Sentiment analysis on twitter data using deep learning algorithms

Afsana Afrin Brishty	**170104086**
Faiza Anan Noor	**170104093**
Anika Bintee Aftab	**170104098**

Project Report

Course ID: CSE 4238

Course Name: Soft Computing Lab

Semester: Fall 2020

Department of Computer Science and Engineering

Ahsanullah University of Science and Technology

Dhaka, Bangladesh

03, October 2021

Sentiment analysis on twitter data using deep learning algorithms

Submitted by

Afsana Afrin Brishty	**170104086**
Faiza Anan Noor	**170104093**
Anika Bintee Aftab	**170104098**

Submitted To

Nibir Chandra Mandal

Sanzana Karim Lora

Department of Computer Science and Engineering

Ahsanullah University of Science and Technology

Department of Computer Science and Engineering

Ahsanullah University of Science and Technology

Dhaka, Bangladesh

03, October 2021

ABSTRACT

With the advancement of web technology and its growth, there is a huge volume of data present on the web for internet users, and a lot of data is generated too. The Internet has become a platform for online learning, exchanging ideas, and sharing opinions. Social networking sites like Twitter, Facebook are rapidly gaining popularity as they allow people to share and express their views about topics, have discussions with different communities, or post messages across the world. There has been a lot of work in the field of sentiment analysis of Twitter data. This project focuses mainly on sentiment analysis of Twitter data which is helpful to analyze the information in the tweets where opinions are highly unstructured, heterogeneous, and are either positive or negative or neutral in some cases. Sentiment classification is a task where text documents are classified according to their sentiment. In this study, we provide a comprehensive survey and an analysis of our experiments on different techniques of deep learning models like Convolutional Neural Network (CNN), Bi-directional RNN, Long Short Term Memory (LSTM) and Bi-directional LSTM together with evaluation metrics used for measuring their efficiency and efficacy such as Accuracy, F1-Score, Precision, Recall, Confusion Matrix, etc. The goal of this research is to find out why these models work well for sentiment analysis and how these models work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The area of computer science and artificial intelligence concerned with the interaction between computers and natural language is called Natural Language Processing (NLP). The natural language contains large diverse vocabularies, words with several different meanings and speakers with different accents. Human interaction and mutual understanding are therefore complicated. For the most part, humans can easily deal with these kinds of problems and skillful use of language is a major part of what makes us human. For this reason, designing a machine that is capable of understanding, speaking and responding properly to natural language has intrigued engineers and scientists for centuries [1].
Nowadays, the age of the Internet has changed the way people express their views, opinions. It is now mainly done through blog posts, online forums, product review websites, social media, etc. Nowadays, millions of people are using social network sites like Facebook, Twitter to express their emotions, opinions and share views about their daily lives. Through the online communities, we get interactive media where consumers inform and influence others through forums. Social media is generating a large volume of sentiment-rich data in the form of tweets, status updates, blog posts, comments, reviews, etc. Moreover, social media provides an opportunity for businesses by giving them a platform to connect with their customers for advertising. People mostly depend upon user-generated content online to a great extent for decision-making. For example, if someone wants to buy a product or wants to use any service, then they firstly look up its reviews online, discuss it on social media before making a decision. The amount of content generated by users is too vast for a normal user to analyze. So there is a need to automate this, various sentiment analysis techniques are widely used.
There are two distinct focuses made in NLP, language processing and language generation. Language processing refers to analysis of language for the purpose of producing a meaningful interpretation, language generation refers to the task of generating natural language. Another distinction made in NLP is between analysis of spoken language and written lan-

guage. The emphasis in this paper will be on the processing of language in the written form, especially sentiment analysis. Sentiment analysis (SA) tells users whether the information about the product is satisfactory or not before they buy it. Marketers and firms use this analysis data to understand their products or services in such a way that it can be offered as per the users requirements.Textual Information retrieval techniques mainly focus on processing, searching, or analyzing the factual data present. Facts have an objective component but there are some other textual contents that express subjective characteristics. These contents are mainly opinions, sentiments, appraisals, attitudes, and emotions, which form the core of Sentiment Analysis (SA). It offers many challenging opportunities to develop new applications, mainly due to the huge growth of available information on online sources like blogs and social networks. For example, recommendations of items proposed by a recommendation system can be predicted by taking into account considerations such as positive or negative opinions about those items by making use of SA.

Sentiment analysis is a subfield in NLP where the goal is to determine the attitude of a speaker or writer. Document-level sentiment classification is a functional task in sentiment analysis, and is crucial to understand user-generated content in social networks or product reviews. The task calls for identifying the overall sentiment polarity of a document [1].

Due to the development of the internet and technology, there has been an enormous amplification in the generation of user data because of the prevalence of numerous blogs, social networking sites and review forums, etc. Sentimental analysis is a natural language processing technique used to extract the feelings or attitudes of general masses regarding a given subject. The central aim of this project is to perform sentiment analysis on Twitter data, keeping the impact of tweets in the fields of education, entertainment, local and global news in mind. Hence, the goal of our project is to do an empirical study of different machine learning techniques in sentiment analysis with the help of tweets put forward by users. In short, we will be classifying whether the tweets of a particular user are negative or positive or neutral. The dataset that will be used for training our models will be the ”Twitter and Reddit Sentimental analysis Dataset” from Kaggle which contains 163K Tweets along with sentiment labels and the models trained on this dataset will be used later to classify whether a specific tweet is positive or negative or neutral. We will follow different approaches of deep neural models in extracting text features such as bag-of-words and will use multifarious performance metrics, such as accuracy, precision, etc for evaluating the efficacy of our proposed models which are- CNN, Bi-directional RNN, LSTM and Bi-Directional LSTM. Lastly, we will compare the results and will analyze which algorithm works best for our purpose.

1.1 Motivation

Sentiment analysis is extremely important because it allows businesses to understand the sentiment of their customers towards their brand. By automatically sorting the sentiment behind social media conversations, reviews, and more, businesses can make better and more informed decisions.It's estimated that 90% of the world's data is unstructured, in other words it's unorganized. Huge volumes of unstructured business data are created every day: emails, support tickets, chats, social media conversations, surveys, articles, documents, etc). But it's hard to analyze sentiment in a timely and efficient manner.
The overall benefits of sentiment analysis include:

- **Sorting Data at Scale:** Can you imagine manually sorting through thousands of tweets, customer support conversations, or surveys? There's just too much business data to process manually. Sentiment analysis helps businesses process huge amounts of data in an efficient and cost-effective way.
- **Real-Time Analysis:** Sentiment analysis can identify critical issues in real-time, for example is a PR crisis on social media escalating? Is an angry customer about to churn? Sentiment analysis models can help you immediately identify these kinds of situations, so you can take action right away.
- **Consistent criteria:** It's estimated that people only agree around 60-65% of the time when determining the sentiment of a particular text. Tagging text by sentiment is highly subjective, influenced by personal experiences, thoughts, and beliefs. By using a centralized sentiment analysis system, companies can apply the same criteria to all of their data, helping them improve accuracy and gain better insights [2].

1.2 Challenges

Here we are exploring the most complex natural language processing (NLP) issue: sentiment analysis challenges, and how to overcome them. Sentiment analysis has become an integral part of marketing. Not only can sentiment analysis accuracy help organizations establish how they are perceived, but it can also help them identify potential pitfalls in their marketing operations and branding content that can be dealt with on time.
When it comes to sentiment analysis challenges, there are quite a few things that companies struggle with in order to obtain sentiment analysis accuracy. Sentiment or emotion analysis can be difficult in natural language processing simply because machines have to be trained

to analyze and understand emotions as a human brain does. This is in addition to understanding the nuances of different languages. As data science continues to evolve, sentiment analysis software is able to tackle these issues better. Here are the main roadblocks in analyzing sentiment.
Main challenges of Sentiment Analysis of text data-

- **Polarity:** Words such as “love” and “hate” are high on positive (+1) and negative (-1) scores in polarity. These are easy to understand. But there are in-between conjugations of words such as “not so bad” that can mean “average” and hence lie in mid-polarity (-75). Sometimes phrases like these get left out, which dilutes the sentiment score.

- **Sarcasm:** People use irony and sarcasm in casual conversations and memes on social media. The act of expressing negative sentiment using backhanded compliments can make it difficult for sentiment analysis tools to detect the true context of what the response is actually implying. This can often result in a higher volume of “positive” feedback that is actually negative.

- **Emojis:** The problem with social media content that is text-based, like Twitter, is that they are inundated with emojis. NLP tasks are trained to be language specific. While they can extract text from even images, emojis are a language in itself. Most emotion analysis solutions treat emojis like special characters that are removed from the data during the process of text mining. But doing so means that companies will not receive holistic insights from the data.

- **Idioms:** Machine learning programs don’t necessarily understand a figure of speech. For example, an idiom like “not my cup of tea” will boggle the algorithm because it understands things in the literal sense. Hence, when an idiom is used in a comment or a review, the sentence can be misconstrued by the algorithm or even ignored. To overcome this problem a sentiment analysis platform needs to be trained in understanding idioms. When it comes to multiple languages, this problem becomes manifold.

- **Negations:** Negations given by words such as not, never, cannot, were not, etc. can confuse the ML model. For example, a machine algorithm needs to understand that a phrase that says, “I can’t help going to my class reunion”, means that the person intends to go to the class reunion.

- **Comparative Sentences:** Comparative sentences can be tricky because they may not always give an opinion. Much of it has to be deduced. For example, when somebody writes, “the Galaxy S20 is larger than the Apple iphone12”, the sentence does not

mention any negative or positive emotion but rather states a relative ordering in terms of the size of the two phones [3].

Chapter 2

Background Study

Pak and Paroubek(2010) [4] proposed a model to classify the tweets as objective, positive and negative. They created a twitter corpus by collecting tweets using Twitter API and automatically annotating those tweets using emoticons. Using that corpus, they developed a sentiment classifier based on the multinomial Naive Bayes method that uses features like Ngram and POS-tags. The training set they used was less efficient since it contains only tweets having emoticons.
Parikh and Movassate(2009) [5] implemented two models, a Naive Bayes bigram model and a Maximum Entropy model to classify tweets. They found that the Naive Bayes classifiers worked much better than the Maximum Entropy model.
Go and L.Huang (2009) [6] proposed a solution for sentiment analysis for twitter data by using distant supervision, in which their training data consisted of tweets with emoticons which served as noisy labels. They build models using Naive Bayes, MaxEntropy and Support Vector Machines (SVM). Their feature space consisted of unigrams, bigrams and POS. They concluded that SVM outperformed other models and that unigram was more effective as features.
Barbosa et al.(2010) [7] designed a two-phase automatic sentiment analysis method for classifying tweets. They classified tweets as objective or subjective and then in the second phase, the subjective tweets were classified as positive or negative. The feature space used included retweets, hashtags, link, punctuation and exclamation marks in conjunction with features like prior polarity of words and POS.
Bifet and Frank(2010) [8] used Twitter streaming data provided by Firehouse API, which gave all messages from every user which are publicly available in real-time. They experimented with multinomial naive Bayes, stochastic gradient descent, and the Hoeffding tree. They arrived at a conclusion that the SGD-based model, when used with an appropriate learning rate was better than the rest used.
Agarwal et al. (2011) [9] developed a 3-way model for classifying sentiment into positive,

negative and neutral classes. They experimented with models such as: unigram model, a feature based model and a tree kernel based model. For the tree kernel-based model they represented tweets as a tree.The feature based model uses 100 features and the unigram model uses over 10,000 features. They arrived at a conclusion that features which combine prior polarity of words with their parts-of-speech(pos) tags are most important and play a major role in the classification task. The tree kernel based model outperformed the other two models.

