WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

LLM Based News Research Tool Using LangChain with Enhancing Similarity Search and Token Limit

Raushan Kumar[1], Dr. Subbulakshmi P[2], Khushi Habbu[3]

[1,3]B. Tech Computer Science, *School of Computer Science and Engineering, Vellore Institute of Technology, Chennai*
[2]Assistant Professor Senior, *School of Computer Science and Engineering, Vellore Institute of Technology, Chennai*

ABSTRACT—

In the digital age, with an overwhelming abundance of news sources and information outlets, people increasingly struggle to effectively navigate and absorb news content. This has led to a growing need for advanced technologies that assist in the categorization, distillation, and extraction of insights from vast amounts of news data. This abstract introduces a novel approach known as LangChain, which leverages blockchain infrastructure and Language Model (LM) technology to develop a sophisticated news research tool.

LangChain is a framework designed specifically for Large Language Models (LLMs) and offers several essential features for document processing. It includes multiple text loaders capable of handling various formats such as text files, CSVs, and URLs. Once documents are loaded, LangChain employs character and recursive text splitters to divide the text into manageable chunks. Text encoding is handled through the Hugging Face and OpenAI modules, which convert text into numeric vectors that can be stored in vector databases. These libraries provide the necessary transformations and embeddings to represent words numerically.

LangChain incorporates fundamental principles of classical information retrieval (IR) for tasks such as retrieval, summarization, search, and keyword extraction. It integrates with FAISS, a library used for similarity search and clustering of dense vectors, facilitating efficient storage and retrieval from FAISS indexes. Techniques such as TF-IDF are utilized for generating similar search results. Additionally, LangChain integrates RetrievalQA with sources chain, a critical component where information chunks are processed through LLMs. Filtered chunks are used to trigger further processing and are continually refined. These refined sections are then used in summarization techniques, enhancing the effectiveness of subsequent queries. Ultimately, the final response is derived from the processed and condensed portions of the input query.

Keywords: LLM, IR, TF-IDF, FAISS, Similarity Search

I. Introduction

There is a great chance that the propose LangChain-based LLM-based news research tool would fundamentally alter how people interact with and understand news content. The Lang Chain framework provides a structured approach to leverage LLM capabilities, ensuring more accurate, language-specific, and context-sensitive news article insights. This has implications for improving the range and efficiency of information extraction across different language situations. Include a Legal Language Model (LLM) in the project for accurate analysis.

To build precise and efficient algorithms for keyword extraction and to ask a query and to retrieve answer from news articles, researchers must overcome numerous challenges that span various aspects of natural language processing and domain-specific knowledge. It is necessary to address a number of issues in natural language processing and information retrieval in order to develop accurate and effective algorithms for query-based information retrieval from news articles and keyword extraction.

Utilize Lang Chain's modular architecture for seamless integration and enhanced natural language processing. Processing data to ensure accuracy and relevance for analysis, tidy and arrange news. Using Language Processing to Develop entities-recognition, concept-extracting, and comprehension algorithms for search and retrieval. These algorithms will enable the efficient use of advanced search tools to obtain relevant legal news using natural language queries. Design of User Interface Make an intuitive dashboard with moveable widgets to enhance the professional user experience.

II. Objectives

Our goal is to create a comprehensive LLM project covering a real-world industrial use case for research analysts. We also want to create a news research tool that allows users to provide the URLs of numerous news stories, and when they ask a question, the system will pull the response from the filtered portions of those articles.

Simply using a key word search is insufficient; for instance, when we search for "apple" on Google, we get both the apple company and standard apple fruit information. To address this, our paper will use vector databases and embedding to precisely understand the context of the search and retrieve pertinent sections from the urls that we will then use to look for answers in it. In many text processing tasks, such as text categorization, information mining, and text retrieval, keyword extraction is essential. Because of its ease of use and ability to accurately determine term relevance based on word frequency, the TF-IDF algorithm has become one of the most popular established approaches. On the other hand, depending exclusively on word frequency data may have drawbacks, especially in situations when words might not appropriately convey their importance within the page.

The process of locating pertinent information within a sizable collection of text documents is called text retrieval, or information retrieval (IR). It entails locating documents pertinent to an information request or query made by a user. Text loaders with many sources can handle a variety of document types, including text files, CSV files, and URLs. Text Splitter with Recursive Characters splits documents into digestible chunks while maintaining semantic consistency by using recursion and character splitters. Within the Langchain framework, this division makes additional analysis easier. Using contemporary pre-trained language models, Hugging Face and OpenAI tools are integrated to build embeddings from textual data. This makes it possible to translate language into numerical representations, which improves analyzing skills.

