract/.

N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

relevant text extraction. Iitd assistech. URL https://github.com/assistech-iitdelhi/relevant_text_extraction.

sshleifer/distilbart-cnn 12-6. summarization tool. URL https://huggingface.co/sshleifer/distilbart-cnn-12-6.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. 2017. URL https://arxiv.org/pdf/1706.03762.pdf.