Davidov et al.,(2010) [10] proposed an approach to utilize Twitter user-defined hashtags in tweets as a classification of sentiment type using punctuation, single words, n-grams and patterns as different feature types, which are then combined into a single feature vector for sentiment classification. They made use of the K-Nearest Neighbor strategy to assign sentiment labels by constructing a feature vector for each example in the training and test set.

Po-Wei Liang et.al.(2014) [11] used Twitter API to collect twitter data. Their training data falls in three different categories (camera, movie , mobile). The data is labeled as positive, negative and non-opinions. Tweets containing opinions were filtered. The Unigram Naive Bayes model was implemented and the Naive Bayes simplifying independence assumption was employed. They also eliminated useless features by using the Mutual Information and Chi square feature extraction method. Finally, the orientation of a tweet is predicted. i.e. positive or negative.

Pablo et. al. [12] presented variations of Naive Bayes classifiers for detecting polarity of English tweets. Two different variants of Naive Bayes classifiers were built namely Baseline (trained to classify tweets as positive, negative and neutral), and Binary (makes use of a polarity lexicon and classifies as positive and negative. Neutral tweets neglected). The features considered by classifiers were Lemmas (nouns, verbs, adjectives and adverbs), Polarity Lexicons, and Multiword from different sources and Valence Shifters.

Turney et al [13] used a bag-of-words method for sentiment analysis in which the relationships between words was not at all considered and a document is represented as just a collection of words. To determine the sentiment for the whole document, sentiments of every word were determined and those values are united with some aggregation functions.

Kamps et al. [14] used the lexical database WordNet to determine the emotional content of a word along different dimensions. They developed a distance metric on WordNet and determined the semantic polarity of adjectives.

Xia et al. [15] used an ensemble framework for Sentiment Classification which is obtained by combining various feature sets and classification techniques. In their work, they used two types of feature sets (Part-of-speech information and World Relations) and three base classifiers (Naive Bayes, Maximum Entropy and Support Vector Machines) . They applied ensemble approaches like fixed combination, weighted combination and Meta-classifier combination for sentiment classification and obtained better accuracy.

Luoet. al. [16] highlighted the challenges and efficient techniques to mine opinions from Twitter tweets. Spam and wildly varying language makes opinion retrieval within Twitter a challenging task.

Currently, with the increasing numbers and size of data, researchers have been developing a deep neural networks approach to be applied in the sentiment analysis task. Santos and Gatti [17] performed sentiment analysis by applying deep convolutional neural networks architecture joined with three different level representations, such as character-level, word-level and sentence-level representations. The proposed method has been applied on two different corpora, namely Stanford Sentiment Treebank and Stanford Twitter Sentiment corpus. Ali, et al [18] carried out sentiment analysis using movie reviews dataset by utilizing four deep learning algorithms, namely Multilayer Perceptron (MLP), Convolutional Neural Network (CNN), Long short-term memory (LSTM) and CNN_LSTM. The experiments have shown that CNN_LSTM outperformed other deep learning methods. Jianqiang, et al [19] proposed a method by combining Glove (Global Vectors) as word representation and Deep Convolutional Neural Networks (DCNN) to conduct sentiment analysis utilizing Twitter dataset. Their study revealed that the Glove-DCNN method performed better than the bag-of-words model with Support Vector Machine (SVM) classifier. Wang et al [20] performed sentiment analysis by applying two Deep Learning algorithms, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). The study revealed that the model from the combination between CNNs and RNNs achieved higher accuracy than other existing models [21]. This study applies several different deep neural network algorithms to classify the sentiment on Twitter dataset. In addition, this work aims to classify the sentiment of the dataset into positive, negative and neutral as it is a multiclass classification. To obtain the best model, we train our dataset using some different deep neural network algorithms, including Convolutional Neural Network (CNN), Bi-directional RNN, Long short-term memory (LSTM), and Bidirectional LSTM.

Chapter 3

Project Objectives

In this study, we will apply different deep learning algorithms on twitter dataset to classify the sentiment. Next we will measure the performance of each model and evaluate the comparison among them to bring out which model performs better than the others. So our work has been organized as follows:

- Data Pre-Processing
- Developing Deep Learning Model
- Measuring Performance

3.1 Data-Preprocessing

The raw data having polarity is highly susceptible to inconsistency and redundancy. Preprocessing of tweet include following points:

- **Removing Null Cells:** In the dataset, some cells were null. Those cells were removed as they do not impact the model development and accuracy.
- **Removing Non Letters:** No letters like special symbols (*,+,- etc) were removed for simplifying the feature extraction process.
- **Tokenization:** This is a process which refers to splitting a phrase, sentence, paragraph, or an entire text document into smaller units called tokens. Tokenization is a way of separating a piece of text into smaller units called tokens. Here, tokens can be either words, characters, or subwords. Hence, tokenization can be broadly classified into 3

types – word, character, and subword (n-gram characters) tokenization. For example, consider the sentence: "Never give up". The most common way of forming tokens is based on space. Assuming space as a delimiter, the tokenization of the sentence results in 3 tokens – Never-give-up. As each token is a word, it becomes an example of Word tokenization.

- **Stemming:** Stemming is a process which refers to the process of reducing a word to its word stem that affixes to suffixes and prefixes or to the roots of words known as a lemma.For example: "Flying" is a word and its suffix is "ing", if we remove "ing" from "Flying" then we will get base word or root word which is "Fly".
- **Splitting data:** The dataset was splitted into 80% and 20% for training and testing.
- **Using Bag of words:** A bag-of-words model, or BoW for short, is a way of extracting features from text for use in modeling, such as with machine learning algorithms. It is called a "bag" of words, because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.

So the total pre-processing can be shown by the following diagram:

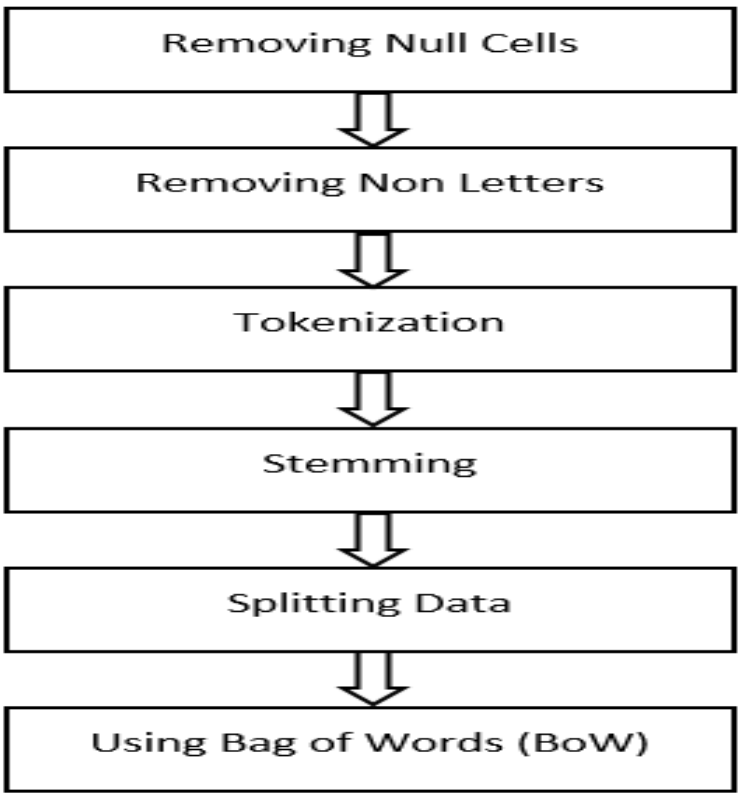

Figure 3.1: Diagram of Data Pre-Processing.

3.2 Deep Learning Model

3.2.1 Convolutional Neural Network

Convolutional Neural Network (CNN) is a deep learning method that is developed from multilayer perceptron (MLP). CNN consists of three main layers such as convolution, pooling and fully connected layer. The convolution layer is the primary CNN building block which consists of a set of independent filters. Pooling layer aims to reduce the complexity of convolved layer representation. The vector results from several convolutions and pooling operations on the multilayer perceptron are known as fully connected layers that are used to do a particular task such as classification.

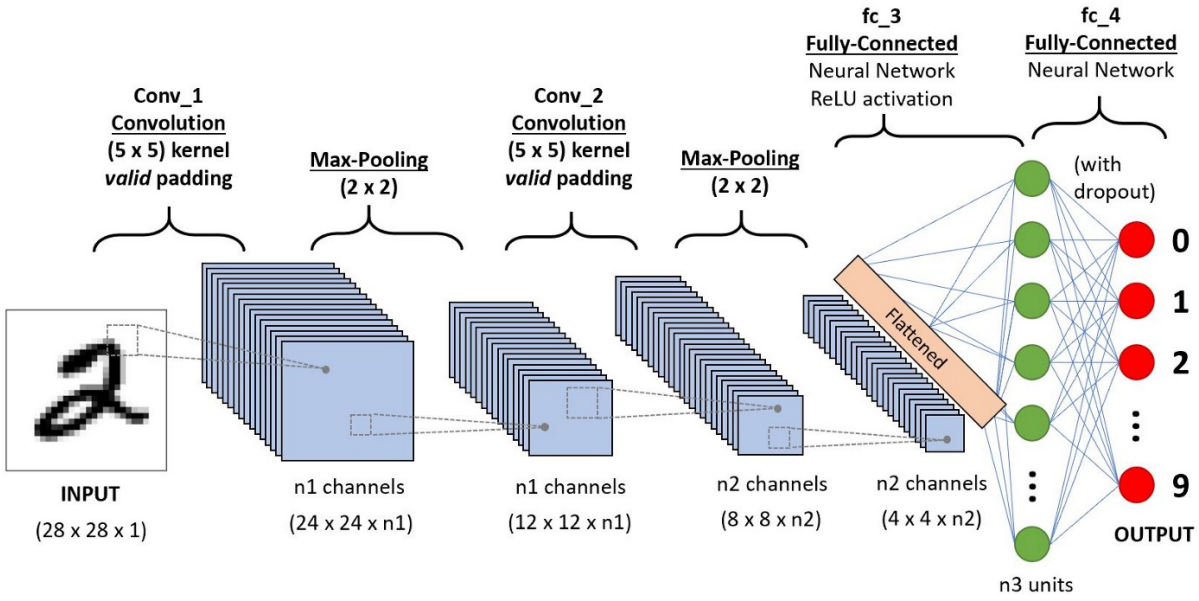

Figure 3.2: A CNN sequence to classify handwritten digits.

3.2.2 Bi-directional Recurrent Neural Network (Bi-RNN)

In a bidirectional RNN, 2 separate sequences are considered. One from right to left and the other in the reverse order. Considering the word sequence “I love mango juice”, the forward layer would feed the sequence as such. But, the Backward Layer would feed the sequence in the reverse order “juice mango love I”. Now, the outputs would be generated by concatenating the word sequences at each time and generating weights accordingly. This can be used for POS tagging problems as well [22].

