A SUMMARIZATION APPROACH FOR EVENT CENTERED BENGALI TWEETS

by

Chanchal Khatua

21PCS007

COMPUTER SCIENCE & ENGINEERING DEPARTMENT

NATIONAL INSTITUTE OF TECHNOLOGY AGARTALA

INDIA-799046

MAY, 2023

A SUMMARIZATION APPROACH FOR EVENT CENTERED BENGALI TWEETS

Project Report submitted to
National Institute of Technology Agartala
for the award of
Master of Technology

by

Chanchal Khatua
Enrolment No: 21PCS007

Under the Guidance
of
Dr. Dwijen Rudrapal
Assistant Professor, CSE Department, NIT Agartala, India

COMPUTER SCIENCE & ENGINEERING DEPARTMENT
NATIONAL INSTITUTE OF TECHNOLOGY AGARTALA
MAY, 2023

DISSERTATION APPROVAL SHEET

This dissertation entitled "*A SUMMARIZATION APPROACH FOR EVENT CENTERED BENGALI TWEETS*", by *Chanchal Khatua*, Enrolment Number *21PCS007* is approved for the award of ***Master of Technology*** in ***Computer Science & Engineering***.

Dr. Dwijen Rudrapal	Dr. Suman Deb
Supervisor	Head Of Department
Assistant Professor	Assistant Professor
Computer Science & Engineering Department	Computer Science & Engineering Department
NIT Agartala	NIT Agartala

External Examiner

Date:04/05/2023
Place:NIT Agartala

DECLARATION

I declare that the work presented in this dissertation titled "**A SUMMARIZATION APPROACH FOR EVENT CENTERED BENGALI TWEETS**", submitted to the Computer Science & Engineering Department, National Institute of Technology Agartala, for the award of the ***Master of Technology*** degree in ***Computer Science & Engineering***, represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

MAY, 2023
NIT Agartala

Chanchal Khatua

(21PCS007)

CERTIFICATE

It is certified that the work contained in the dissertation titled "*A SUMMARIZATION APPROACH FOR EVENT CENTERED BENGALI TWEETS*", by *Chanchal Khatua*, Enrolment Number *21PCS007* has been carried out under my supervision and this work has not been submitted elsewhere for a degree.

Dr. Dwijen Rudrapal
Supervisor
Assistant Professor
Computer Science & Engineering Department
NIT Agartala

Acknowledgement

I would like to express my heartfelt gratitude to everyone who contributed directly or indirectly during this dissertation work.
First and foremost, I would like to thank my supervisor and mentor **Dr. Dwijen Rudrapal** for providing me with invaluable guidance, insightful feedback, and unwavering support throughout the project. His advice, encouragement, and critics are the source of innovative ideas, inspiration and causes behind the successful completion of this dissertation. The confidence shown to me by him is the biggest source of inspiration for me. It has been a privilege working with them from the last few months.

I would like to acknowledge the support of my family and friends for their unwavering support and encouragement throughout the project. Their love, patience, and understanding have helped me stay focused and motivated during the challenging times.

I am highly obliged to all the faculty members of Computer Science and Engineering Department for their support and encouragement. I also thank our Director and HOD **Dr. Suman Deb**, for providing excellent computing and other facilities without which this work could not achieve its quality goal.

Chanchal Khatua

Dedicated to

To God, for every one of the opportunities given to me along my life, for giving me the strength to overcome obstacles and allowing to achieve each of my goals.

To my Loving Family for offering their unconditional support in any situation, for every word of encouragement and advice to the obstacles, for always being with me, for trusting me and for being my guides and incomparable friends. The LOVE!

To my dissertation supervisor **Dr. Dwijen Rudrapal**, for sharing his valuable knowledge, encouragement & showing confidence on me all the time. Each of the faculties of the department to contribute in my development as a professional and help me to achieve this goal.

To all those people who have somehow contributed to the creation of this project and who have supported me.

Abstract

With the ubiquitous expansion of internet and explosion of social medias, huge tweets are generated in regular basis. It is emergent to extract important information from those tweets. There are many event-based summarization techniques available, and they are all based on English Tweets and most of them are specially a particular type of event like sports and disaster. Countless Bengali tweets are regularly produced due to the large population of native Bengali. Yet, there are no such works currently available in Bengali. Additionally, it's important to monitor those tweets and locate important information. Due to the lack of suitable datasets, a dataset is developed based on 46 events in Bengali, which are divided into different categories. Encoder decoder model is used with LSTM network to summarize Bengali tweets based on events. Rouge-1, Rouge-2 and Rouge-L are used for measuring efficiency, and the results were 71%, 39% and 27% respectively. This study, related work, text summarization's classifications, descriptions about dataset, proposed methodology, implementation procedure, evaluation matrices and result, and discussion are covered thoroughly in details.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the widespread nature of the Internet's growth and the emergence of social media, it is easier for people to communicate and share user-generated content in various forms. However, this has resulted in an overwhelming amount of unstructured and disorganized data that requires a specific approach for better understanding. Data mining has emerged as the most crucial approach for organizing such data, especially textual data.[16] One of the essential sub-fields of data mining is text mining, which involves summarizing single or multiple documents into concise texts that provide an overview of the content. Manually summarizing all documents is not feasible due to time constraints, thus necessitating the need for Automatic Text Summarization (ATS). ATS provides a brief and understandable summary by extracting the essence of the original text.

1.1 Background

Natural Language Processing (NLP) is an interdisciplinary field of computer science, artificial intelligence, and linguistics that focuses on the interactions between computers and human languages. It has various applications, including web search improvement, email spam filtering, machine translation, text summarization, and sentiment analysis, etc. One important application of NLP is text summarization, which is the process of reducing a longer text into a shorter summary while retaining its main ideas and concepts. This is useful for quickly understanding the key points of a document or article, and it can be especially helpful for news stories, academic research, and other long-form content. It is a challenging problem in NLP for text mining, but it offers several benefits to users. Text summarization has been extensively studied in various languages, including English, but there is a growing need for summarization systems in other languages as well. Bengali, the official language of Bangladesh and one of the most widely spoken languages in India, is an example of a language that has received less attention in the field of text summarization. Bengali text summarization is challenging due to the complexity of the language, including its use of compound words, idiomatic expressions, and other features that make it difficult for computers to accurately understand and summarize. However, recent advances in NLP and machine learning have made it possible to develop automated Bengali text summarization systems that can accurately summarize a wide range of documents and articles.

Twitter is one of the most widely used social media platforms, where users generate a vast amount of text data every day in multiple languages. While English is the primary language, Bengali tweets are also generated by users in West Bengal, Tripura (India), and Bangladesh, where Bengali is the native language. Additionally, users from different regions may use various dialects of Bengali. To keep track of current events, Bengali tweets are collected related to trending topics and a text summarization algorithm is applied to the tweets of specific events, which allows us to extract the essential information and generate a concise summary, or “gist” of the event. This approach provides an efficient way to process the vast amount of text data on Twitter and enables us to quickly understand the main points of a particular event.

These systems typically use algorithms and models that are trained on datasets of Bengali tweet text, allowing them to identify the most important information and generate summaries that are coherent, informative, and accurate. The development of such systems is critical for improving access to information for Bengali speakers and enhancing the overall efficiency of content processing and dissemination in the Bengali language.

1.2 Motivation

In the past decade, there has been a gradual increase in the use of text summarization techniques. Initially, statistical approaches such as tf-idf and linguistic approaches like clustering and TextRank were popular. With the advent of deep learning, newer techniques and models have emerged that provide better results. Although text summarization has primarily been done for English text, there has been a recent trend in exploring other regional languages as well. After analyzing the Bengali text summarization scenario, most of the work has been focused on news articles, which are typically lengthy and contain extraneous information. To address this issue, events based tweet summarization is chosen. This is a challenging task as there is no existing dataset for this purpose, but it is also an opportunity to innovate and create something new.

1.3 Goal

This thesis aims to provide a comprehensive overview of text summarization techniques, as well as a detailed description of the dataset that has been created for Bengali tweet summarization. The main focus of this project is to gather and distill important information from numerous tweets related to a specific event or hashtag. To accomplish this goal, a Seq2Seq model is proposed for abstractive Bengali tweet summarization. This model will be trained on the carefully curated dataset, with the aim of generating concise and informative summaries that accurately capture the key points of the tweets. Ultimately, the goal of this work is to advance the field of text summarization and provide a useful tool for efficiently summarizing large volumes of Bengali tweets related to specific events or topics.

1.4 Problem Statement

The aim of this study to expand this area of research by developing an abstractive text summarization model using deep learning techniques. Absent of such datasets, Bengali tweets are collected related to specific events or hashtags, and used this data to generate concise and informative summaries ranging from average 10–15 tokens, depending on the number of tweets related to the event. This dataset will be carefully curated and used to train the model, with the goal of improving upon previous research in the field of text summarization.

Chapter 2

Text Summarization Classification

The field of text summarization has a significant impact on data scientists, and the search for an effective approach to automate the process is gaining traction. Automatic text summarization involves creating a concise and coherent summary of a lengthy text document. Humans excel at this type of task by reading the entire content, comprehending the meaning, and generating a brief summary with an overall understanding of the content. The goal of automatic text summarization is to produce a summary that is as good as that created by a human. The automatic process is not simply about generating words or phrases that capture the essence of the source document. Instead, it involves creating a meaningful new document that is easily readable. To further enhance the quality of automatic text summarization, various techniques are being developed and implemented.

Automatic text summarization can be categorized based on several criteria, including input size, purpose, and output types. The classification of Text Summarization diagram is mentioned in Figure 2.1

2.1 Based on Input size

There are two types of summarization according to source input size.

- **Single Document :** In single document summarization[20], the produced summary is based on a single document. Single document summarization is typically used to create shorter versions of longer documents such as articles, reports, or essays. The goal is to capture the main points of the document concisely, making it easier for readers to understand the key takeaways without having to read the entire document.

- **Multiple Documents :** Multi-document summarization[10] is used when there are multiple documents on a similar topic, such as news articles about a specific event or scientific papers on a particular research area. The goal is to synthesize the main points from all the documents and create a summary that captures the key information concisely. Multi-document summarization is a more challenging task than single document summarization because it requires identifying the most relevant information across multiple sources and combining them in a coherent manner. It often involves more advanced natural language processing techniques and algorithms such as topic modeling, clustering and information retrieval.

2.2 Based on Purpose

According to the purpose, there are primarily three types of summarization processes. Each type of summarization has its strengths and limitations, and the selection of a suitable summarization approach depends on the specific requirements of the application.