Facebook developed an open-source library called FAISS (Facebook AI Similarity Search) to efficiently search for similarities and cluster massive datasets. It is especially made to function well with high-dimensional vectors, which makes it appropriate for jobs involving large-scale vector similarity computations, such as natural language processing and picture similarity search. Information chunks are repeatedly passed through LLMs using map reduce method over stuff method i.e. the drawback od exceeding token limit or usage of token limits in API is solved in the important step where Lang chain integrates RetrievalQA with sources chain later in the process. Filtered pieces are constantly refined and act as catalysts for further processing. Summarization techniques are used to combine these portions, making the following queries more effective. After processing the input query iteratively, the final response is derived from the condensed and refined components

An accurate predictive model can be valuable to businesses and consumers to determine the fair price of a diamond. The goal of the project is to build a model that can accurately predict the price of a diamond potentially based on its weight, quality and dimension measurements. We aim at forecasting the prices of diamond using various regression techniques like linear regression, knn, decision tree and random forest.

III. Literature Survey

Natural language processing (NLP) has been the focus of recent research, which has shown how important it is to improving news analysis and summarization in a variety of linguistic and topic domains. Numerous research have used natural language processing (NLP) approaches to address basic tasks like sentiment analysis, trend detection, and summarization, which have the potential to improve the effectiveness of information consumption and decision-making processes.

Researchers such as Khan et al. and Saxena et al., for example, have used natural language processing (NLP) approaches to classify Bengali crime reports and to examine trends in the stock market. By offering insightful information to a wide range of stakeholders—from news outlets and law enforcement to investors and financial specialists—these initiatives hope to help make well-informed decisions and allocate resources.

Furthermore, NLP has shown to be helpful in resolving linguistic and domain-specific issues that arise in news analysis. NLP techniques can be customized to meet the needs of particular linguistic settings and subject areas, as shown by studies like Ghasiya and Okamura's analysis of cybersecurity news stories and Lwin and Nwet's work on Myanmar language extractive summarization. These initiatives enable users to browse and understand complicated material more efficiently, improving their capacity to stay informed and make well-informed decisions. They accomplish this by automating the summarizing process and recognizing important themes and trends.

Moreover, NLP's adaptability goes beyond conventional news outlets to include a variety of channels, including social media and online discussion boards. In order to obtain relevant news, researchers like Jayasiriwardene and Ganegoda have investigated the extraction of keywords from tweets. This shows how NLP technology may assist consumers in staying informed about current events in real-time. Furthermore, Mayopu et al.'s efforts to categorize news produced by AI models like ChatGPT highlight how crucial it is to maintain thdependability and legitimacy of news sources in a time when information is widely disseminated through digital channels.

The growing corpus of research discussed herein, in conclusion, emphasizes the importance of natural language processing (NLP) in transforming news analysis and summarization across languages and topic domains. In the constantly changing world of digital news, natural language processing (NLP) provides a range of effective tools for trend detection, insight extraction, and well-informed decision-making. These tools are made possible by utilizing machine learning algorithms and sophisticated linguistic processing capabilities.

IV. System Overview

A. Background and data description

An enormous amount of news stories has been produced and disseminated across numerous channels in recent years due to the exponential rise of digital media and online news platforms. It becomes very necessary to have effective tools and methods to browse, comprehend, and extract useful

insights from news material in this enormous sea of information. Because of the size, complexity, and diversity of contemporary news data, traditional approaches of news analysis frequently prove inadequate.

By utilizing huge language models, including Hugging Face's transformers and OpenAI's GPT, the LLM-based news research tool overcomes these difficulties. Due to their training on enormous datasets with billions of words, these models are able to understand the complex patterns, semantic linkages, and contextual subtleties seen in natural language processing. Emphasizing information retrieval (IR) methods is one of the tool's primary features. IR algorithms are essential for activities including content retrieval, keyword extraction, summarization, and search. Through the use of cutting-edge information retrieval techniques, the application helps users to effectively search through enormous databases of news stories, extract pertinent information, and find insightful information concealed within the data.

There is a great chance that the proposed LangChain-based LLM-based news research tool would fundamentally alter how people interact with and understand news content. The Articles.csv from Kaggle totaling 2692 news articles. The source https://www.thenews.com.pk.

It features business and sports-related news pieces from 2015 onward. It includes the specific article's heading, content, and date. The location where the comment or article was published is also included in the content. We just used this to get understanding on how information is retrieved from URLs or csv files i.e. documents and understand the similarity, relevance, extraction among them After processing the input query iteratively, the final response is derived from the condensed and refined components. Thus, understanding how this work and what is relevance score and cosine similarity with using tf-idf, bm25 and faiss and then building a news research tool.

B. Methodology

Multiple Source Text Loaders: Create text loaders that can handle a variety of document types, including CSV files, text files, and URLs. Assure effective textual data extraction and processing from a variety of sources for the Langchain architecture.