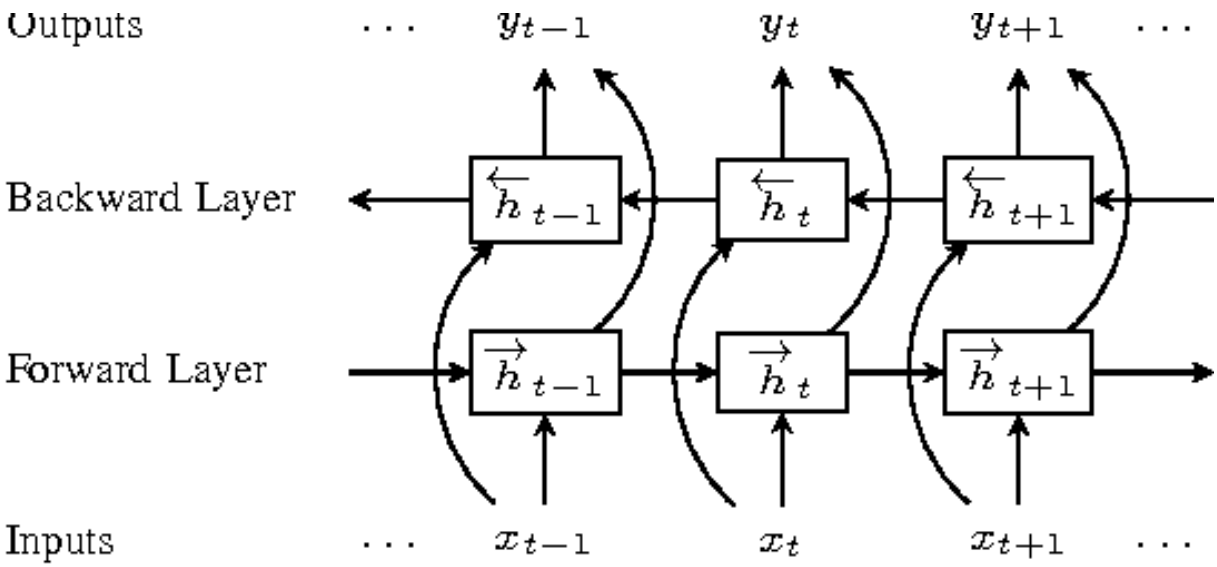

Figure 3.3: Bi Directional RNN for word sequence.

3.2.3 Long-Short Term Memory (LSTM)

LSTM is an improved model of RNN that is developed to overcome long time-dependencies problems. An LSTM cell consists of three gates: input gate, forget gate and output gate. These gates are produced by a sigmoid function over the input sequences and the previous hidden state. These gates receive information across LSTM through cell state. The fundamental component of LSTM is cell state as a transport of information from memory from previous block to memory from current block.

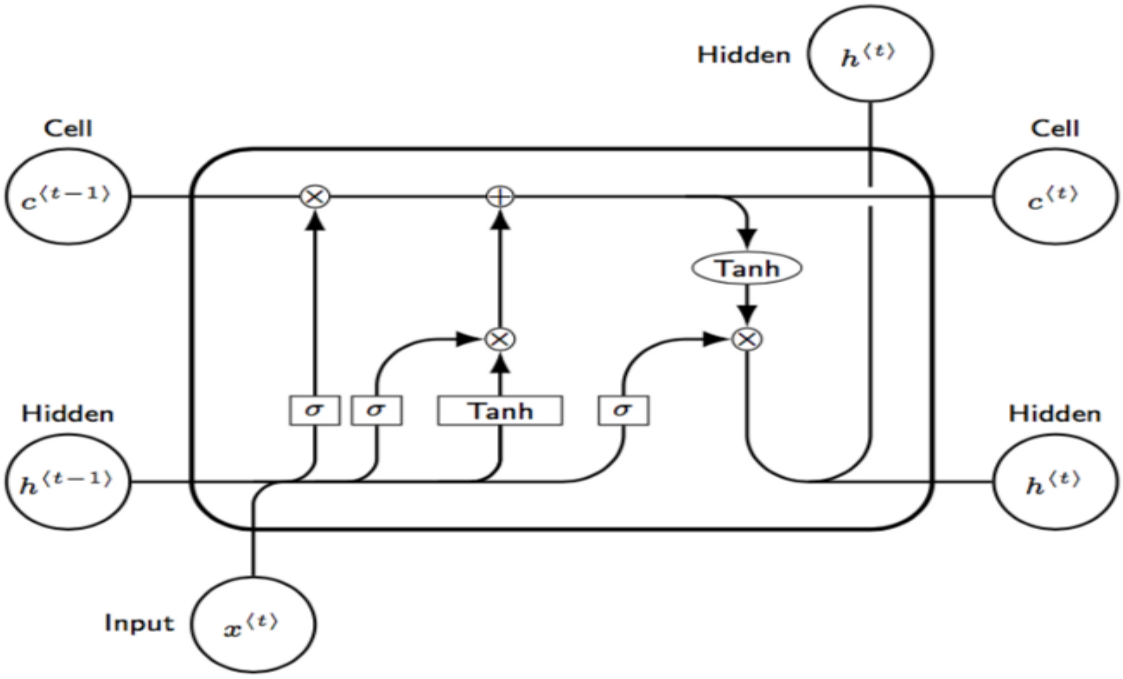

Figure 3.4: LSTM Architecture.

3.2.4 Bi-directional Long-Short Term Memory (Bi-LSTM)

Bidirectional LSTM (BLSTM) is an improved model of LSTM that connects two hidden layers to the same output. The objectives of BLSTM are to enhance performance of the model on sequential problems. Using bidirectional inputs will in two ways, one from past to future and one from future to past and what differs this approach from unidirectional is that in the LSTM that runs backward preserve information from the future.Using the two hidden states combined it will be possible in any point in time to preserve information from both past and future.

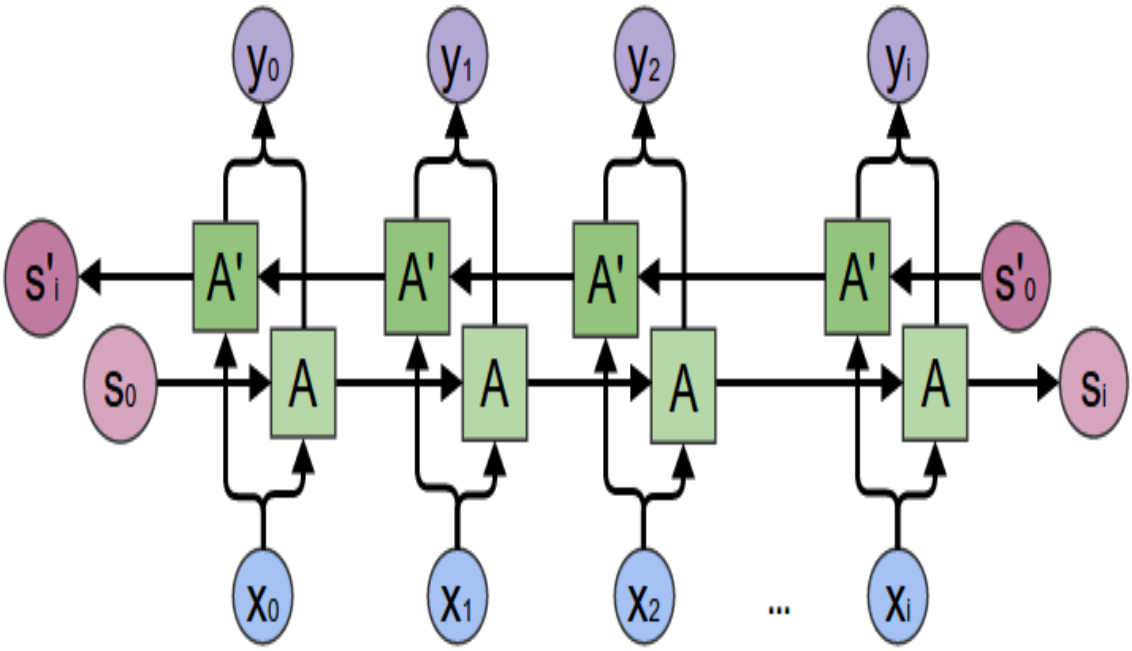

Figure 3.5: Bi-directional LSTM Architecture.

3.3 Performance Metrics

3.3.1 Accuracy

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Accuracy is a great measure but only for the symmetric datasets where values of false positives and false negatives are almost same.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{3.1}$$

3.3.2 Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

3.3.3 Recall (Sensitivity)

Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

3.3.4 F1 score

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if we have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall [23].

$$F1 - Score = \frac{2 * (Recall * Precision)}{Recall + Precision} \tag{3.4}$$

3.3.5 Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing [24].

Let's now define the most basic terms, which are whole numbers (not rates):

- **True Positives (TP):** These are cases in which we predicted yes (they have the disease), and they do have the disease.
- **True Negatives (TN):** We predicted no, and they don't have the disease.

- **False Positives (FP):** We predicted yes, but they don't actually have the disease. (Also known as a "Type I error.")
- **False Negatives (FN):** We predicted no, but they actually do have the disease. (Also known as a "Type II error.")

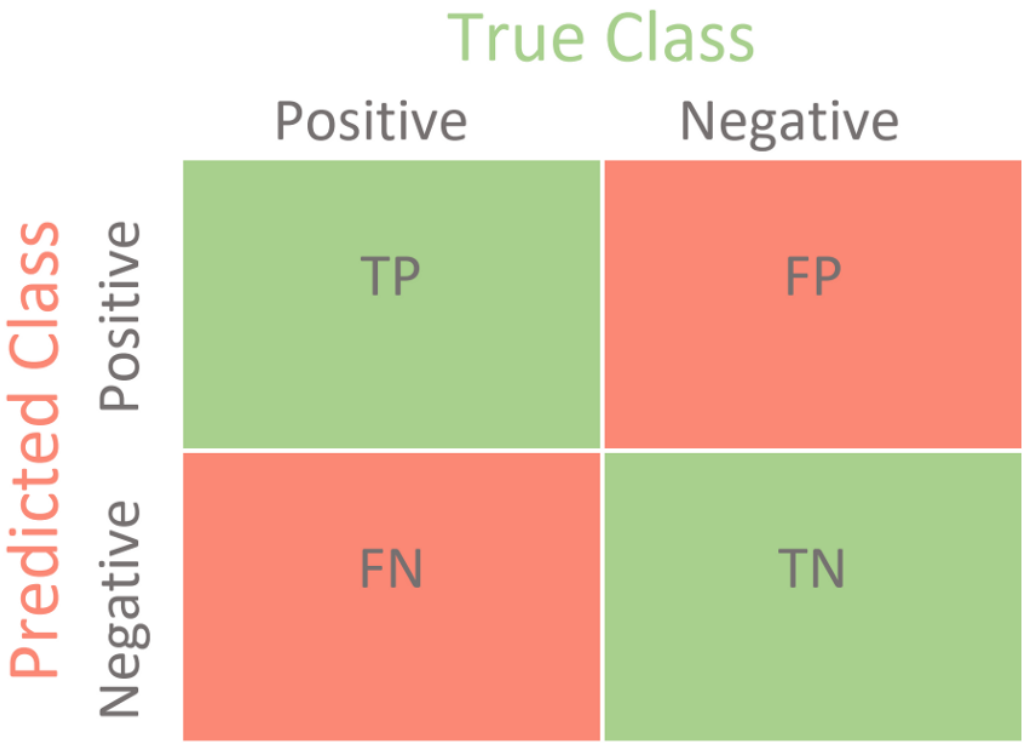

Figure 3.6: General structure of a confusion matrix.