- **Generic:** It is a type of summarization process that aims to summarize all texts[2], regardless of the domain or topic. The objective of generic summarization is to extract the most important information from a document and condense it into a shorter version that captures the essence of the original text. Generic summarization does not assume any specific domain knowledge or jargon, and the scope of its source data considers all documents to be homogeneous. It is commonly used in news summarization, email summarization, and social media summarization.

- **Domain-Specific Summarization:** It is a type of summarization process that aims to generate a summary for a specific domain or topic, such as finance, medicine, or weather. Unlike generic summarization, domain-specific summarization leverages domain-specific knowledge, terminology, and jargon to generate more accurate and informative summaries. Domain-specific summarization systems typically use techniques such as text classification, named entity recognition, and topic modeling to identify relevant information from the source documents. The extracted information is then condensed into a summary that captures the most important aspects of the original text. It is commonly used in industries such as finance, healthcare, and legal, where the ability to quickly and accurately summarize large volumes of text is critical. The use of domain-specific summarization systems can help professionals stay informed about the latest developments and make more informed decisions based on the insights gleaned from the summaries.

- **Query Related:** Query-based summarization[7] is a type of summarization process that generates a summary based on a specific query or search term. The objective of query-based summarization is to provide the most relevant information related to the user's query while filtering out irrelevant information. It typically uses natural language processing techniques to understand the user's query and identify the most relevant information from the source documents. The extracted information is then condensed into a summary that directly addresses the user's query. It is commonly used in search engines, where the system provides a brief summary of the most relevant web pages related to the user's search query. The use of query-based summarization can help users quickly find the information they need without having to read through a large volume of text.

2.3 Based on Summary Procedure

According to procedure of the summarization, it is divided into process is two types.

- **Extraction Based:** Extractive summarization is a type of summarization process where the summary is generated by selecting the most important sentences or phrases from the original document. The selected sentences or phrases are combined to form a shorter version that conveys the essence of the original document. Extractive summarization involves ranking sentences based on their importance and relevance to the document's overall content. The ranking process can be done using various methods such as frequency analysis, sentence position, and semantic analysis. Once the sentences are ranked, the top-ranked sentences are selected to form the summary, and it is commonly used in news summarization and other applications where the original text is factual and objective. The main advantage of extractive summarization is that it retains the exact wording and syntax of the original text, which can be useful in legal or regulatory contexts. However, extractive summarization can sometimes result in a summary that is less readable or coherent, as it may include disjointed sentences or phrases.

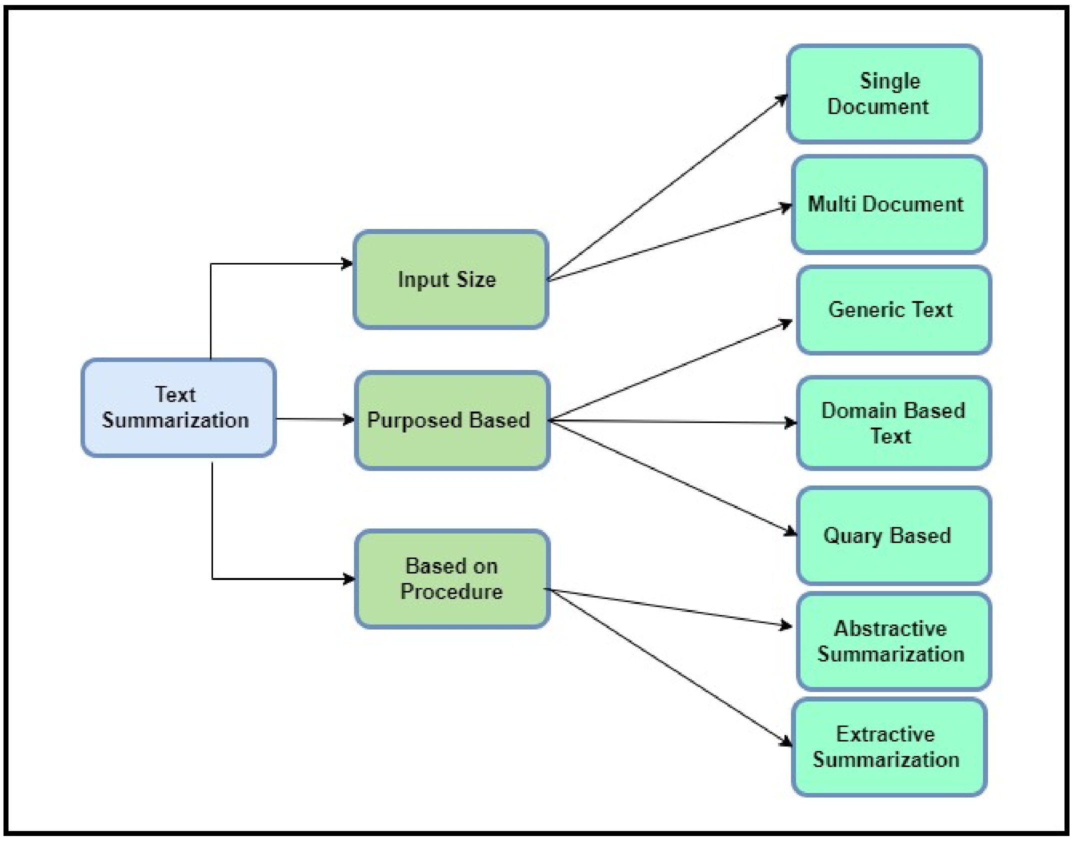


Figure 2.1: Classification of Text Summarization

- **Abstraction Based:** Abstractive summarization[19] is a type of summarization process where the summary is generated by paraphrasing and synthesizing the key information from the original document. Unlike extractive summarization, abstractive summarization can generate summaries that do not necessarily contain the exact wording or syntax of the original text. It involves generating new sentences that capture the main ideas and concepts of the original document. The process can be done using various techniques, including natural language processing and machine learning algorithms. The goal is to generate a concise and readable summary that conveys the essential information of the original document, and it is commonly used in applications such as email summarization, where the original text is subjective and contains opinions and insights. The advantage of abstractive summarization is that it can generate more concise and readable summaries that capture the main ideas of the original text, even if the wording and syntax are different. However, abstractive summarization is a more challenging task than extractive summarization, as it requires the system to generate new text that accurately reflects the content of the original text.

In this study, Bengali event base tweet summarization is abstractive approach and dataset contain generic type text means all types domain of tweet are present.

Chapter 3

Related Work

Twitter has become one of the most popular social media platforms, with millions of users sharing their thoughts and opinions on a wide range of topics. As the volume of tweets increases rapidly, it has become increasingly difficult for users to keep track of all the information shared on the platform. Therefore, automatic tweet summarization has become an essential task to enable users to quickly and efficiently comprehend the content of large volumes of tweets. However, summarizing Bengali tweets poses a unique set of challenges due to the complexity of the Bengali language and the lack of resources and datasets. Bengali summarization of news articles and social media content has received notable attention in the research community, with several noteworthy studies exploring the effectiveness of abstractive summarization techniques.

3.1 Related Work on Bengali Text Summarization

The RNN with bidirectional LSTM within 512 dense layers are used for Bengali news headline generations[20]. Rectified Linear Units(ReLU) [1] used as activation function to overcome vanishing gradient problem in each input layer and the dropout value is set to 0.5 to reduce the overfitting. The another work for abstractive Bengali news summarization, RNN Encoder-Decoder Architecture which is common Seq2Seq model with Bahdana attention mechanism are used and Word2Vec are used as word embedding.[25] A seq2seq encoder-decoder model based on Long Short-Term Memory (LSTM) with local attention is introduced by Bhattacharjee et al.[2] where Word Embedding layer converts the input sequence to numbers, then it supplied in reverse order to the LSTM encoder. The authors employed a greedy LSTM decoder, which is distinct from a beam search decoder. The same previous model bidirectional RNN with encoder and decoder with LSTM is proposed for Bengali news summarization applying Bahdanau Attention mechanism. The “bn_w2v_model” and ”bn” are used for word embedding, specially ”bn” for sentence analysis and according to Islam et al.[17] “bn_w2v_model” performed better. The another proposed model based on bidirectional RNN encoder-decoder are used for Bengali social media summarization and here Fouzia et al.[11] divided proposed model into three parts. At first, machine translation model is used to convert other language into Bengali text and chain to chain model is an architectural model of the RNN applied in deep learning is the LSTM. It’s long-term memory component is learned weight, while its short-term memory component is present value of a gated cell. The encoder and decoder are patched into an LSTM cell in the chain-to-chain model. Table 1 gives summary about Bengali abstractive summarization

Table 1: Bengali Text summarization

Ref.	**Language**	**Proposed model**	**Result analysis**
[20]	Bengali	RNN with bidirectional LSTM	Precision score 81%
[25]	Bengali	RNN Encoder-Decoder Architecture with Bahdana Attention	Owned data Compare with BANS dataset
[2]	Bengali	Encoder Decoder with LSTM & Local Attention Mechanism	ROUGE and BLEU scores
[17]	Bengali	Encoder Decoder with LSTM & Attention Mechanism	Model entire loss 0.0007
[11]	Bengali	RNN with bidirectional LSTM	Not Given

However, the domain of Bengali tweet summarization presents a unique challenge, particularly in the context of event-based summarization. Despite this, only two summarization datasets are currently available, and there is a significant dearth of publicly accessible datasets for Bengali tweet summarization. Most of Bengali text summarization are based on news article and sometime pre-exiting dataset are used or creating owned dataset like here data is collected from newspaper for headline generation[20]. This dataset contains all type of 9757 news headlines with 19,510 words and it contains 39,315 characters. In another work, data are gathered for short length summary from various newspaper contain 500 long articles in various categories and a preexist dataset BANS from ”kaggle.com” is also used [25]. It contains 19 k short summaries with 19 k summaries. The BANS dataset is created by Bhattacharjee et al.[2], and it contains article with 5 to 76 number of words, 3 to 12 words in created summary and also compared owned data set with BNLPC data set, which is a small data sets contain 200 articles with 600 summaries[2]. In Another work, 3000 documents [17] and its summary are collected from online newspaper, social media. Here the data sets are created from 1000 Bengali data from social media text, blog, article, and summaries by Fouzia et al. [11].

3.2 Related Work on Tweet Summarization

The existing literature on Bengali tweet summarization emphasizes the need for improved methods and resources to enable effective summarization of tweets based on events. Despite the challenges posed by the complexity of the Bengali language and the diversity of social media content, recent studies have demonstrated the potential of deep learning-based approaches to address these issues. In last 5 years, many methodology and algorithm are proposed for event based tweets summarization like, here Vijay et al. [26] target sports event tweet for factual instance summarization, getting opinion or exacting viewpoint of sports event and also the specific event from the tweet using cosine similarity technique and cluster manger. Various Machine learning algorithm like SVM, naive bayes, logistic regression are used to categorize opinion. In the other sports events based tweet summarization[5], author used Hidden Markov Models but applied algorithm is small different from standard Hidden Markov Model like it is combing information from multiple events and its summary also depend on time stamp of tweets.

