

Internship Assignment Report

Aimen Asad

211-1505

Submitted To: [KDD.lab](#)



Introduction.....	3
1.1 Project Overview.....	3
1.2 Objectives.....	3
Data Exploration.....	4
2.1 Data Sources.....	4
2.2 Data Exploration.....	4
2.3 Data preprocessing.....	5
Web Application Development.....	7
3.1 User Interface.....	7
3.2 Summarization Feature.....	7
Results and Discussion.....	10
Strengths.....	10
Limitations.....	10
Improvements.....	11

Introduction

1.1) Project Overview

This project aimed to develop a user-friendly web application for summarizing scientific articles retrieved from PubMed, a central repository for biomedical literature. The application caters to researchers, students, and anyone seeking a quick and efficient way to grasp the key points of PubMed articles.

1.2) Objectives

The primary objectives of this project were:

- **To create a web application:** This application would offer a user interface accessible through a web browser for easy interaction and wider accessibility.
- **To integrate PubMed article summarization:** The application would take PubMed articles as input and generate concise summaries highlighting the main points of the research.
- **To improve information access:** By providing summaries, the application would enable users to quickly grasp the core findings of PubMed articles, saving them valuable research time.

Data Exploration

2.1) Data Source

To train this summarization application, I needed a bunch of scientific articles and summaries. I found these on a website called Hugging Face. Hugging Face is kind of like a library for AI projects, and they have a collection of datasets ready-made for different tasks, like summarizing text.

The dataset I used came from PubMed, which is a giant database of scientific research. This was a good choice because it has lots of articles on many different areas of science, so my summarization application could learn from a variety of topics.

2.2) Data Exploration

To gain a deeper understanding of the PubMed Summarization Dataset, I began by loading it into the `KDD_task.ipynb` Jupyter notebook. This notebook served as the initial exploration platform. Here, I focused on uncovering key characteristics of the data that would inform the summarization model's development.

The exploration encompassed the following steps:

1. **Feature Overview:** Firstly examined the dataset's structure, identifying the features (columns) it contained. This provided insights into the type of information available for each article, such as full text, corresponding summary, and potentially additional metadata like title or publication date.
2. **Data Size:** I determined the number of rows (samples) present in each split (train, test, and validation sets). This information is crucial for assessing the model's

training efficiency and generalizability. Larger datasets often require more training time but can potentially lead to better performance.

3. **Examining Examples:** Lastly delved into the content by printing the first example from each split (train, test, validation). This allowed us to visualize the format of the data and get a firsthand sense of the articles and summaries within each set.
4. **Visualization Attempts:** I attempted to explore the data using visualizations (e.g., histograms, scatter plots) to identify potential patterns or relationships between features. However, due to the potentially large size of the dataset, these visualizations could have been computationally expensive and time-consuming. Considering the project's scope and time constraints, I was unable to do it.

By following these steps, I gained valuable insights into the structure, content, and size of the PubMed Summarization Dataset. This initial exploration laid the groundwork for the subsequent data preprocessing and model training stages.

2.3) Data preprocessing

Since I was new to this whole Natural Language Processing (NLP) thing, I had to do some research to figure out how to get the text data ready for the summarization model. I used Google like crazy to learn about different techniques and then put what I found into a Python script called `preprocessing.py`. The goal was to clean up the text and make it easier for the model to understand.

Preprocessing Steps:

1. **Lowercasing:** Through my online research, I learned about the importance of text normalization, which included converting all text to lowercase letters. This ensures consistency and avoids treating words like "Cancer" and "cancer" as different

entities in the model's eyes.

2. **Removing Non-Alphabetic Characters and Multiple Spaces:** I discovered that punctuation marks, special characters, and excessive whitespace can hinder the model's understanding of the text. Leveraging online resources, I implemented regular expressions (`re`) to perform these cleaning tasks:
 - `re.sub(r'^a-z\s', '', text)`: This expression removes any characters that are not lowercase letters (`a-z`) or whitespace (`\s`).
 - `re.sub(r'\s+', ' ', text).strip()`: This expression replaces multiple spaces with a single space and then removes leading and trailing spaces from the text using `.strip()`.
3. **Tokenization:** My research revealed that tokenization, the process of breaking down text into individual words (tokens), is crucial for further NLP tasks. I employed the `nltk.word_tokenize` function to achieve this step.
4. **Stopword Removal:** I learned that stop words, common words like "the," "a," "an," etc., don't contribute much meaning in the context of summarization. Using the `stopwords` library and a pre-downloaded list of English stop words (`stopwords.words('english')`), I removed these words from the tokenized text.
5. **Lemmatization:** Further exploration online introduced me to lemmatization, the process of reducing words to their base forms. For example, "running" becomes "run." I utilized the `WordNetLemmatizer` class from NLTK to achieve this step.