Chapter 4

Methodologies

In our project, we have tried to find out whether a text of Twitter is positive, negative or neutral through applying different types of machine learning classification algorithms. First of all, we need to process the dataset so that we can apply the algorithms to this. For the feature extraction, we have applied the Bag-of-Words algorithm. A bag-of-words model is a way to describe the occurrence of words within a document of texts. It involves a vocabulary of known words and a measure of the presence of known words. The Bag-of-Words model is used to preprocess the text by converting it into a bag of words, which keeps a count of the total occurrences of the most frequently used words. Whenever we apply any algorithm in NLP, it works on numbers. We cannot directly feed our text into that algorithm. So, a bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things, firstly a vocabulary of known words and secondly, a measure of the presence of known words. It is called a "bag" of words because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not wherein the document.

Trained Model: We have trained four different deep neural network models. They are- CNN, Bi-RNN, LSTM and Bi-LSTM. These models have been applied in our processed dataset.

CNN: For the training of CNN model we have used "selu" and "softmax" as activation function and "sgd" as optimizer. The whole training has been done for 20 epochs. The embedding size, batch size, momentum and learning rate has been taken as 32, 64, 0.8 and 0.1 respectively. The vocabulary size was taken as 5000 for the training.
The CNN model flow diagram and gradual accuracy have been shown with the increase of epochs below:

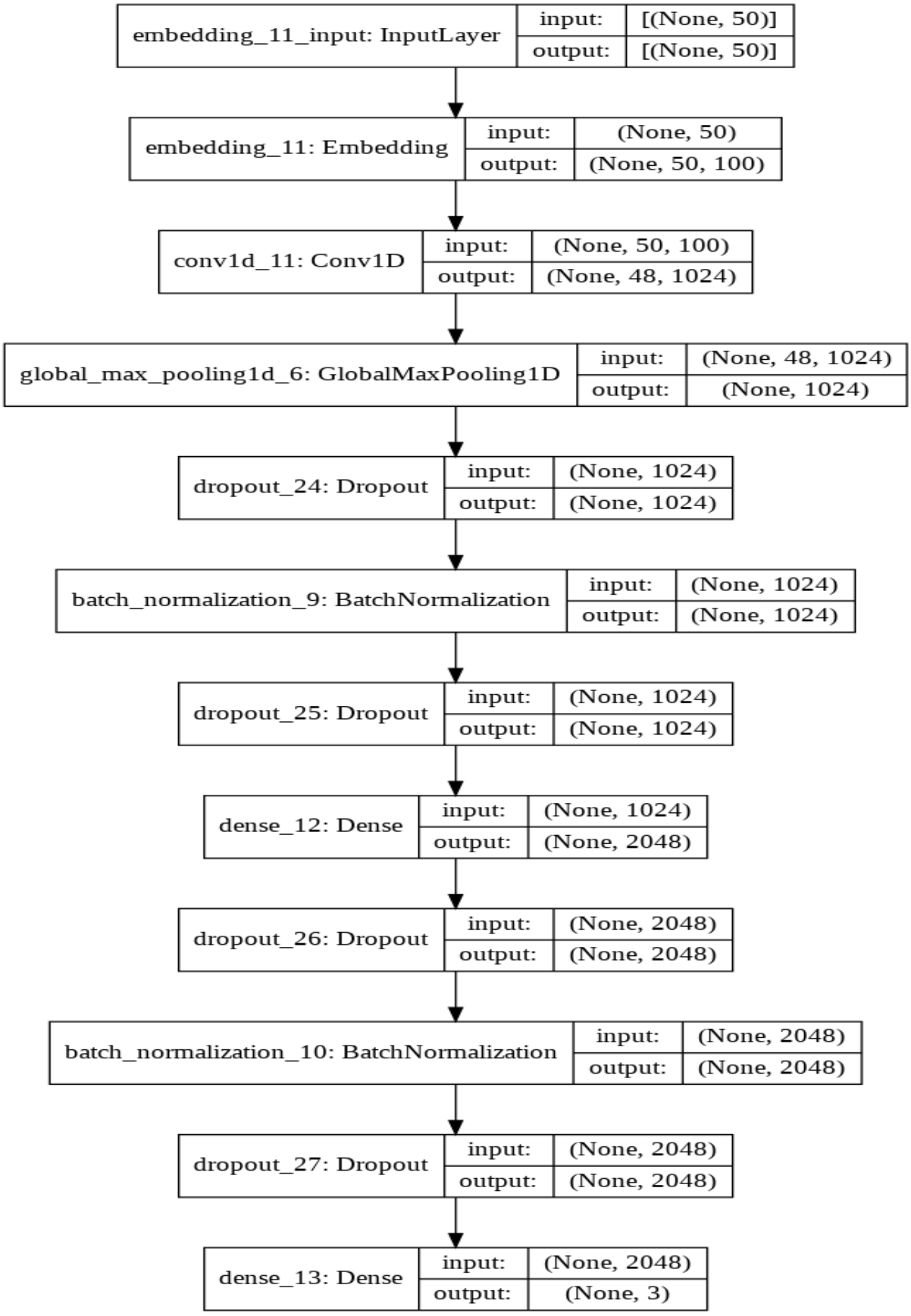

Figure 4.1: CNN model flow diagram.

Epoch No.	Train Accuracy	Train Loss	Val Accuracy	Val Loss
1	0.604903	2.226045	0.698104	0.667987
2	0.721817	0.682396	0.717279	0.625016
3	0.754819	0.612267	0.738418	0.599447
4	0.773064	0.578435	0.737130	0.597592
5	0.785633	0.557197	0.738449	0.581624
6	0.793395	0.540097	0.758667	0.549682
7	0.801444	0.528547	0.764098	0.545999
8	0.806271	0.519843	0.745689	0.568296
9	0.811753	0.508428	0.763668	0.540761
10	0.816907	0.498876	0.776953	0.519036
11	0.820210	0.491653	0.786402	0.501794
12	0.823780	0.484290	0.786679	0.498931
13	0.827472	0.478720	0.791495	0.492157
14	0.829824	0.472952	0.810793	0.466675
15	0.833281	0.467527	0.802479	0.476028
16	0.836001	0.461375	0.806928	0.468650
17	0.839294	0.456736	0.809075	0.464419
18	0.840398	0.452500	0.816899	0.453623
19	0.843610	0.445665	0.830398	0.434568
20	0.844745	0.444176	0.829508	0.434379

Figure 4.2: Gradual accuracy with the increase of epochs(CNN).

Bi-RNN: For the training of Bi-RNN model we have used “selu” and “softmax” as activation function and “sgd” as optimizer. The whole training has been done for 20 epochs. The embedding size, batch size, momentum and learning rate has been taken as 32, 64, 0.8 and 0.1 respectively. The vocabulary size was taken as 5000 for the training. The Bi-RNN model flow diagram and gradual accuray has been shown with the increase of epochs below-

LSTM: For the training of LSTM model we have used “selu” and “softmax” as activation function and “sgd” as optimizer. The whole training has been done for 20 epochs. The embedding size, batch size, momentum and learning rate has been taken as 32, 64, 0.8 and 0.1 respectively. The vocabulary size was taken as 5000 for the training.
The LSTM model flow diagram and gradual accuray has been shown with the increase of epochs below-

Bi-LSTM: For the training of Bi-LSTM model we have used “selu” and “softmax” as activation function and “sgd” as optimizer. The whole training has been done for 20 epochs. The embedding size, batch size, momentum and learning rate has been taken as 32, 64, 0.8 and 0.1 respectively. The vocabulary size was taken as 5000 for the training.
The Bi-LSTM model flow diagram and gradual accuray has been shown with the increase of epochs below-

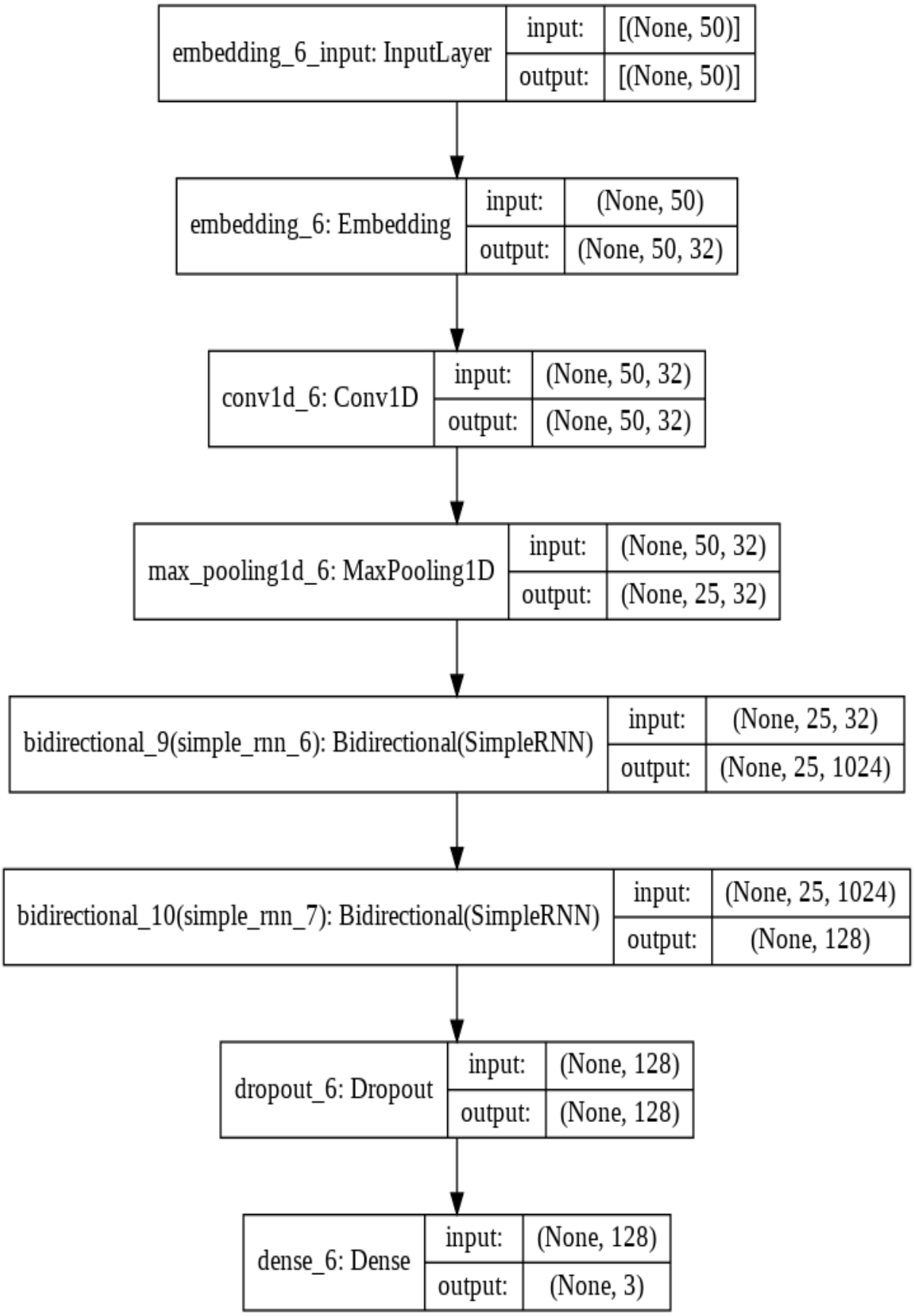

Figure 4.3: Bi-RNN model flow diagram.

