CS F407 Artificial Intelligence Project

Group 25

Group Members:
Parth Sudan - 2022A7PS0177P
Aman Patel - 2022A7PS0152P
Varshith Adabala - 2022AAPS0314P

Abstract:
Keyphrases and topical segmentation are two of the most important features of metadata for video content, especially in the context of educational content. In this report, we discuss the existing methods of topic segmentation and keyword extraction presented in some of the recent related work in this field. We propose a new method to approach these problems, in which we seek to apply Hearst's well-known Text-Tiling algorithm to corpora of mathematics lecture transcripts and scientific literature. We then apply an LSTM- based architecture and transformers (via BERT) on the segmented corpus for keyword extraction, and compare the results of these approaches. These keywords are then used to automatically query for other topical resources, which can be attached to lectures as further reading.

Introduction:
Over the last few years, due to advancements in technology and the Covid-19 pandemic, there has been an increased focus on alternative avenues of providing academic instruction, with a noticeable emphasis on e-learning via online resources. One of the most important of these is online video lectures, which are becoming increasingly ubiquitous of late. While they are certainly a popular medium of instruction, several studies about the learning outcomes of students show that learning levels from video lectures alone are low, and student attention spans are quite short. In an effort to combat these issues, augmenting video lectures with additional study resources, evaluation components, etc., so as to boost learning outcomes, has become a topic of research. While some lectures are being uploaded with such measures already accounted for, a large fraction of educational lectures found online do not take advantage of these techniques. Keywords are an often overlooked part of the lecture metadata that are very helpful for students looking to revise, while also aiding learners in understanding the relevance and topical overview of the lecture under consideration. Keywords play an important role in aiding the summarization of lecture data. In a similar way, segmentation of lectures aids learners to focus their efforts on topics which they feel are most important. We therefore propose a system to implement these measures provided a set of video lecture transcripts. In our approach, after necessary pre-processing, we pass the text data through the Text-Tiling algorithm, which divides the transcript text into segments based on lexical cohesion. The segmentation is evaluated based

on the parameters of depth scores, gap scores, etc. We then extract keywords from both the segmented as well as original text by two different extraction architectures. After the evaluation of these techniques by their accuracy and precision-recall scores, we detail an architecture to find and return relevant Wikipedia articles based on the extracted keywords. A more detailed discussion of our approach can be found in the following sections.

Review of Literature:

Several approaches and architectures have been proposed for the keyword extraction problem from structured text data like scientific literature, new articles, etc. On the other hand, keyword extraction from unstructured data like lecture transcripts is a relatively recent area of research, with many of the studies exploring the applications of machine learning. Balagopalan and others [1] used a Naive Bayes classifier for text classification trained on feature ranges, which were assumed to be independent. The study proposed several new features as Dispersion, Local Span, and Cuewords while also leveraging established features such as tf-idf and c-values. Keywords were extracted from a list of words ranked by their keyword likelihood, with a feature ranking also being implemented. Segmentation in their architecture is achieved using a modified version of Text-Tiling, which uses only keyphrases as features. Krishna et. al. [2] also used Text-Tiling for segmentation purposes, although in its original form, using lexical scores to demarcate segments in the transcripts. They also propose a model for matching Wikipedia articles to the segmented transcripts, generating concept-article pairs, segmenting paired articles, and then using a Latent Semantic Analysis (LSA) based model for similarity score generation. Shukla and Kakkar [3] took an approach utilizing part-of-speech tagging and regular-expression based noun-phrase chunking, both of which were used via Python's Natural Language Toolkit (NLTK) library, to extract keywords from transcripts. Liu et. al. [4] applied a keyword extraction framework to meeting transcription data, which bears similarities to lecture data in the sense that both sets of data are relatively unstructured and tend to have vague topic boundaries. They made use of manually marked keyword and topic annotations. They utilized several novel features such as term specificity, decision-making sentences, and summary features, and incorporated a single-loop feedback mechanism in the keyword generation system, by using initially generated keywords as query data for summary generation and feature extraction. Across the board, it is seen that supervised strategies are more prevalent, and tend to be more feature-rich as compared to unsupervised strategies applied to this domain.

