DAYANANDA SAGAR UNIVERSITY

KUDLU GATE, BANGALORE – 560068

Bachelor of Technology
in
COMPUTER SCIENCE AND ENGINEERING

Major Project Phase-II Report

NLP BASED NEWS SUMMARIZER

By

Chatterjee Keshav- ENG19CS0071
Harshita Chaurasia- ENG19CS0115
Shushant Rishav- ENG19CS0306

Under the supervision of

Dr. Meenakshi Malhotra
Associate Professor Dept. of CS&E

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY,
BANGALORE

(2022-2023)

DAYANANDA SAGAR UNIVERSITY

School of Engineering
Department of Computer Science & Engineering
Kudlu Gate, Bangalore – 560068
Karnataka, India

CERTIFICATE

This is to certify that the Phase-II project work titled **"NLP BASED NEWS SUMMARIZER"** is carried out by **Chatterjee Keshav Kanchan(ENG19CS0071), Harshita Chaurasia (ENG19CS0115), Shushant Rishav (ENG19CS0306) a** bonafide students of Bachelor of Technology in Computer Science and Engineering at the School of Engineering, Dayananda Sagar University, Bangalore in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Engineering, during the year **2022-2023**.

Dr. Meenakshi Malhotra	**Dr. Girisha G S**	**Dr. Udaya Kumar Reddy K R**
Associate Professor Dept. of CS&E, School of Engineering Dayananda Sagar University	Chairman CSE School of Engineering Dayananda Sagar University	Dean School of Engineering Dayananda Sagar University
Date:	Date:	Date:

Name of the Examiner	**Signature of Examiner**
1.	
2.	

DECLARATION

We, **Chatterjee Keshav Kanchan (ENG19CS0071), Harshita Chaurasia (ENG19CS0115), Shushant Rishav (ENG19CS0306)** are students of eighth semester B. Tech in **Computer Science and Engineering**, at School of Engineering, **Dayananda Sagar University**, hereby declare that the Major Project Stage-II titled **"NLP BASED NEWS SUMMARIZER"** has been carried out by us and submitted in partial fulfilment for the award of degree in **Bachelor of Technology in Computer Science and Engineering** during the academic year **2022-2023**.

Signature

Student
Name1: Chatterjee Keshav Kanchan
USN : ENG19CS0071

Name2: Harshita Chaurasia
USN : ENG19CS0115

Name3: Shushant Rishav
USN : ENG19CS0306

Place : Bangalore
Date :

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to School of Engineering & Technology, Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor's degree in this institution.

We would like to thank ***Dr. Udaya Kumar Reddy K R, Dean, School of Engineering & Technology, Dayananda Sagar University*** *for his constant encouragement and expert advice.*

It is a matter of immense pleasure to express our sincere thanks to ***Dr. Girisha G S, Department Chairman****,* ***Computer Science and Engineering****,* ***Dayananda Sagar University,*** *for providing right academic guidance that made our task possible.*

We would like to thank our guide ***Dr.Meenakshi Malhotra.****,* ***Associate Professor****,* ***Dept. of Computer Science and Engineering****,* ***Dayananda Sagar University****, for sparing his/her valuable time to extend help in every step of our project work, which paved the way for smooth progress and fruitful culmination of the project.*

We would like to thank our ***Project Coordinators Dr. Meenakshi Malhotra*** *and* ***Dr. Pramod Kumar Naik*** *as well as all the staff members of Computer Science and Engineering for their support.*

We are also grateful to our family and friends who provided us with every requirement throughout the course.

We would like to thank one and all who directly or indirectly helped us in the Project work.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
RNN	Deep Learning
GUI	Graphical User Interface
LSTM	Long short-term memory
MBPS	Megabits per second
DAS	Data Analytics Suite
TF-IDF	Term Frequency-Inverse Document Frequency
BERT	Bidirectional Encoder Representation from Transformers

LIST OF FIGURES

ABSTRACT

There is enormous amount of electronic data around us. For instance, there are plenty of news articles but a small amount of useful information in the articles and it is hard to read all the articles and find informative news manually. One of the solutions to this problem is to summarize the news in the article. A summary concentrates on a lengthy document by highlighting salient features. It helps the reader understand entirely just by reading the summary so that the reader can save time and decide whether to go through the entire document. Summaries should be shorter than the initial article to make sure to select only pertinent information to include in the article. The main goal of a newspaper article summary is for the readers to understand what the article is all about without needing to read the entire article.

The project goal is to build a Django-based automatic text summarization model with RNN and deep learning algorithms such as LSTM that can output one or two-sentence summarizations for a given article with high accuracy.

CHAPTER 1

INTRODUCTION

CHAPTER 1 INTRODUCTION

News summarization is the process of reducing the length of an article into a shorter form and still providing all the important information of the article. By just reading the summary of the article a user will be able to get all information present in the article. There are many reasons why news summaries are important it saves time because they provide a quick and concise overview of the latest events, allowing people to stay informed without having to spend a lot of time reading lengthy articles. This also enhances the understanding of the news by reducing lengthy news into short and precise pieces of information. They make people aware of news in different domains because people can read multiple news articles in a very short span of time and still be able to read important information.

The advancement in the areas of NLP has enhanced the effectiveness of news summarization. Currently, experts are incorporating both NLP and ML methods to construct advanced news summarization systems.

1.1. OBJECTIVE

The aim of this project is to develop a news summarizer model that can efficiently extract the most important information from news articles and generate concise summaries. The primary objective of this model is to provide users with a quick and efficient way to stay informed about current events.

The news summarizer model will be integrated into a website to offer users a convenient and user-friendly platform for accessing news content. The website will provide a good user experience by featuring a user-friendly interface, categorized news feeds, and options for choosing news from multiple categories.

1.2. SCOPE

In recent years the quantity of text data available has increased dramatically from a variety of sources. This huge volume of literature features a wealth of information and knowledge that must be adequately summarized in order to be useful. The goal of the proposed project is to create a web application that will provide a summary of a news article based on searched keywords entered by the reader It will provide news from all the major categories i.e., Crime, Sports, and Politics while preserving the actual context of the articles. The model will be able to summarize the content by reducing it to 70-80% depending on the category of news. User shall have an option to read entire article or just by summary.

CHAPTER 2
PROBLEM DEFINITION

CHAPTER 2 PROBLEM DEFINITION

The rapid increase in daily news has made it challenging for people to stay current and aware. Many people prefer to read news belonging to a particular category like sports, politics, crime, etc. The complexity and length of news articles exacerbate this issue, leading to an excess of information and missed important happenings. People in today's quick-paced society usually don't have the patience or time to read in-depth articles, resulting in a lack of focus and enthusiasm for the news.

Thus, the demand for news summarization across multiple categories is growing. These methods would give individuals a brief and clear summary of the critical information in a news article across multiple categories, enabling them to keep up with current events without dedicating hours to reading every article and reading their news of interest.

CHAPTER 3
LITERATURE REVIEW

CHAPTER 3 LITERATURE REVIEW

The author talks about text summarization techniques where the study has been conducted to find whether the article should be summarized or not. The author performs experiments using news articles and applies data mining techniques such as C4.5 and Naïve Bayes to model and execute the automatic summarization. It majorly focuses on the need to summarize predictions.[1]

As many people and devices generate enormous data in their work the authors have proposed a method to model the pattern of users' summarization needs on news articles. The author presents a solution to distinguish useful news from unnecessary articles is to implementing automatic summarization. In order to know the appropriate length of an article to be summarized and to measure the good metric of article length they collect data from users. To get a rough idea about whether this article should be summarized or not and learn to predict using the data mining tool WEKA.[1]

There are four steps in the approach of the proposed method firstly to collect the URL of an article selected by users for the summarization secondly to extract the attributes from the collected data. Further to apply data mining techniques using the WEKA tool and generate a decision model. Finally, to measure the generalization ability of the model using test data.[1]

This paper presents a method to predict users' desire for summarization based on some training samples with readers' explicit labelling. To save computing resources this paper presents what kind of article should be summarized for user's needs. They have collected data from five graduate studies using news articles and after that, the model has a pattern of service requests and the model's precision is nearly 90% considered to be ideal by removing ads manually. Few limitations have been discussed in the work as it has considered only one news portal. Secondly, due to the difficulty of processing they've not considered a few attributes such as the meaning of a text, the importance of news articles and so on to reduce the complexity of attributes. [1]

The main goal of a newspaper article summary is, for the readers to walk away with knowledge of what the newspaper article is all about without the need to read the entire article. The system uses a tailor-made web crawler which crawls websites for searching relevant articles to male ad-hoc keyword-based extraction of news articles. The paper presents computational linguistic techniques mainly triplet extraction, semantic similarity calculation and OPTICS clustering with DBSCAN used alongside. The performance evaluation is done using ROUGE metric.[2]

The author talks about the importance of text mining to manage useful information from unstructured data such as news reports, emails and web pages. As text mining techniques have mostly been developed for the English language because most electronic data is in English. The summarization process various methods such as filtering, highlighting and organizing information which is meaningful. The key tasks in text mining are to automatically extract on-line articles from news sites based on a keyword.[2]

To achieve automatic summarization is to devise an application which automatically collects digital online news articles based on keywords and then summarized them using natural language processing techniques. The mechanism of this paper is first to collect data for building corpora of news articles. Further to develop a mechanism for breaking down actual sentences and semantically similar sentences to derive non-redundant summaries of them. By using the ROUGE score evaluating the systems' generated summaries. It has proposed to use sentence tokenization which refers to the practice of dividing a text into a group of sentences of tokens. Secondly, Triplet Extraction where the triplet consists of a subject and an object. Lastly, the paper proposes another method for named entity recognition and stemming where stemming is the process of reducing inflected words to their word stem, base or root form.[2]

As it is observed the summarization doesn't pertain to the query submitted by the user's custom search engine. The works present a system to automatically collect, collate and summarize online news articles based on a user-submitted query using sentence tokenization, Stemming and lemmatization. The scores were evaluated on the DUC 2001 dataset resulting in an average F-score of 0.179 .[2]

The paper implements an unsupervised approach. Using graph-based ranking algorithms and modifying how sentence centrality is computed. BERT, a state-of-the-art neural representation learning model to better capture sentential meaning. Experimental results on three news summarization datasets representative of different languages and styles show that the approach outperforms strong baselines by a wide margin .[3]

The most important thing while single-document summarization is the task of generating a shorter version by retaining its content. In this paper, they have argued upon centrality measures that can be improved in two important aspects. Firstly, to capture the better sentential meaning and compute sentence similarity, they have employed BERT which is a neural representation learning model that has obtained state-of-the-art results. This paper discusses the approach of three single-document news summarization datasets representative of different languages. Position information has been used in summaries regularly, particularly in news reports, the author talks about creating a baseline for summary by selecting n sentences of the document or as a feature in learning-based systems.[3]

This paper has approached in centrality-based summarization, where the first one classifies as an undirected text graph which is a prominent class of approaches in unsupervised summarization that uses graph-based ranking algorithms to determine a sentence eminence for inclusion in summary. A document (or a cluster of documents) is represented as a graph, in which nodes correspond to sentences. A node's centrality can be measured by computing its degree or running a ranking algorithm such as PageRank. Secondly, they have outlined a directed text graph where the idea lies in textual units and RST which is the notion of nuclearity has been leveraged extensively in document summarization. The author has discussed sentence similarity computation where they have represented BERT as finetuned where they've computed similarity between sentences in a document, further they have used BERT as a sentence encoder and obtained representations for sentences in the document and used the cosine similarity.[3]

