IT - 594 Deep Neural Natural Language Processing Project Proposal

Pathik Patel [202003002] **Divya Patel [202001420]**
Kunj Patel [202001421] **Suyash Vyas [202001002]**

1. Dataset

Link - scientific_lay_summarisation-elife-norm · Datasets at Hugging Face

The **scientific_lay_summarisation-elife-norm** dataset is a collection of biomedical articles paired with non-technical lay summaries. This dataset is hosted on Hugging Face's Datasets Hub. Here are some key details about this dataset:

1. **Task**: The primary task associated with this dataset is text summarization. More specifically, it's used for scientific lay summarization, which involves generating non-technical summaries of scientific literature.
2. **Languages**: The dataset is in English.
3. **Dataset Structure**: Each record in the dataset contains several fields:
    - article: The full text of the biomedical article.
    - summary: The non-technical lay summary of the article.
    - section_headings: The headings of the sections in the article.
    - keywords: The keywords associated with the article.
    - year: The year the article was published.
    - title: The title of the article.
    - article_length: The length of the article.
    - summary_length: The length of the summary.
4. **Number of Records**: The dataset contains 4,828 full biomedical articles paired with non-technical lay summaries.

The dataset is designed to help develop and evaluate models for the task of scientific lay summarization. This involves taking a scientific article and producing a summary that is *understandable to a layperson*, without losing the article's original meaning.

2. Simplification

Abstractive summarization is a method of text summarization that generates a short and concise summary, capturing the key ideas of the source text. Unlike extractive summarization, which simply selects and rearranges sentences from the original text, abstractive summarization can introduce new phrases and sentences that may not appear in the source text.This technique involves understanding the context, identifying key pieces of information, and re-creating them in a new way. It's like trying to guess the meaning of the whole text and presenting that meaning in a condensed form. It can provide a more natural and coherent summary, making them particularly useful for simplifying complex text therefore in this project we will be focussing on abstractive summarization instead of extractive summarization.

3. Scope of simplification

In our project, we aim to distill the essence of each section of a given scientific paper into a concise, easy-to-understand format. We refer to this process as ‘simplification’. It involves condensing complex scientific concepts into fewer words, while ensuring the core message remains intact. Our goal is to use plain, non-technical or lay language to make the content accessible to a wider audience(for which purpose dataset is selected), including those without a scientific background. This is not a mere extraction of existing sentences, but a thoughtful reinterpretation of the original text.

Step 1: Text Extraction from PDF This step involves the process of extracting text data from PDF files. The extraction process reads the content of the PDF file page by page and converts it into a format that can be manipulated as text data.

Step 2: Text Segmentation into Sections and Subsections The extracted text is then segmented into sections and subsections. This is achieved by identifying specific patterns or markers that denote the beginning and end of each section and subsection using regex.

Step 3: Text Summarization of Sections and Subsections The final step involves summarizing the text content of each section and subsection. This is done using a trained transformer model, which processes the text data and generates a concise summary that captures the key points of the original content.

In our experiments, we observed that summarizing an entire paper in one go often resulted in a summary that was too brief and did not adequately capture the depth and breadth of the information contained in the full article. To address this issue, we decided to divide the research paper into separate sections and summarize each section individually. This approach has several advantages:

- **Detailed Summaries:** By summarizing each section individually, we can generate more detailed summaries that better capture the specific information and insights contained in each part of the paper.
- **Preserving Structure:** This approach also helps to preserve the original structure of the paper, making it easier for readers to understand the flow of ideas and arguments.
- **Context Preservation:** Summarizing each section separately helps to maintain the context of each part of the paper, which can be lost when summarizing the entire document at once.

4. Pre-processing Module

Tokenization Using Stanford CoreNLP: We use Stanford CoreNLP to tokenize text as suggested by the paper. This involves taking a collection of text documents and splitting them into individual sentences and words.

Cleaning: Lowercasing the words and removing unnecessary characters.

BERT Tokenization: It tokenizes the source and target text and converts them into subtoken sequences for giving input to BERT model.

5. Problem Formulation:

Input: Preprocessed data extracted from biomedical articles in the dataset.
Output: Non-technical lay summaries that convey the key information in a comprehensible manner.

Negative Log-Likelihood (NLL) Loss:
The loss function used in the decoder is the loss function from the vanilla transformer from attention is all you need paper.

$$L = -\frac{1}{N}\sum_{n=1}^{N} log\ p(y_n \mid y_{<n}, x)$$

Where:

L is the NLL loss.

