

# Bang-lish Sentiment Classification Using Deep Learning

by

Abrar Saleheen

19101331

Saleh Ahmed Siam

19101140

Sanjeda Akter

19101258

Akash Ghosh

19101425

Adiba Anbar Ahona

19101257

A thesis submitted to the Department of Computer Science and Engineering  
in partial fulfillment of the requirements for the degree of  
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering  
School of Data and Sciences  
Brac University  
January 2023

© 2023. Brac University  
All rights reserved.

# Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

**Student's Full Name & Signature:**



---

Abrar Saleheen  
19101331



---

Saleh Ahmed Siam  
19101140



---

Sanjeda Akter  
19101258



---

Akash Ghosh  
19101425



---

Adiba Anbar Ahona  
19101257

# Approval

The thesis titled “Banglish Text Classification using Deep Learning” submitted by

1. Abrar Saleheen (19101331)
2. Saleh Ahmed Siam (19101140)
3. Sanjeda Akter (19101258)
4. Akash Ghosh (19101425)
5. Adiba Anbar Ahona (19101257)

Of January, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science and Engineering on January 17, 2023.

## Examining Committee:

Supervisor:  
(Member)



---

Arif Shakil  
Lecturer  
Department of Computer Science and Engineering  
Brac University

Co-Supervisor:  
(Member)

**Annajiat  
Alim  
Rasel** Digitally signed by  
Annajiat Alim Rasel  
DN: cn=Annajiat Alim  
Rasel, o=Brac University,  
ou=CSE Department,  
email=annajiat@bracu.ac.  
bd, c=BD  
Date: 2023.01.14 23:04:00  
+06'00'

---

Annajiat Alim Rasel  
Senior Lecturer  
Department of Computer Science and Engineering  
Brac University

Head of Department:  
(Chair)

---

Sadia Hamid Kazi, PhD  
Chairperson and Associate Professor  
Department of Computer Science and Engineering  
Brac University

Thesis Coordinator:  
(Chair)

---

Golam Rabiul Alam, PhD  
Professor  
Department of Computer Science and Engineering  
Brac University

## **Ethics Statement**

This thesis was prepared based on the outcomes of our in-depth investigation, we the authors hereby truly proclaim. The sources that were used are highly acknowledged and credited in this study. This research paper has never been sent to any organization for certification.

# Abstract

E-commerce websites and social media platforms have become integral parts of people's social lives. Through posts, comments, and reviews on social media and online shopping websites, people can share their ideas. Understanding people's opinions and evaluating input requires the ability to classify sentiment. A variety of deep learning methods have been employed over time to categorize sentiment. Bang-lish, which consists of Bengali words printed in English letters, has gotten very little notice nonetheless. In addition, the majority of Bengali-speaking individuals utilize Bang-lish to post evaluations on e-commerce platforms. Understanding customers' ideas are crucial for sellers who want to improve their goods. Bang-lish, however, is challenging to understand and evaluate since it lacks a set grammar. Convolutional neural networks (CNN) and gated recurrent units (GRU), two types of deep learning, are combined in this study's proposed hybrid framework. Additionally, during the data preprocessing stage, we created a spell check algorithm for the most frequently used words and eliminated Bang-lish stop-words. In binary sentiment classification, our suggested model achieved 89% accuracy, 88% precision, 89% recall, and an 89% F1 score.

**Keywords:** Sentiment classification; Spell Corrector; Hybrid Model; Convolutional Neural Network; Gated Recurrent Unit.

## Acknowledgement

The merciful Almighty God—the Most High, the Most Benevolent, and the Most Merciful—deserves all praise for having granted us the opportunity to study for a bachelor’s degree here at BRAC University. It would not have been possible for us to travel the path to earning a bachelor’s degree without the assistance of many people. We would like to take this opportunity to thank them for their encouragement and guidance.

We appreciate Arif Shakil Sir, our supervisor, and Annajiat Alim Rasel Sir, our co-supervisor, for their kind assistance and counsel in our endeavor. Then, a special thanks to our first thesis supervisor Late Hossain Arif Sir, Assistant Professor, Department of Computer Science and Engineering, without whose guidance and invaluable time we could not have progressed with this project.

Finally, we should thank our parents because we might not have been successful without their constant support. We are now just a few steps away from graduating thanks to their kind prayers and support.

# Table of Contents

<b>Declaration</b>	<b>i</b>
<b>Approval</b>	<b>ii</b>
<b>Ethics Statement</b>	<b>iv</b>
<b>Abstract</b>	<b>v</b>
<b>Acknowledgment</b>	<b>vi</b>
<b>Table of Contents</b>	<b>vii</b>
<b>List of Figures</b>	<b>ix</b>
<b>List of Tables</b>	<b>x</b>
<b>Nomenclature</b>	<b>xi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Thoughts Behind Prediction Model . . . . .	1
1.2 Problem Statement . . . . .	2
1.3 Research Objectives . . . . .	3
<b>2 Literature Review</b>	<b>4</b>
2.1 Sentiment Analysis . . . . .	4
2.2 Convolutional Neural Network(CNN) . . . . .	4
2.3 Gated Recurrent Unit(GRU) . . . . .	4
2.4 Previous Research . . . . .	5
<b>3 Formation &amp; Deployment of Primary Dataset</b>	<b>12</b>
3.1 Web Scrapping . . . . .	12
3.2 Data Collection . . . . .	14
3.3 Data Labeling . . . . .	16
3.4 Data Pre-Processing . . . . .	16
3.4.1 Remove emojis . . . . .	17
3.4.2 Remove punctuation marks . . . . .	17
3.4.3 Tokenization . . . . .	17
3.4.4 Lower casing . . . . .	17
3.4.5 Stop word removal . . . . .	17
3.4.6 Misspelled word handling . . . . .	18



3.4.7	One hot encoding . . . . .	20
<b>4</b>	<b>Analysis of Sentiment Classification Models</b>	<b>23</b>
4.1	Sentiment Analysis Models . . . . .	23
4.1.1	Neural Networks (NN) . . . . .	23
4.1.2	Neural Language Model (NLM) . . . . .	24
4.1.3	Convolutional Neural Network (CNN) . . . . .	24
4.1.4	Recurrent Neural Network (RNN) . . . . .	25
4.1.5	Long Short-Term Memory (LSTM) . . . . .	27
4.1.6	Gated Recurrent Unit (GRU) . . . . .	29
4.2	Proposed Model . . . . .	30
4.2.1	Embedding Layer . . . . .	30
4.2.2	CNN Layers . . . . .	30
4.2.3	Drop out Layer . . . . .	31
4.2.4	Gated Recurrent Unit (GRU) Layer . . . . .	31
4.2.5	Flatten Layer . . . . .	31
4.2.6	Output Layer . . . . .	31
4.2.7	Hyper-parameters . . . . .	32
<b>5</b>	<b>Result Analysis and Evaluation</b>	<b>35</b>
5.1	Result Assessment . . . . .	35
5.1.1	Confusion Matrix . . . . .	35
5.1.2	Recall . . . . .	36
5.1.3	Precision . . . . .	36
5.1.4	F1 Score . . . . .	37
5.1.5	Accuracy . . . . .	37
5.2	Performance Comparison . . . . .	39
<b>6</b>	<b>Conclusion</b>	<b>40</b>
6.1	Final remarks . . . . .	40
6.2	Limitations . . . . .	40
6.2.1	Access . . . . .	40
6.2.2	Prior research . . . . .	40
6.2.3	Sample Size . . . . .	41
6.2.4	Data aggregation . . . . .	41
6.2.5	Time restraint . . . . .	41
6.3	Future Works . . . . .	41
6.3.1	Dataset enhancement . . . . .	42
6.3.2	Multi-class Sentiment Analysis . . . . .	42
6.3.3	Support for Multiple Languages . . . . .	42
6.3.4	Deploying a Bang-lish ChatBot . . . . .	42
	<b>Bibliography</b>	<b>44</b>

# List of Figures

3.1	our scraper is collecting reviews . . . . .	13
3.2	Data saved as CSV file . . . . .	13
3.3	Result of the webscraper . . . . .	13
3.4	Snapshot of our dataset . . . . .	15
3.5	Word Cloud of our dataset . . . . .	15
3.6	Visual Representation of the data variations . . . . .	16
3.7	Data Pre-Processing . . . . .	18
3.8	Workflow of our dataset . . . . .	22
4.1	The basic structure of a Neural Network . . . . .	23
4.2	The common layer of a Convolutional Neural Network . . . . .	25
4.3	Pathflow of RNN . . . . .	26
4.4	Different type of RNN . . . . .	27
4.5	Common activation function of RNN . . . . .	27
4.6	Architecture of LSTM . . . . .	28
4.7	Components of Gated Recurrent Unit (GRU) . . . . .	30
4.8	The basic architecture of the suggested model . . . . .	32
5.1	Metric of Confusion Matrix . . . . .	35
5.2	Confusion Matrix . . . . .	36
5.3	Accuracy plot . . . . .	37
5.4	accuracy plot of LSTM(left) and CNN(right) . . . . .	38
5.5	accuracy plot of RNN(left) and GRU(right) . . . . .	38
5.6	accuracy plot of BiDirectional LSTM(left) and BiDirectional GRU(right) . . . . .	38
5.7	accuracy plot of CNN - LSTM . . . . .	38

# List of Tables

3.1	Example of Handling Misspelled Words . . . . .	19
3.2	One Hot Encoding . . . . .	21
5.1	Performance summary of proposed model . . . . .	37
5.2	Performance Comparison of Algorithms . . . . .	39

# Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

*CNN* Convolutional Neural Network

*GRU* Gated Recurrent Unit

*BPTT* Back Propagation Through Time

*DL* Deep Learning

*LSTM* Long Short Term Memory

*NLM* Neural Language Model

*NN* Neural Network

*RNN* Recurrent Neural Network

# Chapter 1

## Introduction

### 1.1 Thoughts Behind Prediction Model

Information technology is expanding quickly, and social media is undergoing a disruptive transformation. Incredibly popular websites where people can post text, photos, or videos to express their thoughts or feelings include Facebook, YouTube, Twitter, and Instagram. As the influence of the internet grows expeditiously, social media and other online-based marketing have become the norm. More specifically, the dynamic information can be found as messages in discussion forums, reviews, or comments. In that context, studying the thoughts of the common people toward various entities and commodities makes for improved contextual advertising, recommendation systems, and market trend analysis [1].

Sentiment classification refers to a natural language processing application that tries to understand the emotions of texts. To comprehend the sentiment of any text, researchers use a variety of deep learning and machine learning algorithms. Although it is rare, sentiment analysis of Bang-lish (Bengali words transcribed with English letters) material has become essential. Most Bengali-speaking people use Bang-lish in their daily writing on social media or reviews on e-commerce websites. Moreover, business organizations need to understand the reviews to improve their product quality.

Very few machine learning algorithms are considered to perform better for sentiment analysis than others. Naïve Bayes [2] is one of them. The Naïve Bayes algorithm can be improved by combining unigram and bigram features, along with limitations in the accuracy gap between positive and negative reviews. Recurrent neural networks can be used to create a multilingual sentiment analysis model, translating reviews into other languages and then utilizing the model to assess the sentiments. However, the vast majority of these resources cannot be fully applied to other fields, jobs, or languages. Sentiment analysis is one such procedure that necessitates extra work when translating data between languages [3].

For instance, the accuracy of sentiment analysis algorithms relies on the test dataset; for example, Naive Bayes outperforms K-NN when applied to movie reviews. On the other hand, for hotel reviews, both of them perform similarly [1]. Additionally, based on Deep Learning, a DNN model for forecasting the fineness of digital goods was developed in a study [4]. The collected dataset was compiled from popular online marketplaces in Bangladesh. Nowadays, deep learning models are used for sentiment classification to achieve higher accuracy than machine learning models. Some

researchers also introduced a unique attention-based CNN model for Bang-lish sentiment analysis [5]. Moreover, others suggested LSTM-CNN-SVM on English text [6].

Though portraying Bengali with English letters to interact online is becoming more and more common among the public, this research focuses on sentiment analysis of Bang-lish text using Convolution Neural Network (CNN) and Gated Recurrent Unit (GRU). Naturally, Bengali is naturally a quite complex language to learn, but it is even harder to grasp when it comes to typing. As a result, people often opt to type their opinions using Bang-lish. Apart from that, there is little to no study on Bang-lish texts. Also, proper data pre-processing is vital for getting better accuracy in sentiment classification. We developed a dataset-specific misspelled word handling algorithm. While CNN can extract essential features from text data, the recurrent neural network (RNN) interprets temporal or sequential information. However, RNN has a vanishing gradient issue, which prevents it from analyzing lengthy sentences. The GRU solves the vanishing gradient problem by using two gates. For this research, we have combined CNN and GRU to estimate the sentiment of Bang-lish text.