The paper has an automatic evaluation using ROUGE F1 and has reported unigram and bigram overlap, where they have developed unsupervised summarization which has very modest data requirements and a revised graph-based ranking algorithm.[3] Summarization of text is one of the most challenging tasks in the field of natural language processing. In this paper the data from English and Telugu newspapers is extracted, now compare the missing news and then summarize the bi-language data.[4]

Information is of two types informative and non-informative. That is most relevant in query processing for single and multiple documents. Multi-document requires high redundancy and then extraction of the data from two different language newspapers and summarizing it is more complicated. The Summary of a document is a representation of the text which seeks to render the exact idea of its contents. The main types of automatic summarization include extraction-based, abstraction-based and maximum entropy-based. They can be divided into three types: surface-level, intermediate-level and deep parsing techniques.[4]

The paper discusses the preprocessing of text using python scripts and also scraps the data from English and Telugu news articles. The data is preprocessed by using word and sentence tokenization, removing the irrelevant data and then using the text rank approach. Translate the Telugu article to English and again implement the bi-language summarization. The author has described that preprocessing is the most crucial step in data mining and analyzing the data. Further, stemming has been discussed as the information retrieval to describe the process for reducing the inflected word to their word stem. Later on, the noisy data and consistent data are removed so it has classified into correct and incorrect data and the TF-IDF weighting approach is used to eliminate the common words in the document by using term frequency and inverse document frequency.[4]

This paper represents a machine learning approach using NLP for mining newspaper information and using bi-language and summaries that will be meaningful information such that the end user can read the missed data efficiently. The paper has experimented by first English text is summarized and then the Telugu text translated into English. The translated text is then summarized. Future enhancements are represented by using a supervised learning approach with multi-language summarization to obtain better accuracy.[4]

Text summarizing is one of the jobs that has received a lot of attention. Research is currently shifting to abstractive summarization and real-time summarization. The results of the analysis provide an in-depth explanation of the trends that are the focus of their research.[5]

Text summarization is the act of condensing a long text into a shorter, more accurate, and fluent set of phrases that is easy to grasp. Text summarization can be used in news stories, emails, research papers, and online search engines to get a quick summary of the results. The majority of text summary applications are extractive, with only a few producing abstractive summaries. Extractive summarization selects the most important and correct sentences but suffers from incoherency. Abstractive provides a word-by-word summary that is readable but may lose significant data in large manuscripts.[5]

The paper discusses two types of text summarization one is extractive summarization and other is abstractive summarization. They have discussed the method of automatic summarization which is abstractive, multi-lingual, recommendation based and hybrid method. Further, they have identified digital libraries and their respective journal papers and defined the search strategy and search strings. The author has mentioned finding the majority of required citations if yes then gather the final list of papers from cited websites and journals and important papers. A systematic Literature Review was used to perform this review research on text summarizing and has performed selection criteria for research papers concerning news summarization such as site-based, search string based and timeline-based.[5]

This paper has discussed the methods to navigate around the summarization of news articles that is a user-friendly interface for sorting through queries and filters and to use a web crawler to automate the extraction of online-based news articles. Then characteristics of an improved, efficient summary extraction technique: The system must adapt to each type of data format, learning as it encounters new forms and minimizing processing time. Further the characteristics of abstractive news summarization: Sentences and paragraphs must be connected, and a logical and grammatical flow must be established. The pointergenerator model is easier to train than the sequence-to-sequence attention system and resulted in the ideal method by approaching abstractive news summarization.[5]

The dual-attention mechanism extracts both important information highlighted by linking tweets and the salient content in news. DAS outperforms state-of-the-art unsupervised models and achieves comparable results with supervised models. The paper propose positiondependent word salience, which reflects the effect of local context on a sentence.[6]

Comments or linking tweets, together with news, open opportunities to study what information news attracts readers' attention. In recent years, many news websites have closed their online reader comments sections. Linking tweets can be employed to extract important information in news, while the news is still needed to explore for left salient content. This paper proposes a tweet-aware unsupervised extractive summarization model with a dual-attention mechanism, named DAS. They capture important information highlighted by linking tweets and salient content in news simultaneously. In this work, the author has proposed an unsupervised tweet-aware news summarization model with the dual-attention mechanism. Results from both attention modules are projected to the word sequence of news, which reflects the relative importance of each token in news.[6]

The author has focused on the News-Tweet Attention module figuring out the key points in news highlighted by linking tweets. They have followed BERT to set up the encoder due to its great performance on many tasks. The attention mechanism typically works on capturing the alignment relationships between encoder input and decoder output. Later on, News-self Attention module, The autoencoder structure is similar to the encoderdecoder framework, but the inputs of the encoder and decoder are the same, i.e., the news article itself. And Word and Sentence Salience Estimation. The experiments are conducted on the data provided by Wei and Gao. They have 15,547 news-tweet pairs as training instances, which effectively enlarges this dataset to a relatively large one. They have compared DAS with both supervised and unsupervised baselines.[6]

All methods are evaluated by pyrouge package, which is a python wrapper of the official ROUGE metric. Their DAS can generate summaries with better word coverage than all the baselines and this is because of linking tweets. propose an unsupervised extractive summarization model named DAS.[6]

In this paper, the author has introduced a new way of digesting news articles by introducing the task of segmenting a news article into multiple sections and generating the corresponding summary for each section. They have created and made available a dataset of 27k news articles with sections and heading-style section summaries.[7]

In this paper, they have proposed a new task of Segmentation- based News Summarization. Given a news article, authors aim to identify its potential sections and at the same time generate the corresponding summary for each section. This new task provides a novel alternative to summarizing a news article. The paper argue that segmenting news articles can lead to a more organized way of understanding long articles and facilitates a more effective reading style. The framework can integrate many pretrained language generation models, including BART.[7]

They have worked upon document summarization which automatically generates a shorter version text and retains it important information. Document summarization can be classified into different paradigms by different factors. Extractive approaches form summaries by concatenating the most important spans in a document. In abstractive summarization, various text rewriting programs generate summaries using words or phrases that are not in the original text. Secondly, Text Segmentation and Outline Generation as it has been used widely in natural language processing and information extraction. TextTiling is one of the first unsupervised topic segmentation algorithms. The TopicTiling algorithm uses the Latent Dirichlet Allocation to find topical changes within documents. Later on, the authors have given engrossment in The SEGNEWS Benchmark where they studied to take CNN website as their article source and filter articles with no sub-topic structures. Since the first segment is an overview of the news, editors do not assign a summary to it. The resulting dataset contains 26,876 news articles.[7]

They have built a new benchmark dataset SEGNEWS to study and evaluate the task. Furthermore, The paper proposed a design for a segmentation-aware attention mechanism.[7]

The paper proposes a new as neutral summary generation from multiple news articles of varying political leanings to facilitate balanced and unbiased news reading. They have observed NLG models can hallucinate not only factually inaccurate or unverifiable content but also politically biased content.[8] Media framing bias can reinforce political polarization and undermine civil society’s rights. “Allsides” mitigates this problem by displaying articles from various media in a single interface along with an expert-written roundup of news articles. The author introduces to fill the research gap by implementing NEUS and aims to generate a biasfree summary.[8]

Firstly, they have studied extensively media bias where various fields were included. In the context of news reports, framing is about how an issue is characterized by journalists and how readers take the information to form their impression. Media Bias Detection is based upon natural language processing in which computational approaches are used for detecting media bias in a political context. They have proposed automatically neutralizing and summarizing partisan articles to produce a neutral article summary. News aggregation, by playing articles from different news outlets on a particular topic, is the most common approach to mitigate media bias. Multi-document summarization (MDS) aims to condense a set of documents into a short and informative summary. Many works have studied particular subtopics of the MDS task, such as agreement-oriented MDS. To add to this, they propose the NEUS task and create a new benchmark and have performed a few casestudy observations.[8]

Summarization models can reduce the framing bias to a certain degree but they cannot handle the lexical framing bias. This suggests that having good summarization performance (ROUGE1-R) does not mean the model is also neutral. One of the major contributors to high bias in the MDS models is probably hallucination because MDS models portray drastically poor hallucination performance than all the other models. This model demonstrates a stronger tendency to paraphrase rather than directly copy, and has a comparatively more neutral framing of the issue.[8]

CHAPTER 4
PROJECT DESCRIPTION

CHAPTER 4 PROJECT DESCRIPTION

The solution is to build a web application and a news summarizing model in which the news is summarized by the model and is displayed on the website in a categorical manner where users can freely browse news from any category and can read the summarized news. Also if the user desires to read the full news article they can click on the “Read more” button which redirects the user to the website the news is fetched from. The size of the News article is reduced with the help of the model which was built using the deep learning algorithm LSTM.

4.1. System Design:

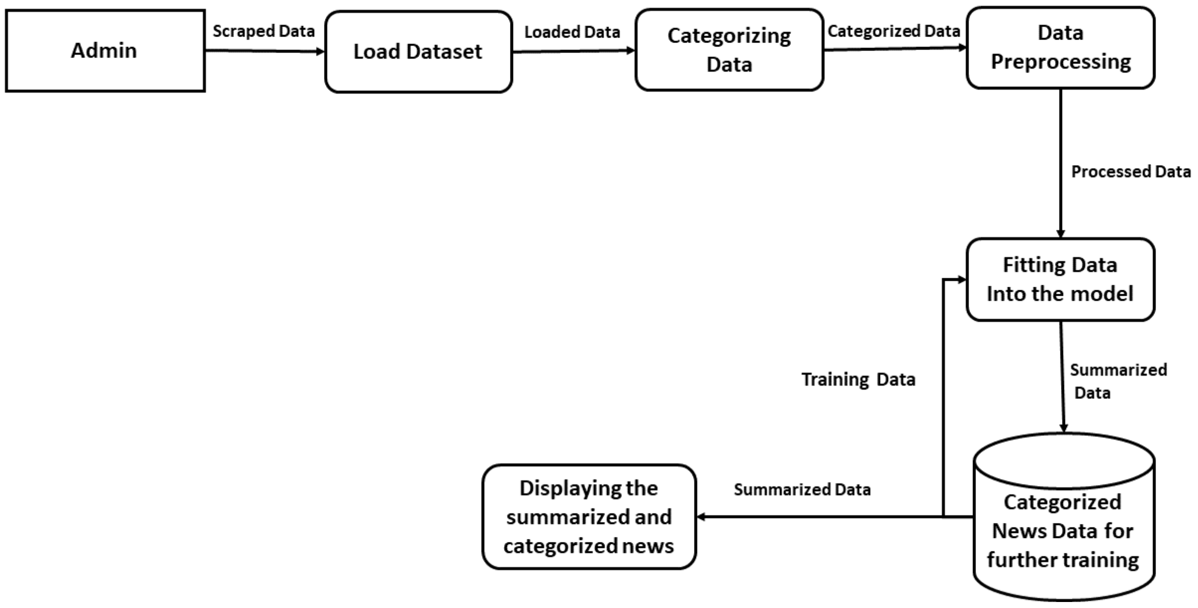


Figure 4.1.1 Dataflow Diagram for News Summarizer

Referring to Figure 4.1.1, the data flow for the project involves scraping data from various websites using a web scraper tool, loading it into a database, categorizing it, and generating summaries using a Bi-directional LSTM model. The summaries are saved to a CSV file which will be used to periodically retrain the model for improved accuracy. The output from model is then saved to disk and integrated with Django to display the generated summaries in real-time on the frontend dashboard or interface.