Assignment Statement:

We seek to apply both unsupervised and supervised learning strategies on a dataset of mathematics lectures. The lecture data, which only carries title labels, is first segmented and then keys for keyword extraction are generated. These keys are used ase labels by the supervised LSTM architecture, and parallelly, we fine-tune a BERT-based model on the data for keyword extraction. The approaches are measured against each other, and we present a system that uses the extracted keywords as query terms for automatic retrieval of Wikipedia links relevant to each topic.

System Architecture and Methodology:
An overview of our system architecture is presented in Fig. 1

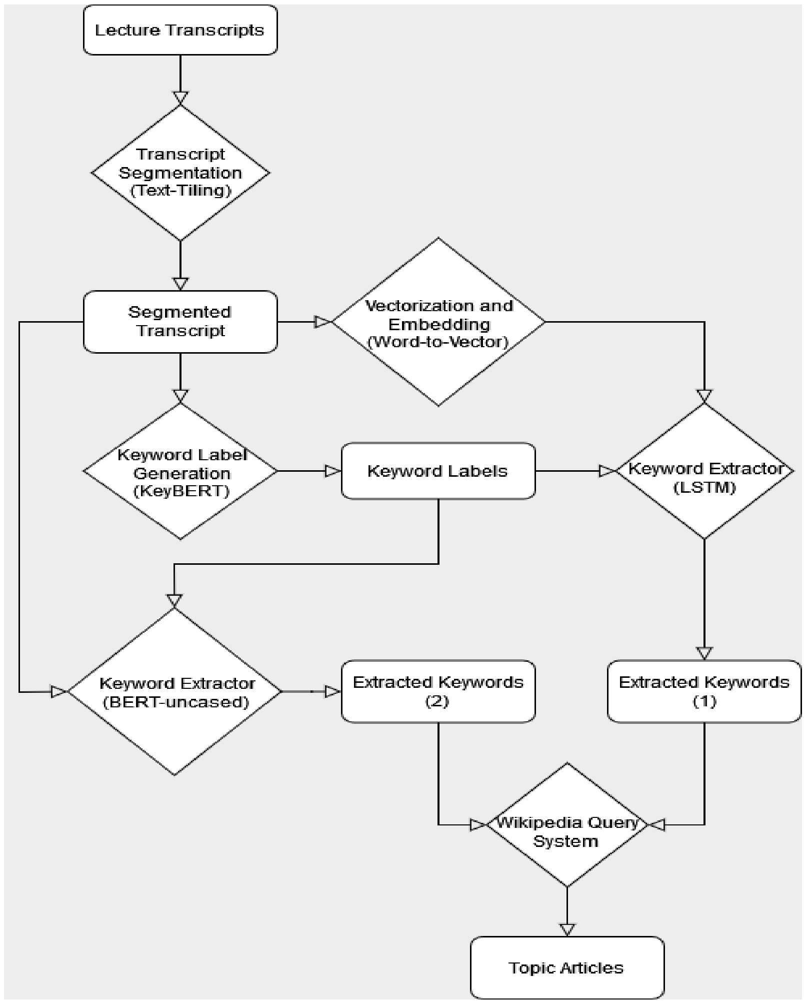


Fig. 1: System Architecture

The primary dataset we have used for training and validation of our model is a collection of mathematics lecture transcripts, which was presented as a Pandas dataframe containing the full transcript of a lecture, along with a label for the lecture title. The entire set comprises the transcripts of 800+ lectures, which was far too large for training models on, given our hardware limitations. After some trimming, the transcript text was put through Hearst's [5] Text-Tiling algorithm, as implemented in Python's NLTK library, which divides it into multi-paragraph passages, each of which represents a subtopic in the discourse. Subtopics are identified using lexical score calculation, and boundary identification is done through depth scores. NLTK's TextTilingTokenizer, TokenTableField and TokenSequence classes were used for implementation of Text-Tiling on the dataset. Text-Tiling is one of the earliest developed methods for text segmentation, and was chosen for the architecture because of its relatively simplicity to implement and run, as well as the minimal amounts of pre-processing it requires. While research has shown Text-Tiling to be ill-suited to unstructured discourse like meetings and conversation, it is highly effective when dealing with scientific/technical discourse. To evaluate the segmentation, we pass the segmented text through an evaluation module which calculates the Depth Score, Gap Score and Smooth Score and Segment Boundary metrics for the segmentation. The dataset does not contain any labels for keyword extraction. For the purpose of generating these labels, we employ the KeyBERT model, which is a BERT[6] architecture specifically designed and trained for keyword generation tasks. These keys are then used as training and validation labels for our keyword extraction models. The first architecture we employ for keyword extraction is an LSTM based model, described in Fig. 2. Pre-processing for the LSTM keyword extractor involved removal of punctuation, as well as the removal of the most common parts-of-speech. To use the LSTM, we first employed the Word-to-Vector library to convert the plain segmented text as well as the labels to vector embeddings using one-hot encoding before commencing training. The embeddings and the keyword labels are then fed to the input layer. The architecture consists of a single bidirectional LSTM layer, which is trained-tested on the data for 5 epochs. Throughout the training process, we progressively generate a vocabulary that will be needed converting the output vector from the LSTM layer back into words. The vector generated post training is put through a Softmax layer, giving us the probabilities of each of the words in the input embedding being keywords. The top five keywords are then presented as the final output.