## 1.2 Problem Statement

Sentiment classification has been the subject of a wide variety of studies. However, there hasn't been much study of Bengali literature that uses English alphabetic characters. Bengali is an immensely difficult language. Typing in Bengali might be difficult for many people. So, people write in Bang-lish. People use this Bang-lish form of text in informal communication, such as commenting on social media posts, messaging a friend, and writing reviews of products they have bought. There are no fixed rules or grammar for writing in Bang-lish. There is no incorrect spelling in this form of writing. For example, প্রোডাক্ট টি ২ দিনেই নষ্ট হয়ে গেছে can be written as product ti 2 dine nosto hoye gese or product ti 2 dene nosto hoi gece. Moreover, people use shorter forms of words to express their opinions. Again, people use many trendy and weird words and phrases that have no specific meaning, for example, xoss, 100, jotiil, purai matha nosto, tnx etc. Libraries like nltk, spacy do not support this form of language. These make data pre-processing very hard. On online platforms, Bangladeshi people usually give reviews in 3 different ways-

1. Pure English (This product is very good.)
2. Pure Bengali (এই পণ্য খুব ভাল)
3. Bang-lish(ei product ti khub valo).

Many sentiment analysis models were developed for analyzing the sentiment of the reviews using deep and machine learning models, which give higher accuracy, but very few have been developed for Bang-lish text. Due to this problem, many sellers cannot receive overall feedback on their products.

Among those few works on Bang-lish sentiment analysis, this paper [10] used a unique attention-based CNN model combined with RNN for Bang-lish sentiment analysis. They got 87% accuracy on binary sentiment analysis, which needs to be improved. Finally, not being an actual language, Bang-lish texts are highly

unstructured. Again, no processing tools are available for this kind of language. The hybrid CNN-GRU model is never used on this type of data.

### 1.3 Research Objectives

To fill up this research gap, we apply proper data preprocessing techniques to prepare our dataset for feeding into the deep learning model. Without proper cleaning and processing of the data, the deep learning model might not perform well, which will eventually lead us to misinterpret the sentiment. Every language is different and has its own unique characteristics, but Bang-lish is not an actual language. Therefore, everybody has their own way of writing words in Bang-lish. In our preprocessing part, we handle the misspelled words for our dataset. Moreover, we prepared a Bang-lish stop-word list and removed the stop-words from our dataset.

Our major contribution:

- A CNN-GRU hybrid model that predicts the sentiment of Bengali words and sentences written in English letters
- Misspelled word handling algorithm for Bang-lish text

Engineers will be able to create tools for analyzing sentiment regardless of the language's structure, such as highly unstructured Bang-lish, according to the results of our research. Our study concentrated on online product reviews, but this may open up new possibilities for an investigation into the detection of fake news, the diagnosis of cyberbullying, the recognition of emotive states, and other topics from unstructured Bang-lish text.

# Chapter 2

## Literature Review

### 2.1 Sentiment Analysis

A strategy for determining a text or piece of literature's positivity, negativity, or neutrality is sentiment analysis. The effectiveness of statement analysis techniques, which can range from straightforward linear models to more complex deep neural models, has been the subject of numerous studies. Inductor models have recently been praised as the most complex approach for a variety of languages(English, Arabic, French, German, Turkish, etc.) and have shown great potential in sentiment analysis. In other words, sentiment analysis extracts sentiments related to the polarity of positive or negative for selected parts of a piece of writing rather than classifying the entire page positively or negatively [7]. It would have included tools for market research, risk management, and competitive analysis.

### 2.2 Convolutional Neural Network(CNN)

Convolutional neural network methods with multiple layers can be utilized to comprehend the characteristics of data hierarchies. Over the past few years, CNN has greatly advanced both the architecture and implementation of processing natural language (NLP) [8]. Neural networks are used to recognize patterns. These patterns are vector-encoded numerical representations of actual data, such as pictures, sounds, texts, or time series. Convolutional neural networks are neural networks that employ convolutional layers to evaluate local features. For the extraction of featured tasks, CNN is effective.

### 2.3 Gated Recurrent Unit(GRU)

The term "rectified linear units" refers to another popular variant of recurrent neural networks (GRUs). The GRU has a number of parameters because it lacks an output gate, but it is nearly identical to an LSTM with a forget gate. GRU and LSTM were found to perform similarly on a few tasks, such as polyphonic musical modeling, building voice signal models, and processing natural language. GRU has been proven to perform better on some smaller, less frequent datasets.



## 2.4 Previous Research

This paper [5] introduced a unique attention-based CNN model for assessing the sentiment of Bang-lish text and analyzing the performances of its variation, as well as the least aspects of NLP, making it platform-agnostic. The input layer of their proposed model is followed by hidden layers, convolutional layers, and max-pooling layers. These layers feed the attention layer of the proposed model. A contact and output layer comes after the RNN layer. They used ReLU as their model’s activation function and a convolution filter with a size of 3. They also used 256 hidden nodes to create the CNN layer. The output size of their RNN layer is 100, while the mini-batch size is 50. They used the optimizer using the Adam algorithm to modify the model’s parameters during training. A binary-labeled dataset and a multiclass-labeled dataset were used to divide the experimental data into two sections. All the characters were initially changed to lowercase during the data pre-processing. Then, they transformed the shorter words into complete words. Moreover, some of the most widely used and important words in their dataset were all spelled the same way. In comparison to multiclass classification, their model performed better with binary classification.

One of the most common daily necessities these days is internet marketing. The biggest problem is that consumers cannot choose a high-quality product by reading every online product review. Product reviews can help an e-commerce portal provide better services, but they take a lot of time and effort. The authors wanted to use Bangla, which is spoken by 228 million people worldwide, in their study [4]. However, Bengali users of online platforms appear to be more at ease using Phonetic Bangla and occasionally English. Because of this, they combined the English, Phonetics Bangla, and Bangla texts to produce a special dataset and removed things like stop words, punctuation, and irrelevant characters. The feature vectors were then extracted using the quick text model that had already been trained for the Bangla language. The training, verification, and test datasets for sentiment analysis contained 4085 (or 80 %), 715 (14 %), and 307 (or 6 %) cases. The training, validation, and test datasets for the Product Review Classification, on the other hand, contain 3574 (=70 %), 460 (=9 %), and 1073 (=21 %) occurrences, respectively. In their proposed model, there are hidden layers between neuronal layers. Additionally, they have chosen to activate Tanh, SoftMax, and Sigmoid and used the optimization model using the Adam algorithm, which has a 0.001 learning rate. The testing performance found for sentiment analysis is 0.84 and 0.69 for optimization algorithms.

Some people utilize shortened phrases to avoid having to type entire words. Any word or phrase that is used in a shortcut may be distorted. It has been noted that the sentence’s original spelling can occasionally be overlooked when using a shortcut. Bengali message-writing capabilities are not available on all mobile phones or laptops; these features must be installed individually. Fast Type, a sophisticated word prediction system, shines despite the usual drawbacks of conventional methods. CNNs’ approach has successfully translated all Banglish and Shortcut words into English. This method seems to be the best as it doesn’t require any feature extraction, according to this paper [9]. They first attempted to identify several word

representations before utilizing computer vision to translate shortcuts and Banglish words into the original word. To suggest the following word in a sentence is a hybrid model for word prediction issues. So they gathered 71 texting abbreviations for the AI project. They utilized 7–10 distinct photos to test the model after training with 10–15 different word representations. They primarily used convolutional neural networks in this case, labeling the shortcuts (CNNs) with the words from the folder name. They claimed that their suggested technique can only provide an accuracy of 65-70 % while processing time is very lengthy due to the possibility of numerous shortcut words being used in the main word. It will stop linguistic distortion for everyone. In addition, the technique is useful for those who enjoy communicating with abbreviations. Apart from that, the approach is useful for those who prefer to converse with abbreviations rather than full words. However, they mentioned that they will attempt to expand this dataset in the future and also include BEM with ScWd (shortcut word).

As users express their ideas, opinions, and sentiments on a variety of themes, microblogging websites are becoming excellent sources for posting massive amounts of customer content. The purpose of this paper is to discover the feelings or viewpoints expressed in Bangla microblog entries [10]. The majority of studies on categorizing the emotions expressed in microblog posts are conducted in English, while work in the Bangla language is primarily focused on news and blogs. For machine learning, they employed the maximum entropy (MaxEnt) and support vector machine (SVM). The tweets in this article’s dataset were all retrieved using the Twitter API v1.1 polling method. Tokenization, normalization, and Part of Speech Tagging were the three steps they used to do preprocessing. Using NLTK POS-Tagger, English tokens are tagged, whereas the Bangla Pos-Tagger Package handles POS Tagging for Bangla. Additionally, they bootstrapped the classifier using a dataset of 100 unlabeled tweets while preserving a 50:50 split between positive and negative tweets. Due to the duration of each tweet, Twitter users prefer to use emoticons more frequently than words (n-grams) to convey their feelings. Together, unigrams and stemming improve classifier accuracy by 1 % to 2 %. Although stemming shrinks the feature space, data sparsity is a problem. MaxEnt provides somewhat better accuracy than SVM for a few characteristics. In conclusion, they use a semi-supervised bootstrapping method to tackle the task of extracting sentiment from Bangla blog postings [9]. They want to tackle the issue in the future and see if adding a neutral class makes it possible to obtain even more precise findings. As they used unigrams with emoticons as features in their SVM algorithm, they were able to reach a pleasing accuracy of 93 %. From the findings, they draw the conclusion that the features of emoticons are significant in the training of classifiers for the binary classification problem.

Since the advent of the internet, it has become more and more common to convey thoughts on social networking sites in a number of languages. Users often combine terms from several languages to express themselves since they feel more at ease speaking in their regionalized language. The approach put forth in this study [11] is adaptable and reliable enough to accommodate additional identification languages as well as unusual foreign or superfluous words. Authors have made an effort to create a method for extracting sentiments from sentences that combines coded En-

glish with some Indian languages (Tamil, Telugu, Hindi, and Bengali). Processing of natural language methods for Indic languages is currently being developed. Using code-mixed words in English and four additional Indian languages, the authors of this research suggested a unique technique for identifying sentiments. The procedure has been separated into sentiment mining approaches and language identification due to its intricacy. After the text has been initially tokenized, the tokens are first categorized by a few classifiers that have been pre-trained with various n-grams. The tokenized and categorized words are then changed back to how they were original. The sentiment behind the term is then examined by the GetSentiment function. They assert that the proposed technique is flexible and robust enough to handle new identifying languages as well as unusual foreign languages.

A word embedding technique based on DNN was recommended in the paper to get around the problems with shallow representation [12]. For text classification, the authors have developed a hybrid model using BIGRU and CNN. This study primarily focused on RNN, attention mechanisms, and CNN to determine how well the suggested model worked. The model approach to the word embedding method represents a sentence in the given text. Through the use of BIGRU and the attention mechanism, the model enhanced the contextual semantics. Deep feature acquisition was followed by classification using the CNN model and SoftMax classifier. A train set and a test set are created from the dataset using a 9:1 ratio. For word vector training, they used the Word2Vec method, the ReLU function as the activation function, and a 0.001 learning rate. The research got a higher F1 value than a single CNN model, which will help in the future in-depth analysis of classification work.

The authors of this study [13] focused on the problems of fine-grained sentiment classification and multi-class text analysis, and they offered the hybrid bidirectional recurrent convolutional neural network attention-based model as a potential solution. Bidirectional long short-term memory, CNN with the attention method, and word2vec have all been utilized for text classification. Their hybrid method selects the useful local properties from the sequences generated by the Bi-LSTM in order to take into account both the context of the phrases and their long-term dependencies on one another. Authors have developed the bilinear attention function, which combines a variety of convolution filters to capture the local semantic characteristics of n-grams at various granularities. There are six layers in the architecture of an attention-based bidirectional recurrent convolutional neural network. In an earlier study, near 96 % accuracy in the Convolutional Neural Network (CNN) model was attained in an earlier study. The accuracy of the hybrid CNN-RNN model is similarly close to 95 %. However, the accuracy of the Hybrid Bidirectional Recurrent Convolutional Neural Network Attention-Based Model has exceeded all previously published results, coming in at close to 97 %.

In a study, authors utilized a variety of datasets, including tweets, reviews, and other online content [6]. The study focuses on providing an assessment of broad application models rather than solving an issue in a specific domain. In this study, the three models: CNN, LSTM, and SVM were combined and assessed. A hybrid CNN-LSTM model and a hybrid LSTM-CNN model were tested, and both models were coupled with the replacements of the classifier. The classifier is an SVM model. It

is clear from the result comparison section that the recommended combined LSTM-CNN model consistently generates outcomes with the highest accuracy. While other studies show an average accuracy of almost 84, the Cornell movie evaluations have an accuracy rate of 87 %. The accuracy rate for IMDb data is 93.4 %. The accuracy rate for book or music reviews is 91.1 %.

The authors of the study compared a hybrid deep learning strategy with a few conventional approaches [14]. A fastText word embedding package is included in the deep learning model. Reviews and blog entries have been used to gather the data for this investigation. FAIR Lab uses a Constant Bag Of Words to introduce this little text. The writers of the article created a unique script in Python to clean the data. The suggested approach first determines the data’s language, then converts it to English before shortening it into positive or negative. After the evolution, it is clear from the comparison section that this model’s accuracy level is higher than that of LSVM, fastText, or SAB-LSTM, which is close to 90 %.