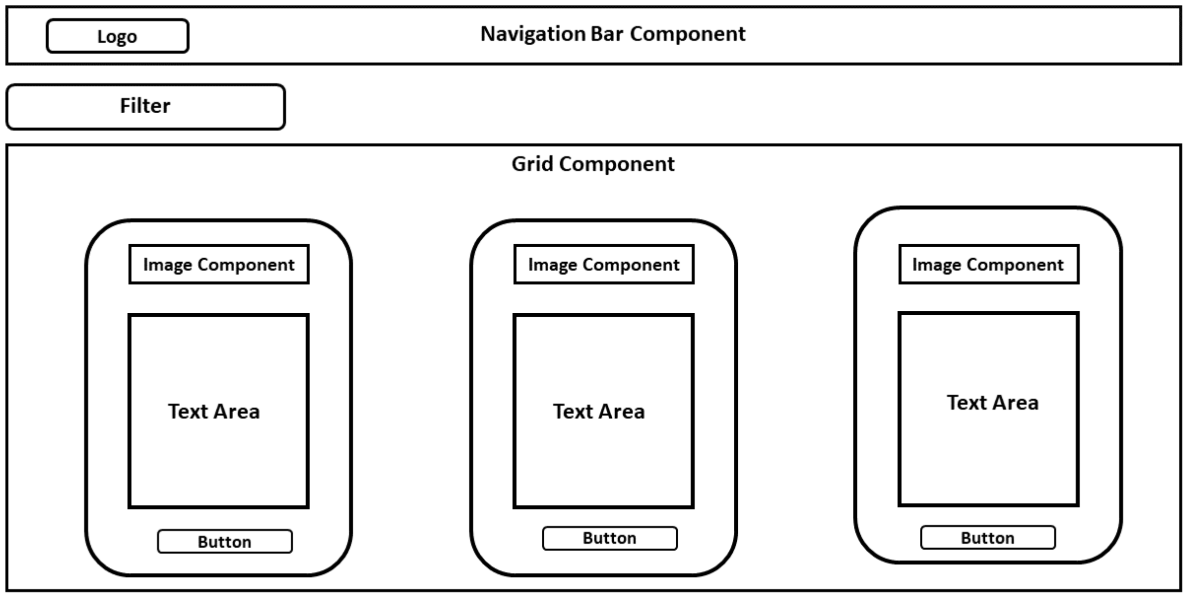


Figure 4.1.2 Screen Design for News Summarizer

The main screen is Figure 4.1.2 which is the design of the deployed application that consists of a navigation bar component displaying the different categories such as world affairs, business, technology and miscellaneous for better navigation on the web app and the logo of the web application.

The subsequent element in this system is a filter, which enables the user to choose from a range of news sources.

The third component comprises a grid containing multiple cards with news images, titles, and text areas that display date and time, heading, brief summaries, enabling users to quickly gain insights into the news. The "Read More" button on the card facilitates seamless navigation to the main article page.

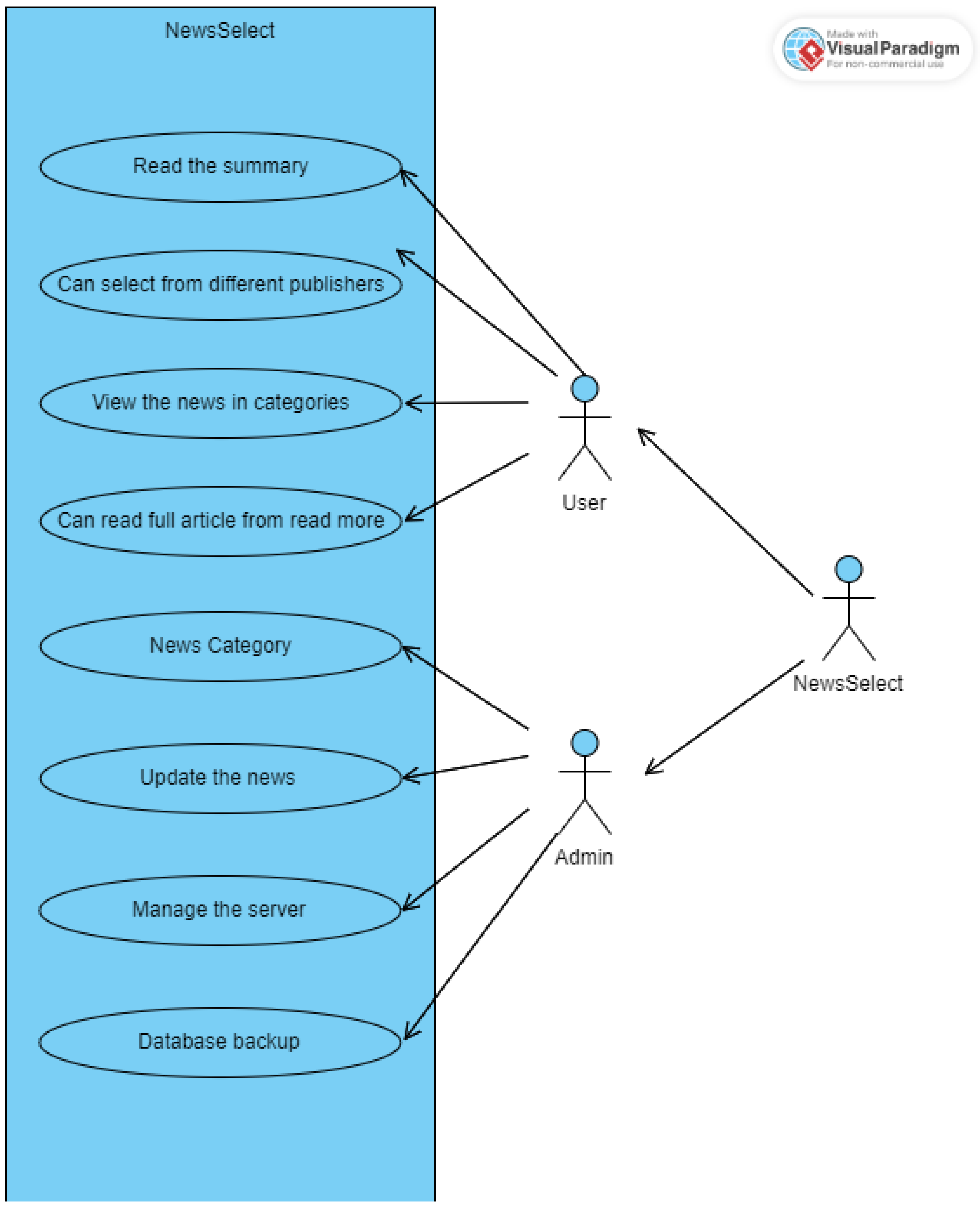

NewsSelect
Made with Visual Paradigm For non-commercial use
Read the summary
Can select from different publishers
View the news in categories
Can read full article from read more
News Category
Update the news
Manage the server
Database backup
User
NewsSelect
Admin

Figure 4.1.3. Use Case diagram of the project

Referring to Figure 4.1.3, the system allows users to read news summaries, select news from different publishers, view news in different categories, and read full articles by clicking the "Read More" button. The news can be categorized, updated, and managed by the admin. The server can be managed, and the database can be backed up for data protection by the admin.

4.2. ASSUMPTIONS AND DEPENDENCIES

- Data should be available to synchronize with the database of the system so that it'll be available to the model for text (news) summarization.
- GPU should be available for running the model.
- Availability of the server to host the website.

CHAPTER 5

REQUIREMENTS

CHAPTER 5 REQUIREMENTS

5.1. Functional Requirements

5.1.1. User-Interface

5.1.1.1. The user will be given a choice to choose between news in different categories on selecting the categories such as world affairs, technology, politics and miscellaneous.

5.1.1.2. Based on user preference the news will be displayed.

5.1.1.3. By using the interface, the user will be able to view the entire news or concise summary of the preferred article.

5.1.2. Training data

5.1.2.1. For training the system there will be a need of training data set.

5.1.2.2. The dataset consists of 4515 sample with five features each namely author_name, headlines, URL of article, short text and complete article.

5.1.2.3. In the project the dataset has been preprocessed by removing the unnecessary spaces, contractions and special characters.

5.1.3. Testing data

5.1.3.1. After the model is train, there will be a need of testing dataset for the model.

5.1.3.2. The dataset consists of 98,280 samples with two features headlines and complete article.

5.2. Non-Functional Requirements

5.2.1. Reliability:

5.2.1.1. The application must inform the admin in situations of a crash or error.

5.2.1.2. Should be up for working and should be able to fetch news from multiple sources.

5.2.2. Usability:

5.2.2.1. The application must be user friendly and easy to use.

5.2.2.2. The application must provide relevant information to its users.

5.2.3. Availability:

5.2.3.1. The application should be available whenever required.

5.2.3.2. News should be available in the application whenever required.

5.3 Software/System Requirements

5.3.1. Hardware Requirements

5.3.1.1. Minimum 16 GB RAM

5.3.1.2. At Least 8 GB of available disk space.

5.3.2. Software Requirements

5.3.2.1. 64-bit Windows 8-11 / macOS 10.14 or higher / 64-bit Linux

5.3.2.2. Visual Studio Code or similar code editor.

5.3.2.3. Python3.7 and above

5.3.3. Bandwidth Requirements

5.3.3.1. Bandwidth: 2-5 Mbps

CHAPTER 6
METHODOLOGY

CHAPTER 6 METHODOLOGY

6.1 About the dataset:

The dataset being used in this project is taken from Kaggle, comprises two CSV files that contain a collection of news articles and their corresponding headlines.

Over 44,000 news articles from various sources were collected, such as The Times of India, NDTV, and Hindustan Times, among others, are included in the first file named "news_summary.csv" Each article encompasses Author name, Headlines, URL of Article, Short text, Complete Article, the time period of data set lies between Feb – Aug 2017.

The second file named "news_summary_more.csv" consists of over 98,000 news articles from the same sources as the first file, but with longer summaries. Each article in this file contains a headline, a longer summary, and the full text of the article.

The dataset features being used in the project are ctext and text.

Data cleaning was performed to remove duplicates, errors, and missing values. Further, text preprocessing was done by removing punctuation, converting text to lowercase, removing stop words, and expanding contractions.

In next step, exploratory data analysis was performed to gain insights into the characteristics of the data.

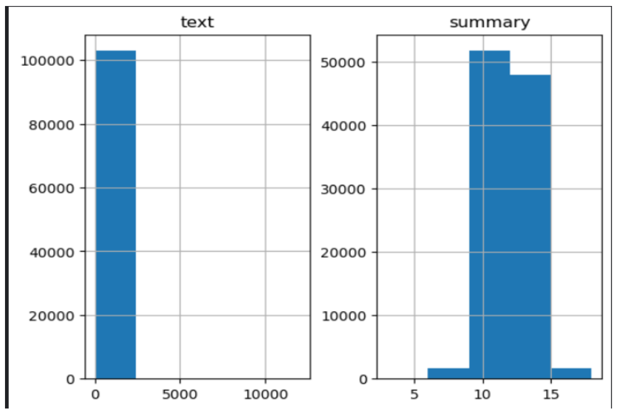


Figure 6.1 Exploratory Data Analysis of dataset

This project primarily used three models that were:

1. seq2seq model with lstm - using LSTM networks for both the

encoder and decoder.

2. Build seq2seq model with bidirectional lstm - using Bidirectional LSTM

networks for both the encoder and decoder.

3. Build hybrid seq2seq model - using Bidirectional LSTM networks for the

encoder and LSTM networks for the decoder.

The use of a sequence to sequence model with LSTM was explored, wherein the encoder and decoder components were implemented with LSTM Networks. Through experimentation, the model achieved a validation accuracy of 48% on 50 epochs, indicating some degree of success in utilizing this architecture for the given task. Subsequently, a more advanced sequence to sequence model was developed using Bidirectional LSTM Networks for both the encoder and decoder. After training the model on 50 epochs, an impressive accuracy of 82% was achieved, indicating that this architecture was more effective for the task at hand.