N is the total number of tokens in the target sequence.

y is the nth token in the target sequence.

$y_{<n}$ represents all the tokens before the nth token in the target sequence.

x is the input sequence.

$p(y_n|y_{<n}, x)$ is the probability of the n_{th} token given the input sequence and all the previous tokens in the target sequence, as predicted by the Transformer decoder. This probability is obtained by applying a softmax function to the decoder's output logits.

The NLL loss is computed over all tokens in the target sequence. The division by N is to normalize the loss by the number of tokens. This is particularly useful when dealing with sequences of varying lengths.

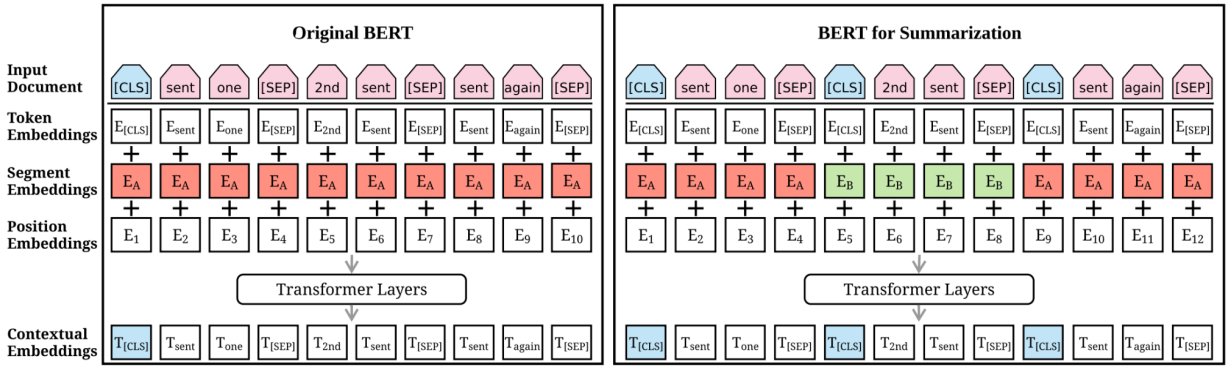
6. Methodology

Reference paper - https://arxiv.org/pdf/1908.08345.pdf (Text Summarization with Pretrained Encoders)
Our aim is to implement the methodology described in above given paper for summarization.

Encoder : Pretrained BERT (bert-base-uncased)
Decoder : Randomly Initialized Transformer Decoder (Vanilla Transformer decoder from paper "Attention is all you need")

The main idea of this paper - "Text Summarization with Pretrained Encoders" is to use BERT for text summarization this proposes framework for both extractive and abstractive models but we will implement only the abstractive model for our case. For abstractive summarization, a new fine-tuning schedule is proposed by authors. This schedule uses different optimizers for the encoder and the decoder to address the mismatch between the two. The encoder is pretrained and therefore has a good understanding of language, while the decoder is not pretrained and needs to learn how to generate summaries from scratch. Using different optimizers helps to balance the learning rates of the encoder and the decoder, preventing overfitting on the encoder side and underfitting on the decoder side.

Modification on top of BERT



BERTSUM modifies the original BERT model to make it suitable for text summarization by summing three kinds of embeddings for each token, inserting multiple [CLS] symbols after [SEP].

Summation of Three Kinds of Embeddings: Each token in the input document is assigned three kinds of embeddings: token embeddings, segmentation embeddings, and position embeddings.

Token Embeddings indicate the meaning of each token.
Segmentation Embeddings are used to discriminate between two sentences.
Position Embeddings indicate the position of each token within the text sequence. The summed vectors of these embeddings are used as input embeddings to Transformer layers.

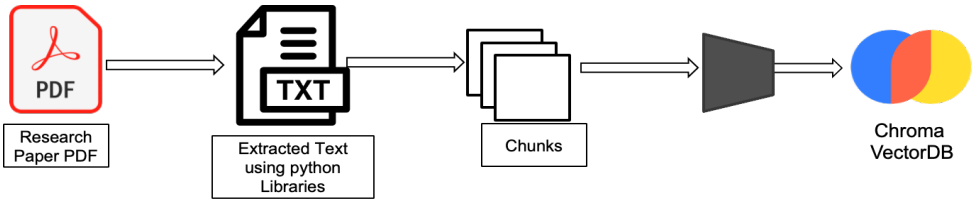
Research Paper Co-Pilot

(Using out-of-the-box LLMs no fine-tuning, just an experiment not main project)