By diligently researching and implementing these preprocessing steps, I ensured the text data was cleaned, normalized, and ready to be utilized for effective training of the summarization model. This approach allowed me to leverage existing knowledge and

readily available resources to overcome my initial limitations in NLP and successfully prepare the data for the next stage of the project.

Web Application Development

Flask, a popular Python web framework, was chosen for its lightweight and flexible nature. Flask provided a solid foundation for building the web application without unnecessary overhead.

3.1) User Interface

The application employs two primary HTML templates:

- `home.html`: This template serves as the landing page, offering users the ability to upload a PubMed article or paste text directly into a text area.
- `result.html`: This template displays the user-submitted article text alongside the automatically generated summary.

By leveraging Flask's routing capabilities, the application dynamically switches between these templates based on user interaction.

3.2) Summarization Feature

The core functionality of automatically summarizing PubMed articles resides in the `summarize_text` function. This function takes the user-provided text and the desired number of sentences as input.

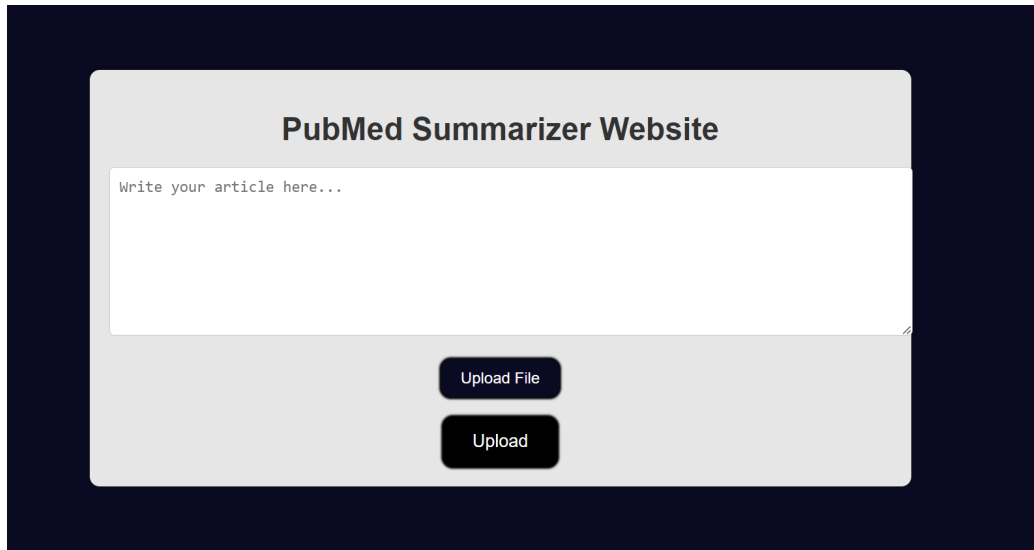
Summarization Process

-
1. **Parsing the Text:** The `sumy` library is employed for the summarization tasks. It starts by parsing the input text using `PlaintextParser` and `Tokenizer`.
 2. **Summarization Algorithm:** The `LsaSummarizer` class, implementing Latent Semantic Analysis (LSA), is used to generate the summary. LSA is an extractive summarization technique that identifies key sentences based on their semantic relationships within the document.
 3. **Summary Construction:** The function extracts the desired number of sentences (`sentence_count`) from the summarization result generated by `LsaSummarizer`.
 4. **Returning the Summary:** The function returns a string containing the concise summary, ready for display in the user interface.