Further a hybrid sequence to sequence model was implemented, which utilized Bidirectional LSTM Networks for the encoder and only LSTM Networks for the decoder. The model was trained for 50 epochs, and it achieved a validation accuracy of 47%. This result indicates that using only LSTM Networks for the decoder was less effective than using Bidirectional LSTM Networks for both encoder and decoder.

The utilization of Bidirectional LSTM Networks in the encoder and decoder of a sequence to sequence model proved to be a more effective architecture for the given task. That's why the project was re-trained over Bidirectional LSTM further by tuning the hyper-parameters.

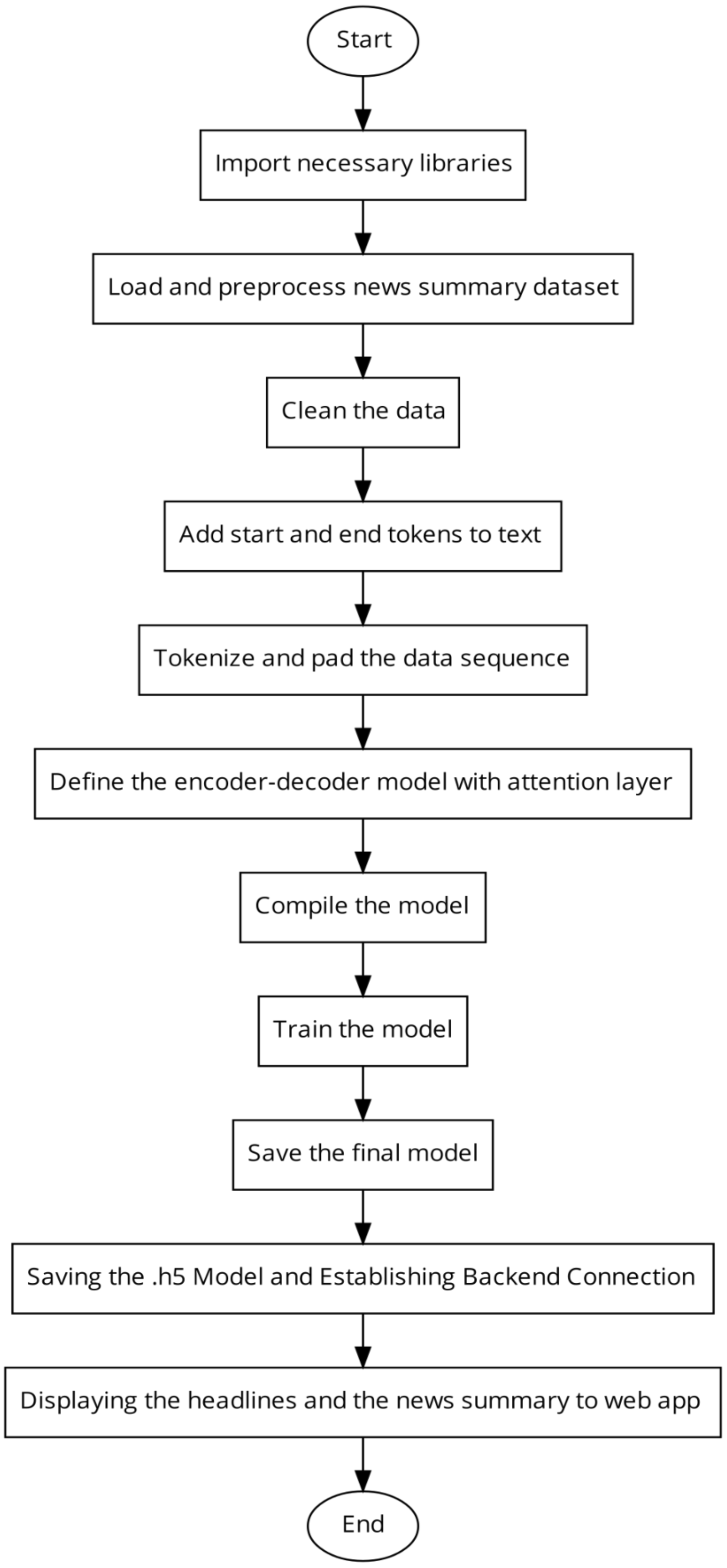

Start
Import necessary libraries
Load and preprocess news summary dataset
Clean the data
Add start and end tokens to text
Tokenize and pad the data sequence
Define the encoder-decoder model with attention layer
Compile the model
Train the model
Save the final model
Saving the .h5 Model and Establishing Backend Connection
Displaying the headlines and the news summary to web app
End

Figure 6.2 Flowchart of the project

Referring Figure 6.2,the process begins with importing necessary libraries and loading the news summary dataset. The data is preprocessed, cleaned, and tokenized, with start and end tokens added to the text. The encoder-decoder model with an attention layer is defined, compiled, and trained on the preprocessed data. The final model is saved, along with establishing a backend connection. Finally, the headlines and news summaries are displayed on a web app before the process ends.

CHAPTER 7
EXPERIMENTATION

CHAPTER 7 EXPERIMENTATION

The project aims to develop a deep learning model for text summarization. The main steps are as followed:

1) Importing the modules

   First, the required libraries are imported, including the necessary layers, models, and preprocessing modules from TensorFlow Keras.

2) Data Preparation

   Further for the project, news summary dataset is loaded and preprocessed by removing any rows with missing values, converting all columns to string data type, and creating a new data frame with columns "text" and "summary." The project used regex to get all contractions in the text and expand it. After which all matched sub-strings from the text have to be sorted out. The final step was to convert the remaining text into lowercase.

3) Data Cleaning
   - i In the first step all of the punctuations in the text are removed
   - ii In the next step all the stop-words from the text have to be removed
   - iii In the next step, all of the hyphen(–) which are present in titles, has to be
   - iv replaced with an empty string. As this hyphen(–) is different from the normal hyphen(-).
   - v In the next step all of the unnecessary characters such as white space, and new lines are removed.

4) This project uses sostok & eostok tokens which ensures the learning algorithm knows from where the text start and end.

5) Tokenization

   In the next step the text and summary columns are then tokenized and padded to create input and target sequences for the model. The model is defined using an encoder-decoder architecture with attention. The encoder takes the input sequence and returns the final encoder state, which is passed to the decoder along with the target sequence. The decoder uses the encoder state and attention mechanism to generate the output sequence and after that the maximum length for input and target sequences is calculated based on system configurations. This approach ensures that the summary makes sense and is not randomly generated from a piece of text.

a) The constraints at the time of the project:

max_text_len = 300 and max_summary_len = 60

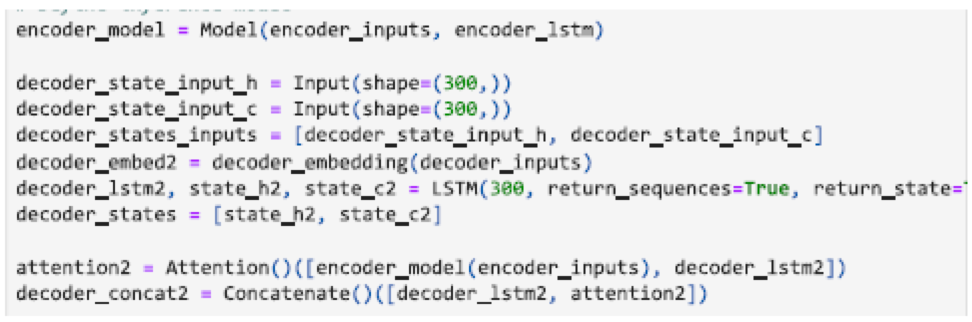
6) Dataset size: 27084

7) Model Training

The model was then compiled with the Adam optimizer and the sparse categorical cross-entropy loss function. Early stopping was used to prevent overfitting. The model is further trained on the preprocessed data .The final model is saved as a .h5 file.

8) Inference Model

This project defined an inference model to generate the summaries for new input texts. The encoder model takes the input sequence and returns the final encoder state.



```
encoder_model = Model(encoder_inputs, encoder_lstm)

decoder_state_input_h = Input(shape=(300,))
decoder_state_input_c = Input(shape=(300,))
decoder_states_inputs = [decoder_state_input_h, decoder_state_input_c]
decoder_embed2 = decoder_embedding(decoder_inputs)
decoder_lstm2, state_h2, state_c2 = LSTM(300, return_sequences=True, return_state=
decoder_states = [state_h2, state_c2]

attention2 = Attention()([encoder_model(encoder_inputs), decoder_lstm2])
decoder_concat2 = Concatenate()([decoder_lstm2, attention2])
```

Figure 7.8 Inference model

9) Decoder Model

This project implemented a decoder model that took the encoder state and the target sequence and generates the output sequence using the attention mechanism.

10) Plotting the loss and accuracy of model

Then project was implemented to plot the accuracy and loss history of the model using matplotlib.

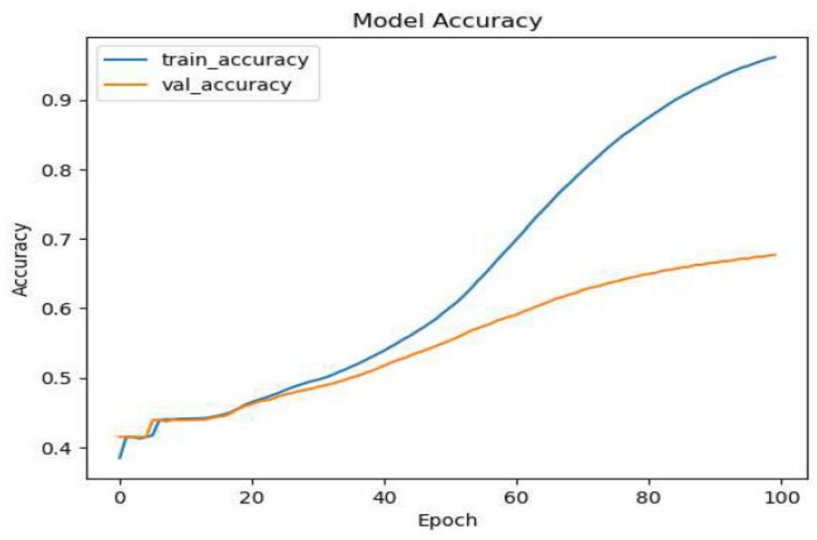


Figure 7.10.1 Accuracy of Bi directional LSTM

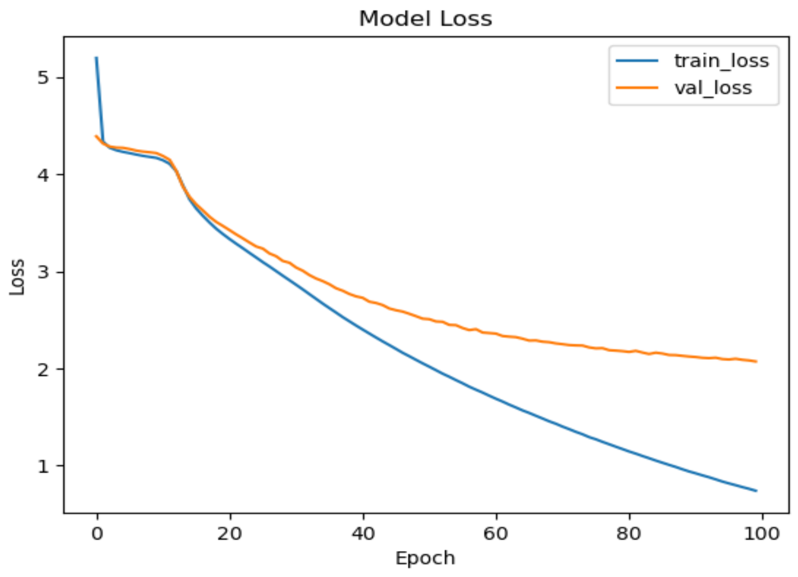


Figure 7.10.2 Loss of Bidirectional LSTM

11) Saving the .h5 Model and Establishing Backend Connection

The project was deployed using Django and the news summary is displayed in the web application ,where the users can read the summaries of the article.