Epoch No.	Train Accuracy	Train Loss	Val Accuracy	Val Loss
1	0.508156	1.000955	0.564981	0.942920
2	0.710118	0.692847	0.828220	0.490722
3	0.858459	0.435264	0.880622	0.363945
4	0.895777	0.343191	0.901178	0.308309
5	0.910934	0.298392	0.917562	0.266135
6	0.920782	0.269870	0.923299	0.249618
7	0.927532	0.250746	0.927134	0.236925
8	0.932758	0.237671	0.934313	0.225220
9	0.937370	0.225771	0.928576	0.237118
10	0.940121	0.215190	0.933638	0.228713
11	0.942709	0.207639	0.937749	0.215932
12	0.945398	0.201161	0.938302	0.214050
13	0.947076	0.195419	0.940480	0.214090
14	0.948160	0.190306	0.937535	0.214060
15	0.949377	0.186220	0.940112	0.213621
16	0.949735	0.182432	0.940664	0.213904
17	0.951627	0.177687	0.939467	0.210896
18	0.952537	0.174961	0.943026	0.202068
19	0.954102	0.170388	0.942934	0.207262
20	0.954204	0.167160	0.943609	0.203025

Figure 4.4: Gradual accuracy with the increase of epochs(Bi-RNN).

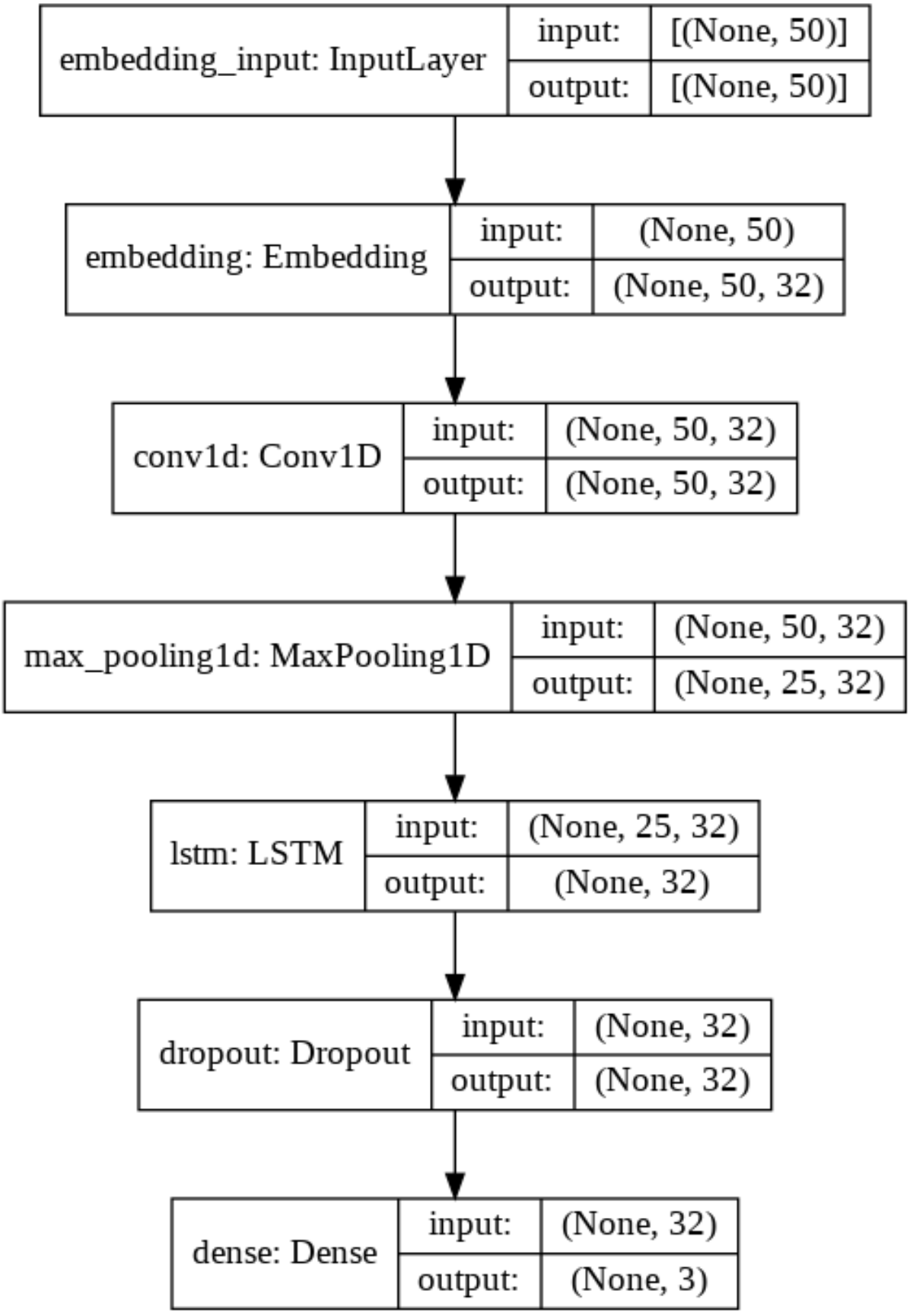

Figure 4.5: LSTM model flow diagram.

Epoch No.	Train Accuracy	Train Loss	Val Accuracy	Val Loss
1	0.521829	0.988584	0.536510	0.970798
2	0.619926	0.844013	0.705222	0.649803
3	0.798754	0.536421	0.845800	0.448465
4	0.858408	0.436204	0.852059	0.430079
5	0.874720	0.403803	0.878873	0.382990
6	0.881316	0.386435	0.880684	0.368545
7	0.887043	0.369738	0.885408	0.356916
8	0.890490	0.359524	0.891821	0.345136
9	0.892474	0.353638	0.889243	0.350936
10	0.894417	0.349862	0.893385	0.338778
11	0.895890	0.343603	0.892342	0.345478
12	0.896964	0.339727	0.895901	0.332182
13	0.897915	0.335937	0.896055	0.332073
14	0.898743	0.332773	0.895226	0.330169
15	0.899224	0.330968	0.896392	0.329542
16	0.899551	0.327125	0.896883	0.324876
17	0.900471	0.323797	0.897374	0.325638
18	0.900390	0.322332	0.894336	0.328991
19	0.901136	0.320008	0.893324	0.331572
20	0.901617	0.316167	0.896975	0.320841

Figure 4.6: Gradual accuracy with the increase of epochs(LSTM).

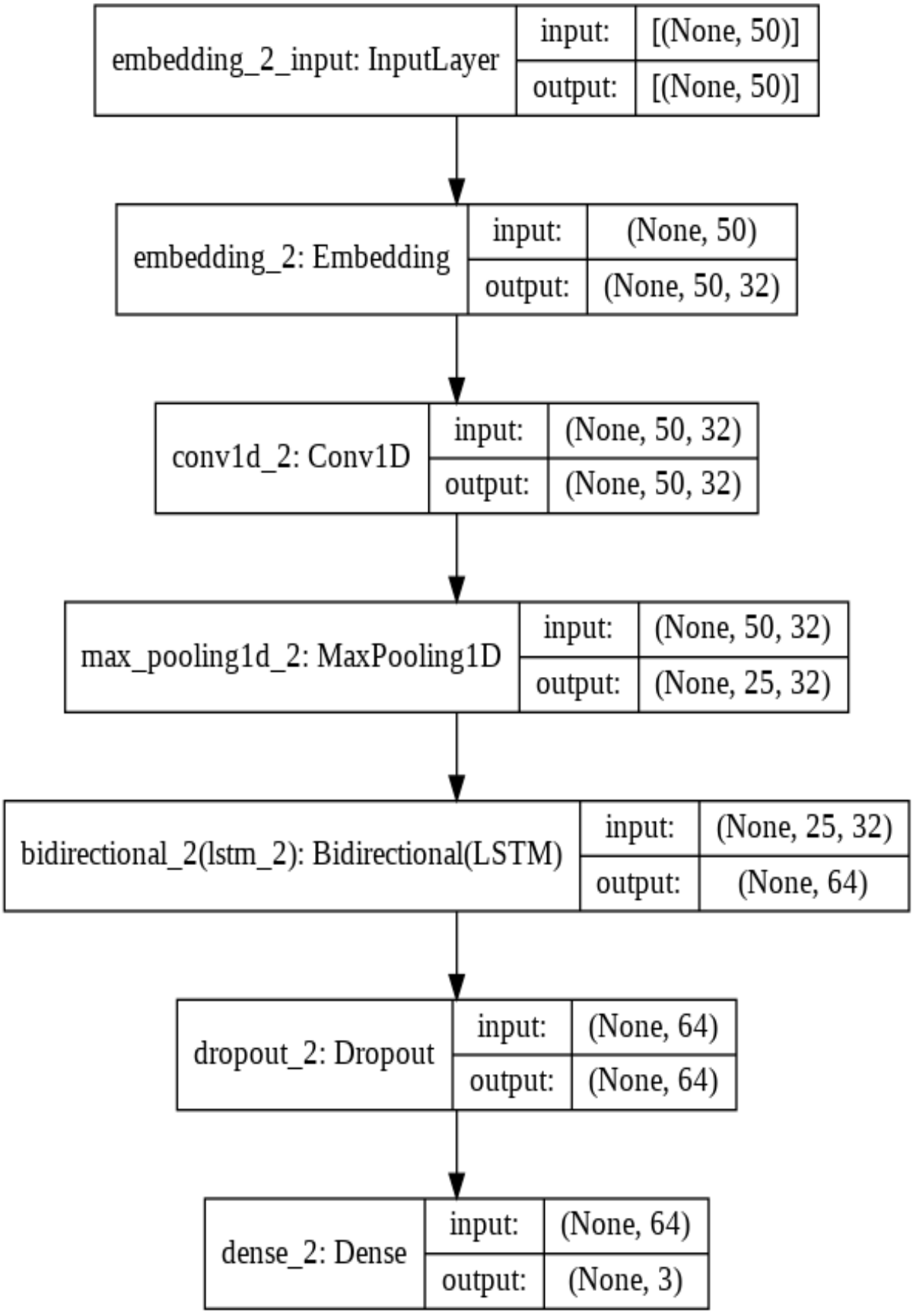

Figure 4.7: Bi-LSTM model flow diagram.

Epoch No.	Train Accuracy	Train Loss	Val Accuracy	Val Loss
1	0.554873	0.947039	0.621280	0.840133
2	0.731134	0.642027	0.791035	0.533381
3	0.811078	0.506662	0.840738	0.470172
4	0.839928	0.463416	0.852028	0.437951
5	0.856373	0.436633	0.860250	0.419388
6	0.865516	0.416663	0.869884	0.399717
7	0.872992	0.400598	0.876511	0.385553
8	0.876714	0.389503	0.881635	0.375642
9	0.880928	0.377238	0.881543	0.367664
10	0.884548	0.369677	0.885777	0.360180
11	0.886123	0.362458	0.887372	0.356062
12	0.888833	0.355601	0.889151	0.349806
13	0.890562	0.349832	0.888660	0.346716
14	0.891482	0.345990	0.890379	0.343751
15	0.892709	0.342237	0.886451	0.346922
16	0.894335	0.337752	0.892434	0.338295
17	0.895430	0.335380	0.891054	0.339176
18	0.896370	0.331705	0.893140	0.334980
19	0.896728	0.329321	0.893416	0.333527
20	0.897792	0.327176	0.892588	0.333950

Figure 4.8: Gradual accuracy with the increase of epochs(Bi-LSTM).