Text Splitter with Recursive Characters: Use recursion and character splitters to divide documents into more manageable sections. During the splitting process, semantic integrity and coherence are preserved for further analysis.

Integration of Hugging Face for Embeddings with OpenAI: Combine the Hugging Face and OpenAI tools to create embeddings from textual data. Modern pre-trained language models can be used to convert sentences into numerical representations for additional analysis.

Investigation of Algorithms for Information Retrieval: Examine several information retrieval (IR) algorithms that can be used for retrieval, summarization, search, and keyword extraction. Analyze the performance of various algorithms in relation to research and news analysis activities.

FAISS Indexing and Retrieval Implementation: Use FAISS (Facebook AI Similarity Search) to store, index, and retrieve embeddings more quickly. Create indexes with FAISS to facilitate scalable and quick similarity searches for news items and related queries.

RetrievalQA Development Using Sources Chain: Create and execute RetrievalQA using Sources Chain, the foundational element of the news research instrument. Create algorithms that handle and filter obtained chunks iteratively, adding feedback loops to improve search results.

Combining LLMs to Perform Multi-pass Analysis: Combine LLMs with the Langchain framework to analyze retrieved text chunks across several passes. Make use of LLMs to produce summaries, extract important data, and incrementally improve query replies.

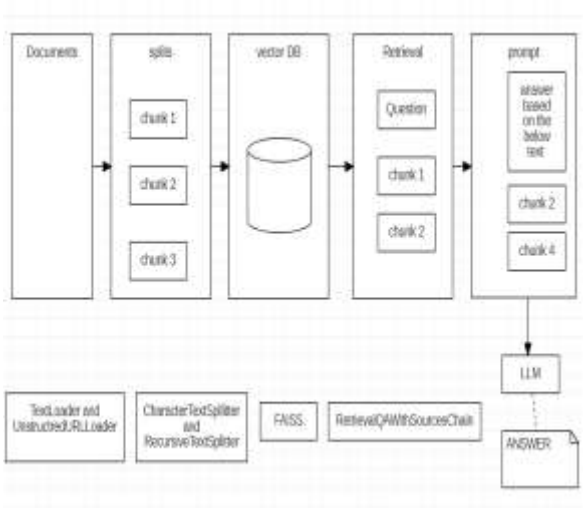

Figure.1: Proposed System Diagram

There are two methods to summarize chunks ,stuff method (simple append method) or map reduce method , a drawback in stuff method being that it exceeds token limit and usage of more space as summary chunk is huge so we can solve it by using map reduce method where based on splitter's and separators we get chunks with similarity search , let's take if we get 4 chunks all those 4 are sent again through LLMs and filtered chunks with necessary info only based on keyword extraction similarity search with faiss and those filtered chunks are summarized and along with input query ,sent in final

LLM and we get an answer , in this way we save token limit from exceeding its easier to get precise answer with lesser cost (as token size is calculated) following below is the figure to understand map reduce method.

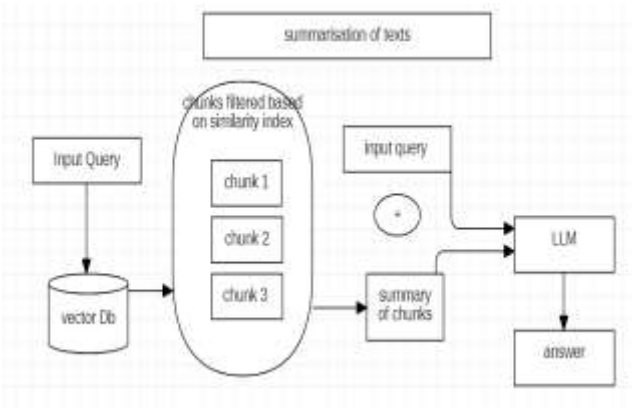

Figure.2: Summarization of chunks

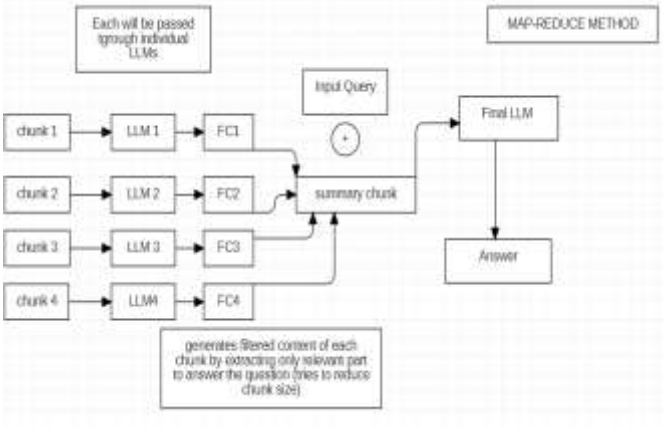

Figure.3: Map Reduce Method

One of the key components of the Langchain framework is the RetrievalQA With Sources Chain, designed for question-answering (QA) tasks that involve retrieving relevant information from multiple sources. This chain leverages pre-trained language models (LLMs) to understand and process queries, search across diverse sources such as documents and databases, and extract pertinent responses. Among its notable features, the RetrievalQA With Sources Chain excels in question understanding by utilizing LLMs to interpret user inquiries more accurately, enhancing information retrieval. It supports source integration by connecting with various information repositories, including text documents, databases, and websites, facilitating comprehensive search capabilities.