12) Curating News Content from Multiple Portals

    The project is designed to procure real-time articles from reputable news portals such as Hindustan Times, The News Minute, News18, The Guardian, and The Print. The complete text is loaded onto a CSV file on the backend, along with accompanying images, headlines, dates, times, full article text, and "Read More" URLs for AJAX retrieval. These data are then transmitted to the website, where they are processed using a Bi-Directional LSTM text summarization model, and displayed to the user interface in the form of news summaries. The project has a frontend filter component that allows users to select and read news exclusively from their preferred news portals, such as Hindustan Times, The News Minute, News18, The Guardian, and The Print. This feature, coupled with the project's ability to procure real-time articles, enables users to personalize their news feed according to their interests and preferences. Moreover, the news summaries on the website are updated every 30 minutes to ensure that users have access to the latest news content.

13) Categorizing News Content for Efficient Reading

    The project incorporates a category system that includes topics such as world affairs, politics, technology, and miscellaneous. Users can conveniently navigate through these categories by selecting the desired option from the website's navigation bar and read the news summaries.

The process of creating a mechanism to extract real-time data for categorized news can be broken down into five main steps, which are as follows:

Step 1 Involves importing necessary libraries and defining some variables.

Step 2 Entails scraping news data from a website. This is done by creating headers to simulate a browser request, sending a request to the website using the URL and headers created in the previous step, parsing the content of the response using BeautifulSoup, and storing it in a dictionary called "dict." A function called "storedata" is created to store the parsed news data in the "dict" dictionary.

Step 3 Involves fetching more news data through Ajax calls. The script finds the "start_id" value from the previous response, which is used to fetch additional news data through Ajax calls. For each iteration of the loop, an Ajax request is sent to the website to fetch more news data. The "storedata" function is called to parse the Ajax response and store the news data in the "dict" dictionary. After every 10 iterations, the news data is saved to a CSV file.

Step 4 Entails fetching news content from each news article. The program loops through the headlines and read_more links from the previously scraped news data. For each news article, a request is sent to its corresponding read_more link to fetch its full content. The content is parsed using BeautifulSoup, and the relevant data is stored in the "d" DataFrame. If the read_more link corresponds to a specific news website, the program uses a specific set of instructions to parse the content.

Step 5 Involves data cleaning and storage. In this final step, the "d" DataFrame is cleaned to remove any duplicate rows. A new column called "ctext" is added to the DataFrame to store the full content of each news article. The cleaned DataFrame is saved to a CSV file, and the original file is deleted.

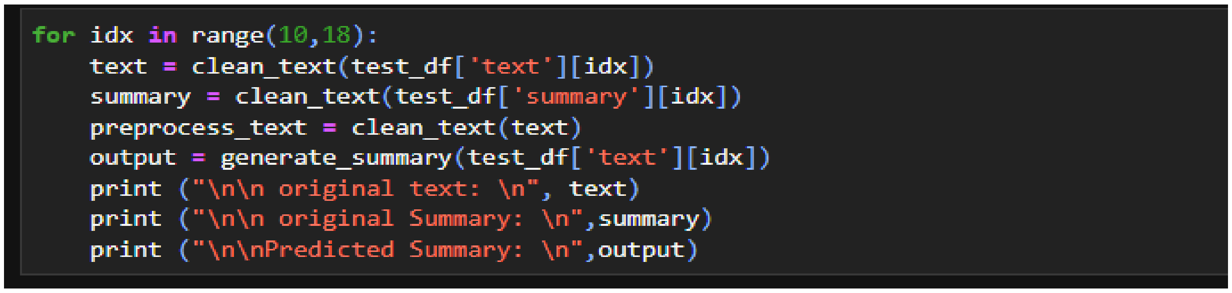
CHAPTER 8

TESTING AND RESULTS

CHAPTER 8 TESTING AND RESULTS

8.1 TESTING

Testing the dataset on Bi-directional model



```
for idx in range(10,18):
    text = clean_text(test_df['text'][idx])
    summary = clean_text(test_df['summary'][idx])
    preprocess_text = clean_text(text)
    output = generate_summary(test_df['text'][idx])
    print ("\n\n original text: \n", text)
    print ("\n\n original Summary: \n",summary)
    print ("\n\nPredicted Summary: \n",output)
```

Figure 8.1 Sample Code for Experimentation

Accuracy:

-

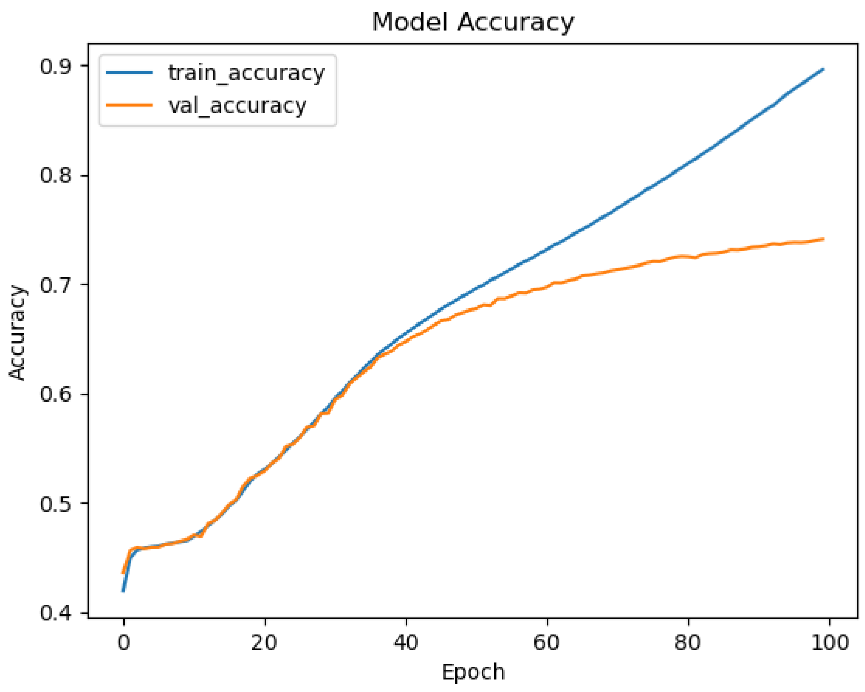


Figure 8.2 Accuracy for Bi-directional LSTM

Fig 8.2 depicts accuracy graph for bidirectional at the start of training, the model's accuracy on the training data is 41.93%, and its validation accuracy is 43.60%. After 100 epochs, the training accuracy improves to 89.62%, while the validation accuracy improves to 74.08%.

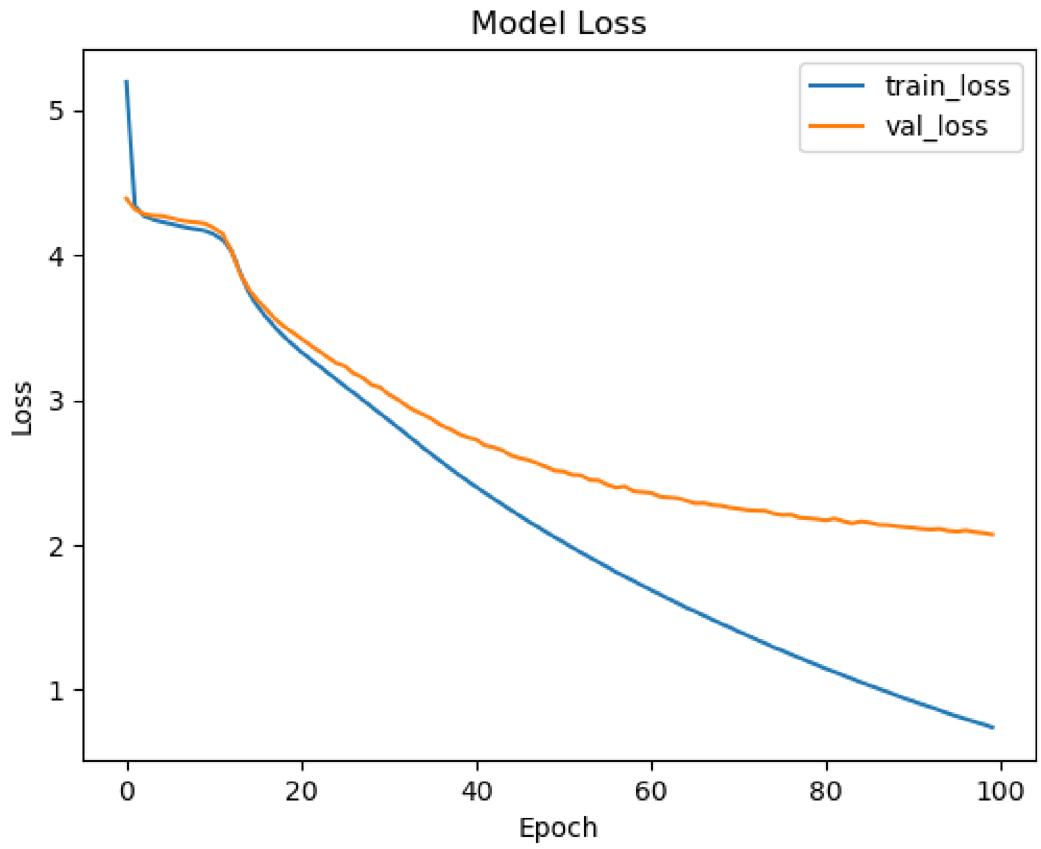


Figure 8.3 Loss for Bi-directional LSTM

Figure 8.3 depicts the training loss decreases from 5.196 to 0.742, while the validation loss decreases from 4.389 to 2.072. This indicates that the model has learned to make better predictions as it was trained with more data.