Another specific event based tweet summarization, Garg et al.[13] created 10 datasets for tweet summarization base on 10 each disaster events and disaster are both natural or Human-mad and got good F1 score. The other disaster based tweet summarization, Garg et al.[12] used EnDSUM which is a disaster based summarizer and here dataset are created using 6 different type of disaster events. Dutta et al. [9] have used various algorithms like ClusterRank, COWTS, FreqSum, LexRank, LSA, LUHN, MEAD, Sumbasic for summarize microblog during emergency event. It is extractive summarization and about 1000 tweets are collected for dataset based on 5 emergency events. Among of those algorithms, LUHN provide good ROUGH score. In another crisis base tweets summarization, Rudra et al.[22] applied AIDR classifier to detect various categories of crisis event and then COWTS is applied for extractive summary. After that COWABS is used for abstractive summarization.

Chakraborty et al.[6] did a sightly different type of tweet summarization where they collected tweet based 36 news articles to related to the particular event. LexRank, LSA, and MDS are used techniques and author also have created weight graph as called TSG(tweet similarity graph) which contain weight between tweets based on their contain similarity. Dusart et al.[8] also created a dataset and used previous mentioned methodology.

Chakma et al. [4] have achieved outstanding result using deep learning Pertained model BERT with own created dataset. Pointer Generator with coverage mechanism also is applied on datasets and compare the summary generated by both mechanism. Before applying both mechanism, four clustering mechanism are used to identify possible event and then re-ranking those tweets base on semantic similarity. Li et al. [18] also did re-rank tweets base on order of timestamp and cosine similarity. TweetSift is used for event prediction which is an efficient and effective real time tweet topic classifier and then applying BERT for predict summary of events. Goyal et al. [15] used encoder-decoder with LSTM model for abstractive base event summarization and applied clustering algorithm to detect sub-events from real-time tweet steam and each sub-events organized according to their hierarchy under main events.

In query-focused summarization of tweets stream, Geng et al. [14] developed a microblogging stream clustering algorithm in the basis of each time slot. At starting of clustering, K-means clustering method is used, and every time similarity is calculated between exiting cluster and new tweet stream cluster. If similarity is grater than the predetermined threshold value, then both cluster will merge and also removed the outdated cluster from main cluster to track new or trending events. Q-LexRank methodology is applied for query-focused summarization by selecting the sentences with the priority.

In another study, Rudrapal et al. [23] ranked tweets based on priority and remove less informative sentence from the event. Then Partial Textual Entailment methodology is applied for event summarization. In this study, Bilal et al.[3] used fined tuned and non-fined tuned model for abstractive based tweet summarization and Covid-19, UK Elections as event are used only for it. However, the lack of available datasets for Bengali tweet summarization hinders progress in this field, limiting the ability of researchers to develop and evaluate novel summarization techniques. Consequently, there is a pressing need for the development of publicly accessible datasets for Bengali tweet summarization, which would facilitate the creation of effective and robust summarization models for this important domain. Here we collected Bengali tweets from various type of trending event or topic in range of previous 10 days and summarize those tweets to extract essential information using abstraction summarization.

Chapter 4

Dataset Preparation and Annotation

4.1 Preparation

Tweets are collated base trending keywords or event from “www.tweetarchivist.com” in form of Excel file which is a tweeter’s third party API and then other language tweet are filtered from those Excel files. To prepare the tweets for annotation, a series of preprocessing steps are performed to standardize the text and remove any extraneous information that was not relevant to the task of summarization. This included removing URLs, hashtags, extra space, punctuation, comma, Bengali and English digit, and mentions, as well as standardizing the text by performing tokenization and save into normal text format file. In addition to tokenization, quality checks are also performed on the tweets to ensure that they met certain criteria. Specifically, tweets are removed which are not in the Bengali language or were shorter than 5 words. To address the challenging nature of summarizing tweets, summaries are created based on the most important information from a particular event. Analyzing all tweets in the dataset to create summaries would have been a daunting task, as it would have required identifying and extracting the most relevant information from a large volume of noisy data. Instead, we chose to group tweets that belong to the same event and create a summary that captures the most important aspects of that event. This approach ensured that the summaries were concise and informative, while also reducing the amount of noise in the data. As a result, the annotations in this dataset are tailored to specific events, rather than attempting to summarize all tweets in the dataset. This makes the dataset useful for applications where users are interested in obtaining a summary of a specific event or topic, rather than trying to analyze all tweets in the dataset. The summaries for the Bengali tweet summarization dataset were manually created by human annotators. The created summaries are maximum 20 tokens long and some summaries may be as short as 5 words, depending on the number of tweets in the related event. The annotators were trained in the task of summarization and instructed to capture the most important information from the tweets in a concise and readable manner. Each tweet is associated with a summary that was created by a human annotator, with the goal of capturing the most important information from the tweet in a concise and readable manner. The collected tweets of Gujarat bridge collapse is in Figure 4.3 in Morabi.

Table 2: Details Description of Dataset.

Parameter	**Details**
No of events	46
Total tweets	5094
Min tweets of an event	6
Max tweets in an event	604
Total words	57637
Unique words	9924
Maximum input length of event	6227
Minimum input length of event	63

4.2 Description

The Bengali tweet summarization dataset is a collection of 5,094 Bengali tweets and their corresponding summaries based on 46 events, with a total of 57,637 words in the dataset. The dataset includes about 9,924 unique words and summaries that range from 4 to 31 words in length and its depend on number of tweets or essential information of an event. Figure 4.1 and Figure 4.2 give details about word count frequency distribution about original text and summary text, respectively. The maximum input length of event is 6,227 token sequence and minimum is 63, ensuring that the dataset can be used for both long-form and short-form summarization tasks and total 550 unique word is present in summary. Table 2 give that Description summary of the dataset.