Integration with Flask Routes

- The `/` route handles both GET and POST requests.
- For GET requests, it renders the `home.html` template, presenting the user with the upload/paste options.
- Upon receiving a POST request, the route retrieves the uploaded file or submitted text from the form data.
- The retrieved article text undergoes preprocessing using the `preprocess` function from the `preprocessing.py` script.
- The preprocessed text is then fed to the `summarize_text` function to generate the summary.
- Finally, the route renders the `result.html` template, displaying both the original article text and the generated summary.

Index.html



The image shows a web interface for a 'PubMed Summarizer Website'. It features a title bar, a text input area with the placeholder 'Write your article here...', and two buttons labeled 'Upload File' and 'Upload'.

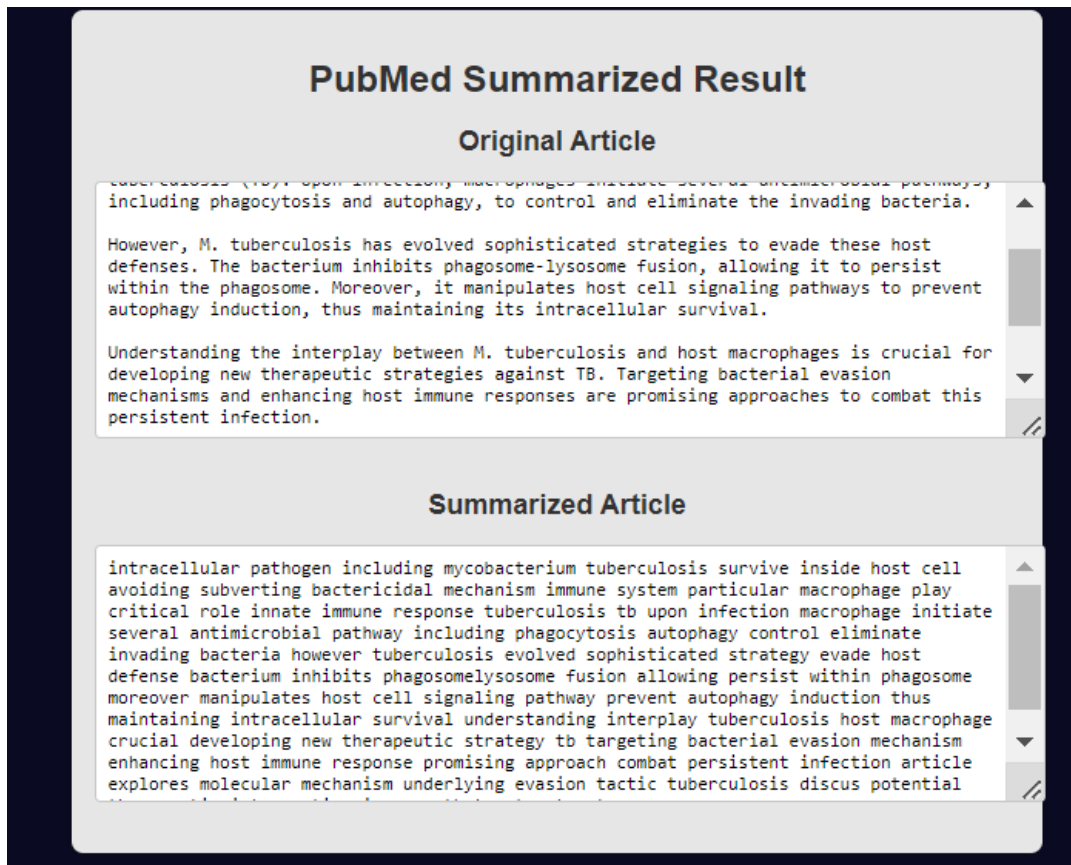
PubMed Summarizer Website

Write your article here...