8.2. RESULTS

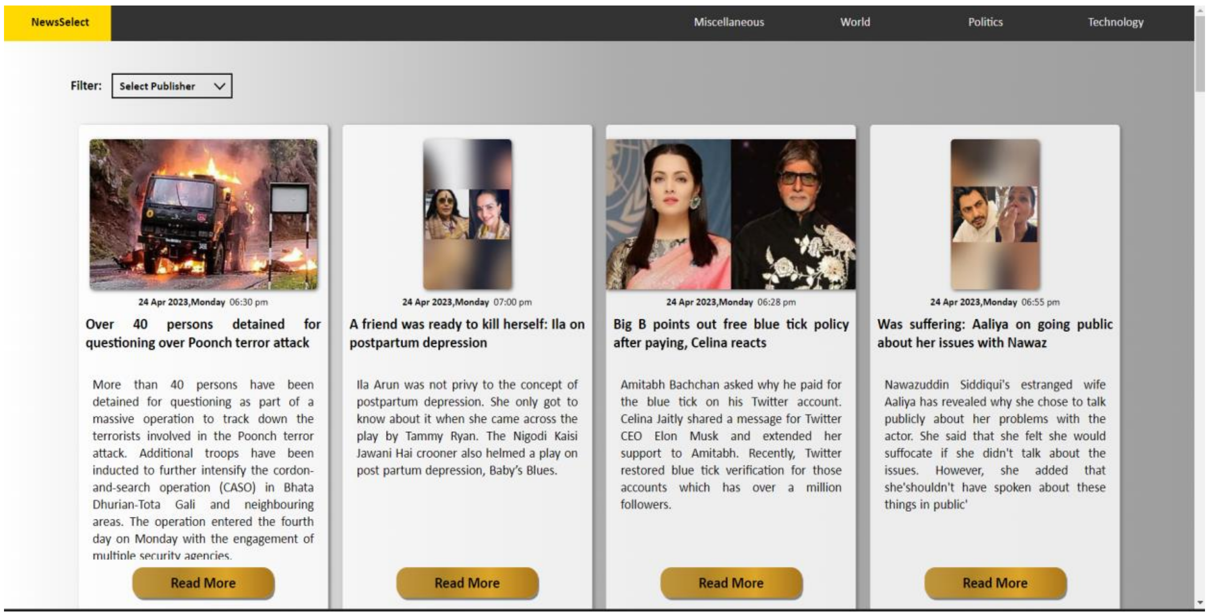
The model has been trained on 100 epochs having metrics such as batch size =128, maximum length of the input text = 300, maximum summary length = 60, and which have achieved accuracy of 89% as mentioned in Fig 8.1 and followed by the loss 2.072 as shown in the Fig8.2.

Below is the example of one such text which Bi-directional model has predicted the summary from the text given to it achieving a validation accuracy of 74%.

```
original text:
 new delhi aug 2 pti bomb squads and canine teams were today rushed in to check a suspect object that
was recovered at the cargo hold area of the igi airport here later declared safe after it was found that th
e consignment only contained some auto spare parts cisf director general o p singh said the suspect ite
m has been declared safe and there is nothing to worry about at 715 am an xray image of a consignme
nt of maruti spare parts raised suspicion among the staff of the domestic cargo terminal according to s
ources in the bureau of civil aviation security bcas the terminal staff immediately alerted the cisf who
along with a team of bcas and a bomb detection and disposal squad rushed to the spot


 original Summary:
 bomb squads and canine teams were rushed to check a package recovered in the cargo hold area of de
lhi airport on wednesday officials later declared the area safe adding the suspect item was nothing but
some spare parts of company maruti its xray image had raised suspicion among airport staff


Predicted Summary:
  xray image of a consignment of maruti spare parts raised suspicion among the staff of the domestic c
argo terminal. The terminal staff immediately alerted the cisf who along with a team of bcas and a bo
mb detection and disposal squad rushed to the spot. It was found that the consignment only contained s
ome auto spare parts.


original text:
 as many as 2297 residential buildings under the cpwd in delhi have been identified as unsafe the lok s
abha was informed on wednesdaythe number of unsafe or dangerous buildings under the new delhi mu
nicipal council stands at 124 union minister of state for home affairs hansraj gangaram ahir saidalso 17
buildings have been found unsafe in areas falling under three municipal corporations with 14 in north t
wo in east and one in south delhithe central public works department cpwd has informed that 2297 bui
ldings have been identified as unsafe residential buildings in delhi ahir said in a written replythe three
municipal corporations new delhi municipal council and cpwd intimated that every year the survey of
unsafedangerous building is carried out and necessary action for demolition of the same is undertaken
it statedthe reply also said that this is a continuous process every year and corrective action is taken


 original Summary:
 a union minister of state for home affairs informed the lok sabha on wednesday that delhis 2297 resid
ential buildings under the central public works department were found to be unsafe another 124 buildi
ngs under the new delhi municipal council were identified as dangerous the authorities said that they c
onduct surveys of unsafe buildings every year and undertake required actions


Predicted Summary:
```

as many as 2297 residential buildings under thecpwd in delhi have been identified as unsafe. 17 buildi ngs have been found unsafe in areas falling under three municipal corporations with 14 in north two in east and one in south delhit. Every year the survey of unsafedangerous building is carried out and nece ssary action for demolition is undertaken.

Figure 8.3 Testing the model (Generated Results)



24 Apr 2023,Monday 06:30 pm

Over 40 persons detained for questioning over Poonch terror attack

More than 40 persons have been detained for questioning as part of a massive operation to track down the terrorists involved in the Poonch terror attack. Additional troops have been inducted to further intensify the cordon-and-search operation (CASO) in Bhata Dhurian-Tota Gali and neighbouring areas. The operation entered the fourth day on Monday with the engagement of multiple security agencies.

24 Apr 2023,Monday 07:00 pm

A friend was ready to kill herself: Ila on postpartum depression

Ila Arun was not privy to the concept of postpartum depression. She only got to know about it when she came across the play by Tammy Ryan. The Nigodi Kaisi Jawani Hai crooner also helmed a play on post partum depression, Baby's Blues.

24 Apr 2023,Monday 06:28 pm

Big B points out free blue tick policy after paying, Celina reacts

Amitabh Bachchan asked why he paid for the blue tick on his Twitter account. Celina Jaitly shared a message for Twitter CEO Elon Musk and extended her support to Amitabh. Recently, Twitter restored blue tick verification for those accounts which has over a million followers.

24 Apr 2023,Monday 06:55 pm

Was suffering: Aaliya on going public about her issues with Nawaz

Nawazuddin Siddiqui's estranged wife Aaliya has revealed why she chose to talk publicly about her problems with the actor. She said that she felt she would suffocate if she didn't talk about the issues. However, she added that she'shouldn't have spoken about these things in public'

Figure 8.4 User Interface snippet

8.3. DISCUSSION OF RESULTS

The bidirectional LSTM networks used in the project yielded meaningful summaries with an accuracy of 89.62% and a loss of 2.07. The successful deployment of the model enabled it to fetch current news data from multiple news portals and effectively employ the bidirectional LSTM model to produce summaries for the same. While the model has demonstrated strong performance, further enhancements could be achieved through the modification of hyper-parameters.

CHAPTER 9

CONCLUSION AND FUTURE WORK

9.1. CONCLUSION

In conclusion, the news summarizer is a tool that uses advanced algorithms to condense news articles into shorter summaries and was implemented using Seq2seq with Bi-directional LSTM, with an accuracy of 89.62% and the testing accuracy **81.62**%. By leveraging machine learning and natural language processing techniques, this tool can help readers save time and quickly understand the key points of a news story. However, there are some limitations but with further development, the accuracy of the models can be improved along with the results of summarized news. Overall, it is a useful tool for those who need to stay up-to-date on current events but have limited time to read lengthy articles.

9.2. SCOPE FOR FUTURE WORK

The scope for this technology is promising with natural language processing and machine learning algorithms continuing to improve, we can expect news summarizers to become even more accurate and efficient. To enhance the model's performance, it is possible to improve its training by incorporating newly procured data throughout the month. In addition, advancements in voice recognition and text-to-speech technology can make them more accessible for individuals with visual impairments or those who prefer to consume news content through audio formats. Overall, the future of news summarization technology is bright, and we can expect to see continued innovation in this field.

REFERENCES

[1] Ji Eun Lee, Hyun Soo Park, Kyung Joong Kim, Jae Chun No (2013). "Learning to Predict the Need of Summarization on News Articles 2013" 17th Asia Pacific Symposium on Intelligent and Evolutionary Systems, IES2013, Procedia Computer Science 24:274 – 279:
2013.

[2] Laxmi B. Rananavare and P. Venkata Subba Reddy (2018), "Automatic News Article Summarization 2018" International Journal of Computer Sciences and Engineering, vol. 6, Issue-2: 2018.

[3] Hao Zheng and Mirella Lapata (2019). "Sentence Centrality Revisited for Unsupervised
Summarization, 2019";

[4] Dr. K. S. Deepthi, Janni Ramdhar Thanmayi Sharon, Sai Teja Alubelli (2020)." Multi-Document Bi-Lingual News Summarization 2020". IJRAR March 2019, Volume 6, Issue 1; 2020.

[5] Sara Tarannum, Piyush Sonar, Aashi Agrawal, Krishnai Khairnar (2020). "NLP based Text Summarization Techniques for News Articles: Approaches and Challenges 2020". International Research Journal of Engineering and Technology (IRJET); Volume: 08 Issue: 12; Dec 2021.

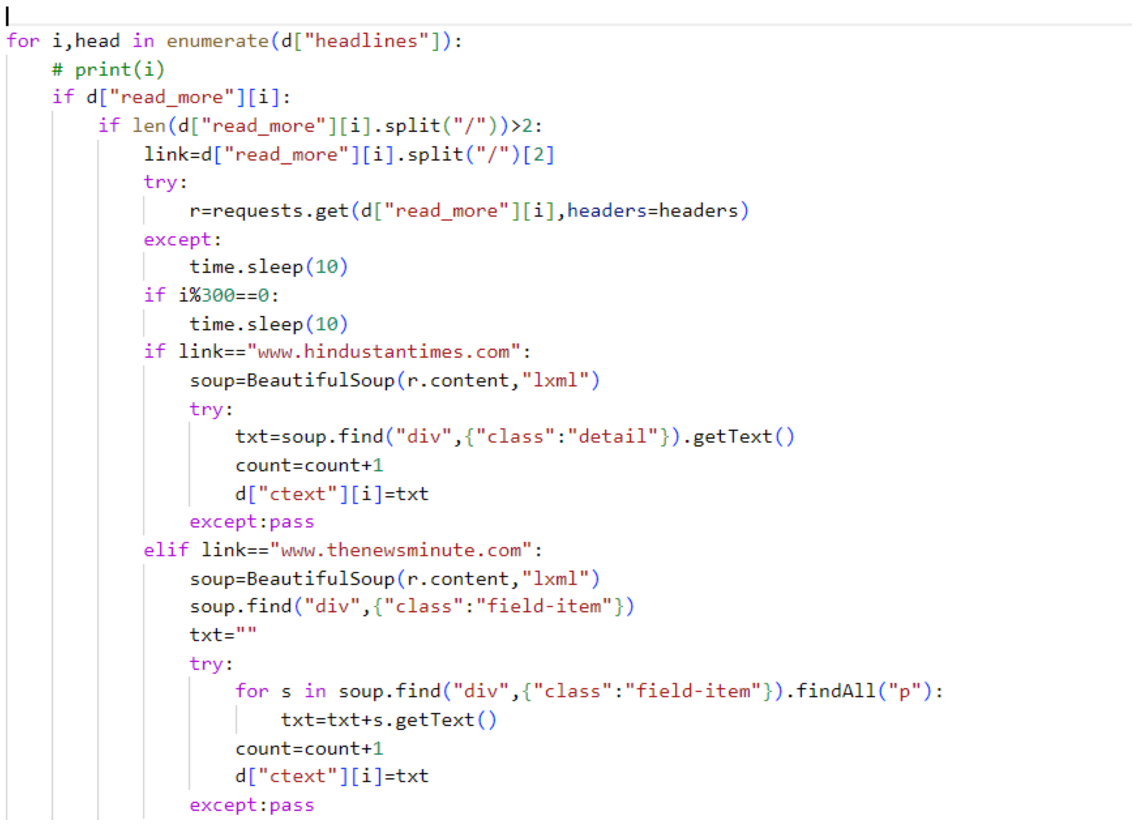
[6] Xin Zheng, Aixin Sun, Karthik Muthuswamy (2021). "Tweet-aware News Summarization with Dual-Attention Mechanism 2021"; Companion Proceedings of the Web Conference 2021 (WWW '21). Association for Computing Machinery: 2021.

[7] Yang Liu, Chenguang Zhu, Michael Zeng (2022). "End-to-End Segmentationbased News Summarization 2022"; Findings of the Association for Computational Linguistics: ACL 2022.