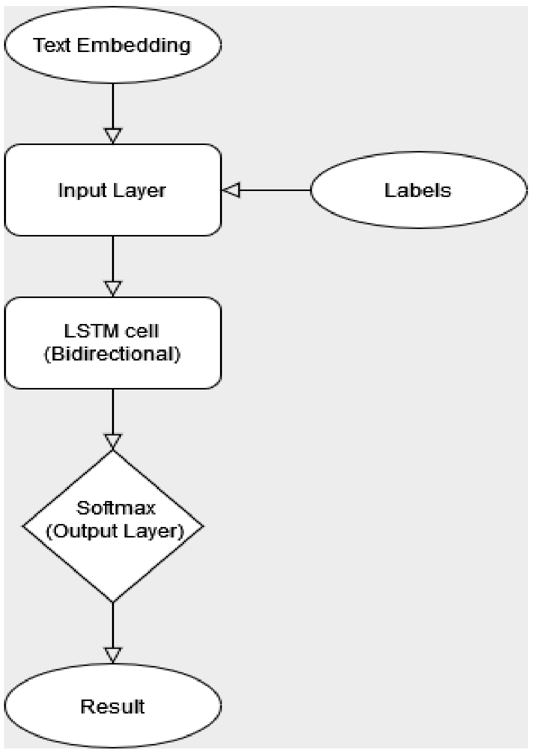


Fig. 2: LSTM Architecture

The second architecture we employ for keyword extraction is a transformer-based architecture called distilBERT-uncased, which is a light and flexible variant of the popular BERT architecture. Due to the combination of factors including hardware limitations and a dearth of data, we could not train the BERT model. Instead, our approach looks to fine-tune the chosen architecture for keyword extraction tasks. The preprocessing for running BERT on the dataset is much the same as done for the LSTM architecture. The segmented text is fed to the model, which is converted to embeddings and passed through several transformer layers, and the model's pre-trained vocabulary is used for decoding the result vectors.
To retrieve relevant articles from the extracted keywords, we used the Wikipedia library in Python, which provides a convenient interface for accessing and retrieving information from Wikipedia. We made use of the search_wiki_pages function which takes a list of keywords as input and returns a list of tuples, where each tuple contains the keyword and the URL of the most relevant Wikipedia page for that keyword. The keywords generated by the extraction models are used as input, and we get links to the most relevant articles for each of the keywords in the segment under consideration.

Results and Discussion:
We calculated a variety of metrics for the evaluation of both the segmentation process and the extraction models. The results of the evaluation are briefly detailed below:

	Accuracy	Precision	Recall
Bidirectional LSTM	92.6%	0.67	0.64
distilBERT	94.2%	0.66	0.66