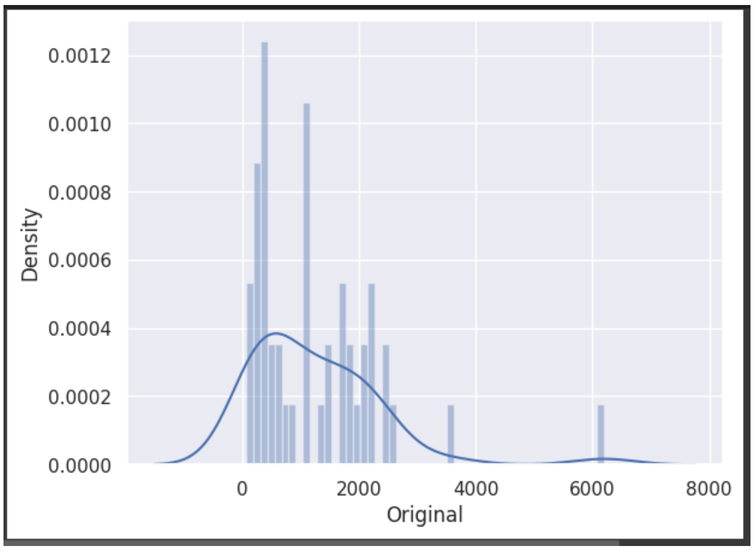


Figure 4.1: Word Count Frequency of Event in Original Text

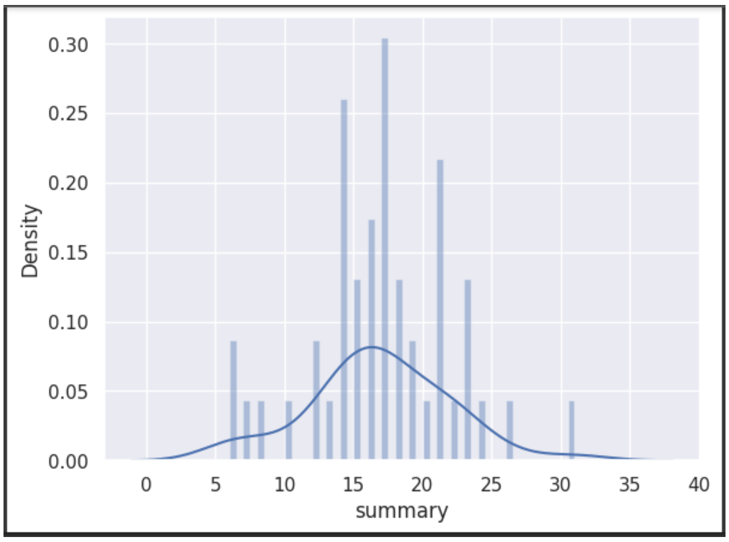


Figure 4.2: Word Count Frequency of Summary

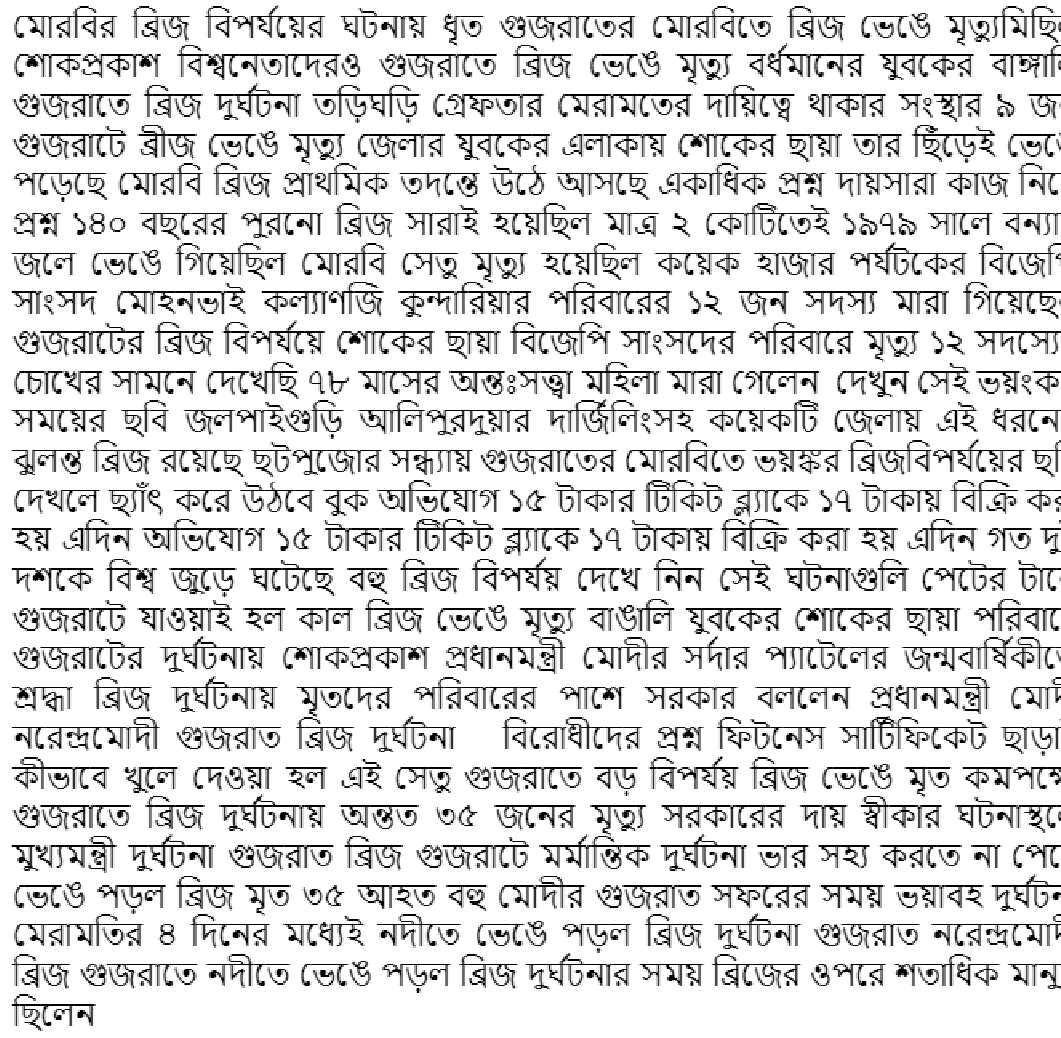

মোরবির ব্রিজ বিপর্যয়ের ঘটনায় ধৃত গুজরাতের মোরবিতে ব্রিজ ভেঙে মৃত্যুমিছিল শোকপ্রকাশ বিশ্বনেতাদেরও গুজরাতে ব্রিজ ভেঙে মৃত্যু বর্ধমানের যুবকের বাঙ্গালি গুজরাতে ব্রিজ দুর্ঘটনা তড়িঘড়ি গ্রেফতার মেরামতের দায়িত্বে থাকার সংস্থার ৯ জন গুজরাটে ব্রীজ ভেঙে মৃত্যু জেলার যুবকের এলাকায় শোকের ছায়া তার ছিঁড়েই ভেঙে পড়েছে মোরবি ব্রিজ প্রাথমিক তদন্তে উঠে আসছে একাধিক প্রশ্ন দায়সারা কাজ নিয়ে প্রশ্ন ১৪০ বছরের পুরনো ব্রিজ সারাই হয়েছিল মাত্র ২ কোটিতেই ১৯৭৯ সালে বন্যার জলে ভেঙে গিয়েছিল মোরবি সেতু মৃত্যু হয়েছিল কয়েক হাজার পর্যটকের বিজেপি সাংসদ মোহনভাই কল্যাণজি কুন্দারিয়ার পরিবারের ১২ জন সদস্য মারা গিয়েছেন গুজরাটের ব্রিজ বিপর্যয়ে শোকের ছায়া বিজেপি সাংসদের পরিবারে মৃত্যু ১২ সদস্যের চোখের সামনে দেখেছি ৭৮ মাসের অন্তঃসত্ত্বা মহিলা মারা গেলেন দেখুন সেই ভয়ংকর সময়ের ছবি জলপাইগুড়ি আলিপুরদুয়ার দার্জিলিংসহ কয়েকটি জেলায় এই ধরনের ঝুলন্ত ব্রিজ রয়েছে ছটপুজোর সন্ধ্যায় গুজরাতের মোরবিতে ভয়ঙ্কর ব্রিজবিপর্যয়ের ছবি দেখলে ছ্যাঁৎ করে উঠবে বুক অভিযোগ ১৫ টাকার টিকিট ব্ল্যাকে ১৭ টাকায় বিক্রি করা হয় এদিন অভিযোগ ১৫ টাকার টিকিট ব্ল্যাকে ১৭ টাকায় বিক্রি করা হয় এদিন গত দুই দশকে বিশ্ব জুড়ে ঘটেছে বহু ব্রিজ বিপর্যয় দেখে নিন সেই ঘটনাগুলি পেটের টানে গুজরাটে যাওয়াই হল কাল ব্রিজ ভেঙে মৃত্যু বাঙালি যুবকের শোকের ছায়া পরিবারে গুজরাটের দুর্ঘটনায় শোকপ্রকাশ প্রধানমন্ত্রী মোদীর সর্দার প্যাটেলের জন্মবার্ষিকীতে শ্রদ্ধা ব্রিজ দুর্ঘটনায় মৃতদের পরিবারের পাশে সরকার বললেন প্রধানমন্ত্রী মোদী নরেন্দ্রমোদী গুজরাত ব্রিজ দুর্ঘটনা বিরোধীদের প্রশ্ন ফিটনেস সার্টিফিকেট ছাড়াই কীভাবে খুলে দেওয়া হল এই সেতু গুজরাতে বড় বিপর্যয় ব্রিজ ভেঙে মৃত কমপক্ষে গুজরাতে ব্রিজ দুর্ঘটনায় অন্তত ৩৫ জনের মৃত্যু সরকারের দায় স্বীকার ঘটনাস্থলে মুখ্যমন্ত্রী দুর্ঘটনা গুজরাত ব্রিজ গুজরাটে মর্মান্তিক দুর্ঘটনা ভার সহ্য করতে না পেরে ভেঙে পড়ল ব্রিজ মৃত ৩৫ আহত বহু মোদীর গুজরাত সফরের সময় ভয়াবহ দুর্ঘটনা মেরামতির ৪ দিনের মধ্যেই নদীতে ভেঙে পড়ল ব্রিজ দুর্ঘটনা গুজরাত নরেন্দ্রমোদী ব্রিজ গুজরাতে নদীতে ভেঙে পড়ল ব্রিজ দুর্ঘটনার সময় ব্রিজের ওপরে শতাধিক মানুষ ছিলেন

Figure 4.3: Collected Tweets About Morabi Bridge Collapse

4.3 Statistics

The tweets were collected from various domains, including politics, sports, Disaster, entertainment, technology, and related tweets of famous Person to ensure that the dataset is diverse and representative of the types of texts that might need to be summarized. Table 3 represent the diversity of Event.

Table 3: Categories of Event with Distribution.

Categories of Event	**Number of Event**	**Distribution of Event**	**Number of Tweet**	**Distribution of Tweet**
Tweets of famous person	8	15.39%	510	9.26%
Protest	6	13.04%	1363	24.76%
Sports	9	19.57%	935	16.98%
Disaster	5	10.87%	576	10.46 %
Entertainment	3	6.52%	535	6.41%
Politics	7	15.22%	872	15.84%
Other	8	15.39%	485	8.81%

1. **Tweets of famous person:** This category contains 510 tweets, making up 9.26% of the total tweets and 15.39% of the total event. These instances include related tweets of famous person like Adani, Bil Gates, Sourav Ganguli, Amit Shah etc.

2. **Protest:**This category contains, 1363 tweets, representing 24.76% of the total data and 13.04% of total event. The instances in this category include protest like Tet agitation in West Bengal, protest in Jadavpur University etc.

3. **Sports:** This category includes 9 various sport events, such as T20 world cup, Asia cup, ISL, etc. It contains 935 tweets, which represents 16.98% of the total tweets and 19.57% of the total events.

4. **Disaster:** This category includes 5 different natural disasters, such as Earthquake in the Middle East, Cyclone Sitang in Bengal, Bridge collapse in Gujarat, etc. The total contribution of disaster-related tweets in the dataset is 10.87%.

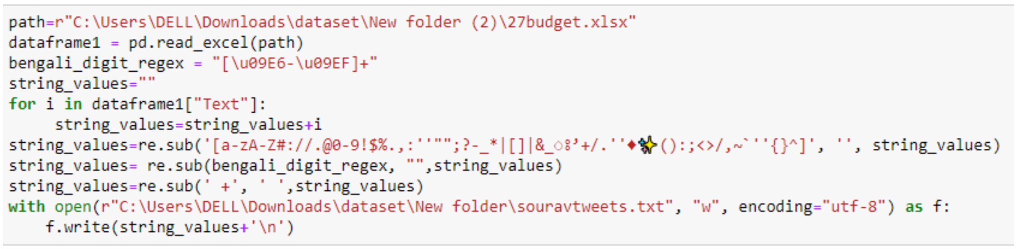
5. **Entertainment:**In this category, only 3 events are included, such as Kolkata Boimela, Phathan movie-related tweets, and Bengali movie-related tweets.

6. **Other:** There are 8 events in the dataset that cannot be classified into any special categories, such as Central Budget tweets, spy balloon in the US, comet in the sky, etc.

4.4 Annotation Challenges

Data set annotation can be a challenging task, and there are several difficulties that can arise during the process.

- One difficulty is dealing with tweets in different languages. It can be time-consuming to filter out tweets in languages other than the desired language. Machine translation may be an option, but it may not be accurate enough for annotation purposes.
- Another difficulty is dealing with various symbols that are used in tweets. Symbols can vary greatly, and it can be challenging to identify and remove them. It may be necessary to continuously update the code to deal with new symbols, and example are given in Figure 4.4. Special characters or symbols in text data can cause issues when processing or analyzing the text, especially when using it in a machine learning model or other types of computational analysis. Special characters can create noise in the data, which can make it difficult to extract meaningful information. For example, in natural language processing (NLP), special characters can interfere with text normalization techniques such as tokenization and stemming, which are used to break text into smaller, more manageable pieces.



```
path=r"C:\Users\DELL\Downloads\dataset\New folder (2)\27budget.xlsx"
dataframe1 = pd.read_excel(path)
bengali_digit_regex = "[\u09E6-\u09EF]+"
string_values=""
for i in dataframe1["Text"]:
     string_values=string_values+i
string_values=re.sub('[a-zA-Z#://.@0-9!$%.,:''"";?-_*|[]|&_০ঃ'+/.''♦✨():;<>/,~`''{}^]', '', string_values)
string_values= re.sub(bengali_digit_regex, "",string_values)
string_values=re.sub(' +', ' ',string_values)
with open(r"C:\Users\DELL\Downloads\dataset\New folder\souravtweets.txt", "w", encoding="utf-8") as f:
    f.write(string_values+'\n')
```

Figure 4.4: Data Cleaning

- Removing symbols from text data can sometimes result in large blank spaces, which can make it challenging to determine where one tweet ends and another begins. This can be a particular challenge when dealing with social media data, such as tweets, which may have variable length and structure. Removing extra spaces is an important step in data cleaning and text processing, particularly when working with natural language text. Extra spaces can occur in text due to a variety of reasons such as formatting, user input errors, or data extraction issues. These extra spaces can cause issues when processing or analyzing the text.