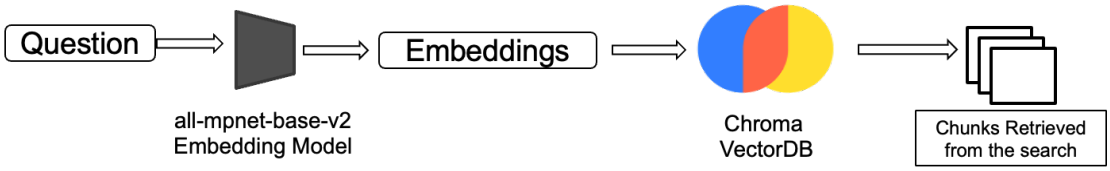
This tool is designed to facilitate question answering on research papers using the LLM (specifically, the Llama-2 model). The process involves converting the research papers into a form that can be queried, which is achieved through the use of embeddings and vector database. These embeddings are then stored in a Chroma database for efficient retrieval.

The process can be broken down into three main components:

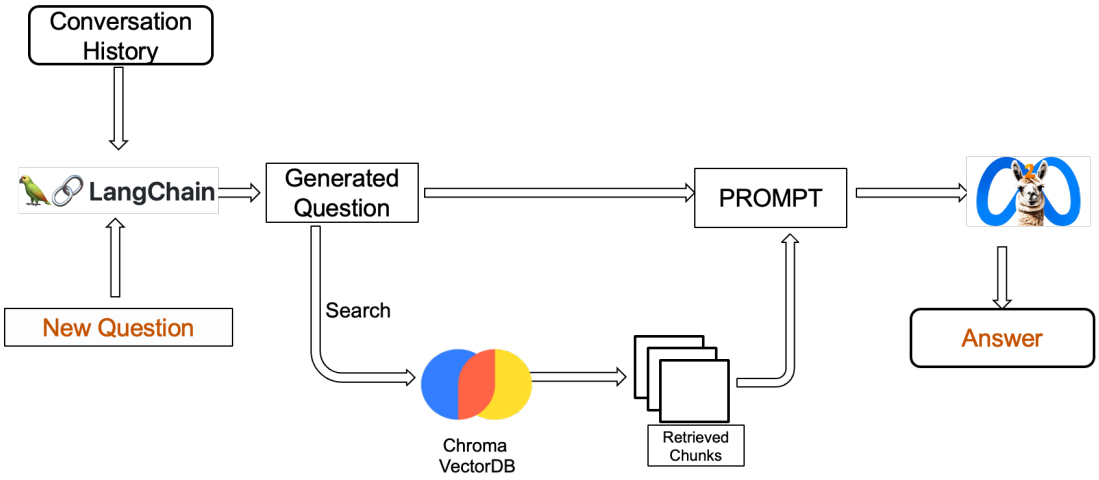
1. **Vector Database Ingestor:** This component is responsible for processing the research papers and storing them in the vector database. The process begins by converting the PDF files of the research papers into text format. The text is then divided into chunks, which are transformed into embeddings using the `all-MPNet-base-v2` model from Hugging Face. These embeddings are finally stored in the vector database, ready to be queried.



2. **Searching Mechanism:** This component handles the retrieval of relevant information from the vector database. When a question is posed, it is first converted into an embedding using the same model that was used for the research papers. This question embedding is then used to retrieve the top-k most similar chunks from the vector database using a vector search algorithm.



3. **Question Answering:** This component is responsible for generating the final answer to the user's question. It utilizes the LangChin question generator to generate questions based on the previous conversation history. These generated questions are then used to search the vector database, which retrieves relevant chunks of text. These chunks, along with the original question, are used to form a prompt that is passed to the Llama-2 Chat model. The Llama-2 model then generates an answer to the user's question based on this prompt.



Timeline for completion of Project

Week 1: Initial implementation and comprehension of the methodology outlined in the reference paper.

Week 2: Exploration and experimentation with various modifications to the core architecture of the transformer, along with an analysis of their impacts.

Week 3: Finalization of the model and collection of results.

Task delegation among members

Pathik Patel	[202003002] – Summarizer Transformer and Copilot
Divya Patel	[202001420] – Summarizer Transformer and Copilot
Kunj Patel	[202001421] – Summarizer Transformer
Suyash Vyas	[202001002] – Summarizer Transformer