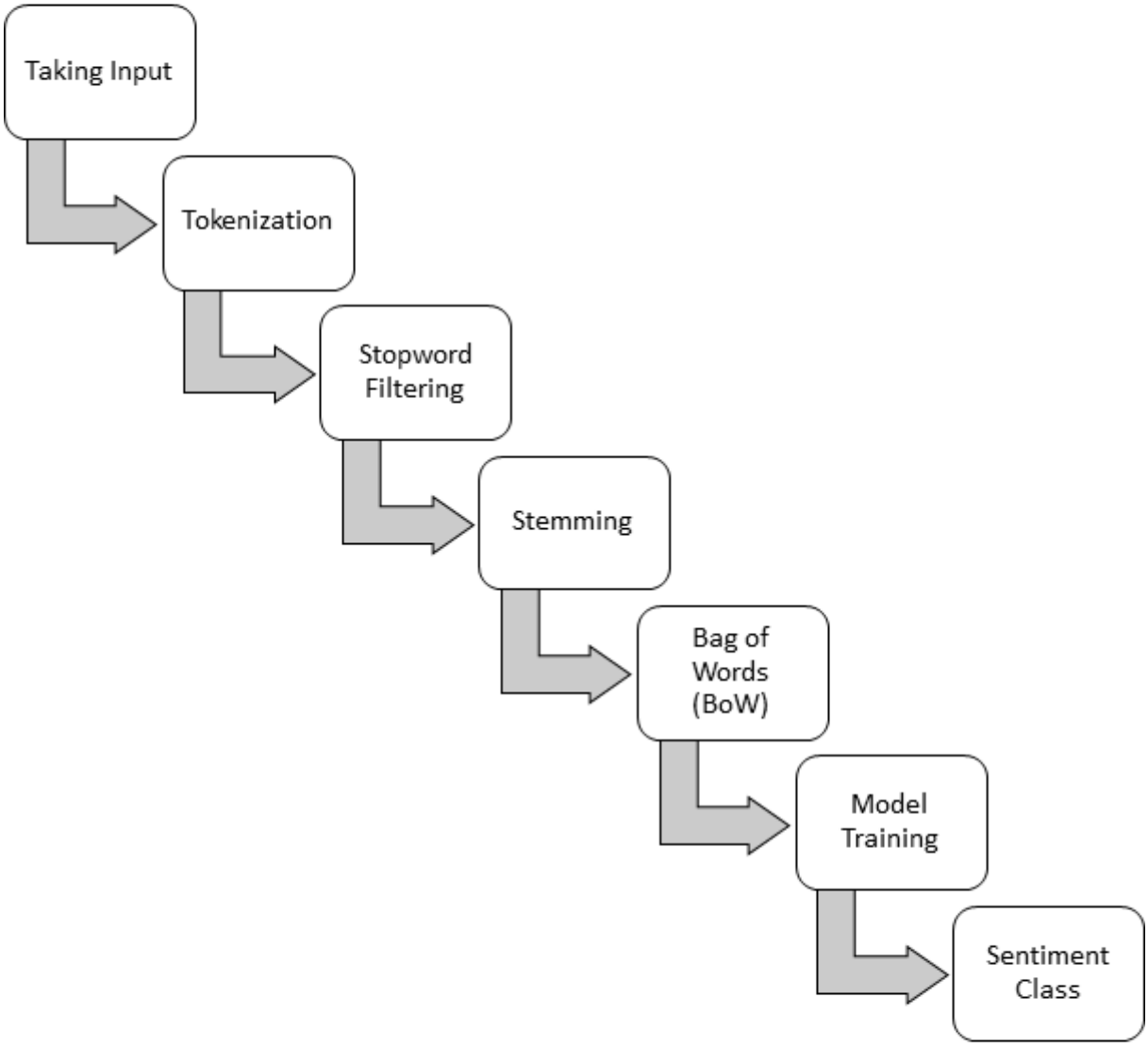

Figure 4.9: Diagram of the whole process.

Chapter 5

Experiments

5.1 Dataset

For our proposed project, we have collected the dataset from Kaggle which is a Twitter dataset. It contains two columns(clean text and category) having 162980 clean texts. This is a dataset for ternary sentiment classification containing substantially more data than other benchmark datasets. The dataset contains 72250 positive(1) texts, 35510 negative(-1) texts and 55213 neutral(0) texts.

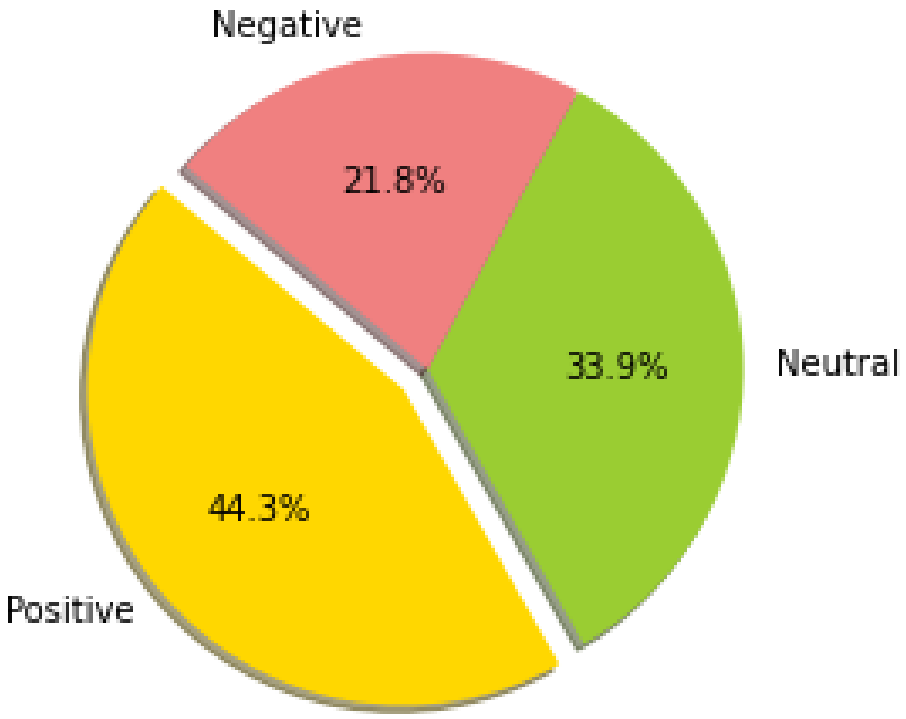

Figure 5.1: Pie chart visualization of the three types clean text

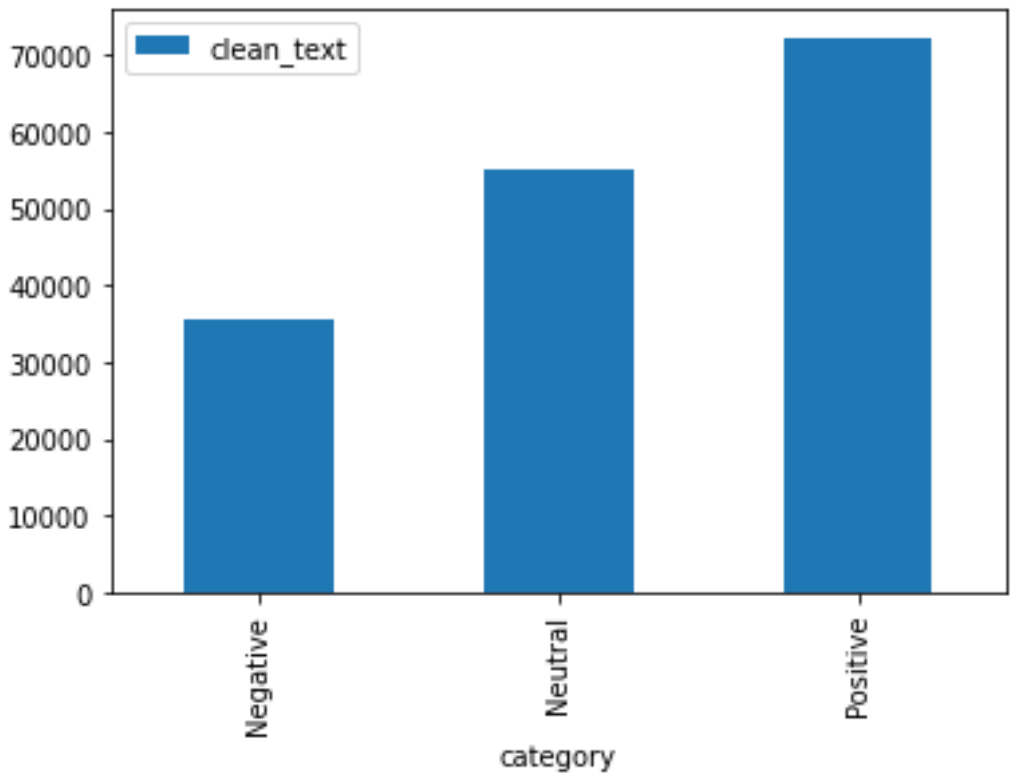

Figure 5.2: Bar chart visualization of the three types clean text

The dataset has not been preprocessed. The unprocessed and unstructured dataset can be easy to understand by humans but machine learning algorithms need structured data to perform well. We have to extract features from the dataset to make it favorable for several classification algorithms.

Table 5.1: Twitter dataset(Sample)

Clean Text	**Category**
what did just say vote for modi welcome bjp told you rahul the main campaigner for modi think modi should just relax	1
vote such party and leadershipwho can take fast and firm action none other than narendra damodardas modi and bjp party	-1
this comes from cabinet which has scholars like modi smriti and hema time introspect	0

For making an independent test set to verify the performance of your model, we have to split the dataset into a test and train the dataset. In our project, we've splitted our dataset where 80% data is for training and 20% data is for testing purposes. So, the ratio of train and test data is 0.8:0.2.

5.2 Evaluation Metric

To visualize important predictive analytics like recall, f1-score, accuracy, and precision, and evaluate the proposed system; we've used a confusion matrix. Confusion matrices are useful because they give direct comparisons of values like True Positives, False Positives, True Negatives and False Negatives.

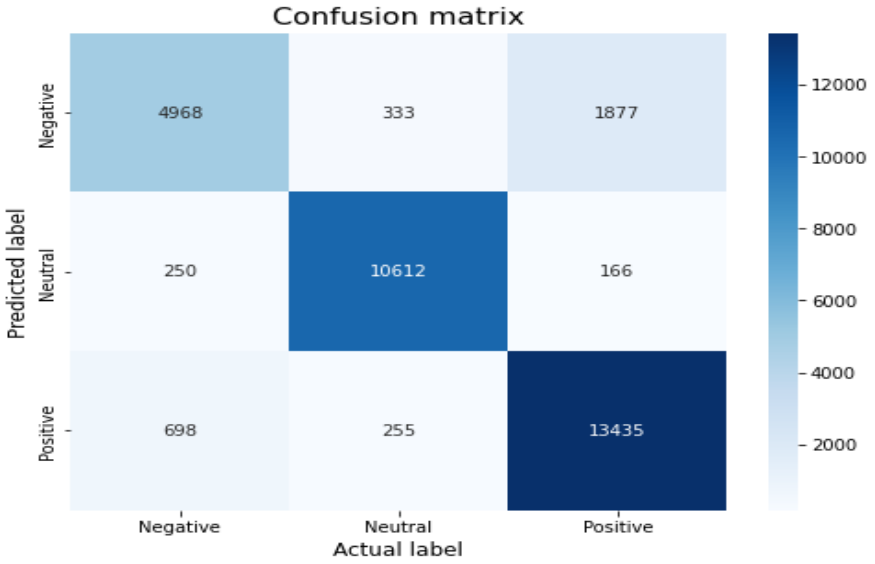

Figure 5.3: Confusion metric after applying Bidirectional LSTM model.