- Creating gold summaries manually can also be challenging. It can be difficult to determine which tweets are the most important or common for a particular event, and there may be disagreement among annotators about what should be included in the summary.

Overall, data set annotation can be a complex and time-consuming process that requires careful consideration and attention to detail. It may be helpful to use a standardized annotation framework and to regularly review and update the annotation guidelines to ensure consistency and accuracy.

Chapter 5

Proposed Methodology

In the domain of automatic text summarization, the Sequence to Sequence model is a popular approach, which consists of two RNNs: an encoder and a decoder. An Encoder-Decoder model with an LSTM network was also used in this experiment.

5.1 Encoder Architecture

The Encoder takes the variable length of input sequence and converts it into a fixed-dimensional vector representation called a "thought vector" or "context vector" using an RNN cell. This vector contains all the relevant information from the input sequence and this context vector use as decoder's main input. The building context vector is the main goal of the encoder. Let $x_i = (x_1, x_2, ..., x_n)$ be the input sequence and h_0 be the initial hidden state of the Encoder. For each input x_i, compute the hidden state h_t using f_t which is the transfer function of a recurrent neural cell like GRU, Vanila or LSTM.

$$h_t = f_t(x_i, h_{t-1}) \tag{1}$$

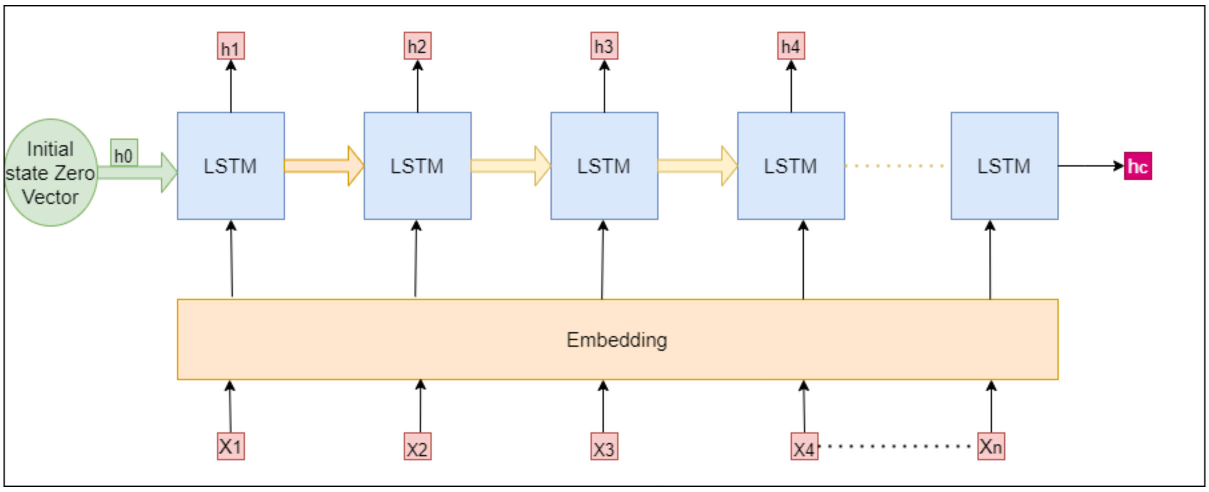


Figure 5.1: Encoder Architecture

The context vector h_c is the final hidden state when $t = n$:

$$h_c = h_n \tag{2}$$

5.2 Decoder Architecture

The decoder is a crucial component of the sequence-to-sequence model, responsible for generating the output sequence given the input sequence. It takes the final hidden state or context vector from the encoder as initial input and the previously generated tokens. The decoder follows an autoregressive approach, where it generates the output sequence one token at a time. At each time step, the decoder takes in the context vector, the previous output token, and the previous hidden state as inputs. It generates the next token in the output sequence by passing the previous token through an embedding layer, which converts the token into a dense vector representation. The dense vector representation is then fed into the RNN cell, which updates the decoder's internal hidden state. The output of the RNN cell is passed through a softmax layer, which generates a probability distribution over the possible next tokens in the output sequence. The decoder uses the context vector to initialize its internal hidden state and starts generating the output sequence. The decoder's internal hidden state serves as its memory and is updated at each time step based on the input token and the previous hidden state. It captures the dependencies between the input and output sequences and helps the decoder generate the next token in the output sequence. Let $y_i = (y_1, y_2, ..., y_m)$ be the target sequence and s_t be the initial hidden state of the Decoder, initialized to the context vectorh_c. For each output y_i, compute the hidden state s_t using the Decoder's RNN or LSTM:

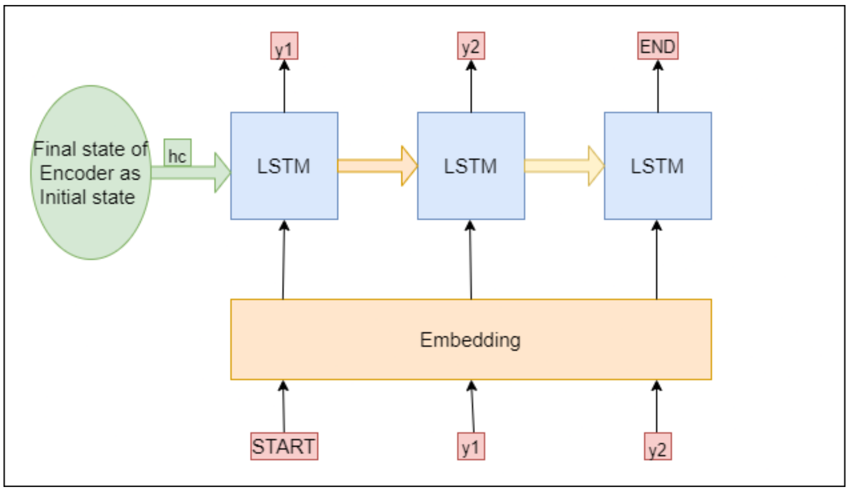


Figure 5.2: Decoder Architecture

$$s_t = f_t(yi - 1, s_{t-1}, h_i) \tag{3}$$

where h_i is a weighted sum of the Encoder's hidden states. The probability distribution over the next word is computed as:

$$P(y_i|y_1, y_2, ..., y_{i-1}, h_c) = softmax(y_{i-1} * W_y * s_t + by) \tag{4}$$

where W_y is a weight matrix and by is a bias vector

5.3 Explain Inference Phase

In order to decode a test sequence after training, an inference architecture must be configured. The LSTM unit in the decoder outputs $y1, y2, y3, \ldots, yk$ at each time step, where k is the length of the output sequence. The output $y1$ is generated at time step t=1, and $y2$ is generated at time step t=2. The decoder is provided with the "start" token as a source of input. The decoder is executed using its inner state for a single step, and the output will be the probability for the next word. The word with the highest likelihood is selected, and the internal states are updated with the current time step. The decoder is provided with the sampled word as input in the following timestamp, and this process continues until the target sequence reaches its maximum length or an "end" token is issued.

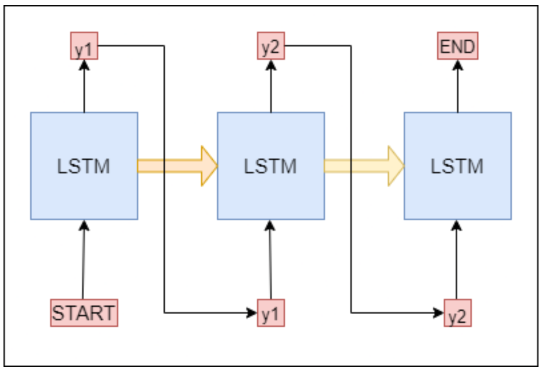


Figure 5.3: Inference Model Architecture

5.4 LSTM Architecture

Recurrent neural networks (RNNs) are extended to LSTM networks[24; 27], which were developed primarily to address Long-Term Dependency problem of RNN and Normally RNN have standard recurrent cells like sigma cell and tanh cell, but it's remember capacity are improved using gate concept into cell in LSTM networks. The information in a sequence of data enters, is stored in, and exits the network using a series of 'gates' that are used by LSTMs. A typical LSTM has three gates: a forget gate, an input gate, and an output gate. These gates can be viewed as filters, each with their own neural network, and a highly distinctive method for these layers to interact.

The forget layer is the first step in the process, and decide what information is relevant given the previous hidden state and the current input data will be determined here. It is basically a sigmoid layer which creates a vector with each input and falling inside the range [0,1]. This layer trains to output a value that is roughly close to 0, when an input component is regarded irrelevant. When It is considered important, then value is roughly close to 1. Consider each component of this vector as a sort of filter or sieve that lets in more information. Following that, these output values are transferred upward and pointwise multiplied with the previous cell state. Because of this pointwise multiplication, elements of the cell state that the forget gate network has considered unnecessary will be multiplied by a value near to 0 and will therefore have less of an impact on the subsequent stages. The next layer Input gate decided what new information have to add, and it has two parts. The first part is another sigmoid decides which value will be change and second part tanh creates vector of new information values. Basically, this step's objective is to determine, on the basis of the earlier hidden state and fresh data entering the network, what new information ought to be added to the network's long-term memory and then Input layers update the previous state. In the output gate layer, a sigmoid layer is used to decide which parts of the cell sate release as output. The cell state is passed though the tanh gate and multiply with the output of sigmoid gate. The multiply result is output of current state. Figure 5.4 give the details about LSTM cell architecture.