The research [15] was carried out using several methods for extracting characteristics from textual data as well as text classification and classifying procedures. By assessing the sentiment of comments from Bangla newspapers, they employed a hybrid strategy with a pre-trained deep learning classifier and achieved 89.89 % accuracy. They integrated the optimizer function “Adam” and apply word embedding “Glove” in their hybrid model. Their dataset contains 13802 data which is primarily collected from the platform Kaggle. Their data mainly contains Bengali news text with 2951 Negative, 2280 Very Negative, 3198 Very Positive, 3928 Positive, 1445 Neutral, and 2951 Very Negative data. Due to the fact that there were many sorts of data, such as disorganized, missing data, incorrectly labeled, grammatically wrong, etc., they applied to preprocess techniques in the model to clean datasets. They first remove the punctuation, then cease word removal, then convert the numerical values to words, then integrate the data, and last tokenize the raw data to make the dataset well-formed. A bidirectional LSTM (also known as a BiLSTM) is a sequential processing technique that uses two LSTMs, one of which moves forward and the other backward. Even though CNN is great for long articles, it is still challenging to distinguish between its components. Following the embedding layer, a 32-filter convolution layer, a kernel shape of 3, two bidirectional layers with 0.5 dropout values each, and an “relu” activation feature were added. The steps in the procedure must fill in the “categorical cross entropy” error function around one another. The layer dropout for the “Adam” optimizer’s use of this model was 0.5. This model was trained using 512 batch sizes and 80 epochs. This model could be as long as 1000. This model also includes the FastText Model and also parameter setting to maximize the performance of this model. They want to use several hybrid models as well as neural network-based models in the future, including ANN, LSTM, RNN, and BERT for improving the model.

The study [16] was done with the procedures of convolutional neural networks with several layers and bidirectional gated recurrent units made into a hybrid neural network model. They looked at an LSTM variant called the GRU (Gate Recurrent Unit), which is naturally better at handling time series jobs and can quickly capture the characteristics of text context information. In this study, word2vec is used to

train huge Chinese news corpora. To represent text characteristics, Out of a sizable amount of text data, word vectors of Chinese words were chosen. Word2vec has automated compressed semantic data into mathematical vectors in comparison to conventional feature extraction machine learning approaches. They enhance the CNN + GRU classification model using the word vector text 141 representation technique by including more convolution layers, including the Attention Mechanism in the GRU layer, and horizontally combining the two parallel models. Because of MLCNN's local feature learning capabilities, BiGRU's dependence on the length, and the possibility of the Attention Mechanism to collect crucial information, we designed the MLCNN and BiGRU-ATT model employing these three components. According to the study of experimental findings, the hybrid network model performed well in the job of classifying news texts. They may then take what they've learned about applying text classification techniques to the blockchain. Using NLP technology, research is being done to expand blockchain research and improve text categorization by basing it on the characteristics of blockchain data.

In this research study [17], Researchers used a neural network model for the categorization of news content. They used a hybrid neural network consisting of MLCNN and BiGRU-ATT with attention mechanisms. GRU has an inherent advantage when computing time series jobs since it is well suited to capturing the features of text context data. To achieve encouraging results, they applied word segmentation and the Word2vec model as preprocessing and feature extraction. There are three components to the hybrid model. For the first section, They split the entire model into two parts. These generated text vector feature representations and created feature representations. The hybrid model was then used to obtain text vector feature representation. Finally, They used a fully connected layer and a softmax classifier to achieve the desired results. The experiment's data set was quite small and hence might have diverged from reality. There will be many data sets collected for experimentation in the next work, bringing research closer to its real applicability. Research is ongoing on the use of deep learning techniques and other ways to simplify and boost the accuracy of text categorization while at the same time reducing complexity.

Bidirectional Encoder Representations from Transformers, or BERT, was introduced in this study [18]. It is a deep learning model built on transformers. The pre-trained BERT model can be improved to produce cutting-edge models for a variety of NLP applications with just one additional output layer. The BERTBASE has 12 encoders and 12 bidirectional self-attention heads, while the BERTLARGE has 24 encoders and 16 bidirectional self-attention heads. They used the pre-training and fine-tuning processes in their methodology. BERT is already trained on a sizable corpus of unlabeled text, such as the Book Corpus and Wikipedia's 2,500 million words (800 million words). Two unsupervised tasks, such as Next Sentence Prediction and Mask LM, were employed in the pre-training (NSP). They connected the task-specific inputs and outputs to BERT during the fine-tuning and adjusted every parameter end-to-end. On eleven natural language processing tasks, this model achieves new state-of-the-art results, such as increasing the GLUE score to 80.5% (7.7 % absolute improvement), MultiNLI accuracy to 86.7 % (4.6 % ), SQuAD v1.1 question answering Test F1 to 93.2 % (1.5 % absolute improvement) , and SQuAD

v2.0 Test F1 to 83.1 % absolute improvement. They demonstrated the value of pre-training in both directions for language representations. Once more, the researchers showed that the need for numerous highly developed task-specific architectures can be reduced by using pre-trained representations. This effort aims to enable a single pre-trained model to successfully perform a number of NLP tasks.

The development of a sequence and transduction model is covered in the work [19], specifically a transformer created just using the attention process and avoiding recurrence and convolutions. Although pre-trained for a brief period, the study asserts that task construction with attention seems to lift that boundary and accomplish greater performance than before. The design consists of a six-layer encoder with two additional sublayers on each layer and a six-layer decoder with three more sublayers. The sub-layers employ both a straightforward attention mechanism and a multi-head self-attention mechanism. Each layer of those encoders and decoders also has a feed-forward network that is connected. The findings of the decoder were transformed into anticipated next-token probabilities once more by adding the learnt linear transformation and SoftMax function. The huge transformer model surpasses the competition on the WMT 2014 English to German-translation problem, earning a new BLEU score of 28.4. In conclusion, the Transformer, which was based solely on attention, substituted multi-headed self-attention for the repetitive layers. The model may be trained much more quickly and perform better than architectures based on recurrent or convolutional layers for translation tasks.

The paper [20] describes BanglaBert, a Transformer-based NLU model specifically for the Bangla language, along with introducing a term called “Embedding Barrier,” which refers to the difficulty of the difference among languages in terms of their vocabulary, writing structure, and even the script while implementing a model with any particular language. To start developing any language model, a collection of vast amounts of excellent text data is required. Here, for the BanglaBert, a dataset of about 18.6GB of pre-processed data was used, and it was pre-trained with the assistance of ELECTRA. Furthermore, fine-tuned using five downstream works as well as keeping check with two other multilingual models (mBERT and XLM-RoBERTa). The study also demonstrates that languages with embedded barriers perform less accurately than their sister languages, which share a lot of resources. On that notion, the research was done in a couple of languages, such as English, Bangla, Hindi, Nepali, and Swahili. English and Swahili shared the same script, whereas Bangla, Hindi, and Nepali shared similar scripts. Even so, the embedding barrier for Bangla was greater than all the other languages mentioned.

To evaluate various BERT fine-tuning techniques on a text classification task and to provide a general BERT fine-tuning solution, extensive experiments were carried out in this literature [21]. Three steps comprise their suggested fine-tuning process: the first step is to further pre-train BERT using inside training data, also known as in-domain data. The second alternative, if there are multiple related tasks available, is to optionally fine-tune BERT with multi-task learning. BERT must be adjusted for the intended task as the final step. Additionally, they examined fine-tuning strategies for BERT on target tasks and cutting-edge discoveries for a range of NLP activities, highlighting the great potential of the fine-tuning strategy. In

this study, the BERT text categorization fine-tuning method was further explored. Their datasets cover three text classification tasks that are performed: sentiment analysis, question classification, and topic classification. The batch size for the additional BERT pre-training was 32, the maximum sequence length was 128, the learning rate was  $5e-5$ , the train steps were 100,000, and the warm-up steps were 10,000. The batch size for fine-tuning was 24. They looked into various fine-tuning techniques in the first experiment. For the long text problem in the fine-tuning procedure, they once more applied the truncation method, which empirically picks the first 128 and the last 382 tokens of long texts. Additionally, they tested few-Shot learning, multitask fine-tuning, and additional pre-training. They presented five findings after the experiments. First off, text classification benefits more from the top layer of BERT. Second, BERT can solve the catastrophic forgetting problem with a proper layer-wise decreasing learning rate. Thirdly, additional pre-training within-task and within-domain can considerably improve its accuracy. Fourth, additional pre-training is better than earlier multi-task fine-tuning. The end result implies that BERT can enhance the work for small-size data.

# Chapter 3

## Formation & Deployment of Primary Dataset

### 3.1 Web Scrapping

The term "web scraping" refers to any technique used to gather information from the Global Internet. This information can be used for numerous purposes, such as collecting product reviews for analysis or market research. By using web scraping, users can quickly and easily get product reviews, pricing information, and other data from websites. Web scraping can save time and effort for researchers who need to collect data from large websites. The most common language used to code web scrapers is Python, since it has several libraries designed specifically for web scraping.

As we need to collect a large product review dataset, it is a very time consuming process to collect it from website to website. As a result, we make a web scraper for collecting reviews. Our web scraper can collect reviews from the daraz website. In our python code for web scraping, when we run our code a request is sent to the url that we have selected. Then the server sends the data and allows us to read the HTML page as a response to the request. Then our code parses the HTML page first and then finds all the related data and then extracts them. We can divide this task into 6 stages.

1. URL creation
2. Inspection of HTML page
3. In search of reviews to be collected
4. Coding phase
5. Deployment of code and collecting reviews
6. Data saving in a .CSV file

We used these following python libraries to implement our web scraping.

1. The Selenium library is utilized for testing websites. It has the capability of automating the browser operations.
2. BeautifulSoup is a package for the Python programming language that may be used to parse HTML texts and generate parse trees. These parse trees are beneficial while attempting to retrieve the data.
3. Requests is use for sending request

In our code, we can change our product category easily. There is a variable named



'searchKey'. If we change the variable, the product category will be changed. For example, if we set "sunglass" here all the sunglasses appear and our code will extract the reviews from those products. So, we can easily get different types of product reviews. It helped us to train our model more accurately. Also, we can select page numbers.

```
#####
page_start = 1
page_end = 2
#####
71 #####
72 # Put Your Search Key Here
73 searchKey = 'sunglass'
74 #####
```

Figure 3.1: our scraper is collecting reviews

After collecting the required data we get the scraped data as a .CSV file.

```
with open('results.csv', 'w', newline='', encoding='utf-8') as f:
    writer = csv.writer(f)
    writer.writerow(['Type', 'Reviews'])
    writer.writerows(reviewsList)
```

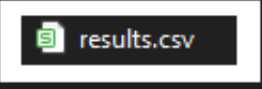


Figure 3.2: Data saved as CSV file

In this picture we can see a product's review on the daraz website. After running our web scraping code, it extracts all the reviews. Then it stores all the reviews in an excel file.

139	review	packaging khub valo chilo...sunglass tau ekdom fresh and same as picture...size o valo sob mile ami satisfied 😊 thanks to daraz and seller ❤️		
140	review	চশমাটির জন্য প্রচুর আগ্রহী ছিলাম, যেমন অর্ডার দিয়েছি তেমনই পেয়েছি daraz কে ধন্যবাদ		
141	review	product vlo but delivery onek late e paisi...pray 11 din por but product man somponno		
142	review	alhamdulillah jemonta ceyeci akdom same setai peyeci r kub taratari hate peyeci.apnarao nite paren potarito hoban na insah allah		
143	review	Rejected product is delivered. i am very upseted about this product, i cannot believe that 🤔🤔🤔.		
144	review	আমার ফোনের কেমেরা ভালো না তেমন, কিন্তু মাল যা দেখেছি তাই পেয়েছি, চশমাটা দামের তুলনায় অনেক সুন্দর। ধন্যবাদ দারাজ কে		
145	review	I got this sunglass in acurate time.. This product is perfect as i see in photos.. if anybody want to buy this sunglass, anybody can buy it without hesitation or problems..		
146	review	Ai sunglasses ar bepar re kisu volar nai ak kothai joss ☐ onk valo akta sunglasses 🍀		
147	review	product motamoti valo... packaging o valoi silo		
148	review	Alhamdulillah dam hishabe sunglass ta khub shundor. Same as picture and thanks for Daraz 🙏🙏🙏Apnara chile nite paren.		
149	review	আলহামদুলিল্লাহ,পারসেলটি হাতে পেলাম,আমার খুব ই পছন্দ হইছে ,,সেলার, ডেলিভারি বয় কে অসংখ ধন্যবাদ		
150	review	Thanks for sending me this authentic product Menspe Quality is very much premium. It tooks 23 days to receive. Thanks yo DARAZ.		

Figure 3.3: Result of the webscaper

It is our result.csv file. Here we can see that the scraper collects all the reviews and stores them in this results.csv file. One interesting thing is that the daraz website restricted our IP address for sending so many requests in a short period of time. To solve this issue, we used VPN to change our ip address. Then it worked perfectly.

## 3.2 Data Collection

A crucial component in the research methodology is the data-gathering procedure. The standardized process of collecting observational data or metrics is known as data collection. For our research purpose, we collected quantitative data, which is expressed in words and analyzed through categorizations and interpretations. We've used second-party data, referring to information shared by other organizations with their customer base, to build up our proposed model. We scraped customer reviews from the e-commerce website Daraz. The manual extraction of web data is frequently a laborious and time-consuming process. Web data extraction refers to a technique for obtaining information and data from websites using bots. Web scraping, contrasted to screen scraping, which only duplicates pixels, demonstrates on screen, extracts implicit HTML code, and, with it, records contained in a database. After that, the scraper can recreate the contents of the whole webpage somewhere else. For our research work, we have used online web scraper tools to extract relevant information. We've chosen product reviews from an e-commerce site, and using some selectors, the valuable information is pulled down.