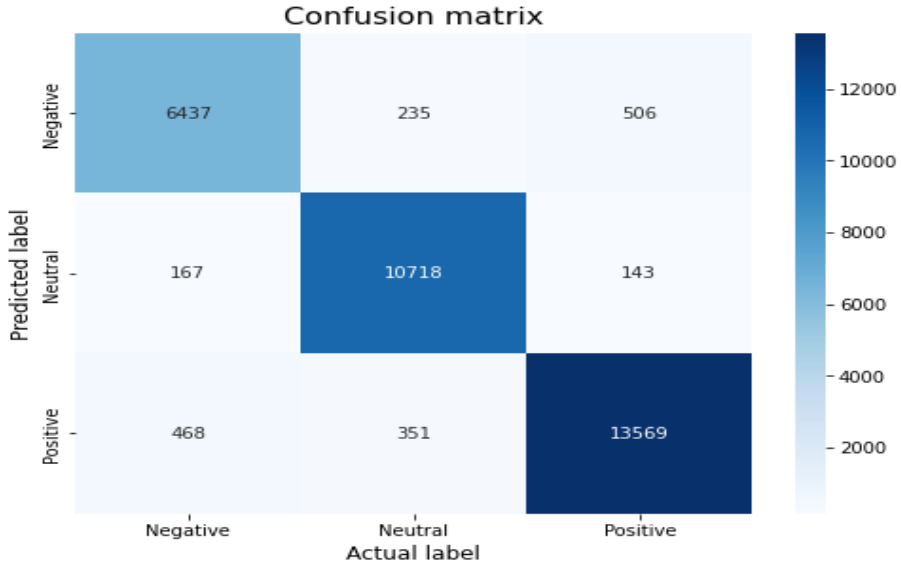

Figure 5.4: Confusion metric after applying Bidirectional RNN model.

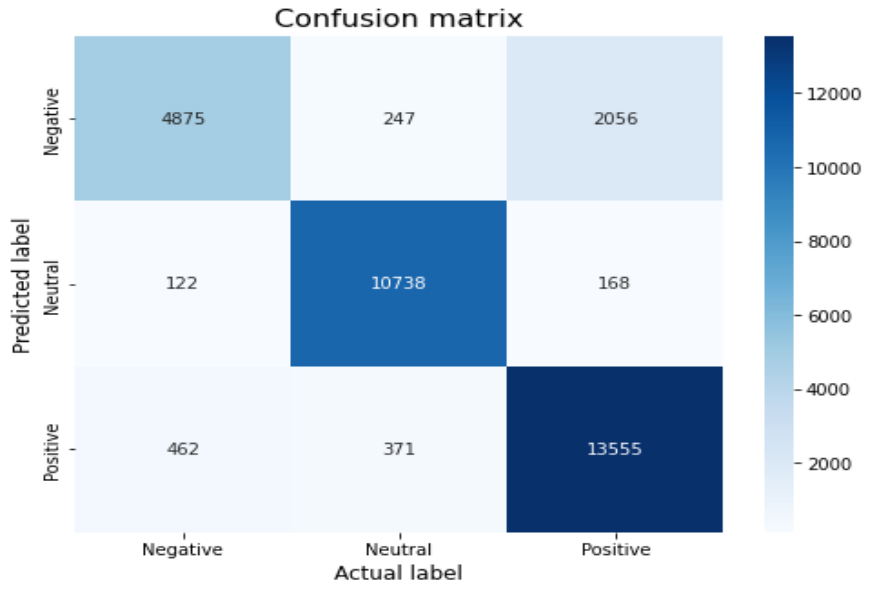

Figure 5.5: Confusion metric after applying LSTM model.

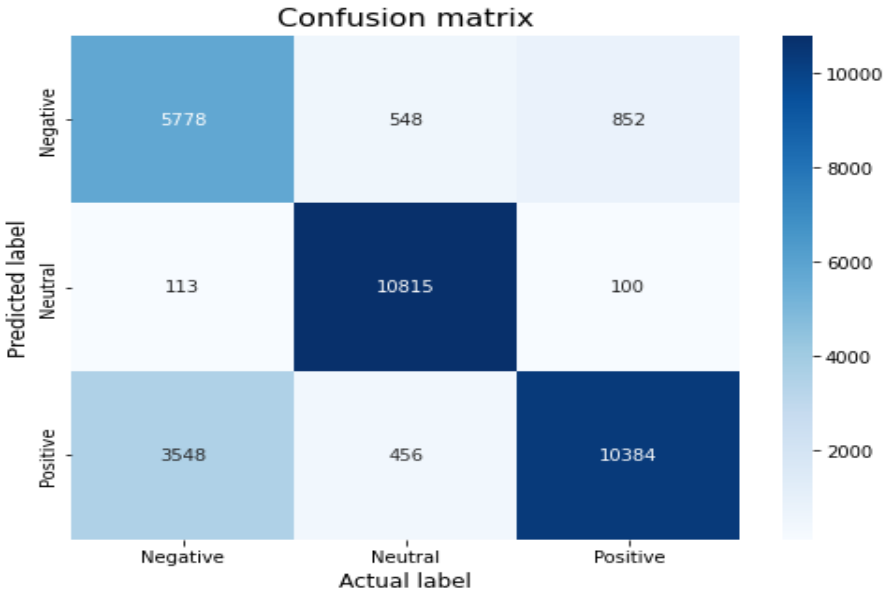

Figure 5.6: Confusion metric after applying CNN model.

5.3 Results

In this section, we present the results for our four deep learning models which are: Convolutional Neural Network (CNN), Bi-directional RNN, Long Short Term Memory (LSTM) and Bi-directional LSTM. The results after applying the models on processed data are shown in the following table.

From this table, we can see that the Bidirectional RNN gives the best result among all models. Bidirectional LSTM and LSTM performed equally well comparatively better than CNN.

Table 5.2: Results after applying deep learning algorithms on data

Model	Accuracy	Precision	Recall	F1 score
Bi-LSTM	89%	89.3%	88.7%	89%
Bi-RNN	94.3%	95%	93.4%	94.2%
LSTM	89.5%	89.8%	89.3%	89.5%
CNN	82.7%	83.1%	82.5%	82.8%

The accuracy and loss curves after applying on our models are shown below:

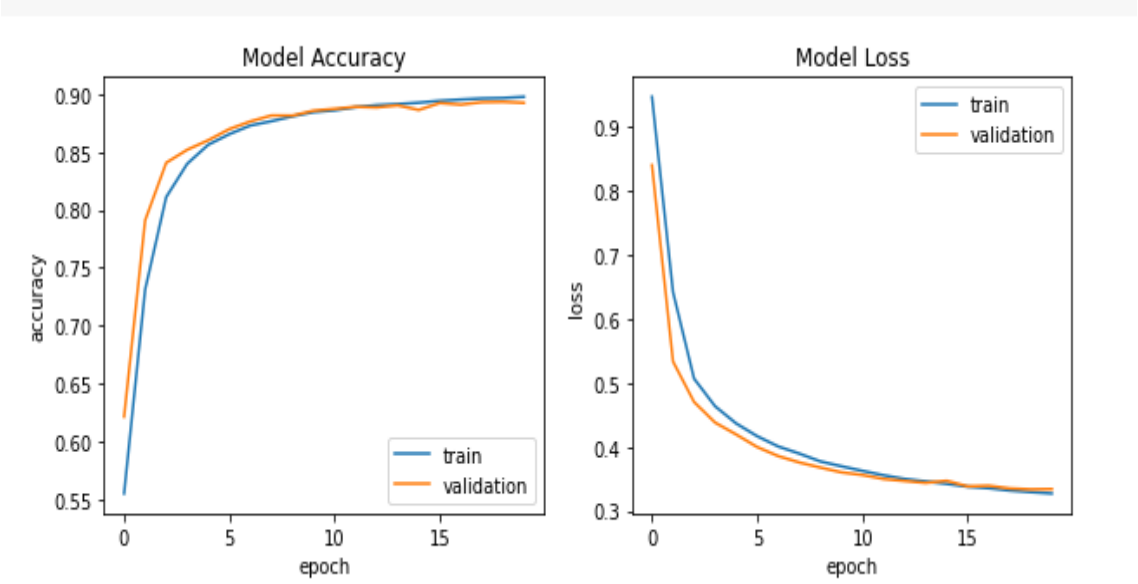

Figure 5.7: Bidirectional LSTM Model Accuracy and Loss curves.

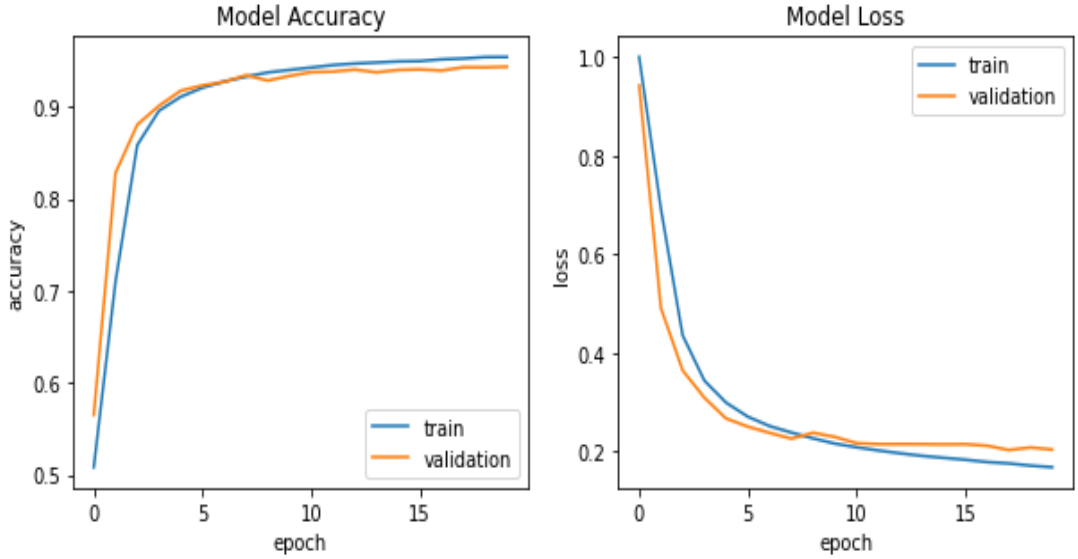

Figure 5.8: Bidirectional RNN Model Accuracy and Loss curves.