The chain is also adept at answer extraction, employing advanced techniques to ensure responses are contextually relevant and coherent. Furthermore, it offers scalability to handle large volumes of data efficiently, making it suitable for complex QA tasks across extensive datasets. Customization is another strength, allowing users to tailor the retrieval and QA processes to fit specific domains and use cases. Overall, the RetrievalQA With Sources Chain significantly boosts the effectiveness and efficiency of Q&A systems by combining LLMs with diverse information sources to deliver accurate and comprehensive answers.

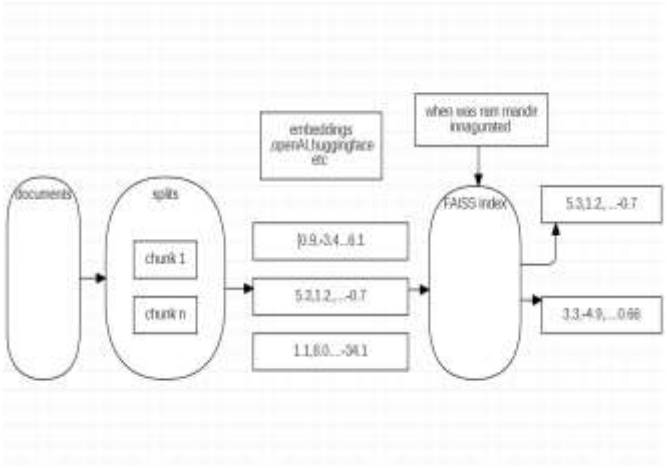

Figure.4: FAISS Index

Python Interface: FAISS offers a Python interface that is simple to add into your machine learning pipelines and works well with well-known libraries like NumPy and scikit-learn. Integration with Embedding Libraries: For applications involving natural language processing or computer vision, FAISS is frequently utilized in conjunction with embedding libraries, such as word embeddings. Pre-trained Models: FAISS makes it simple to install and integrate into applications by providing pre-trained models and indexes for typical use cases. All things considered, FAISS is a strong library that can be used to do clustering and similarity search in high-dimensional spaces. It is extensively utilized in many different applications, such as content-based retrieval systems, recommendation systems, and search engines.

V. Results:

A. IR algorithms

1. A generative statistical model called Latent Dirichlet Allocation (LDA) is frequently employed for topic modelling. Let's look into that first

Figure.5: LDA

The process begins with dictionary creation, where pre-processed tokens are mapped to numeric IDs in a dictionary. This is followed by dictionary filtering, which removes tokens that appear in more than 50% of the documents or fewer than 20 documents, eliminating words that are either too common or too rare to be useful for subject modeling. Next, the corpus is created, representing each document as a list of (word ID, word frequency) tuples, similar to a bag of words model. With the filtered dictionary and corpus, the Latent Dirichlet Allocation (LDA) model is trained.

LDA is a generative statistical model that identifies underlying themes in a set of observations. The LDA model then outputs the top words associated with each identified topic, showing the most probable terms for each subject. Finally, the coherence score is computed using the 'c_v' measure, which assesses the semantic closeness of the high-scoring terms within each topic. In summary, the process involves analyzing a set of headlines with LDA to discover topics and using coherence scores to evaluate the quality of these topics.