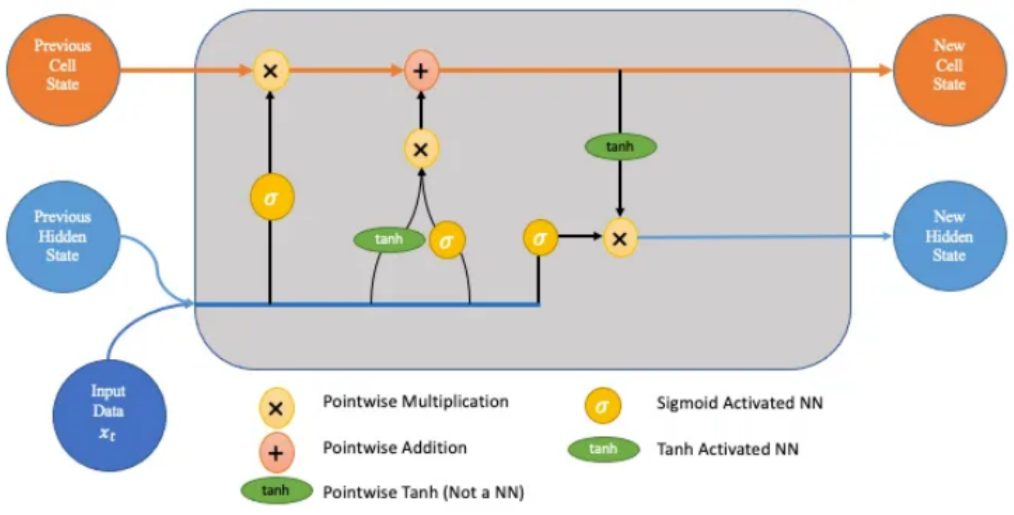


Figure 5.4: LSTM Architecture.

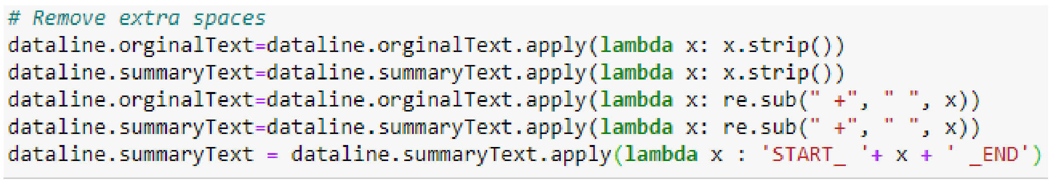
Chapter 6

Implementation Procedure

A sequence-to-sequence model using LSTM(Long Short Term Memory) architecture is developed for event base tweets summarization on our own dataset.

6.1 Dataset prepossessing

The tweets collected require preprocessing, as does the dataset, before applying a deep learning model. During preprocessing, extra spaces, single quotes, and double quotes are removed, and a tag is added to the summary text of the dataset as ”_START” at the beginning and ”_END” at the end, as shown in Figure 6.1.



```
# Remove extra spaces
dataline.orginalText=dataline.orginalText.apply(lambda x: x.strip())
dataline.summaryText=dataline.summaryText.apply(lambda x: x.strip())
dataline.orginalText=dataline.orginalText.apply(lambda x: re.sub(" +", " ", x))
dataline.summaryText=dataline.summaryText.apply(lambda x: re.sub(" +", " ", x))
dataline.summaryText = dataline.summaryText.apply(lambda x : 'START_ '+ x + ' _END')
```

Figure 6.1: Data Prepossessing .

Tags such as _START and _END are added to the summary text to aid the model in identifying the beginning and end of the summary, which is useful for text generation or summarization. At this stage, the dataset is also tokenized by breaking the text into individual words or tokens, which can then be represented numerically using techniques such as one-hot encoding or word embedding. The maximum sequence length for the input and target text is also calculated, which can be important for efficient training of the model, as it allows for optimization of batch size and memory requirements. Overall, data preprocessing is a crucial step in building an effective deep learning model, as it ensures that the model can learn from the data and make accurate predictions.

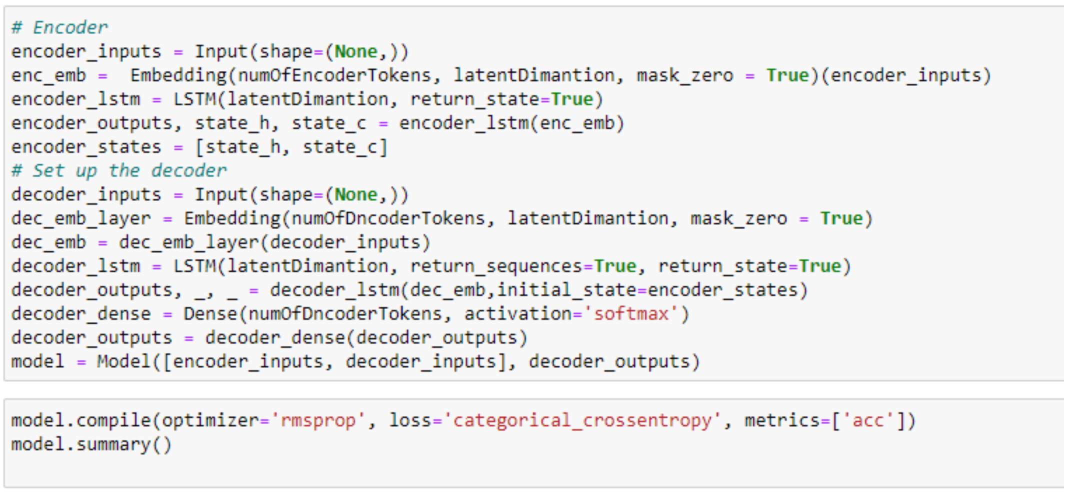
6.2 Experiment Setup

The model was implemented in Keras and consisted of an encoder and a decoder, each with a hidden state vector of size latentDimension 400, which is a hyperparameter that determines the complexity of the model and can be adjusted to optimize performance. Tokenization is the process of breaking down text into smaller units, such as individual words or subwords, that can be more easily processed by the model. By tokenizing the input and output data into num_encoder_tokens and num_decoder_tokens distinct tokens, respectively, the complexity of the data is effectively reduced, making it more amenable to processing by the model. The input data were tokenized into, 9925 distinct tokens, while the output data were tokenized into 550 distinct tokens.

During training, the number of samples per batch size was set to 10, and the model was trained for 135 epochs. To optimize the model, the RMSprop optimizer with a learning rate of 0.001 was used, and the categorical cross-entropy loss function was employed. The dataset was divided into a 9:1 ratio for training and validation purposes, with the training dataset containing 41 events and the validation dataset containing 5 events. Table 4 provides a summary of the parameters and hyperparameters used in this experiment.

Table 4: Parameter and Hyperparameter Details of Model.

Parameter & Hyperparameter	**Details**
numOfEncoderTokens	9925
numOfDecoderTokens	550
latentDmention	400
batchSize	10
Epoch	135
Loss Function	categorical cross-entropy
Optimizer	RMSprop
Training Data	41(90%)
Validation Data	5(10%)



```
# Encoder
encoder_inputs = Input(shape=(None,))
enc_emb =  Embedding(numOfEncoderTokens, latentDimantion, mask_zero = True)(encoder_inputs)
encoder_lstm = LSTM(latentDimantion, return_state=True)
encoder_outputs, state_h, state_c = encoder_lstm(enc_emb)
encoder_states = [state_h, state_c]
# Set up the decoder
decoder_inputs = Input(shape=(None,))
dec_emb_layer = Embedding(numOfDncoderTokens, latentDimantion, mask_zero = True)
dec_emb = dec_emb_layer(decoder_inputs)
decoder_lstm = LSTM(latentDimantion, return_sequences=True, return_state=True)
decoder_outputs, _, _ = decoder_lstm(dec_emb,initial_state=encoder_states)
decoder_dense = Dense(numOfDncoderTokens, activation='softmax')
decoder_outputs = decoder_dense(decoder_outputs)
model = Model([encoder_inputs, decoder_inputs], decoder_outputs)
```

```
model.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['acc'])
model.summary()
```

Figure 6.2: Encoder Decoder Model

6.3 Train Encoder Decoder Model

The encoder-decoder structure used in the code presented in Figure 6.2 creates a deep learning model for sequence-to-sequence text summarization. The model is built to take integer sequences as input and provide summarize sequences in the same format. An encoder and a decoder are the two primary parts of the model. A context vector is produced by the encoder after processing the input sequence. The decoder can use this context

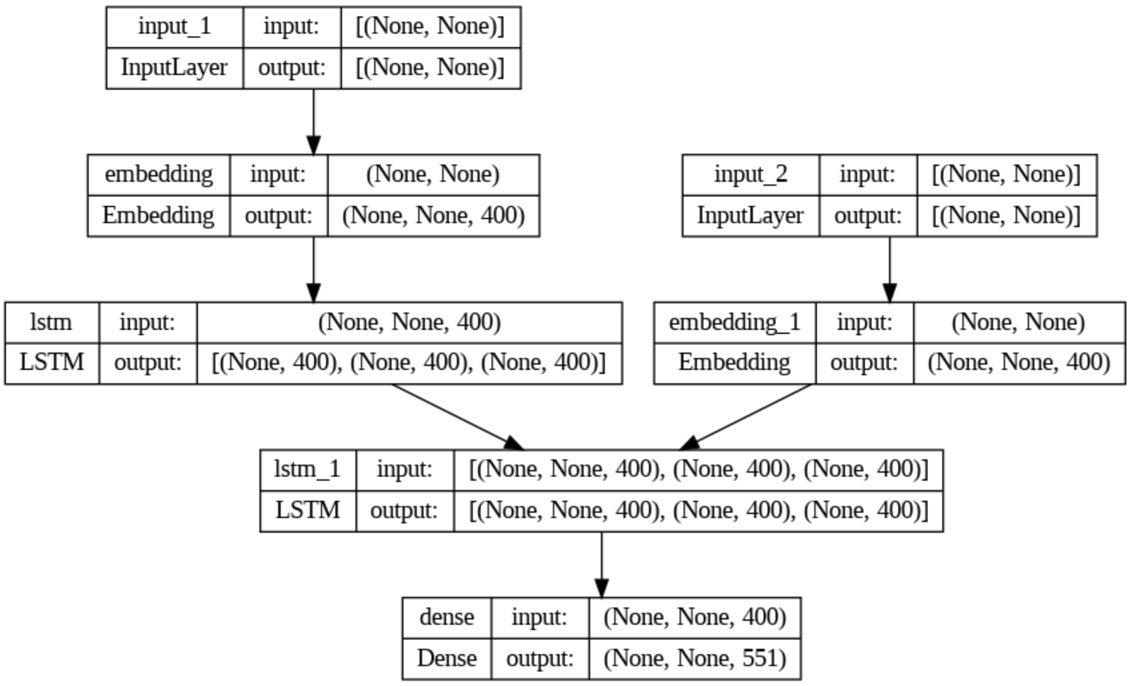

input_1 | input: | [(None, None)]
InputLayer | output: | [(None, None)]
embedding | input: | (None, None)
Embedding | output: | (None, None, 400)
input_2 | input: | [(None, None)]
InputLayer | output: | [(None, None)]
lstm | input: | (None, None, 400)
LSTM | output: | [(None, 400), (None, 400), (None, 400)]
embedding_1 | input: | (None, None)
Embedding | output: | (None, None, 400)
lstm_1 | input: | [(None, None, 400), (None, 400), (None, 400)]
LSTM | output: | [(None, None, 400), (None, 400), (None, 400)]
dense | input: | (None, None, 400)
Dense | output: | (None, None, 551)