Our extracted customer reviews had four different types. They are-

- Pure Bengali - ঘ্রাণ মোটামুটি । অতো ভালনা ।
- Pure English- The product is perfect in this price segment
- Bengali and English mixed -আলহামদুলিল্লাহ বেশ ভালোই, Everybody can buy this
- Bang-lish - product khub e valo lagse amar

As our research objective is to work with Bang-lish data, we excluded the first three types from our dataset and kept only the Bang-lish reviews.



### 3.3 Data Labeling

Data labeling is considered the most valuable step because it helps AI and machine learning algorithms build an accurate understanding of environments and conditions. The process of applying labels to actual data in order to assign context or interpretation to the information is known as data labeling. As we collect reviews from users as data, we need to add labels and prepare the dataset for sentiment analysis. The training dataset is then utilized to train ML models to predict, understand, or accumulate speech. Sometimes ML models lack confidence due to highly valued datasets, so humans need to add labels manually. This is what we did for our research work. Due to the high amount of data, built-in AI models can't give proper accuracy, so we opt for manual labeling. We have labeled positive reviews as 1(one) and kept 0(zero) for negative comments. After that, the labeled dataset was sent through the model to give improved feedback. Our dataset has 5000 Bang-lish reviews, of which 2500 are positive reviews and 2500 are negative reviews.



Figure 3.6: Visual Representation of the data variations

### 3.4 Data Pre-Processing

Real-world data usually includes noise, inconsistent data, emotions, punctuation, and may be in an inoperable layout that can't be utilized directly for predictive models. Data preprocessing is a major prerequisite for cleaning the data and preparing it for a classification model, which enhances the accuracy and performance of the trained model.

Human languages have different meanings but machines can't understand words. Hence words need to be converted into numbers. The performance of a model

depends on the process of data preprocessing and how well the dataset has been trained. For our dataset, we performed six stages in pre-processing our collected data.

### 3.4.1 Remove emojis

While writing in an informal way, people use emojis. As a first step, all emojis are removed from the dataset as it has no special meaning.

### 3.4.2 Remove punctuation marks

For better results, punctuation marks must be removed from words. There are 32 primary pieces of punctuation that should be acknowledged, which are '!""()\*+,-./:;<=>?@[|\`' . To substitute any punctuation in words with an empty set, we can use the string module in conjunction with a regular expression. Replacing any alphanumeric factor, we have eliminated all punctuation marks from our word list.

### 3.4.3 Tokenization

The method of splitting up a paragraph into small components such as phrases or words is known as tokenization. Each component is then treated as a separate token. Tokenization's basic aspect is to endeavor to comprehend the significance of the text by evaluating the individual units or tokens that comprise the paragraph. We have preprocessed the sentences into words. For example, for our dataset, there is one sentence 'product motamuti valo'. After tokenization, the outcome should be 'product', 'motamuti', 'valo', 'shundor'.

### 3.4.4 Lower casing

After tokenizing our review sentences, we converted all the characters in the words into lowercase. When capitalized, terms like "valo" and "Valo," which have the same meaning, are modeled in the vector space model as two distinct terms. As a consequence, the dimensions increase. Therefore, in our dataset, we kept all our words in lowercase to provide uniformity in words.

### 3.4.5 Stop word removal

Stop words are frequently used in documents. These types of words don't signify anything and sometimes result in the deep learning model underfitting or overfitting while training. For example, some of the Bang-lish stop words are e, er, ki, theke, etc. We prepared a stop word list of 480 Bang-lish stop words. We used the Bengali stop word list from a GitHub repository [22] and converted the Bengali words into Bang-lish form. For example, Bengali word বরং translated into borong in Bang-lish. Besides, we added some words which are considered stop words in Bang-lish text. In Bang-lish form, people write single-letter words because of typos or for using suffixes. For example, পণ্যটি can be written as ponno t. This t is also functioning as a stop word. We also added some of these words to our stop word list.

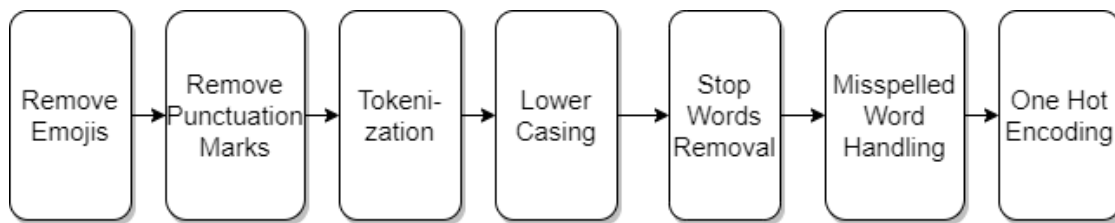


Figure 3.7: Data Pre-Processing

### 3.4.6 Misspelled word handling

As Bang-lish is not a real language, it doesn't have any proper rules to write words and sentences. For example, the Bang-lish word valo can be written as valo, bhalo, vhalo. For some of our dataset's most frequently occurring terms, we must spell them consistently. In order to manually determine the uniform form of these terms, we produced a list of the most commonly used and significant words in our dataset. After that, we developed an algorithm that would be able to find the misspellings of those frequent words and correct them. In the algorithm, we used the phonetics-based algorithm, Double Metaphone, and Levenshtein distance.

- **Levenshtein Distance:** Levenshtein distance basically is a technique to find out the contrast among vocables. More specifically, it is the count of one alphabetical character alteration between two terms at a time. The alterations can include adding, deleting, or substituting characters. This distance is occasionally known as edit distance as well.

Mathematically, two randomly given strings  $a$ ,  $b$  would have the Levenshtein distance,  $lev(a,b)$ , where  $|a|$  and  $|b|$  is the length of the string.

$$lev(a,b) = \begin{cases} |a| & \text{if } |b| = 0, \\ |b| & \text{if } |a| = 0, \\ lev(\text{tail}(a),\text{tail}(b)) & \text{if } a[0] = b[0], \\ & \text{otherwise,} \\ 1 + \min \begin{cases} lev(\text{tail}(a), b) \\ lev(\text{tail}(b), a) \\ lev(\text{tail}(a),\text{tail}(b)) \end{cases} \end{cases}$$

We used the python library `fuzzywuzzy` which implements Levenshtein distance between two words.

- To put simply, Metaphone essentially is a phonological algorithm that outperforms Soundex in aspects of matching phrases that sound similar by incorporating data about variances and discrepancies in both pronunciations along with spelling in order to generate more precise encoding.
- Double Metaphone would be the title of a subsequent edition of the genuine algorithm. This technique takes into account the spelling quirks of several additional languages, as compared to the previous technique, which exclusively

applies to English. Therefore, we sought to incorporate this method to amend the Bang-lish misspelled terms.

The python library fuzzy is used in our algorithm for implementing double Metaphone.

The pseudocode of our algorithm is given below:

```

import fuzzy
from fuzzywuzzy import fuzz
set updated_corpus to []
dphone ← fuzzy.DMetaphone(4)
mw ← pd.read_csv ('most_frequent_words.csv')
mw ← mw["Words"].tolist()
for all i in range corpus: do
  cor ← i.split()
  for all j in range mw: do
    set d_1 to dphone(j)
    for all k in range(len(cor): do
      set d_2 to dphone(cor[k])
      if j is not equal to cor[k]
      and fuzz.tokenratio(d_1[0], d_2[0]) is greater than 74
      and fuzz.token_sort_ratio(j,cor[k])is greater than 74 then
        set cor[k] to j
      end if
    end for
  end for
  set g to " ".join(cor)
  do updated_corpus.append(g)
end for

```

Our algorithm has detected and corrected 3721 misspelled words in our dataset. Table 3.1 provides several specifics with examples:

Misspelled Word	Correct Word
jinis	jinish
sundor	shundor
nosto	noshto
onk	onek
dalivary	delivery
sae	same
vlo	valo
asol	ashol
cilo	chilo
balo	valo
aktu	ektu

Table 3.1: Example of Handling Misspelled Words

### 3.4.7 One hot encoding

Categorical variables are transformed via one-hot encoding into a format that may be used as input by machine learning algorithms. Many machine learning projects require this pre-processing phase, which is crucial. Transforming content from a categorical form to a numerical representation is the aim of one-hot encoding. The ordinal encoding of the category variables is followed by the representation of each integer value as a binary vector with all the other values of the vector being zero except for the location of the integers, which is labeled with a 1. To put it another way, this is a technique used to transform a category feature with string labeling into K nominal attributes such that the result of one of the K (one-of-K) characteristics is 1 and the value of the remainder (K-1) features is 0. Since the features produced as a result of these approaches are dummy features that don't reflect any real-world features, it is also known as dummy encoding. Instead, they were developed to use fake numerical characteristics to encode the various values of category features. Consider a dataset that has details about several fruit varieties. With one-hot encoding, each variety of fruit would have its own column, with a value of 1 signifying that the rows include information about that product and a value of 0 signifying that it doesn't.

Python's Pandas package could be used to achieve one-hot encoding. Data can be one-hot encoded using the "get dummies" function offered by the Pandas package. There are further methods available, such as using the OneHotEncoder class in the sklearn.preprocessing package for one-hot encoding. Dummy data is another name for one-hot encoded data. Because dummy data does not require specific processing of categorical variables, it makes machine learning models easier to utilize.

In order for machine learning libraries to use the values to train a model, the categorical features must be transformed or converted into numerical features using the one-hot encoding approach. It is advised to explicitly transform these category data into numerical features even though many machine learning libraries do it internally (dummy features).

The following are some drawbacks of one-hot encoding:

One-hot encoding performs well for smaller cardinality. One-hot encoding, for instance, will result in a matrix with the values [1,0,0,0,0,0] for Sunday if we want to express categorical variables like week (7 days) in this way. The feature must be converted to one-hot encoding for greater cardinality cases, such as expressing millions of customer ids, because doing so results in a sparse matrix that is unsuitable for many machine learning techniques. One-hot encoding also regards the categorical data as independent, which is a drawback. If there is a link between categorical values, one-hot encoding cannot capture that relationship.

OneHotEncoder may be used to do one-hot encoding for a single-category feature. The transformation of a categorical feature with two values, like gender, is shown in the next code sample. The gender value is translated into two feature columns when OneHotEncoder is used with an empty constructor function. One of the fake features is dropped when OneHotEncoder is constructed with drop='first'. This is



so that all 0 values in the remaining features indicate the dummy feature that was eliminated. Multi-collinearity, a problem for some approaches, is avoided with this technique (for instance, methods that needs matrix inversion).

We converted our corpus into one hot representation. We set the max-length to 131, and the padding value set to "pre". Some examples of our converted text data into one hot representation are given below:

valo service pacchi	→	[ 0, 0, 0, ..., 2297, 966, 1211]
oshadharon valo maaner jinish	→	[ 0, 0, 0, ..., 4210, 6252, 1584]
shundor na	→	[ 0, 0, 0,..., 1051, 6920, 1584]
quality mutamuti dam	→	[ 0, 0, 0, ..., 8164, 10598, 7033]
altu bayadop osobvo nillojjo	→	[ 0, 0, 0, ..., 8963, 11792, 12763]
bag er man baje	→	[ 0, 0, 0, ..., 0, 10845, 3099]

Table 3.2: One Hot Encoding

In-depth discussions of the data pre-processing, models, our algorithm for handling misspelled words, and the suggested model's structure are provided in this section. Figure illustrates the overall workflow of our work.

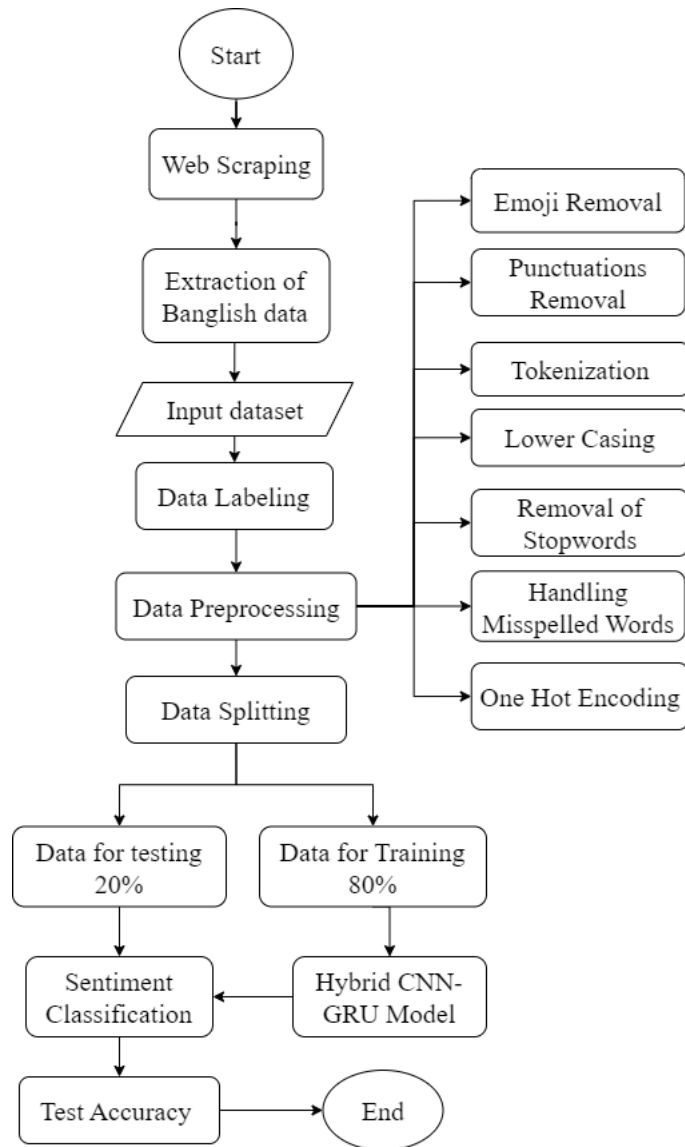


Figure 3.8: Workflow of our dataset

# Chapter 4

## Analysis of Sentiment Classification Models

### 4.1 Sentiment Analysis Models

#### 4.1.1 Neural Networks (NN)

Neural Networks form the base of deep learning (DL), a subfield of Machine learning (ML) where the algorithms are inspired by the structure of the human brain. Neural networks basically works by training with data and recognizing the patterns and then predicts the outputs of a new set of data.