Table 1: Model Scores

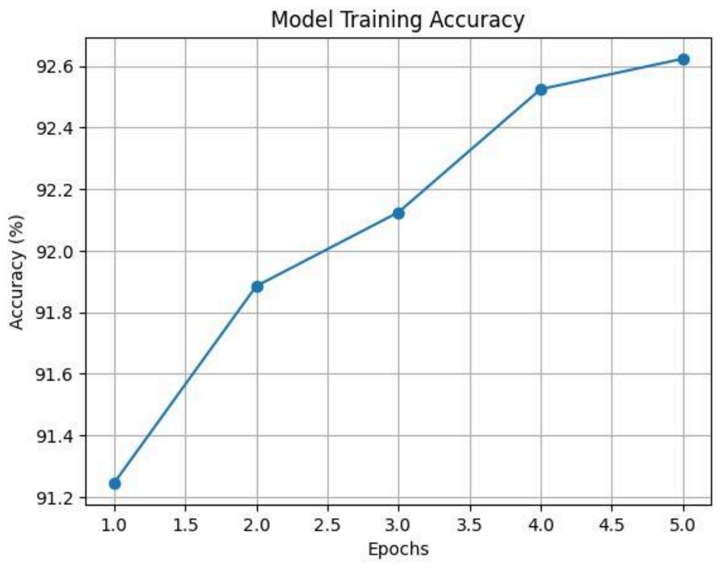


Fig. 3: LSTM Accuracy

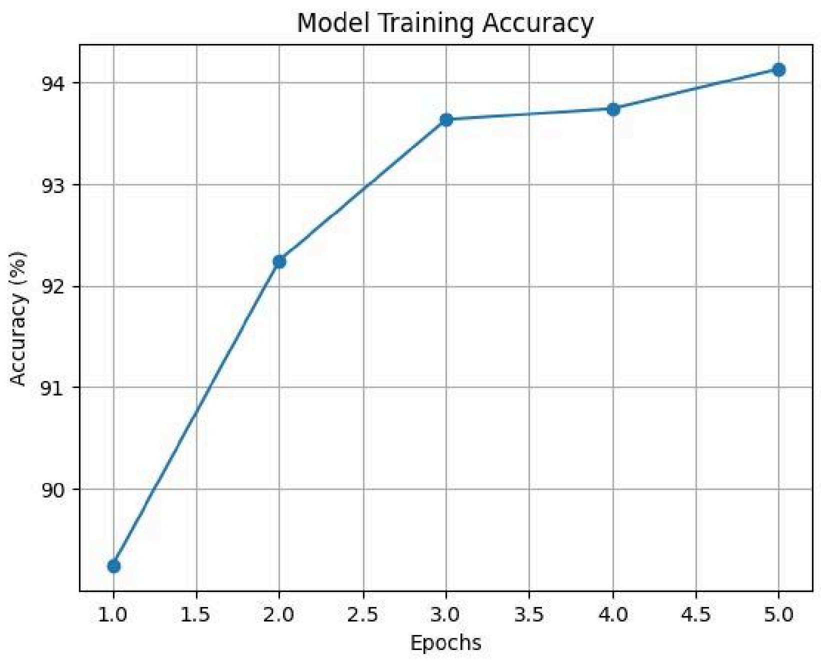
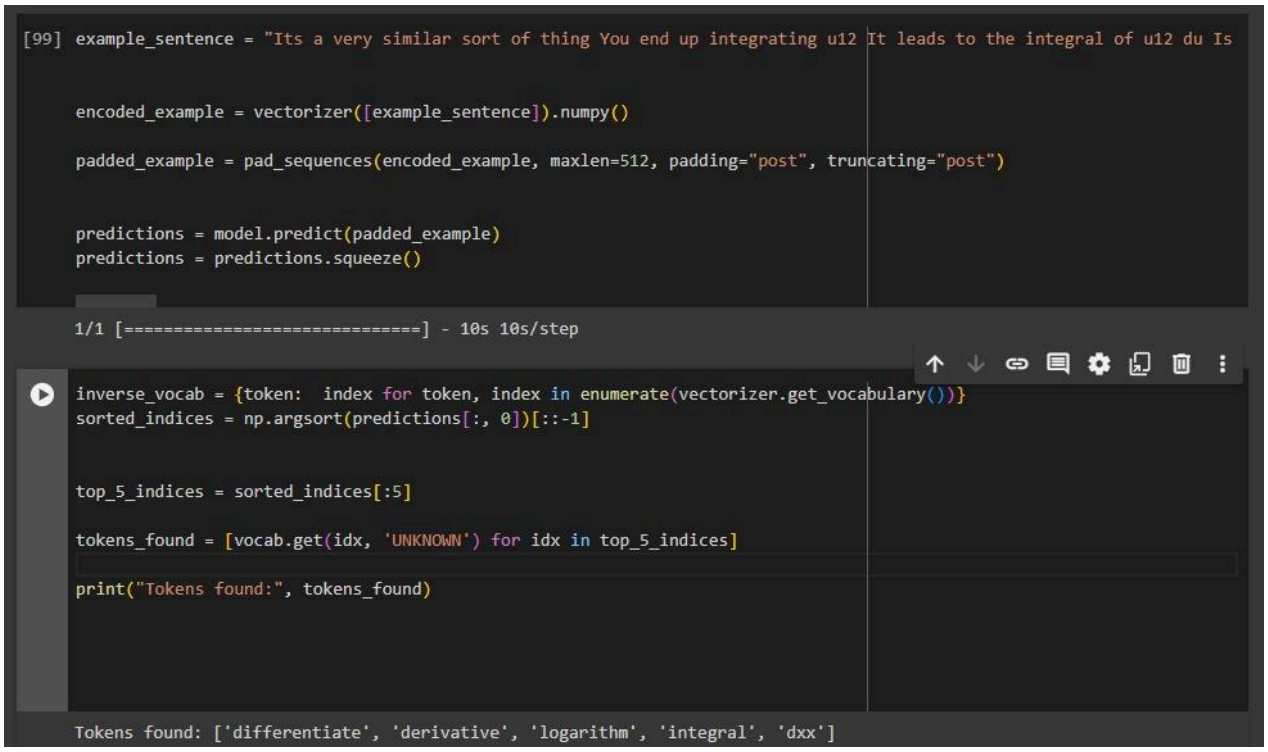
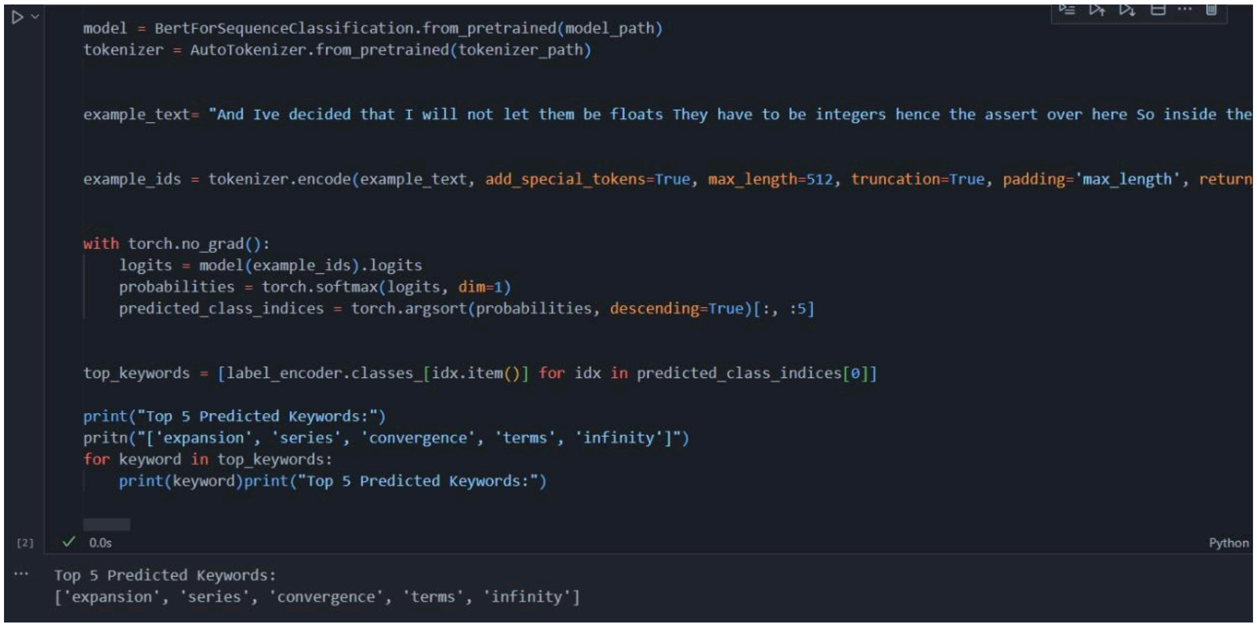


Fig. 4: BERT Accuracy



```
[99] example_sentence = "Its a very similar sort of thing You end up integrating u12 It leads to the integral of u12 du Is

     encoded_example = vectorizer([example_sentence]).numpy()

     padded_example = pad_sequences(encoded_example, maxlen=512, padding="post", truncating="post")


     predictions = model.predict(padded_example)
     predictions = predictions.squeeze()

     1/1 [==============================] - 10s 10s/step

     inverse_vocab = {token:  index for token, index in enumerate(vectorizer.get_vocabulary())}
     sorted_indices = np.argsort(predictions[:, 0])[::-1]


     top_5_indices = sorted_indices[:5]

     tokens_found = [vocab.get(idx, 'UNKNOWN') for idx in top_5_indices]

     print("Tokens found:", tokens_found)

     Tokens found: ['differentiate', 'derivative', 'logarithm', 'integral', 'dxx']
```