Figure 6.3: Model Summary.

vector to create the output sequence by summarizing the input sequence. A dense vector representation of each word in the input sequence is created by the embedding layer of the encoder. An LSTM layer comes after the embedding layer, processing the embedded input sequence and producing a series of hidden states and a final hidden state. The context vector produced by the encoder is used by the decoder to produce the output sequence. The decoder has an embedding layer, just like the encoder, which turns each word in the target sequence into a dense vector representation. The final hidden state of the encoder LSTM layer is used to initialize the decoder LSTM layer. The embedded input sequence is processed by the decoder LSTM, which then generates a series of hidden states that are fed to a dense layer to predict the subsequent word in the target sequence. Given the “thought vectors” obtained from the encoder model, a different decoder model will be used to produce the output sequence. The embedded decoder sequence and previous time step states are inputs to the model's decoder LSTM layer. The decoder LSTM layer's starting states are derived from the "Thought vectors" in an encoder. A dense softmax layer is applied after the decoder LSTM layer to the output sequence, which creates a probability distribution over the target summary. The input sequence, the previous time step states, and the output sequence are used to define the final decoder model. The decoder input sequence and the previous time step states are listed as the model's input, while the produced output sequence and the current time step states are listed as the model's output. Figure 6.3 give details about model summary.

Chapter 7

Evaluation Metrics and Result

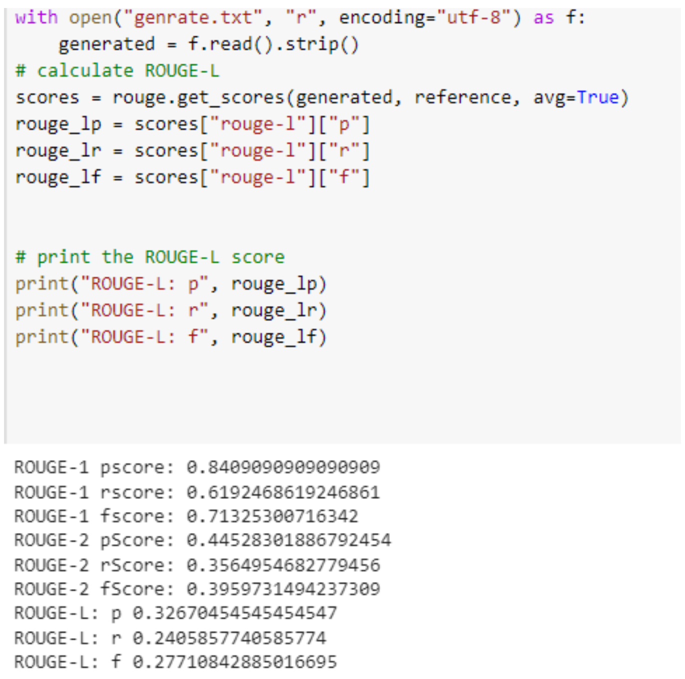
Recall-Oriented Understudy for Gisting Evaluation is known as ROUGE. ROUGE is a set of evaluation metrics used to assess the quality of text summarization or text generation systems. It measures the similarity between a generated summary and one or more reference summaries, based on the overlap of n-grams (sequences of n words) between the two texts. These measures are frequently used to assess the effectiveness of text summarization or text generation systems.

ROUGE-1 (unigram ROUGE) measures the overlap of unigrams (individual words) between the generated text and the reference summary. It calculates the precision, recall, and F1 score based on the number of unigrams shared between the two texts. Precision is the ratio of the number of common unigrams to the total number of unigrams in the generated summary, while recall is the ratio of the number of common unigrams to the total number of unigrams in the reference summary. F1 score is the harmonic mean of precision and recall, and it balances the trade-off between the two measures.

ROUGE-2 measures the overlap of bigrams (two sequential words) between the generated text and the reference summary. It is helpful in evaluating the produced summary's contextual and syntactical coherence. The calculation of precision, recall, and F1 score for ROUGE-2 is similar to that of ROUGE-1, but it uses bigrams instead of unigrams. ROUGH-1 and ROUGH-2 were used as our experiment's success evaluation metrics.

ROUGE-L is a metric for evaluating the quality of automatic summaries. It measures the longest common subsequence (LCS) between the generated summary and the reference summary. LCS is the longest sequence of words that are present in both the generated summary and the reference summary. ROUGE-L takes into account the fact that a good summary should contain not only important words and phrases, but also the correct order of those words and phrases. It calculates the F1 score, which is the harmonic mean of precision and recall. Precision is the number of words in the LCS divided by the number of words in the generated summary, while recall is the number of words in the LCS divided by the number of words in the reference summary. In this experiment, F1 score of ROUGE-1, ROUGE-2 and ROUGE-L is 71%, 39% and 27% respectively mention in Figure 7.1.

Precision = Number of Common Bigrams / Number of Bigrams in Generated summary
Recall = Number of Common Bigrams / Number of Bigrams in Reference summary
F1 score = 2 * precision * recall / (precision + recall)



```
with open("genrate.txt", "r", encoding="utf-8") as f:
    generated = f.read().strip()
# calculate ROUGE-L
scores = rouge.get_scores(generated, reference, avg=True)
rouge_lp = scores["rouge-l"]["p"]
rouge_lr = scores["rouge-l"]["r"]
rouge_lf = scores["rouge-l"]["f"]


# print the ROUGE-L score
print("ROUGE-L: p", rouge_lp)
print("ROUGE-L: r", rouge_lr)
print("ROUGE-L: f", rouge_lf)
```

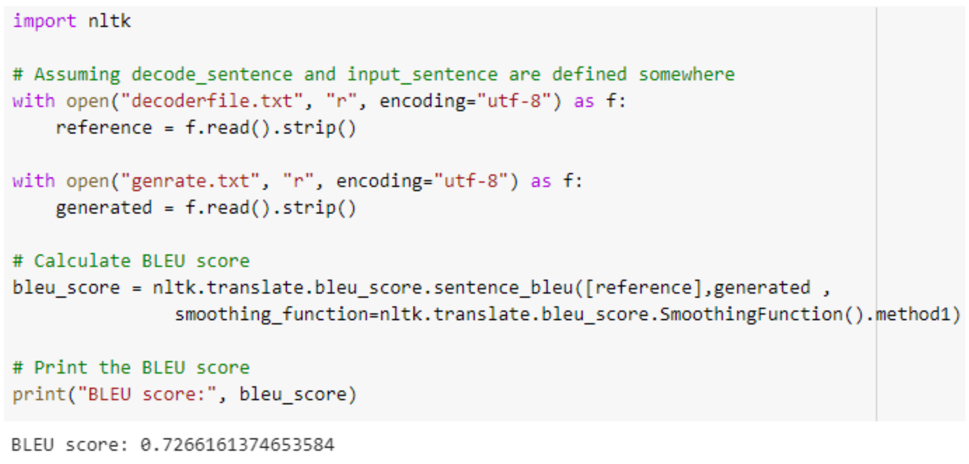
```
ROUGE-1 pscore: 0.8409090909090909
ROUGE-1 rscore: 0.6192468619246861
ROUGE-1 fscore: 0.71325300716342
ROUGE-2 pScore: 0.44528301886792454
ROUGE-2 rScore: 0.3564954682779456
ROUGE-2 fScore: 0.3959731494237309
ROUGE-L: p 0.32670454545454547
ROUGE-L: r 0.2405857740585774
ROUGE-L: f 0.27710842885016695
```

Figure 7.1: Rouge Score

For example, suppose a generated summary and a reference summary for a news article:

Generated summary: The new iPhone model has been released.
Reference summary: Apple has released its latest iPhone model.

At first need to identify the unigrams and bigrams that appear in both summaries to calculate ROUGE-1 and ROUGE-2:
Common unigrams: the, new, iPhone, model, has, been, released
Common bigrams: the new, new iPhone, iPhone model, model has, has been, been released



```
import nltk

# Assuming decode_sentence and input_sentence are defined somewhere
with open("decoderfile.txt", "r", encoding="utf-8") as f:
    reference = f.read().strip()

with open("genrate.txt", "r", encoding="utf-8") as f:
    generated = f.read().strip()

# Calculate BLEU score
bleu_score = nltk.translate.bleu_score.sentence_bleu([reference],generated ,
               smoothing_function=nltk.translate.bleu_score.SmoothingFunction().method1)

# Print the BLEU score
print("BLEU score:", bleu_score)
```

```
BLEU score: 0.7266161374653584
```

Figure 7.2: BLEU Score

In the given scenario depicted in Figure 7.2, the variable “generate” holds the summaries generated by the Encoder Decoder model, while “reference” contains the actual summaries from the entire dataset.The Bilingual Evaluation Understudy(BLEU)[21] score of 0.72 was computed on these two variables, which indicates a high level of accuracy and suggests that our summarization model is performing exceptionally well. This result is indicative of the model’s ability to produce summaries that are similar to the original summaries in the dataset, thus demonstrating its effectiveness.

Performance of this experiment is compared with two exiting datasets in Table 5. The row A1(proposed) of the table shows performance of the proposed model, and approach(A2)[2] is a bengali abstractive news summarization approach based on encoder decoder model with LSTM on “BANS” dataset which is already discussed in Related Work. In other approach(A3), [6] 36 news article tweets collected for the summarization and its ROUGE-2 score is better than the proposed model. Overall, performance of proposed is good among these approaches.