B. Now a ranking algorithm like BM25 with inverted index

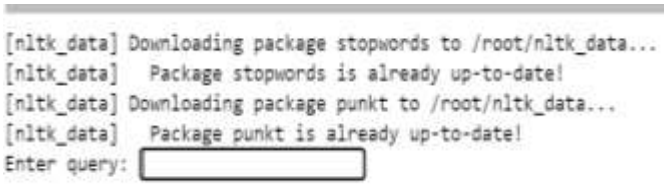

Figure.6: Simple BM25 output

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
Enter query: Python was conceived in the late 1980s ?
Top 5 relevant documents:
1. Document 1 (Score: 5.37149151532058)
2. Document 2 (Score: 1.6479184330021646)
3. Document 3 (Score: 0.9460852522523836)
```

Figure.7: Simple BM25 output with score

We can see the relevance score as per question asked if given as per similarity/relevance using bm25 (also tf-idf) but inverted, we ask a query and among the URLs it gives top 5 documents and most relevant in them as per the score now lets look into few visualizations to understand this extraction methods better.

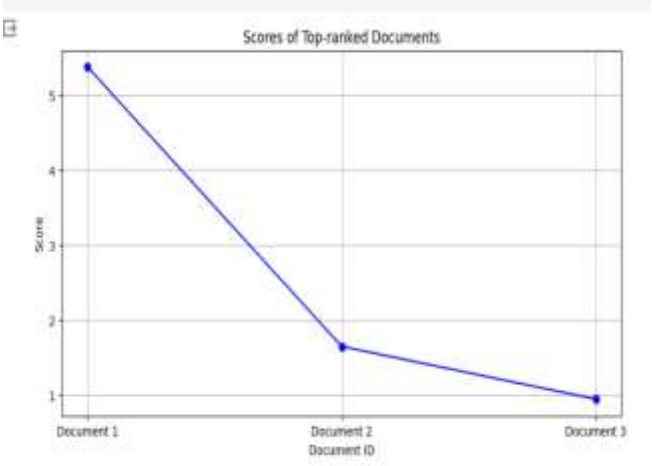

Figure.8: Scores of the top ranked documents

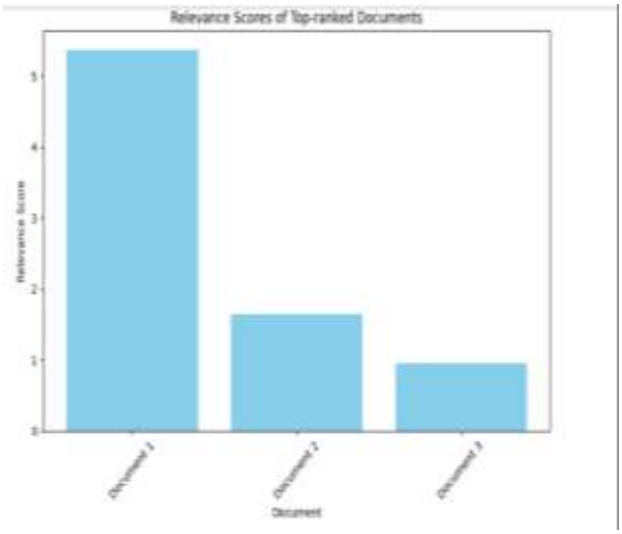

Figure.9: Relevance score of top documents in bar plot

C. TF-IDF with FAISS

Before adding the TF-IDF calculated vectors to the FAISS index, the process involves several steps. First, the dataset is loaded and the text data is preprocessed. Next, a TF-IDF vectorizer is initialized to convert the preprocessed text into a TF-IDF matrix, which is then transformed into a NumPy array. Following this, a FAISS index is initialized, and the TF-IDF vectors are added to the index. Once the vectors are indexed, a query is performed to retrieve the five most relevant news articles, including their headings and types. The similarity of the retrieved articles to the query is assessed using cosine similarity, a measure known for its accuracy in evaluating textual similarity.

Figure.10: Simple TF-IDF with FAISS output

The process begins by loading a dataset from a CSV file named "Articles.csv" into a pandas DataFrame, which may contain articles along with headers and additional data. Next, the heading data is extracted from the DataFrame for textual analysis. The TF-IDF vectorization is then performed using the TfidfVectorizer from scikit-learn, which quantifies each word's importance in a document relative to a collection of documents. The fit_transform() method converts the textual data into a TF-IDF matrix, which is subsequently transformed into a NumPy array. For similarity searching, an FAISS index, specifically IndexFlatL2, is initialized. This type of index supports L2 (Euclidean) distance calculations. Finally, the TF-IDF vectors, converted to float32, are added to the FAISS index for efficient similarity search.

When a user enters a text query, it is first vectorized using the same TF-IDF vectorizer applied to the articles, converting it into a float32 NumPy array. The FAISS index is then used to perform a similarity search, identifying the k nearest neighbors (k=5) to the query vector. The indices of the most similar documents are retrieved, and the corresponding articles, including their titles and additional details, are printed.

In addition to this, cosine similarity is computed by taking dot products between the query vector and each document vector in the TF-IDF matrix, followed by normalization of the query vector. The similarity scores for the retrieved documents are calculated and printed. In summary, this process involves taking a user query, finding the most similar documents based on TF-IDF vector similarity, and presenting both the documents and their cosine similarity scores.