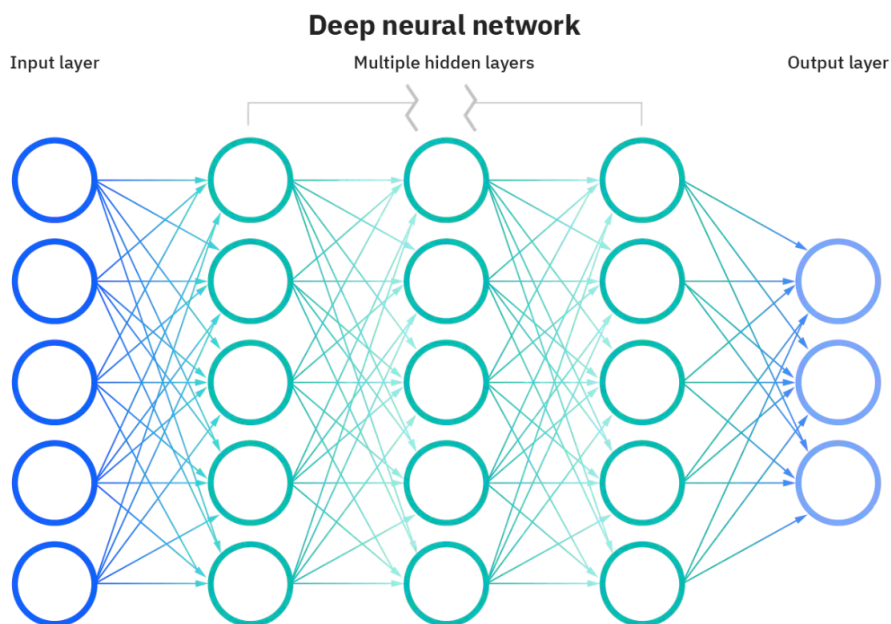


Figure 4.1: The basic structure of a Neural Network

An input layer, an output layer, and a number of hidden layers make up the fundamental building blocks of a neural network. The hidden layers carry out the

necessary calculations needed to achieve the output after the input layer receives the inputs and the output layer predicts the outcome. The primary processing components of the network, or neurons, are present in each layer. Each of the channels that connect these neurons from one layer to another is given a weight, which is a numerical value. The inputs are then multiplied by the associated weights, and the sum is subsequently supplied as input to the hidden layer. Now, in that layer, every single one of those neurons is once again connected to a numerical value known as the bias. Now, the activation function, a threshold function, is applied to this total. A specific neuron's activation status is determined by the activation function. In this way, the neurons transmit the info all the way to the output layer. This method of transmission is known as forward propagation. The neuron with the greatest value is finally selected as the projected value in the output layer. However, since these are merely probabilities, there will inevitably be mistakes. This is where the training phase enters the picture. The errors are compared to the real outputs, and the information is then sent back to the network. The weights and bias are then modified in accordance with this. The weights are refined until they are as accurate as feasible through an iterative cycle of forward and back propagation. Hours, days, or even months could pass throughout this procedure [23].

### **4.1.2 Neural Language Model (NLM)**

Neural language models (NLM) are, in essence, natural language processing (NLP) operations that are accomplished by applying neural networks (NN). Neural language models have an advantage over n-grams due to the fact that NLM is more generalized when it comes to similar settings and can handle longer records. This gives NLM a competitive advantage. In addition, although both NLM and n-gram models are trained with identical datasets, NLM delivers a greater level of predicted accuracy. As a result, contemporary NLP increasingly relies on neural language models in order to improve performance. However, it uses numerical representations of words as input and outputs potential future words via probability distribution, albeit the working process is not wholly different from n-gram models. The distinction resides in the input formats; neural language model utilizes different word embedding approaches (e.g., pre-trained word2vec embeddings) whereas employing the n-gram itself. Hence, NLM is better at generalizing unseen data as they have a good grasp on the semantics behind the text [24].

### **4.1.3 Convolutional Neural Network (CNN)**

A convolutional neural network, more commonly known as a CNN, is essentially an artificial neural network that has been carefully trained to recognize or detect patterns. As a result of this pattern detection, CNNs are extremely useful tools for conducting picture analysis. The use of convolutional layers, which are CNNs' equivalent of hidden layers, is what sets CNNs apart from other types of neural networks. Convolutional neural networks (CNNs) may, and frequently do, include additional layers that aren't part of the convolutional process, but the convolutional layers make up its core. A convolutional layer is one that first receives data, then modifies that data in some way, and finally sends the modified data to the layer that comes after it. Input channels and output channels are the terms that are used

to refer to the inputs and outputs of convolutional layers, respectively.

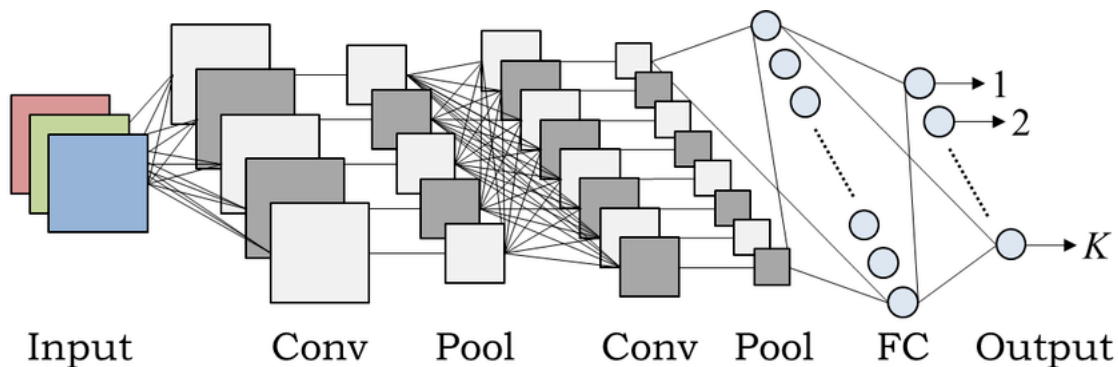


Figure 4.2: The common layer of a Convolutional Neural Network

A convolution operation is the name given to the change that takes place when working with a convolutional layer. In any event, this is how those who work in the field of deep learning refer to it. Convolutional layers are responsible for performing an operation known in mathematics as cross-correlations. This operation is a type of convolution. Convolutional neural networks are a technique that can be used to recognize patterns in photographs. It is necessary for us to specify the number of filters that should be present in each convolutional layer. These filters do an accurate job of discovering the patterns.

#### 4.1.4 Recurrent Neural Network (RNN)

Time series data and artificial neural networks with time series data are known as Recurrent Neural Networks (RNN). These deep learning algorithms are built into popular programs like Siri, Voice Search, and Google Translate. They are often used for order or time issues such as: B. Speech translation, natural language processing (nlp), speech recognition, image annotation. Recurrent neural networks (RNNs), like feedforward and convolutional neural networks (CNNs), use training data for learning. They are notable for their "remembering" which allows data from previous inputs to be used to influence current inputs and outputs. The output of iterative neural networks depends on the previous part of the sequence, unlike typical deep neural networks, which assume that the inputs and outputs are independent of each other. Unidirectional recurrent neural networks are effective in determining the output of a given sequence, but cannot take future events into account in their predictions.

Recurrent networks are characterized by sharing parameters between all layers. Iterative neural networks share a single weight parameter across all network layers, in contrast to feedforward networks, which assign individual weights to each node. Nevertheless, these weights are modified by backpropagation and gradient descent processes to support reinforcement learning.

Recurrent neural networks use the backpropagation through time (BPTT) technique to compute gradients. This technique is designed to be used with sequence data, so it differs slightly from regular backpropagation. Like traditional backpropagation, where the model is trained by computing the error from the input layer to the output layer, BPTT works on the same basic concept. With the help of these calculations, the parameters of the model can be changed and tuned appropriately. In contrast to traditional methods, BPTT adds an error at each time step. In contrast, feedforward networks do not require error summation because the parameters are not shared by all layers.

Exploding gradients and vanishing gradients are two problems RNNs commonly encounter in this procedure. The magnitude of the gradient, which represents the gradient of the loss function along the error curve, characterizes these problems. If the gradient is too small, it continues to get smaller, updating the weight parameter until it becomes negligible, or 0 if the gradient is too small. After that, the algorithm stops learning. Gradient explosion, which occurs when the gradient is too large, creates an unstable model. In this scenario, the model weights eventually become inexpressible (NaN) due to overgrowth. One way he deals with these problems is to reduce the number of hidden layers in neural networks to remove the complexity of RNN models.

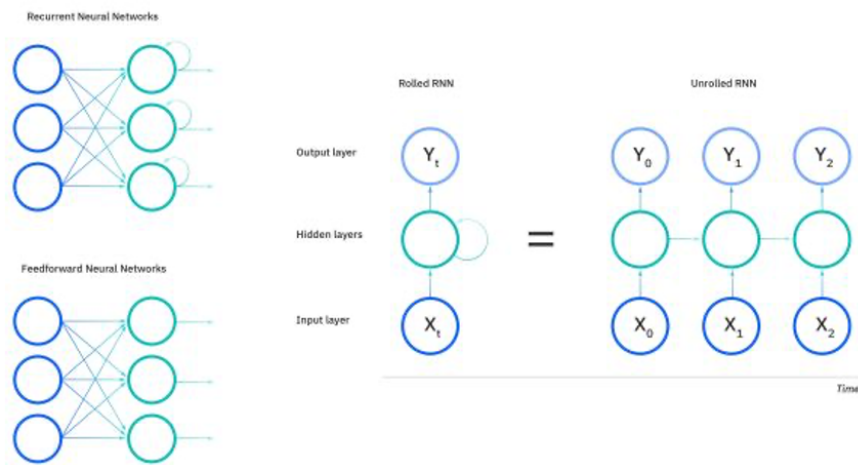


Figure 4.3: Pathflow of RNN

Iterative neural networks don't actually have this limitation, but the diagram above shows it that way. Input data are mapped to output in a feedforward network.. Instead, multiple types of RNNs are used for different use cases, such as music creation, sentiment classification, machine translation, etc., and their input and output lengths can vary.

The diagram above depicts a recurrent neural network in this way, but in reality it is not restricted like a feedforward network, which maps inputs to outputs. Instead, multiple types of RNNs are deployed for different use cases. B. Music production, sentiment classification, machine translation, and their inputs and outputs vary in length.

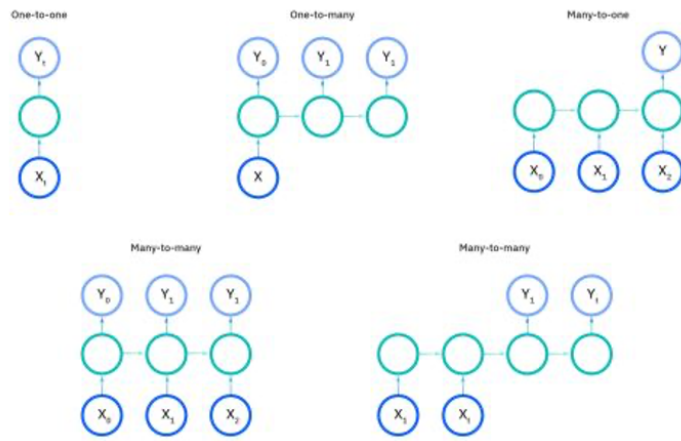


Figure 4.4: Different type of RNN

As explained on the Neural Networks tutorial page, a neuron's activation function determines whether the neuron is turned on. In most cases, nonlinear functions transform the output of a particular neuron into a number between 0 and 1 or -1 and 1.