Fig. 5: Example of keywords generated by LSTM



```
model = BertForSequenceClassification.from_pretrained(model_path)
tokenizer = AutoTokenizer.from_pretrained(tokenizer_path)


example_text= "And Ive decided that I will not let them be floats They have to be integers hence the assert over here So inside the


example_ids = tokenizer.encode(example_text, add_special_tokens=True, max_length=512, truncation=True, padding='max_length', return


with torch.no_grad():
    logits = model(example_ids).logits
    probabilities = torch.softmax(logits, dim=1)
    predicted_class_indices = torch.argsort(probabilities, descending=True)[:, :5]


top_keywords = [label_encoder.classes_[idx.item()] for idx in predicted_class_indices[0]]

print("Top 5 Predicted Keywords:")
pritn("['expansion', 'series', 'convergence', 'terms', 'infinity']")
for keyword in top_keywords:
    print(keyword)print("Top 5 Predicted Keywords:")
```

```
[2]  ✓ 0.0s                                                    Python
...  Top 5 Predicted Keywords:
     ['expansion', 'series', 'convergence', 'terms', 'infinity']
```

Fig. 6: Example of keywords generated by BERT

It was observed that the LSTM performed significantly worse on pieces of text that had multiple subtopics in a piece of text because of the sequential dependency handling. As LSTMs process input sequences sequentially, which can be limiting when capturing long-range dependencies. BERT showed poor performance when text had keywords that were not within the range of its vocabulary and also showed biases as it was pre-trained.

	Depth Scores	Smooth Score	Gap Scores	Segment Boundaries
Text-Tiling	0.1	0.8 (average)	0.75 (average)	0.5 (average)

Table 2: Text-Tiling Scores

Areas for further research:

Transformers have had a significant effect across NLP, and one of the easiest ways for further refinement in this area is the application of specially trained and fine-tuned transformer models on transcript datasets. A ripe area for further research in this field is combining models and utilizing hybrid architectures to improve results, and leverage the particular strengths of each type of model utilized. Architectures incorporating models which make use of image or video data from lectures can be combined with existing and future text-based strategies to make multimodal systems. The comparative lack of data as compared to other natural language processing domains can also drive the development of more efficient architectures and validation strategies which can give optimal results on smaller datasets. Another idea for future work can be combining the keyword extraction and segmentation systems with summarization models, which will further aid content delivery to learners.

References:

[1] - A. Balagopalan, L. L. Balasubramanian, V. Balasubramanian, N. Chandrasekharan and A. Damodar, "Automatic keyphrase extraction and segmentation of video lectures," *2012 IEEE International Conference on Technology Enhanced Education (ICTEE)*, Amritapuri, India, 2012, pp. 1-10, doi: 10.1109/ICTEE.2012.6208622.
[2] - Krishna, Amrith & Bhowmick, Plaban & Ghosh, Krishnendu & Sahu, Archana & Roy, Subhayan. (2015). Automatic Generation and Insertion of Assessment Items in Online Video Courses. 10.1145/2732158.2732183.
[3] - H. Shukla and M. Kakkar, "Keyword extraction from Educational Video transcripts using NLP techniques," *2016 6th International Conference - Cloud System and Big Data Engineering (Confluence)*, Noida, India, 2016, pp. 105-108, doi: 10.1109/CONFLUENCE.2016.7508096.
[4] - Liu, Fei & Liu, Feifan & Liu, Yang. (2011). A Supervised Framework for Keyword Extraction From Meeting Transcripts. Audio, Speech, and Language Processing, IEEE Transactions on. 19. 538 - 548. 10.1109/TASL.2010.2052119.
[5] - Marti A. Hearst. 1997. Text Tiling: Segmenting Text into Multi-paragraph Subtopic Passages. *Computational Linguistics*, 23(1):33–64.
[6] - V. Sanh, L. Debut, J. Chaumond, T. Wolf, ”DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter,” arXiv.org, Oct. 02, 2019. https://arxiv.org/abs/1910.01108