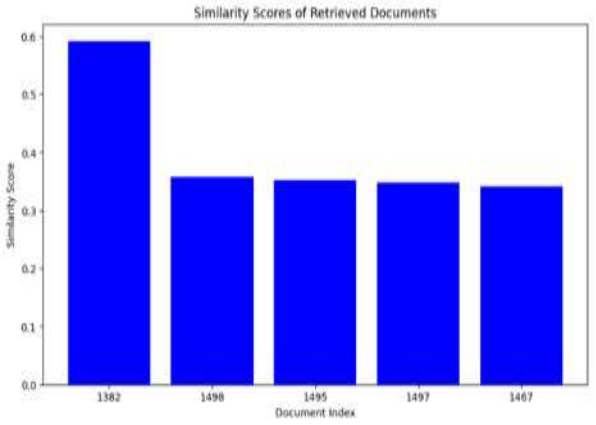

Figure. 11: Similarity scores of retrieved documents

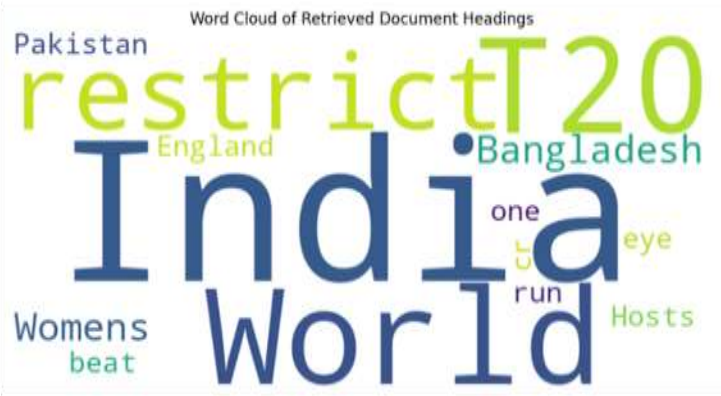

Figure. 12: Word cloud of retrieved documents headings

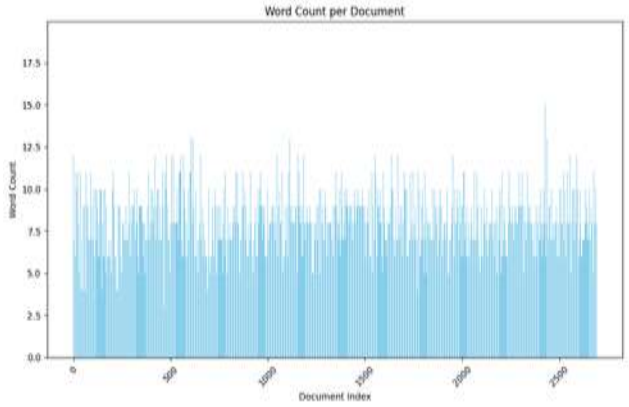

Figure.13: Word count per document

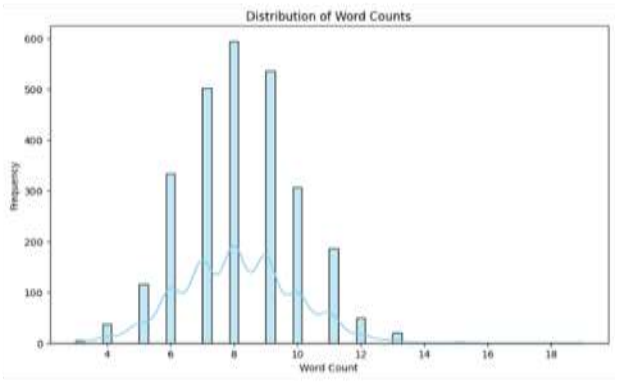

Figure.14: Distribution of word count

The decision between the two methods—using BM25 with inverted index or TF-IDF with FAISS—is based on a number of variables, such as the size and kind of your dataset as well as the particular needs of your application. Here are some things to think about with each strategy:

TF-IDF combined with FAISS is well-suited for handling large datasets due to its ability to perform fast similarity searches in high-dimensional spaces, making it effective for big data applications. FAISS excels in scalability, efficiently managing extremely large-scale datasets. The similarity calculated with TF-IDF and FAISS is based on the embedded representation of documents, allowing for the identification of semantic similarities. However, preprocessing steps such as tokenization and TF-IDF vectorization can be computationally intensive, particularly for very large datasets.

On the other hand, BM25 with an inverted index offers a more straightforward approach to information retrieval. It is simpler to understand and implement, making it ideal for smaller datasets. While BM25 can handle large datasets, maintaining the inverted index can introduce overhead, potentially making it less efficient than FAISS for very large-scale data. BM25 provides flexibility in query processing by considering term relevance, but it requires parameter tuning—such as adjusting k1 and b values—to achieve optimal performance depending on the dataset and specific task.

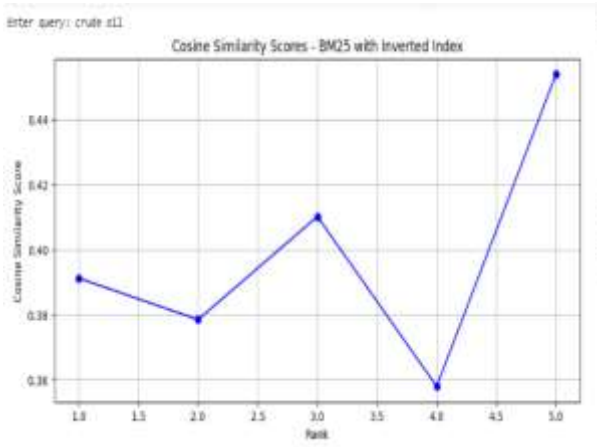

Figure. 15: Cosine Similarity for BM25

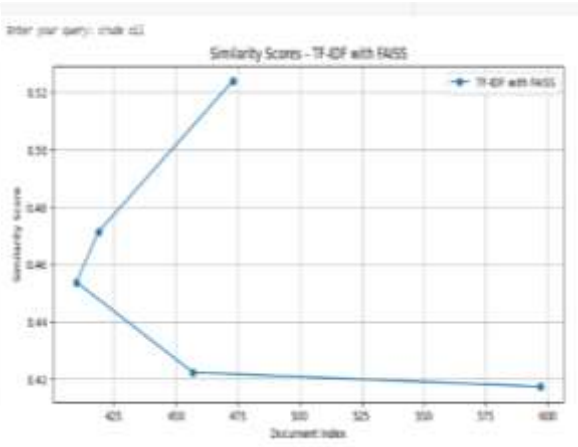

Figure.16: Cosine Similarity for TF-IDF with FAISS

In summary, TF-IDF combined with FAISS is an excellent choice for very large datasets where scalability is a primary concern. This approach enhances the accuracy of keyword and similarity searches by leveraging linguistic patterns and domain-specific knowledge tailored to each category. In this study, a model is proposed that employs keyword search and extraction techniques specifically designed for news articles. It extracts relevant responses from filtered text chunks using a map-reduction technique, which addresses token limit and space issues. By first dividing the components and then using LLM and FAISS to filter out relevant information, the model summarizes the filtered portions with the final LLM. This process ensures precise and efficient results by querying the news URLs and sources.

In our implementation, we begin by leveraging the RetrievalQA Sources Chain with OpenAI's API key and FAISS vector index. Initially, we load and split our documents, creating embeddings using OpenAI's embeddings. These embeddings are then used to build the FAISS vector index. When a query is posed, the RetrievalQA Sources Chain processes it with summaries and prompts to retrieve relevant answers. We have designed an LLM prompt using LangChain that integrates all necessary modules. This setup allows us to handle chunks of text, filter them, and generate accurate responses based on the provided prompts.

Figure.17: RetrievalQA sources chain and LLM prompt