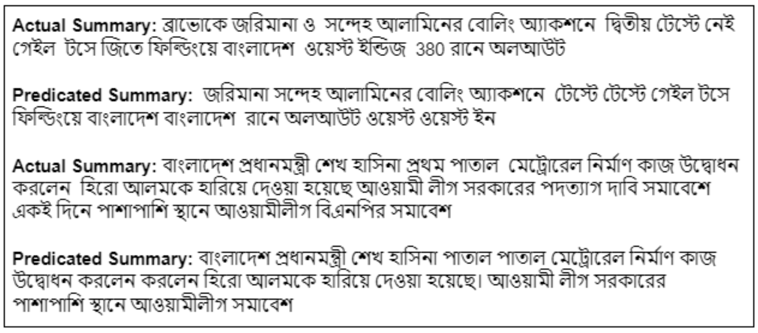
Table 5: Performance Comparision

Approach	**Rouge-1**	**Rouge-2**	**Rouge-L**	**BLEU**
A1	**0.71**	0.39	0.27	**0.72**
A2	0.30	Not Given	**0.31**	0.30
A3	0.66	**0.54**	Not Given	Not Given

Chapter 8

Discussion

A ROUGE-1 score of 0.71 indicates that the generated summaries share around 71% of their unigrams with the reference summaries. This means that the summaries contain many of the same words as the reference summaries, but there may be some differences in wording, word order, or grammatical structure. ROUGE-2 score of 0.39 means that the generated summaries share around 39% of their bigrams with the reference summaries. This is a lower score than ROUGE-1, indicating that the generated summaries may have more issues with contextual and syntactical coherence, as ROUGE-2 takes into account the overlap of sequential words and ROUGE-L score of 0.27 suggests that the generated summaries have a relatively low overlap with the reference summaries in terms of longest common sub-sequences. This indicates that the generated summaries may not capture the main themes or ideas expressed in the reference summaries. Figure 8.2 show that when original summary length is small, then generated summary is more accurate. But when original summary length is long, then model produce inconsistent summary in Figure 8.1. Most Probably the training dataset doesn't have enough examples of longer summaries for a particular category of events, then the model may not have learned to capture the nuances and complexities of longer summaries for that category. This can result in poor summarization performance for longer input sequences in that category.



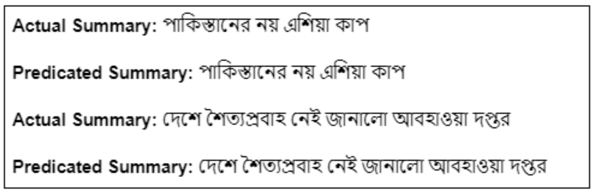
Actual Summary: ব্রাভোকে জরিমানা ও সন্দেহ আলামিনের বোলিং অ্যাকশনে দ্বিতীয় টেস্টে নেই গেইল টসে জিতে ফিল্ডিংয়ে বাংলাদেশ ওয়েস্ট ইন্ডিজ ৩৪০ রানে অলআউট

Predicated Summary: জরিমানা সন্দেহ আলামিনের বোলিং অ্যাকশনে টেস্টে টেস্টে গেইল টসে ফিল্ডিংয়ে বাংলাদেশ বাংলাদেশ রানে অলআউট ওয়েস্ট ওয়েস্ট ইন

Actual Summary: বাংলাদেশ প্রধানমন্ত্রী শেখ হাসিনা প্রথম পাতাল মেট্রোরেল নির্মাণ কাজ উদ্বোধন করলেন হিরো আলমকে হারিয়ে দেওয়া হয়েছে আওয়ামী লীগ সরকারের পদত্যাগ দাবি সমাবেশে একই দিনে পাশাপাশি স্থানে আওয়ামীলীগ বিএনপির সমাবেশ

Predicated Summary: বাংলাদেশ প্রধানমন্ত্রী শেখ হাসিনা পাতাল পাতাল মেট্রোরেল নির্মাণ কাজ উদ্বোধন করলেন করলেন হিরো আলমকে হারিয়ে দেওয়া হয়েছে। আওয়ামী লীগ সরকারের পাশাপাশি স্থানে আওয়ামীলীগ সমাবেশ

Figure 8.1: Summary Comparison-1.



Actual Summary: পাকিস্তানের নয় এশিয়া কাপ

Predicated Summary: পাকিস্তানের নয় এশিয়া কাপ

Actual Summary: দেশে শৈত্যপ্রবাহ নেই জানালো আবহাওয়া দপ্তর

Predicated Summary: দেশে শৈত্যপ্রবাহ নেই জানালো আবহাওয়া দপ্তর

Figure 8.2: Summary Comparison-2.

Chapter 9

Conclusion

The current state of event-based tweet summarization is focused on graphical and statistical-based approaches, with a greater emphasis on English tweets rather than Indian languages. However, there are some abstractive summarization approaches available for Bengali social media text. In this study, the focus is on Bengali tweets that cover various categories of events, and the model's performance on this dataset is confident. However, increasing the dataset size and the number of events in different categories could further improve the model's performance. Currently, various pre-trained models are being used for text summarization, and some related works are included in this study. However, the methodology used in this study cannot perform on real-time tweet data. In the future, an extended version of this study could collect real-time Bengali tweets from the tweet stream, classify them into events, and automatically prepare the data for analysis. Despite the lack of proper datasets and resources, this approach sheds light on the future direction of event-based tweet summarization for Indian languages.

References

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[2] Prithwiraj Bhattacharjee, Avi Mallick, and Md Saiful Islam. Bengali abstractive news summarization (bans): a neural attention approach. In *Proceedings of International Conference on Trends in Computational and Cognitive Engineering: Proceedings of TCCE 2020*, pages 41–51. Springer, 2021.

[3] Iman Munire Bilal, Bo Wang, Adam Tsakalidis, Dong Nguyen, Rob Procter, and Maria Liakata. Template-based abstractive microblog opinion summarization. *Transactions of the Association for Computational Linguistics*, 10:1229–1248, 2022.

[4] Kunal Chakma, Amitava Das, and Swapan Debbarma. Summarization of twitter events with deep neural network pre-trained models. In *Information Management and Big Data: 7th Annual International Conference, SIMBig 2020, Lima, Peru, October 1–3, 2020, Proceedings*, pages 45–62. Springer, 2021.

[5] Deepayan Chakrabarti and Kunal Punera. Event summarization using tweets. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 5, pages 66–73, 2011.

[6] Roshni Chakraborty, Maitry Bhavsar, Sourav Kumar Dandapat, and Joydeep Chandra.

Tweet summarization of news articles: An objective ordering-based perspective. *IEEE Transactions on Computational Social Systems*, 6(4):761–777, 2019.

[7] CR Dhivyaa, K Nithya, T Janani, K Sathis Kumar, and N Prashanth. Transliteration based generative pre-trained transformer 2 model for tamil text summarization. In *2022 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–6. IEEE, 2022.

[8] Alexis Dusart, Karen Pinel-Sauvagnat, and Gilles Hubert. Issumset: a tweet summarization dataset hidden in a trec track. In *Proceedings of the 36th annual ACM symposium on applied computing*, pages 665–671, 2021.

[9] Soumi Dutta, Vibhash Chandra, Kanav Mehra, Sujata Ghatak, Asit Kumar Das, and Saptarshi Ghosh. Summarizing microblogs during emergency events: A comparison of extractive summarization algorithms. In *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2018, Volume 2*, pages 859–872. Springer, 2019.

[10] Alexander R Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R Radev. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. *arXiv preprint arXiv:1906.01749*, 2019.

[11] Fatema Akter Fouzia, Minhajul Abedin Rahat, Md Tahmid Alie-Al-Mahdi, Abu Kaisar Mohammad Masum, Sheikh Abujar, and Syed Akhter Hossain. A bengali text summarization using encoder-decoder based on social media dataset. In *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2020, Volume 2*, pages 539–549. Springer, 2021.

[12] Piyush Kumar Garg, Roshni Chakraborty, and Sourav Kumar Dandapat. Endsum: Entropy and diversity based disaster tweet summarization. *arXiv preprint arXiv:2203.01188*, 2022.

[13] Piyush Kumar Garg, Roshni Chakraborty, and Sourav Kumar Dandapat. Ontorealsumm: Ontology based real-time tweet summarization. *arXiv preprint arXiv:2201.06545*, 2022.

[14] Fei Geng, Qilie Liu, and Ping Zhang. A time-aware query-focused summarization of an evolving microblogging stream via sentence extraction. *Digital Communications and Networks*, 6(3):389–397, 2020.

[15] Poonam Goyal, Prerna Kaushik, Pranjal Gupta, Dev Vashisth, Shavak Agarwal, and Navneet Goyal. Multilevel event detection, storyline generation, and summarization for tweet streams. *IEEE Transactions on Computational Social Systems*, 7(1):8–23, 2019.

[16] Gloria Hristova, Boryana Bogdanova, and Nikolay Netov. Data mining of public opinion: An overview. In *AIP Conference Proceedings*, volume 2505, page 020004. AIP Publishing LLC, 2022.

[17] Md Muhaiminul Islam, Mohiyminul Islam, Abu Kaisar Mohammad Masum, Sheikh Abujar, and Syed Akhter Hossain. Abstraction based bengali text summarization using bidirectional attentive recurrent neural networks. In *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2020, Volume 2*, pages 317–327. Springer, 2021.

[18] Quanzhi Li and Qiong Zhang. Abstractive event summarization on twitter. In *Companion Proceedings of the Web Conference 2020*, pages 22–23, 2020.

[19] Abu Kaisar Mohammad Masum, Sheikh Abujar, Md Ashraful Islam Talukder, AKM Shahariar Azad Rabby, and Syed Akhter Hossain. Abstractive method of text summarization with sequence to sequence rnns. In *2019 10th international conference on computing, communication and networking technologies (ICCCNT)*, pages 1–5. IEEE, 2019.

[20] Abu Kaisar Mohammad Masum, Md Majedul Islam, Sheikh Abujar, Amit Kumer Sorker, and Syed Akhter Hossain. Bengali news headline generation on the basis of sequence to sequence learning using bi-directional rnn. In *Soft Computing Techniques and Applications: Proceeding of the International Conference on Computing and Communication (IC3 2020)*, pages 491–501. Springer, 2021.

[21] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[22] Koustav Rudra, Pawan Goyal, Niloy Ganguly, Muhammad Imran, and Prasenjit Mitra. Summarizing situational tweets in crisis scenarios: An extractive-abstractive approach. *IEEE Transactions on Computational Social Systems*, 6(5):981–993, 2019.

[23] Dwijen Rudrapal, Amitava Das, and Baby Bhattacharya. A new approach for twitter event summarization based on sentence identification and partial textual entailment. *Computación y Sistemas*, 23(3):1065–1078, 2019.

[24] Ralf C Staudemeyer and Eric Rothstein Morris. Understanding lstm–a tutorial into long short-term memory recurrent neural networks. *arXiv preprint arXiv:1909.09586*, 2019.

[25] Mariyam Sultana, Partha Chakraborty, and Tanupriya Choudhury. Bengali abstractive news summarization using seq2seq learning with attention. In *Cyber Intelligence and Information Retrieval: Proceedings of CIIR 2021*, pages 279–289. Springer, 2022.

[26] N Vijay Kumar and M Janga Reddy. Factual instance tweet summarization and opinion analysis of sport competition. In *Soft Computing and Signal Processing: Proceedings of ICSCSP 2018, Volume 2*, pages 153–162. Springer, 2019.

[27] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.