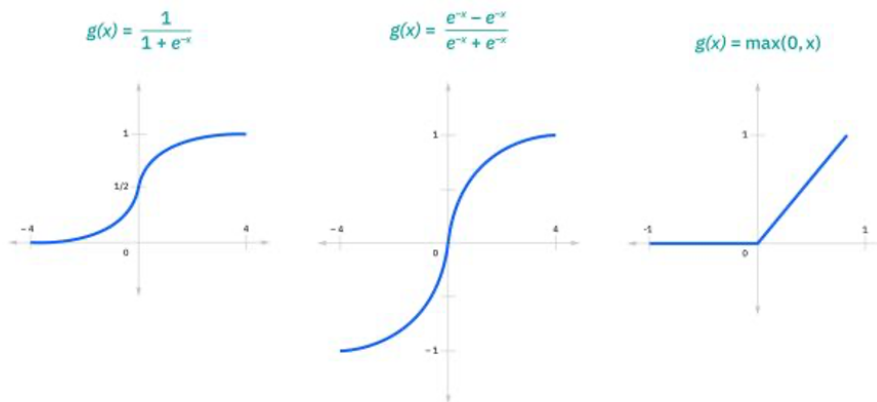


Figure 4.5: Common activation function of RNN

#### 4.1.5 Long Short-Term Memory (LSTM)

A common type of RNN (recurrent neural network) for learning sequential data prediction problems is the LSTM (long short-term memory) network. Like other neural networks, LSTMs contain several layers that support pattern recognition and learning to improve performance. The basic function of LSTM can be seen as keeping the data you need and discarding the data that is needed but not useful for future predictions. There are many different types of LSTM networks, but they

can be broadly categorized into three groups. Bi-LSTM, Bi-LSTM, LSTM forward pass, LSTM reverse pass.

As the names imply, the forward pass and backward pass LSTM are unidirectional LSTM, processing information in just one direction, either on the front side or on the backside, as opposed to bidirectional LSTM, which processes information on both sides to sustain it. Each of the LSTM kinds listed above operates according to a fundamental framework. Of a fundamental LSTM model architecture, several parts may be found.

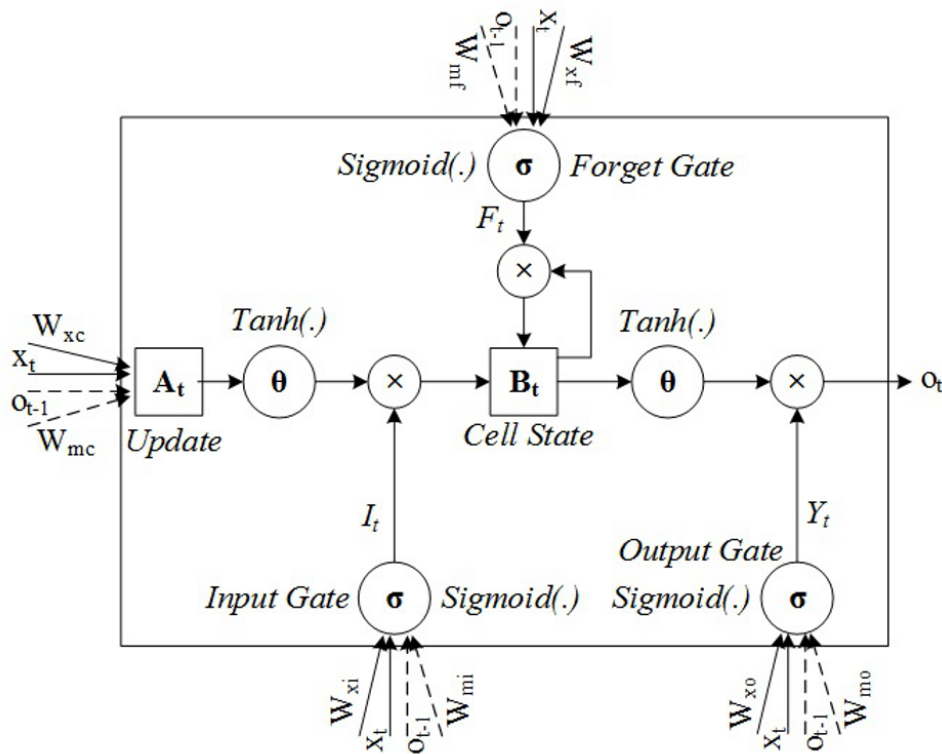


Figure 4.6: Architecture of LSTM

One of the most notable characteristics of LSTMs is their capacity to recall and recognize incoming information and discard that which is unnecessary for the network to learn data and generate predictions. This LSTM property is caused by this gate. It helps determine whether data can pass through the layers of the network. Information from the previous layer and information from the presentation layer are two different forms of input expected from the network.

The Forget gate circuit, where  $h$  and  $x$  are information, is depicted in the above picture. When this data is processed using the sigmoid function, any information with a propensity to zero is removed from the network.

By altering the cell state, the input gate assists in determining the significance of the information. Whereas the forget gate aids in the removal of data from the network, the input gate determines the significance of the data and aids the forget function in the removal of the data that is not significant, allowing subsequent layers to learn the data that is crucial for creating predictions. The sigmoid and tanh functions process the input, with the former determining the information's weight and the



latter lowering the network's bias.

Information about weight gain passes through the cell state calculated by this layer. The input gate and forget gate outputs are multiplied by the cell state. Information that can be lost is amplified by values very close to zero. Input and output values are added here to the cell status to update the cell status with network related information.

The last gate of the circuit helps determine the subsequent hidden state of the network through which the data is transmitted via the sigmoid function. The cell updated from the cell state is sent through the tanh function and multiplied by the initial state sigmoid function. Facilitates the transmission of information in a confidential manner. The hidden state can now choose which information to transfer thanks to the final stage of the circuit.

When executing standard text modeling, the majority of the preprocessing and modeling tasks concentrate on producing data in a sequential manner. Examples of similar tasks include POS tagging, stopword deletion, and text sequencing. These techniques aim to simplify the understanding of data by a model in accordance with the established pattern. The outcomes are possible.

Applying LSTM networks in this situation may offer a unique advantage. We explained how LSTM includes a feature that allows it to memorize the order of the data earlier in the essay. The dataset always contains a lot of unnecessary information, which may be removed by the LSTM in order to shorten computation times and lower costs. This feature is another one that this program possesses. The LSTM is a great tool for text categorization and other text-based tasks mainly because of the capability of eliminating irrelevant information and memorizing the order of the information.

The very first layer of the modeling process is the embedding layer, which employs a certain length vector, and the second layer is the LSTM layer, which contains neurons that will serve as the model's memory unit. Following LSTM, the dense layer—an output layer with sigmoid function—helps in supplying the labels. There are some modifications we can make to the model or the data that will assist improve the accuracy of our work. With sequential data, such as time series data and audio data, LSTM is a frequently utilized network. We may use LSTM to do a variety of tasks in the time series analysis field.

#### **4.1.6 Gated Recurrent Unit (GRU)**

Gated Recurrent Unit is a modification of the RNN hidden layer that makes it much better for long range connections and helps a lot when it comes to vanishing gradient troubles. Gated Recurrent Unit also helps a lot when it comes to solving problems with vanishing gradients. When it comes to issues with fading gradients, Gated Recurrent Unit is also of great assistance and helps a great deal. The following is a more in-depth description of the operational mechanism of the GRU: Calculate the

parameterized current input and previously hidden state vectors for each gate by conducting element-wise multiplication (Hadamard Product) between the relevant vector and the associated weights for each gate. This will yield the parameterized current input and state vectors. The parameterized current input and state vectors are what you get as a result of this step.

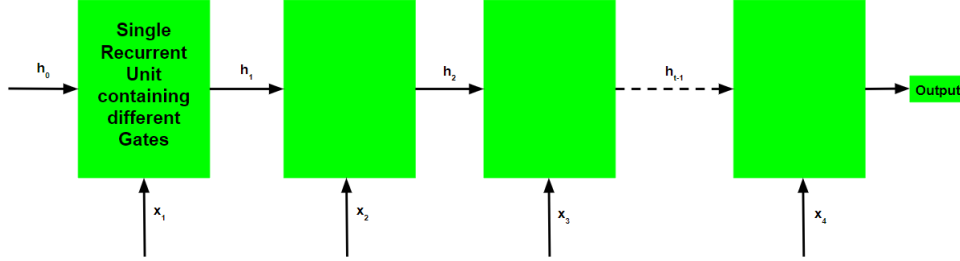


Figure 4.7: Components of Gated Recurrent Unit (GRU)

Apply the proper activation function for each gate by working your way through the parameterized vectors element by element. Further down below, you will see a list of gates, along with the activation function that should be utilized for each individual gate [25].

## 4.2 Proposed Model

### 4.2.1 Embedding Layer

Before feeding into the deep learning model, the dataset must be converted into numbers from textual form. Each word is transformed into a fixed-length, predetermined-size vector by the embedding layer.

### 4.2.2 CNN Layers

#### Convolution Layer

The convolution layer performs the convolution procedure on the input provided by the embedding layer. Our proposed hybrid model has two convolution layers.

$$z^l = h^{(l-1)} * W^l \quad (4.1)$$

The final component of the convolutional layer that increases the output's nonlinearity is an activation function and we utilized ReLU (Rectified Linear Unit) as activation function in the convolution layers.

$$ReLU(z_i) = \max(0, z_i) \quad (4.2)$$

## Max-Pooling Layer

Our suggested model includes a max-pooling layer after each convolution layer. Convolution layer outputs are subjected to the max-pooling process after each convolution layer.

$$h_{xy}^l = \max_{i=0,\dots,sj=0,\dots} h_{(x+i)(y+j)}^{(l-1)} \quad (4.3)$$

## 4.2.3 Drop out Layer

Drop out is a technique where during training, neurons chosen at random are not used. The dropout layer does the dropout task. We added a dropout layer after the CNN layers.

## 4.2.4 Gated Recurrent Unit (GRU) Layer

In order to interpret the lengthy text, our suggested model includes a GRU layer, a form of RNN that resolves the vanishing gradient problem. The dropout layer's output was passed to the GRU layer.

## 4.2.5 Flatten Layer

The flatten layer converts a multidimensional array into a one-dimensional flattened array. We used a flatten layer upon the outputs of the GRU layer.

## 4.2.6 Output Layer

The output layer will perform the sentiment classification task by utilizing sigmoid as the activation function and binary cross entropy as the loss function.

### Sigmoid function

As we are predicting probability and output of sigmoid function exists in between 0 to 1. Equation of Sigmoid Function:

$$\sigma(z_i) = \frac{1}{1 + e^{-z_i}} \quad (4.4)$$

### Binary Cross Entropy

In binary classification, only two true values of y are feasible: 0 or 1. To evaluate loss among real and expected values, it is necessary to compare the actual value (0 or 1). Equation of Binary Cross Entropy:

$$Loss = \frac{1}{n} \sum_{i=1}^N -(y_i * \log p_i) + (1 - y_i) * \log(1 - p_i) \quad (4.5)$$

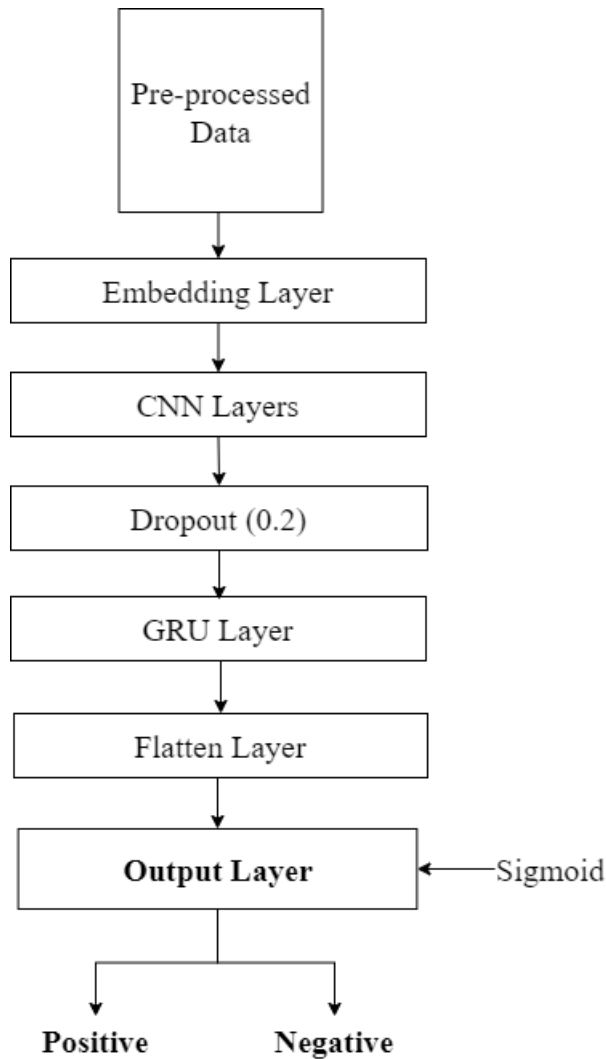


Figure 4.8: The basic architecture of the suggested model

### 4.2.7 Hyper-parameters

The model hyper-parameters proposed are as follows:

#### Embedding vector size:

The number of dimensions in the vector space into which the word is embedded is the size of the embedding vector. Additionally, several dimensions are tangentially connected to other, separate "concepts" that a word may be classified into. The precise number of embedding dimensions often has little impact on job performance. The number of dimensions may have an impact on training time. Choosing a power of two is a frequent heuristic for cutting training time. When moving data, powers of 2 will boost cache use, hence minimizing bottlenecks. Depending on which order of magnitude is selected. Our embedding vector size is 60.