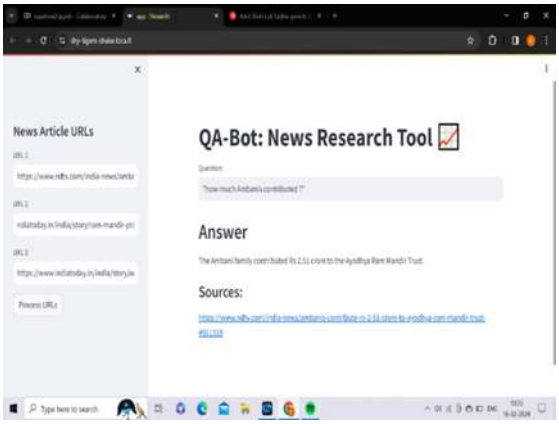

Fig. 18: QA-bot

The QA-Bot is built by integrating the aforementioned components into a cohesive system. This involves combining document loading, splitting, embedding, FAISS indexing, and retrieval processes using various modules and packages. The final implementation is realized through a Streamlit application, where the app.py file is run and hosted on a local network at port 8501. This application serves as an interface for the LLM-based news research tool using LangChain. The QA-Bot efficiently retrieves and provides accurate answers to user queries, demonstrating a practical and budget-friendly project in the fields of NLP, AI, and ML. This project adds significant value for engineering students, especially in the current landscape where AI applications like ChatGPT are rapidly advancing.

VI. CONCLUSION

By utilizing FAISS and TF-IDF, we enhance keyword search accuracy through the application of linguistic patterns and domain-specific knowledge tailored to each category. While TF-IDF methods might overlook domain-specific nuances and fail to fully capture the complexities of different topics, our model optimizes these approaches by integrating domain-specific features. This refinement enables a more accurate representation of responses derived from news articles in each category.

Our application ensures efficient textual data extraction by supporting multiple document formats, including text files, CSV files, and URLs, thus facilitating comprehensive data collection from diverse sources. It maintains semantic integrity and coherence by employing a text splitter with recursive characters, which allows for detailed content analysis. Advanced embedding generation is achieved through a combination of Hugging Face and OpenAI technologies, producing embeddings from textual data using cutting-edge pre-trained language models. These embeddings numerically represent sentences for further processing. Additionally, we have developed an intuitive user interface to enhance user interaction, allowing for seamless query input and improving the overall user experience.

VII. FUTURE WORK

Future research could focus on optimizing the proposed system by incorporating deep learning techniques, which may enhance its capabilities in tasks such as topic modeling, document retrieval, and summarization. This could also include exploring the application of the model for news content in different languages and domains, such as regional news retrieval.

Improvements could be made to the user interface to ensure it is more intuitive, allowing users to easily input queries, examine results, and visualize insights. Additionally, implementing real-time data processing capabilities would enable continuous monitoring and analysis of news articles as they are released, providing timely insights and updates. Domain-specific customization could be introduced, allowing users to tailor the tool to their specific needs and industries. Lastly, integrating with external APIs and data sources, such as social media, financial, and geopolitical data, could enhance the analysis and offer a more comprehensive understanding of news events.

VIII. REFERENCES

[1] Khan, N., Islam, M. S., Chowdhury, F., Siham, A. S., & Sakib, N. (2022, December). Bengali crime news classification based on newspaper headlines using NLP. In *2022 25th International Conference on Computer and Information Technology (ICCIT)* (pp. 194-199). IEEE.