[8] Nayeon Lee, Yejin Bang, Tiezheng Yu, Andrea Madotto, Pascale Fung (2022)."NeuS: Neutral Multi-News Summarization for Mitigating Framing Bias" 20 conferenceeence of the North American Chapter of the Association for Computational Linguistics: 2022.

SAMPLE CODE

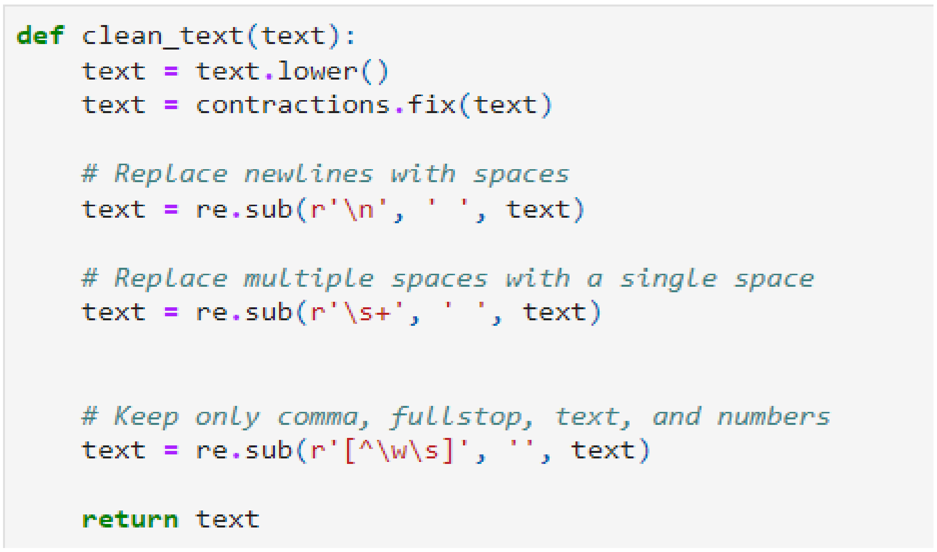
1. Data collection:



```
for i,head in enumerate(d["headlines"]):
    # print(i)
    if d["read_more"][i]:
        if len(d["read_more"][i].split("/"))>2:
            link=d["read_more"][i].split("/")[2]
            try:
                r=requests.get(d["read_more"][i],headers=headers)
            except:
                time.sleep(10)
            if i%300==0:
                time.sleep(10)
            if link=="www.hindustantimes.com":
                soup=BeautifulSoup(r.content,"lxml")
                try:
                    txt=soup.find("div",{"class":"detail"}).getText()
                    count=count+1
                    d["ctext"][i]=txt
                except:pass
            elif link=="www.thenewsminute.com":
                soup=BeautifulSoup(r.content,"lxml")
                soup.find("div",{"class":"field-item"})
                txt=""
                try:
                    for s in soup.find("div",{"class":"field-item"}).findAll("p"):
                        txt=txt+s.getText()
                    count=count+1
                    d["ctext"][i]=txt
                except:pass
```

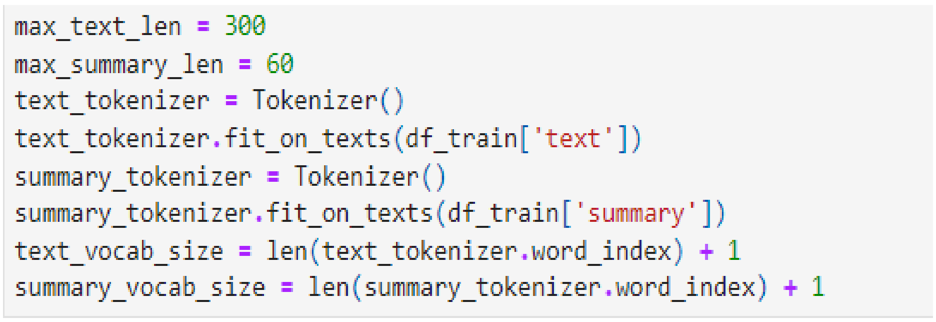
Figure I Data Scraping

2. Data preparation:



```
def clean_text(text):
    text = text.lower()
    text = contractions.fix(text)

    # Replace newlines with spaces
    text = re.sub(r'\n', ' ', text)

    # Replace multiple spaces with a single space
    text = re.sub(r'\s+', ' ', text)


    # Keep only comma, fullstop, text, and numbers
    text = re.sub(r'[^\w\s]', '', text)

    return text
```

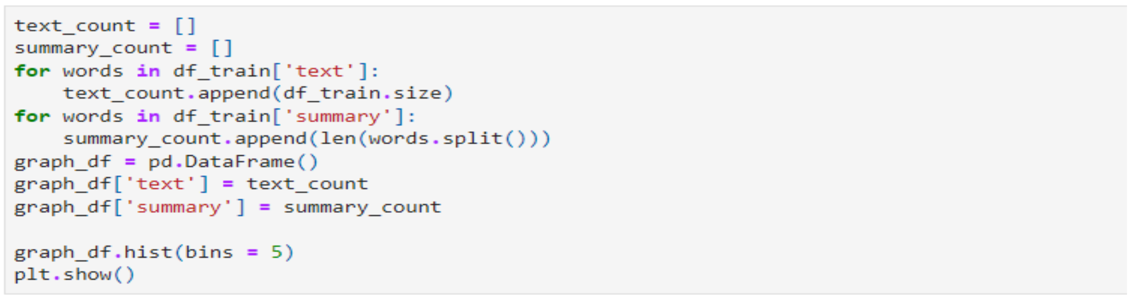
Figure II.a Data Cleaning



```
max_text_len = 300
max_summary_len = 60
text_tokenizer = Tokenizer()
text_tokenizer.fit_on_texts(df_train['text'])
summary_tokenizer = Tokenizer()
summary_tokenizer.fit_on_texts(df_train['summary'])
text_vocab_size = len(text_tokenizer.word_index) + 1
summary_vocab_size = len(summary_tokenizer.word_index) + 1
```

Figure II.b Data Preprocessing

3. Data exploration and analysis:



```
text_count = []
summary_count = []
for words in df_train['text']:
    text_count.append(df_train.size)
for words in df_train['summary']:
    summary_count.append(len(words.split()))
graph_df = pd.DataFrame()
graph_df['text'] = text_count
graph_df['summary'] = summary_count

graph_df.hist(bins = 5)
plt.show()
```

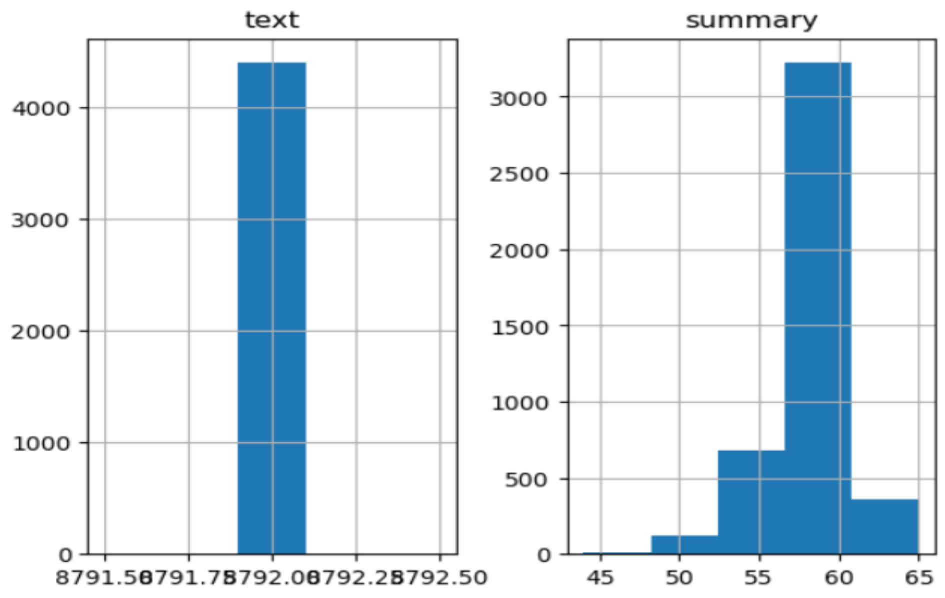


Figure III Exploratory Data Analysis

4. Modeling:

Define model

```
encoder_inputs = Input(shape=(None,))
encoder_embed = Embedding(text_vocab_size, 300, trainable=True)(encoder_inputs)
encoder_lstm = Bidirectional(LSTM(300, return_sequences=True, dropout=0.2))(encoder_embed)
encoder_states = [encoder_lstm[:, i, :] for i in range(2)]
```

```
decoder_inputs = Input(shape=(None,))
decoder_embedding = Embedding(summary_vocab_size, 300, trainable=True)
decoder_embed = decoder_embedding(decoder_inputs)
decoder_lstm = Bidirectional(LSTM(300, return_sequences=True, dropout=0.2))(decoder_embed)
```

```
attention = Attention()([encoder_lstm, decoder_lstm])
decoder_concat = Concatenate()([decoder_lstm, attention])
decoder_dense = Dense(summary_vocab_size, activation='softmax')(decoder_concat)

model = Model([encoder_inputs, decoder_inputs], decoder_dense)
```

```
# model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',metrics=['accuracy'])
model.compile(optimizer='rmsprop', loss='sparse_categorical_crossentropy',metrics=['accuracy'])

# Use early stopping to prevent overfitting
early_stopping = EarlyStopping(monitor='val_loss', patience=3, verbose=1, restore_best_weights=True)
```

Figure IV Encoder/Decoder Model

5. Model evaluation:

```
# Plot accuracy history
plt.plot(history.history['accuracy'], label='train_accuracy')
plt.plot(history.history['val_accuracy'], label='val_accuracy')
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend()
plt.show()
```

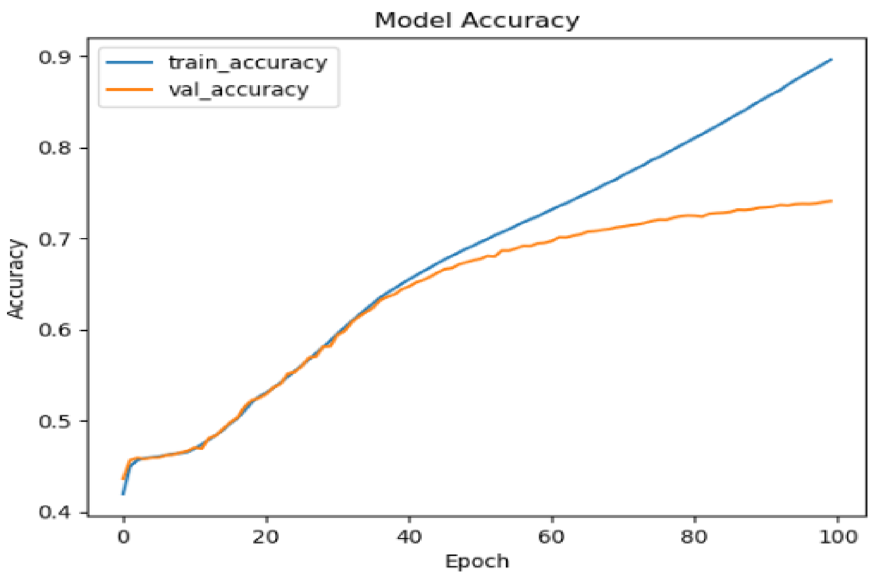
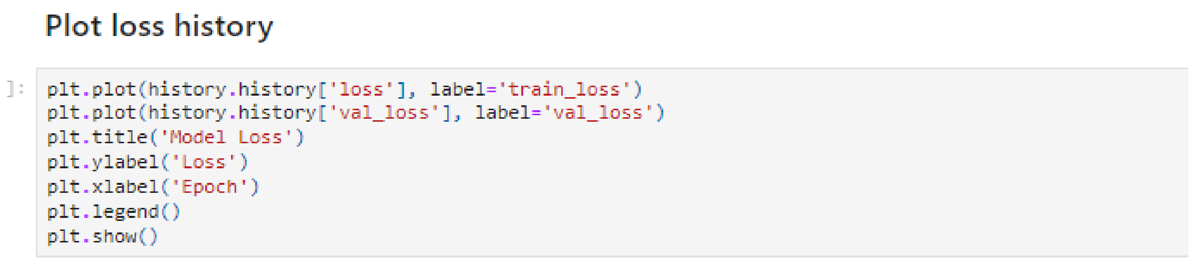