**Kernel size:**

The phrase "step size" describes how far the filter slides, whereas the term "kernel size" describes the size of the filter that revolves around the feature map. In this case, maintaining a kernel size of 3x3 or 5x5 is common. The feature recovery would be local and fine-grained, without knowledge of the neighboring pixels. Therefore, 1 x 1 is taken off the list of possible optimal filter sizes. The kernel size for the convolution layers was set to 3 and 24 respectively.

**Filter size:**

The filter size in a CNN may be thought of as the n-gram size in natural language processing. A group of n-pieces from a particular text sample make up an n-gram. The application may specify that these components be words, letters, or pairings. The N-grams are often gathered from a corpus of text or voice. In the convolution layers, the filter size was set to 64 and 126 respectively.

**Padding:**

Convolutional neural networks (CNNs) benefit from padding, which is defined as the number of pixels added to an image before it is processed by the CNN kernel. If the padding in a CNN is zero, for instance, all excess pixels will be assigned the number 0. For our model, we kept our padding the same.

**Max-pooling:**

The maximum pooling method, sometimes referred to as "max pooling," establishes the highest or maximum value in each block of each feature map. The outputs are feature maps that have been pooled or down-sampled to highlight the feature that is most prevalent in the area. The pooling size is 2 in both of the max-pooling layers.

**Activation function:**

When designing a neural network, one of the decisions that must be made is the activation function to utilize in the network's hidden and output layers. To make a decision whether or not it will activate a neuron, the activation function makes a weighted sum and adds a bias to it. The activation function was developed to give a neuron's output non-linearity. ReLU was used as the activation function in both of the convolution layers.

**Dropout:**

To some extent, the dropout layer is considered as a mask, allowing all other neurons to continue functioning while reducing the input to the layer below. Dropout improves a machine learning model's performance because it reduces overfitting by making the network simpler. The neural networks' neurons are taken out during training. The dropout rate of 0.2 was used in the dropout layer.

**Units:**

A gated recurrent unit (GRU) is a portion of a specific kind of recurrent neural network that is created to make use of connections generated across a number of nodes to perform memory- and clustering-related machine learning tasks, such as speech recognition. Gated recurrent units aid in modifying neural network input weights in order to solve the vanishing gradient problem, which is a common issue with recurrent neural networks. The term "minimal gated unit" refers to a condensed version of the complete gated unit that uses gating performed using the prior concealed state and the bias in different combinations. The units size was set to 20 in the GRU layer.

# Chapter 5

## Result Analysis and Evaluation

We have covered the assessment of the trained model's efficiency in this chapter. We have established some key criteria for our trained model's evaluation: accuracy, F1 score, precision, and recall. To visualize the results of classification, we employed confusion matrix. The side-by-side analysis of the seven model versions has also been covered.

### 5.1 Result Assessment

#### 5.1.1 Confusion Matrix

The confusion matrix summarizes the prediction result with the count values of correct and incorrect predictions. It is the breakdown that gets over the drawback of relying solely on categorization accuracy. True Positive ( $T_P$ ), True Negative ( $T_N$ ), False Positive ( $F_P$ ), and False Negative ( $F_N$ ) are the four metrics in the confusion matrix.

		PREDICTED	
		Positive	Negative
ACTUAL	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Figure 5.1: Metric of Confusion Matrix

We have obtained True Positive = 426, True Negative = 464, False Positive = 50, and False Negative = 60 in our confusion matrix shown in figure 5.2. As True Positive, True Negative values are high, which means our model is able to correctly classify datas, that's the reason our model gives higher accuracy.

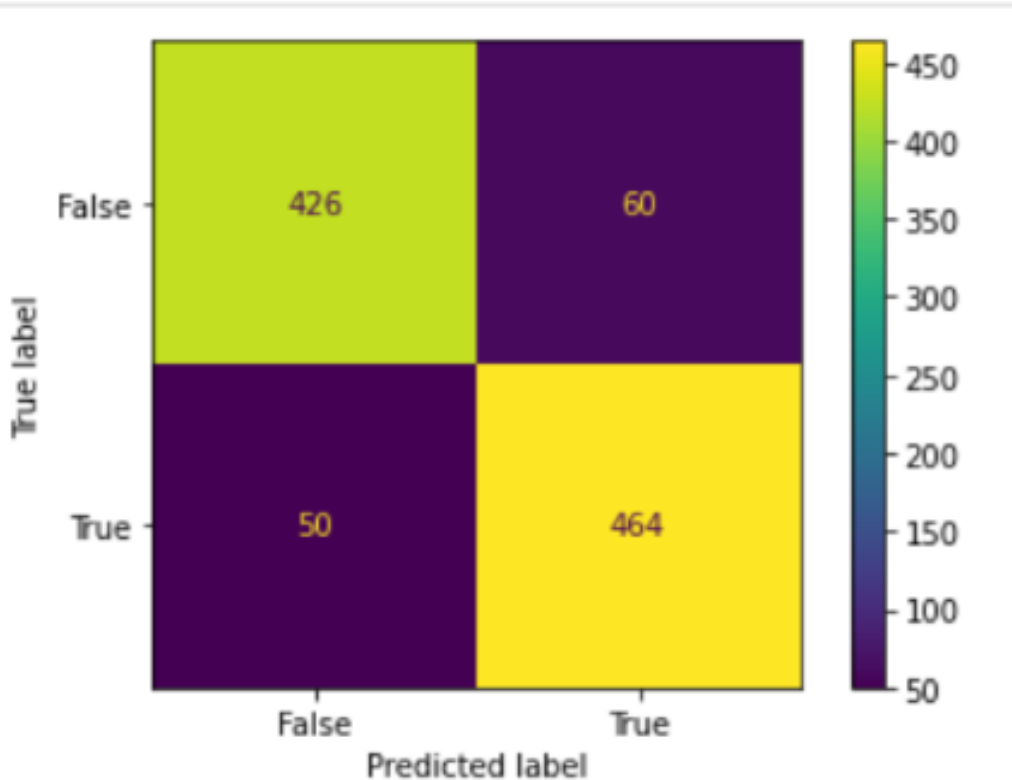


Figure 5.2: Confusion Matrix

### 5.1.2 Recall

The accuracy of a classifier is quantified by its recall, which is a measure of its comprehensiveness. When recall is high, there are fewer false negatives, and when recall is low, there are more false negatives. Increasing recall can sometimes lead to less precision because it gets harder to be accurate as the data point grows.

$$\mathbf{recall} = \frac{T_P}{T_P + F_N} \quad (5.1)$$

In our Banglish text classification, our model has obtained 89% ability to classify text.

### 5.1.3 Precision

Precision is determined by the proportion of the amount of samples correctly labeled as positive to the entire sample labeled as positive (either correctly or incorrectly). Precision is a measure of how well the model can tell if a sample is positive or not. Greater precision means fewer false positives, while less precision results in more of these erroneous readings.

$$\mathbf{precision} = \frac{T_P}{T_P + F_P} \quad (5.2)$$

From the table 5.1, it can be said that our model has gained 88% precision in terms of Banglish text classification.



### 5.1.4 F1 Score

The F1 Score represents the measurement of the optimal compromise between the classification model's precision and its ability to correctly identify occurrences of value. Our model achieved 89% F1 Score in terms of classify Banglish text.

$$\mathbf{F1\ Score} = 2 \times \frac{\textit{precision} \times \textit{recall}}{\textit{precision} + \textit{recall}} \quad (5.3)$$

### 5.1.5 Accuracy

Accuracy leads to the measurement of the proper classification of positive and negative occurrences. Our model gained 89% accuracy ability to classify Banglish text.

$$\mathbf{Accuracy} = \frac{T_P + T_N}{T_P + T_N + F_P + F_N} \times 100 \quad (5.4)$$

The following graph illustrates the test and train accuracy of our model.

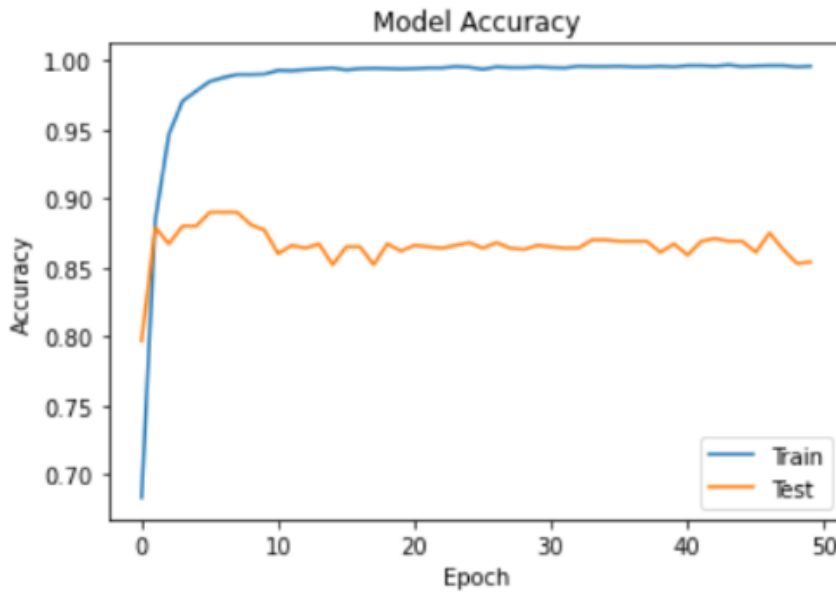


Figure 5.3: Accuracy plot

Parameter	Value
Accuracy	89%
Precision	88%
Recall	89%
F1 score	89%

Table 5.1: Performance summary of proposed model

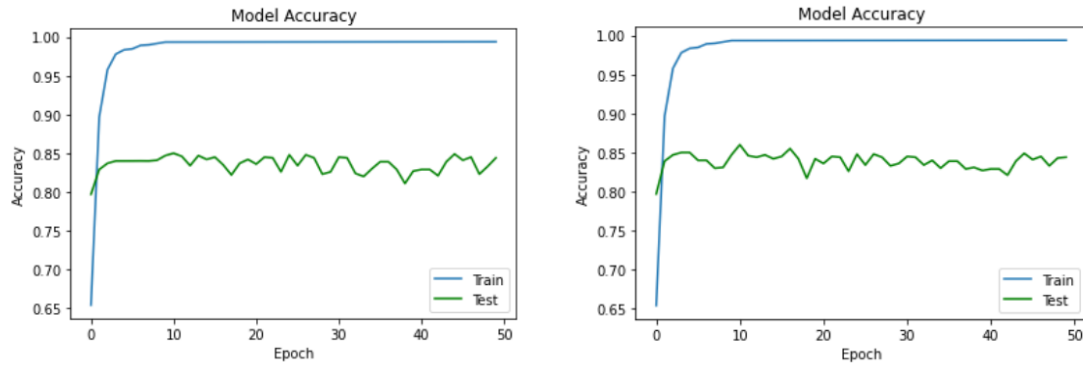


Figure 5.4: accuracy plot of LSTM(left) and CNN(right)

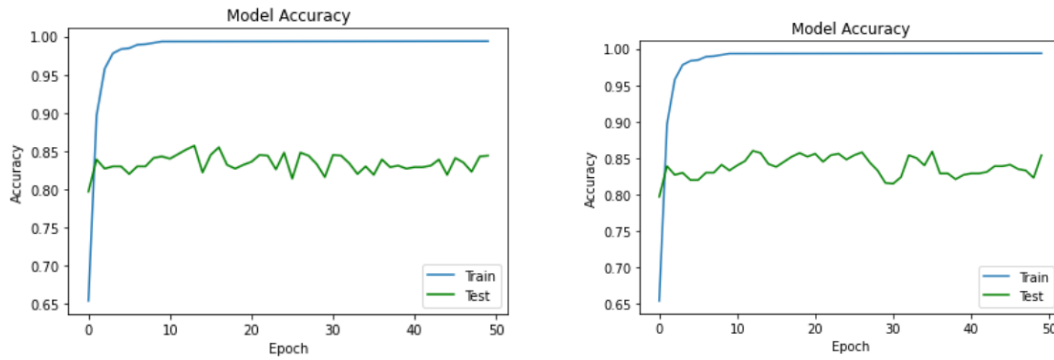


Figure 5.5: accuracy plot of RNN(left) and GRU(right)

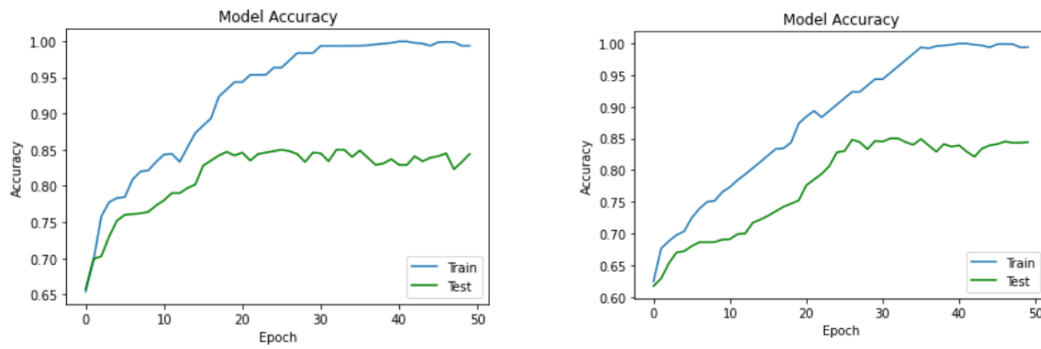


Figure 5.6: accuracy plot of BiDirectional LSTM(left) and BiDirectional GRU(right)

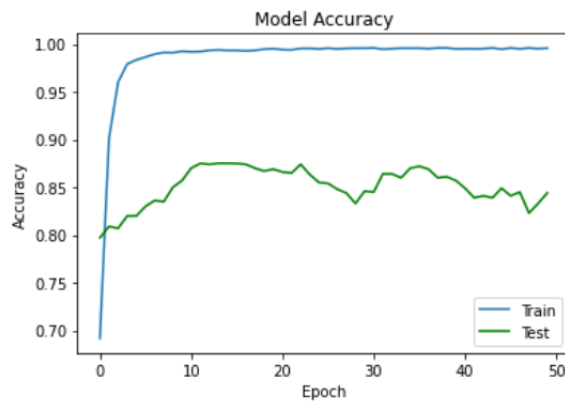


Figure 5.7: accuracy plot of CNN - LSTM

## 5.2 Performance Comparison

The experiment includes testing eight different variants of the model. We found that handling corrections for misspelled words during data pre-processing significantly improved the performance of the prediction model.