[2] Saxena, A., Bhagat, V. V., & Tamang, A. (2021, August). Stock market trend analysis on Indian financial news headlines with natural language processing. In *2021 Asian Conference on Innovation in Technology (ASIANCON)* (pp. 1-5). IEEE.

[3] Ghasiya, P., & Okamura, K. (2021, January). Investigating Cybersecurity News Articles by Applying Topic Modeling Method. In *2021 International Conference on Information Networking (ICOIN)* (pp. 432-438). IEEE.

[4] Lwin, S. S., & Nwet, K. T. (2018, November). Extractive summarization for Myanmar language. In *2018 international joint symposium on artificial intelligence and natural language processing (iSAI-NLP)* (pp. 1-6). IEEE.

[5] Alam, K. M., Hemel, M. T. H., Islam, S. M., & Akther, A. (2020, December). Bangla news trend observation using lda based topic modeling. In *2020 23rd International Conference on Computer and Information Technology (ICCIT)* (pp. 1-6). IEEE.

[6] Kosmajac, D., & Kešelj, V. (2019, March). Automatic text summarization of news articles in serbian language. In *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)* (pp. 1-6). IEEE.

[7] Priyadharshan, T., & Sumathipala, S. (2018, December). Text summarization for Tamil online sports news using NLP. In *2018 3rd international conference on information technology research (ICITR)* (pp. 1-5). IEEE.

[8] Deny, J., Kamisetty, S., Thalakola, H. V. R., Vallamreddy, J., & Uppari, V. K. (2023, May). Inshort Text Summarization of News Article. In *2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1104-1108). IEEE.

[9] Mishra, A., Sahay, A., anjusha Pandey, M., & Routaray, S. S. (2023, March). News text Analysis using Text Summarization and Sentiment Analysis based on NLP. In *2023 3rd International Conference on Smart Data Intelligence (ICSMDI)* (pp. 28-31). IEEE.

[10] Habu, R., Ratnaparkhi, R., Askhedkar, A., & Kulkarni, S. (2023, September). A Hybrid Extractive-Abstractive Framework with Pre & Post-Processing Techniques To Enhance Text Summarization. In *2023 13th International Conference on Advanced Computer Information Technologies (ACIT)* (pp. 529-533). IEEE.

[11] Boorugu, R., & Ramesh, G. (2020, July). A survey on NLP based text summarization for summarizing product reviews. In *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)* (pp. 352-356). IEEE.

[12] Kynabay, B., Aldabergen, A., & Zhamanov, A. (2021, April). Automatic summarizing the news from inform. kz by using natural language processing tools. In *2021 IEEE International Conference on Smart Information Systems and Technologies (SIST)* (pp. 1-4). IEEE.

[13] Mayopu, R. G., Nalluri, V., & Chen, L. S. (2023, November). Classification ChatGPT Generated News and True News Using Support Vector Machines. In *2023 12th International Conference on Awareness Science and Technology (iCAST)* (pp. 228-232). IEEE

[14] Hingle, A., Katz, A., & Johri, A. (2023, October). Exploring NLP-Based Methods for Generating Engineering Ethics Assessment Qualitative Codebooks. In *2023 IEEE Frontiers in Education Conference (FIE)* (pp. 1-8). IEEE.

[15] Jayasiriwardene, T. D., & Ganegoda, G. U. (2020, September). Keyword extraction from Tweets using NLP tools for collecting relevant news. In *2020 International Research Conference on Smart Computing and Systems Engineering (SCSE)* (pp. 129-135). IEEE.