Figure V.a Model Accuracy



Plot loss history

```
]: plt.plot(history.history['loss'], label='train_loss')
   plt.plot(history.history['val_loss'], label='val_loss')
   plt.title('Model Loss')
   plt.ylabel('Loss')
   plt.xlabel('Epoch')
   plt.legend()
   plt.show()
```

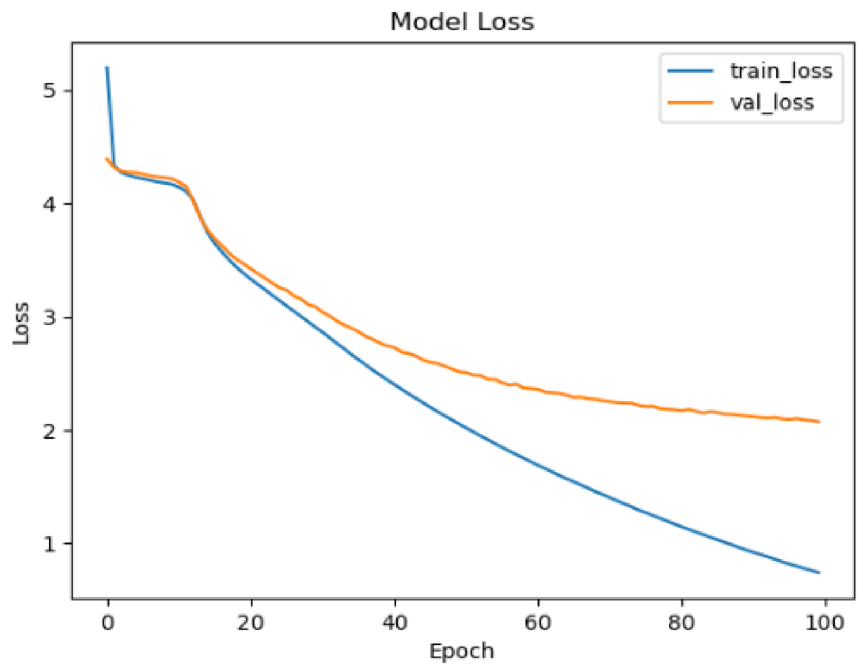
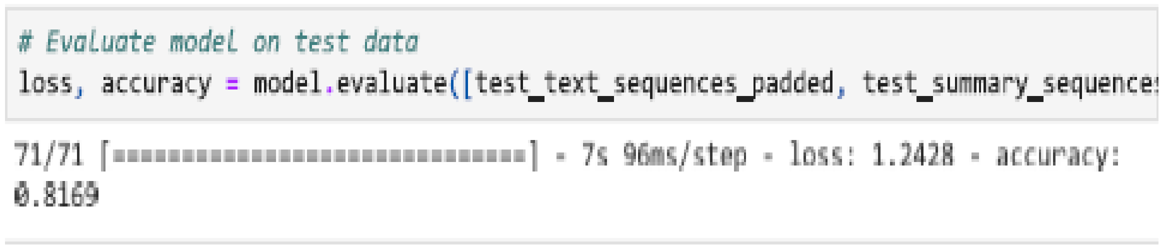


Figure V.b Model Loss

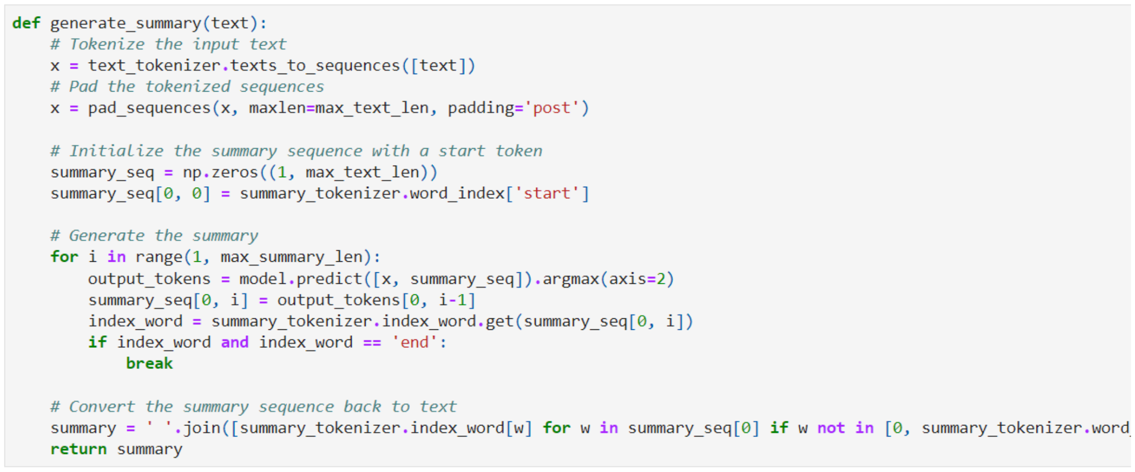


```
# Evaluate model on test data
loss, accuracy = model.evaluate([test_text_sequences_padded, test_summary_sequence
```

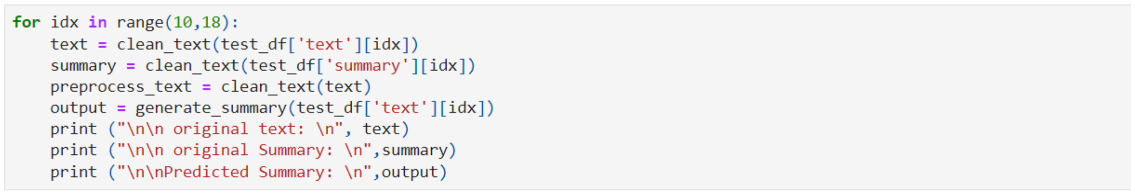
```
71/71 [==============================] - 7s 96ms/step - loss: 1.2428 - accuracy:
0.8169
```

Figure V.c Model Evaluation

6. Deployment:



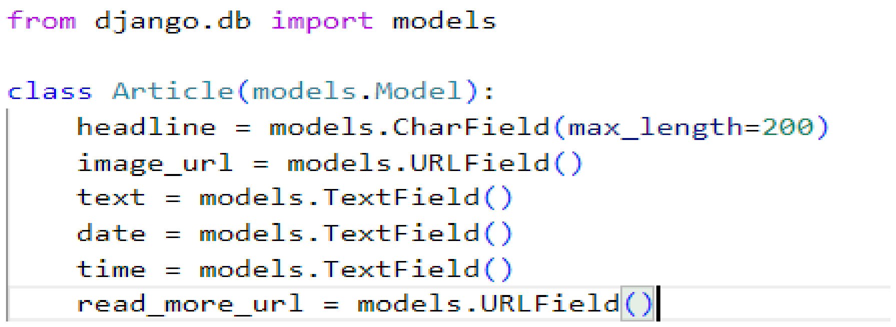
```python
def generate_summary(text):
    # Tokenize the input text
    x = text_tokenizer.texts_to_sequences([text])
    # Pad the tokenized sequences
    x = pad_sequences(x, maxlen=max_text_len, padding='post')

    # Initialize the summary sequence with a start token
    summary_seq = np.zeros((1, max_text_len))
    summary_seq[0, 0] = summary_tokenizer.word_index['start']

    # Generate the summary
    for i in range(1, max_summary_len):
        output_tokens = model.predict([x, summary_seq]).argmax(axis=2)
        summary_seq[0, i] = output_tokens[0, i-1]
        index_word = summary_tokenizer.index_word.get(summary_seq[0, i])
        if index_word and index_word == 'end':
            break

    # Convert the summary sequence back to text
    summary = ' '.join([summary_tokenizer.index_word[w] for w in summary_seq[0] if w not in [0, summary_tokenizer.word
    return summary
```



```python
for idx in range(10,18):
    text = clean_text(test_df['text'][idx])
    summary = clean_text(test_df['summary'][idx])
    preprocess_text = clean_text(text)
    output = generate_summary(test_df['text'][idx])
    print ("\n\n original text: \n", text)
    print ("\n\n original Summary: \n",summary)
    print ("\n\nPredicted Summary: \n",output)
```

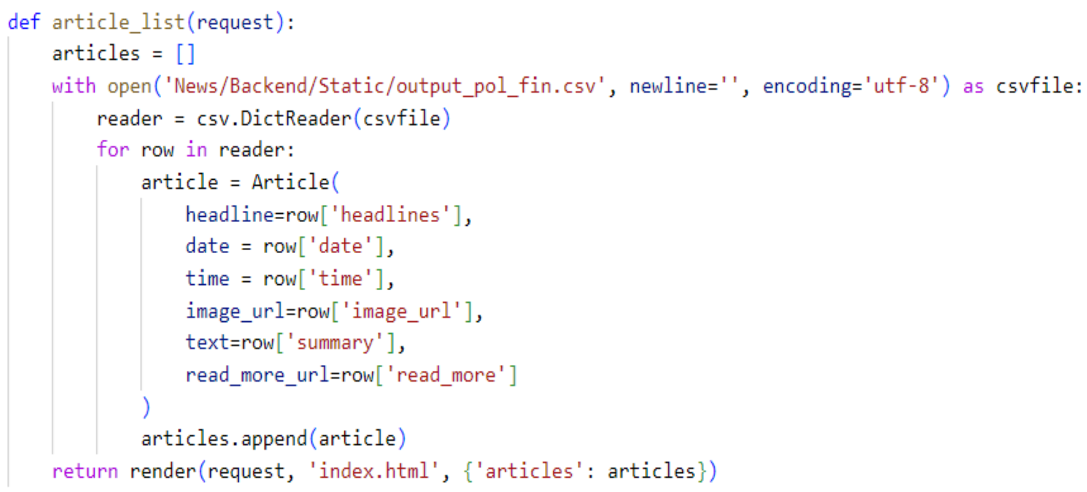
Figure VI Summary

7. Django Deployment



```python
from django.db import models

class Article(models.Model):
    headline = models.CharField(max_length=200)
    image_url = models.URLField()
    text = models.TextField()
    date = models.TextField()
    time = models.TextField()
    read_more_url = models.URLField()
```

Figure VII.a Django framework Model



```python
def article_list(request):
    articles = []
    with open('News/Backend/Static/output_pol_fin.csv', newline='', encoding='utf-8') as csvfile:
        reader = csv.DictReader(csvfile)
        for row in reader:
            article = Article(
                headline=row['headlines'],
                date = row['date'],
                time = row['time'],
                image_url=row['image_url'],
                text=row['summary'],
                read_more_url=row['read_more']
            )
            articles.append(article)
    return render(request, 'index.html', {'articles': articles})
```

Figure VII.b Django framework Views