Upload File

Upload

Result.html



The image shows a web interface for 'PubMed Summarized Result'. It displays the 'Original Article' and a 'Summarized Article' in a text area with a scrollbar. The original article text is: 'Mycobacterium tuberculosis (Tb) upon infection macrophages initiate several antimicrobial pathways, including phagocytosis and autophagy, to control and eliminate the invading bacteria. However, M. tuberculosis has evolved sophisticated strategies to evade these host defenses. The bacterium inhibits phagosome-lysosome fusion, allowing it to persist within the phagosome. Moreover, it manipulates host cell signaling pathways to prevent autophagy induction, thus maintaining its intracellular survival. Understanding the interplay between M. tuberculosis and host macrophages is crucial for developing new therapeutic strategies against TB. Targeting bacterial evasion mechanisms and enhancing host immune responses are promising approaches to combat this persistent infection.' The summarized article text is: 'intracellular pathogen including mycobacterium tuberculosis survive inside host cell avoiding subverting bactericidal mechanism immune system particular macrophage play critical role innate immune response tuberculosis tb upon infection macrophage initiate several antimicrobial pathway including phagocytosis autophagy control eliminate invading bacteria however tuberculosis evolved sophisticated strategy evade host defense bacterium inhibits phagosome lysosome fusion allowing persist within phagosome moreover manipulates host cell signaling pathway prevent autophagy induction thus maintaining intracellular survival understanding interplay tuberculosis host macrophage crucial developing new therapeutic strategy tb targeting bacterial evasion mechanism enhancing host immune response promising approach combat persistent infection article explores molecular mechanism underlying evasion tactic tuberculosis discuss potential'.

PubMed Summarized Result

Original Article

Mycobacterium tuberculosis (Tb) upon infection macrophages initiate several antimicrobial pathways, including phagocytosis and autophagy, to control and eliminate the invading bacteria. However, M. tuberculosis has evolved sophisticated strategies to evade these host defenses. The bacterium inhibits phagosome-lysosome fusion, allowing it to persist within the phagosome. Moreover, it manipulates host cell signaling pathways to prevent autophagy induction, thus maintaining its intracellular survival. Understanding the interplay between M. tuberculosis and host macrophages is crucial for developing new therapeutic strategies against TB. Targeting bacterial evasion mechanisms and enhancing host immune responses are promising approaches to combat this persistent infection.

Summarized Article

intracellular pathogen including mycobacterium tuberculosis survive inside host cell avoiding subverting bactericidal mechanism immune system particular macrophage play critical role innate immune response tuberculosis tb upon infection macrophage initiate several antimicrobial pathway including phagocytosis autophagy control eliminate invading bacteria however tuberculosis evolved sophisticated strategy evade host defense bacterium inhibits phagosome lysosome fusion allowing persist within phagosome moreover manipulates host cell signaling pathway prevent autophagy induction thus maintaining intracellular survival understanding interplay tuberculosis host macrophage crucial developing new therapeutic strategy tb targeting bacterial evasion mechanism enhancing host immune response promising approach combat persistent infection article explores molecular mechanism underlying evasion tactic tuberculosis discuss potential

Result and discussion

Strengths

- **Flexibility and User Choice:** A key strength of the application is its ability to accommodate user preferences. Users can choose between pasting the article text directly or uploading a file from their computer, catering to different comfort levels and working styles.
- **NLP-powered Summarization:** The application effectively utilizes NLP techniques for text summarization. This allows users to obtain summaries of complex scientific articles without needing to read them in their entirety.
- **User-Friendly Interface:** The application prioritizes a user-friendly interface built with Flask. This makes it accessible to a wider range of users, including researchers and students with varying levels of technical expertise.

Limitations

- **Limited Visualization:** Currently, the application does not generate visual aids like graphs or charts to complement the summaries. While the text summaries are informative, incorporating visualizations could further enhance user understanding, especially for data-driven articles.
- **Deployment Scope:** At this stage, the application functions as a standalone web application. Deploying it on a public server would make it more accessible to a broader audience and allow for wider use within the research community.

Future Improvements

- **Enhanced User Interface Features:** The user interface can be further refined to provide a more interactive experience. Consider adding a pop-up notification confirming successful file upload or incorporating progress bars to indicate processing time.
- **Visual Representation:** As mentioned earlier, integrating visual elements like charts or graphs could significantly improve the user experience. Visualizations can effectively represent trends or relationships within the summarized information, aiding user comprehension.
- **Deployment and Accessibility:** Deploying the application on a publicly accessible server would allow researchers and students to benefit from its summarization capabilities from anywhere with an internet connection. This wider availability can significantly increase the application's impact.

By addressing these limitations and exploring future improvements, you demonstrate a thoughtful approach to project development and a clear vision for its potential.