Performance (in Percentage)		
Algorithm	Accuracy (Before applying word corrector algorithm)(%)	Accuracy (After applying word corrector algorithm)(%)
LSTM	84	85
CNN	85.5	86
RNN	84.5	85.5
GRU	85.7	86
Bidirectional LSTM	83.7	85
Bidirectional GRU	83.8	85
CNN-LSTM	87.15	87.5
CNN-GRU	88	89

Table 5.2: Performance Comparison of Algorithms

By applying a word correction algorithm to our binary data set, the previously developed CNN-LSTM model [5] for Bang-lish sentiment classification improved its accuracy to 87.5%, while our proposed model improved upon that to 89%.

# Chapter 6

## Conclusion

Our work's constraints and potential outcomes will be discussed here as a wrap-up to the study. The potential areas of improvement for our research will also be discussed.

### 6.1 Final remarks

To provide a brief summary, we developed a hybrid model for the classification of sentiments based on Banglish text. The CNN and GRU standalone models performed worse than our recommended model, which had better results. We improved the model's overall performance by fixing the misspellings of some of the most crucial and often occurring terms in our dataset. We believe there is a substantial amount of opportunities and challenges in studying Banglish text. We are positive that our findings will pave the way for future studies of Banglish and other related languages like Hinglish (Hindi words written in English letters).

### 6.2 Limitations

During the course of research, one must anticipate coming into a number of constraints associated with the procedure. The following is a list of the study's limitations.

#### 6.2.1 Access

This is an important illustration of the constraints that were placed on the research for the dissertation. During the process of writing the dissertation, we have encountered roadblocks in gaining access to materials that are significant to our research, such as papers, persons, or other resources.

#### 6.2.2 Prior research

Insufficient prior study on the subject of the dissertation is another potential barrier that must be overcome. Some research projects will have a portion titled "literature review," which will require reading through previously published material. On the

other hand, there are also instances in which you can discover that very little work has been done in the past in a specific field related to your research. Due to the fact that Bang-lish is not a conventional language, there have been relatively few research conducted on it. This is also the situation in our case. However, Banglish is being used by the vast majority of the public across all kinds of social media and online shopping platforms.

### **6.2.3 Sample Size**

The term "sample size" refers to the amount of pieces of information that are incorporated into a research project; these details are meant to be representative of a specific group of materials. In quantitative research, sample size might be a limitation since the statistical results can be less precise if the sample size is small. If the sample size is too large, however, the statistical results will be more accurate. Because we had to handle the sample by hand throughout the investigation, the number of study subjects was restricted. A total of approximately 8000 samples were collected; however, only approximately 5000 were included in the dataset; the other samples were discarded.

### **6.2.4 Data aggregation**

There are many different approaches to data collection that can be taken for study. As a consequence of this, every strategy for data collection will have certain inherent restrictions. The process of collecting the data was difficult because the data cleaning had to be done manually, and there were a lot of components that weren't necessary that we had to get rid of before we could create our final dataset. In addition, it was rather difficult to create a balance between favorable and negative assessments of the product.

### **6.2.5 Time restraint**

It is possible that the amount of time available could be considered a limitation in some circumstances, however this is very dependent on the type of research being conducted. For us, it was necessary to finish the study within a specified amount of time. In spite of this, we were able to successfully complete the study and make the most of the time we had available.

## **6.3 Future Works**

As was said in the prior chapter, the results that our suggested hybrid deep learning model achieved in this criterion were superior to those that were attained by any other state-of-the-art model. This model, however, has the potential to be utilized in a wide variety of other fields after the necessary adjustments have been made. A portion of that being that -

### **6.3.1 Dataset enhancement**

as was indicated earlier, it's probable that the data we have is not sufficient enough to get the best results possible. As a result, the first step that has to be taken is to increase the number of samples in the dataset. In addition, the accuracy and precision of the model may improve as the dataset that it is based on grows greater.

### **6.3.2 Multi-class Sentiment Analysis**

In this investigation of a limited scope, the primary emphasis was placed on the categorization of the text-based sentiments as either positive or negative (Either positive or negative). In the future, when we have access to a sufficient amount of data, we will be able to take it up a notch by classifying sentiments into other categories, expanding beyond the two primary types of emotional expression.

### **6.3.3 Support for Multiple Languages**

Although the focus of our work was solely on Bengali words written using English syntax, this research can be applied to any language with a similar grammatical structure, including Hindi and Japanese (in the forms Hinglish and Japan-english/Romaji). Our research can serve as a foundation for future research like this. In addition to this, if studies have previously been conducted, then they will be able to contrast their models with ours.

### **6.3.4 Deploying a Bang-lish ChatBot**

Finally, the idea for the future would be to install a chatbot that is capable of communicating in Bang-lish, and we are hoping that this will enable us to take advantage of many new chances. At the moment, each and every chatbot is run on the raw form of its own language. As a result, the customer's experience with this chatbot may feel more individualized and tailored to their specific requirements.

# Bibliography

- [1] L. . Dey, S. . Chakraborty, A. . Biswas, B. . Bose **and** S. . Tiwari, *Sentiment Analysis of Review Datasets Using Naïve Bayes' and K-NN Classifier*, 8 **july** 2016. DOI: 10.5815/ijieeb.2016.04.07. **url:** <http://dx.doi.org/10.5815/ijieeb.2016.04.07>.
- [2] H. Kang, S. J. Yoo **and** D. Han, *Senti-lexicon and improved Naïve Bayes algorithms for sentiment analysis of restaurant reviews*, **april** 2012. DOI: 10.1016/j.eswa.2011.11.107. **url:** <http://dx.doi.org/10.1016/j.eswa.2011.11.107>.
- [3] E. F. Can, A. Ezen-Can **and** F. Can, *Multilingual Sentiment Analysis: An RNN-Based Framework for Limited Data*. **june** 2018. **url:** <https://arxiv.org/pdf/1806.04511.pdf>.
- [4] M. H. Munna, M. R. I. Rifat **and** A. S. M. Badrudduza, *Sentiment Analysis and Product Review Classification in E-commerce Platform*, 19 **december** 2020. DOI: 10.1109/iccit51783.2020.9392710. **url:** <http://dx.doi.org/10.1109/iccit51783.2020.9392710>.
- [5] R. Basri, M. Mridha, M. A. Hamid **and** M. M. Monowar, *A Deep Learning based Sentiment Analysis on Bang-lish Disclosure*, **march** 2021. DOI: 10.1109/nccc49330.2021.9428849. **url:** <http://dx.doi.org/10.1109/nccc49330.2021.9428849>.
- [6] C. N. Dang, M. N. Moreno-García **and** F. . De la Prieta, *Hybrid Deep Learning Models for Sentiment Analysis*, T. . Jia, **editor**, 12 **august** 2021. DOI: 10.1155/2021/9986920. **url:** <http://dx.doi.org/10.1155/2021/9986920>.
- [7] T. Nasukawa **and** J. Yi, *Sentiment analysis*, **october** 2003. DOI: 10.1145/945645.945658. **url:** <http://dx.doi.org/10.1145/945645.945658>.
- [8] W. Wang **and** J. Gang, *Application of Convolutional Neural Network in Natural Language Processing*, **july** 2018. DOI: 10.1109/iciscae.2018.8666928. **url:** <http://dx.doi.org/10.1109/iciscae.2018.8666928>.
- [9] S. Das, M. S. Islam **and** I. Mahmud, *A Deep Learning Study On Understanding Banglish and Abbreviated Words Used in Social Media*, **may** 2021. DOI: 10.1109/iccics51141.2021.9432339. **url:** <http://dx.doi.org/10.1109/iccics51141.2021.9432339>.
- [10] S. Chowdhury **and** W. Chowdhury, *Performing sentiment analysis in Bangla microblog posts*, **may** 2014. DOI: 10.1109/iciev.2014.6850712. **url:** <http://dx.doi.org/10.1109/iciev.2014.6850712>.
- [11] R. . Bhargava, Y. . Sharma **and** S. . Sharma, *Sentiment analysis for mixed script Indic sentences*, **september** 2016. DOI: 10.1109/icacci.2016.7732099. **url:** <http://dx.doi.org/10.1109/icacci.2016.7732099>.

- [12] X. Chen, C. Ouyang, Y. Liu, L. Luo **and** X. Yang, *A Hybrid Deep Learning Model for Text Classification*, **september** 2018. DOI: 10.1109/skg.2018.00014. **url:** <http://dx.doi.org/10.1109/skg.2018.00014>.
- [13] J. Zheng **and** L. Zheng, *A Hybrid Bidirectional Recurrent Convolutional Neural Network Attention-Based Model for Text Classification*, 2019. DOI: 10.1109/access.2019.2932619. **url:** <http://dx.doi.org/10.1109/access.2019.2932619>.
- [14] A. K. Durairaj **and** A. . Chinnalagu, *Sentiment Analysis on Mixed-Languages Customer Reviews: A Hybrid Deep Learning Approach*, 10 **december** 2021. DOI: 10.1109/smart52563.2021.9676277. **url:** <http://dx.doi.org/10.1109/smart52563.2021.9676277>.
- [15] U. Saha, M. S. Mahmud, A. Chakroborty, M. Akter, M. R. Islam **and** A. A. Marouf, *Sentiment Classification in Bengali News Comments using a hybrid approach with Glove*, **april** 2022. DOI: 10.1109/icoei53556.2022.9777096. **url:** <http://dx.doi.org/10.1109/icoei53556.2022.9777096>.
- [16] J. Duan, H. Zhao, W. Qin, M. Qiu **and** M. Liu, *News Text Classification Based on MLCNN and BiGRU Hybrid Neural Network*, **october** 2020. DOI: 10.1109/smartblock52591.2020.00032. **url:** <http://dx.doi.org/10.1109/smartblock52591.2020.00032>.
- [17] X. She **and** D. Zhang, *Text Classification Based on Hybrid CNN-LSTM Hybrid Model*, **december** 2018. DOI: 10.1109/iscid.2018.10144. **url:** <http://dx.doi.org/10.1109/iscid.2018.10144>.
- [18] J. Devlin, M.-W. Chang, K. Lee **and** K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, **october** 2018. **url:** <https://arxiv.org/pdf/1810.04805v2>.
- [19] A. Vaswani, N. Shazeer, N. Parmar **and others**, *Attention is All you Need*, **june** 2017. **url:** <https://arxiv.org/pdf/1706.03762v5>.
- [20] A. Bhattacharjee, T. Hasan, K. Samin **and others**, *BanglaBERT: Combating Embedding Barrier in Multilingual Models for Low-Resource Language Understanding*, **january** 2021. **url:** <https://arxiv.org/pdf/2101.00204>.
- [21] C. Sun, X. Qiu, Y. Xu **and** X. Huang, *How to Fine-Tune BERT for Text Classification?* 2019. DOI: 10.1007/978-3-030-32381-3\_16. **url:** [http://dx.doi.org/10.1007/978-3-030-32381-3\\_16](http://dx.doi.org/10.1007/978-3-030-32381-3_16).
- [22] D. Gene **and** S. Bengali, *GitHub repository*. 2016.
- [23] C. M. Bishop, *Neural networks and their applications*, **june** 1994. DOI: 10.1063/1.1144830. **url:** <http://dx.doi.org/10.1063/1.1144830>.
- [24] D. Jurafsky **and** J. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, **january** 2000. **url:** <https://nats-www.informatik.uni-hamburg.de/pub/CDG/JurafskyMartin00Comments/JurafskyMartin00-Review.pdf>.
- [25] D. Bahdanau, K. Cho **and** Y. Bengio, *Neural Machine Translation by Jointly Learning to Align and Translate*. **september** 2014. **url:** <https://arxiv.org/pdf/1409.0473v7>.