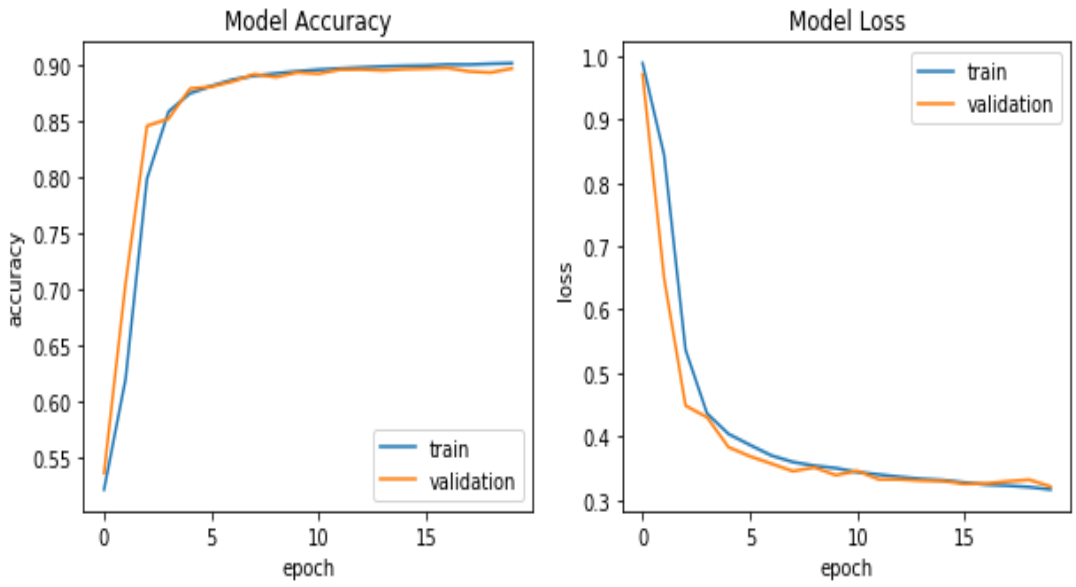

Figure 5.9: LSTM Model Accuracy and Loss curves.

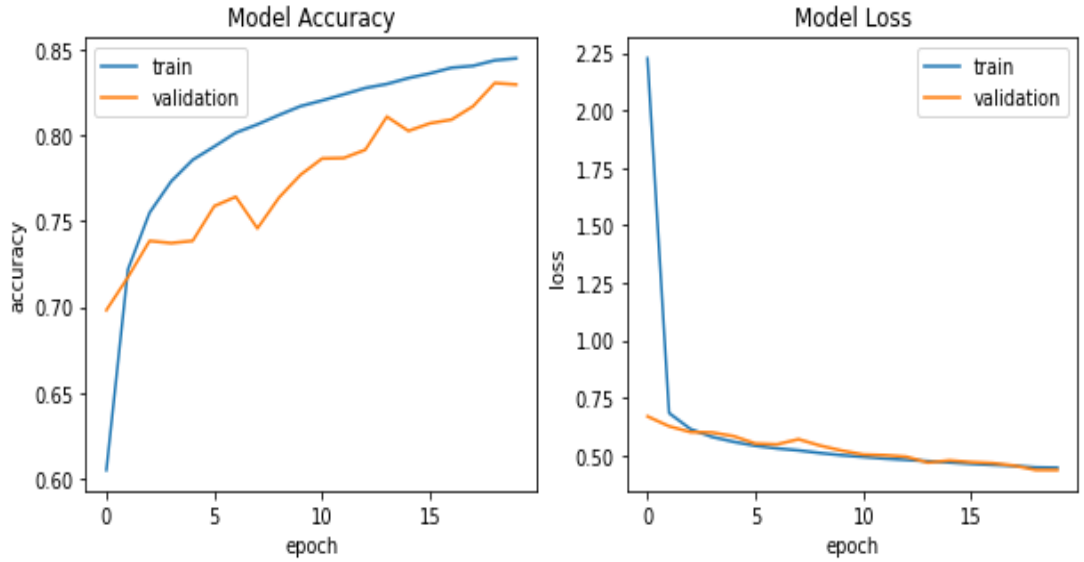

Figure 5.10: CNN Model Accuracy and Loss curves.

Taking a look at the training curves above, it appears to be that the model's preparation is working out in a good way where the loss is decreasing significantly.

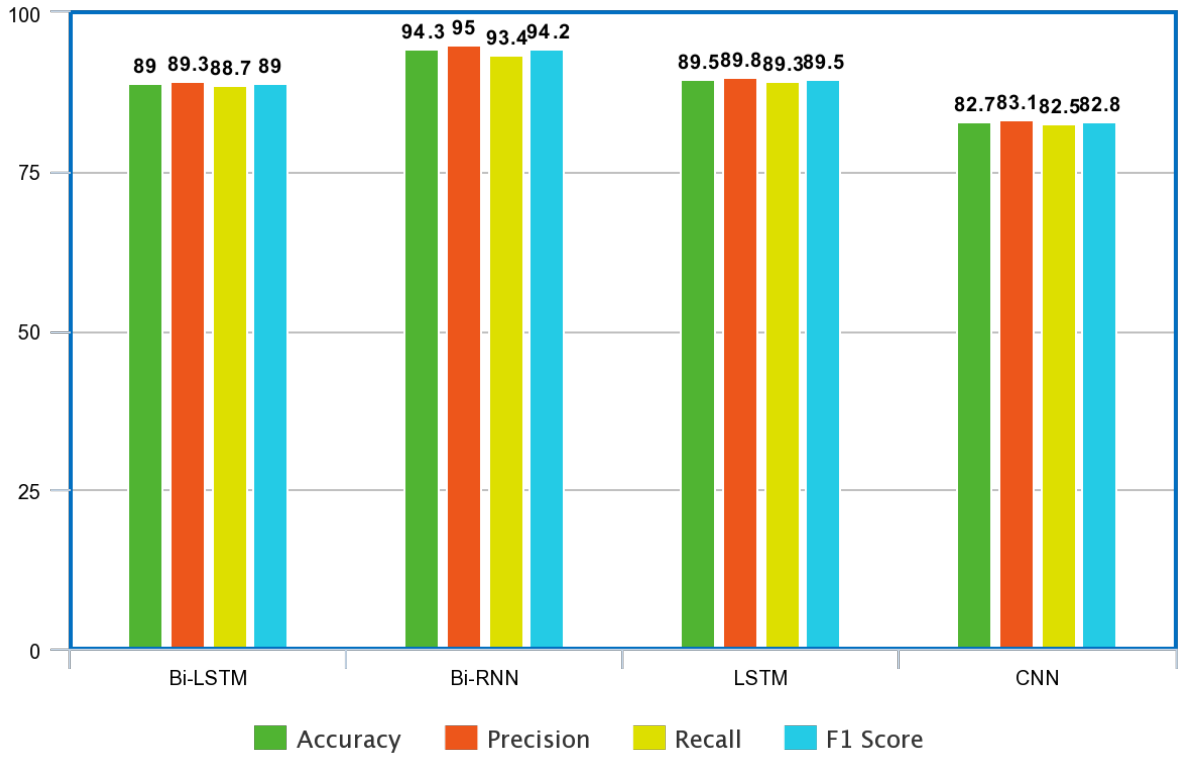

Figure 5.11: Bar chart visualization of our results.

Chapter 6

Conclusion and Future Work

Twitter sentiment analysis is developed to analyze customers' perspectives toward the critical to success in the marketplace. The program is using a deep learning approach which is more accurate for analyzing a sentiment; together with natural language processing techniques will be used. As a result, the program will categorize sentiment into positive, negative, and neutral which is represented in a pie chart and bar chart. So, we've presented results for sentiment analysis on Twitter and gained the highest 94.3% accuracy from Bi-LSTM.

In future work we can make a powerful stride in improving the accuracy and precision results. A strategy to anticipate the area and client's data of specific tweets however mysterious can be distinguished later on. It is proposed to stream continuous live tweets from twitter utilizing Twitter API, and to perform opinion investigation on numerous different dialects. This will assist with improving any investigation on any web based media particularly Twitter for this situation.

References

[1] Shobana, J. and Murali, M. An efficient sentiment analysis methodology based on long short-term memory networks. , Complex & Intelligent Systems. (2021)

[2] Sentiment Analysis: The Go-To Guide. . [Online]. Available: https://monkeylearn.com/sentiment-analysis/. [Accessed: 02-Oct-2021]

[3] Repustate Inc Sentiment Analysis Challenges: Everything You Need to Know. . 11-Jun-(2021) , Repustate

[4] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining.," in LREc, vol. 10, pp. 1320–1326, 2010.

[5] R. Parikh and M. Movassate, "Sentiment analysis of user-generated twitter updates using various classification techniques," CS224N Final Report, vol. 118, 2009.

[6] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," CS224N project report, Stanford, vol. 1, no. 12, p. 2009, 2009.

[7] L. Barbosa and J. Feng, "Robust sentiment detection on twitter from biased and noisy data," in Coling 2010: Posters, pp. 36–44, 2010.

[8] A. Bifet and E. Frank, "Sentiment knowledge discovery in twitter streaming data," in International conference on discovery science, pp. 1–15, Springer, 2010.

[9] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. J. Passonneau, "Sentiment analysis of twitter data," in Proceedings of the workshop on language in social media (LSM 2011), pp. 30–38, 2011.

[10] D. Davidov, O. Tsur, and A. Rappoport, "Enhanced sentiment learning using twitter hashtag and smileys," in Coling 2010: Posters, pp. 241–249, 2010.

[11] P.-W. Liang and B.-R. Dai, "Opinion mining on social media data," in 2013 IEEE 14th international conference on mobile data management, vol. 2, pp. 91–96, IEEE, 2013.

[12] P. Gamallo and M. Garcia, "Citius: A naive bayes strategy for sentiment analysis on english tweets," in Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014, Citeseer, 2014.

[13] P. D. Turney, "Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews,"arXiv preprint cs/0212032, 2002.

[14] J. Kamps, M. Marx, R. J. Mokken, M. De Rijke, et al., "Using wordnet to measure semantic orientations of adjectives.," in LREC, vol. 4, pp. 1115–1118, Citeseer, 2004.

[15] R. Xia, C. Zong, and S. Li, "Ensemble of feature sets and classification algorithms for sentiment classification," Information sciences, vol. 181, no. 6, pp. 1138–1152, 2011.

[16] Z. Luo, M. Osborne, and T. Wang, "An effective approach to tweets opinion retrieval," World Wide Web, vol. 18, no. 3, pp. 545–566, 2015.

[17] Dos Santos C N and Gatti M 2014 Deep Convolutional Neural Networks form Proceedings of COLING 2014, the 25th Int. Conf. on Computational Linguistics: Technical Papers (Dublin, Ireland: Dublin City University and Association for Computational Linguistics) pp 69–78

[18] Ali N M, El Hamid M M A and Youssif A 2019 Sentiment analysis for movies reviews dataset using deep learning models Int. J. of Data Mining & Knowledge Management Process 09 pp 19–27

[19] Jianqiang Z, Xiaolin G and Xuejun Z 2018 Deep Convolution Neural Networks for Twitter sentiment analysis IEEE Access 6 pp 23253–60

[20] Wang X, Jiang W and Luo Z 2016 Combination of Convolutional and Recurrent Neural Network for sentiment analysis of short texts Proc. of COLING 2016 The 26th Int. Conf. on computational linguistics: Technical papers pp 2428–37

[21] [No title]. . [Online]. Available: https://iopscience.iop.org/article/10.1088/1757-899X/1077/1/012001/pdf?fbclid=IwAR01p2cCm0TqnVzm0ay9euPkLnsSZI1ZmyALo0XocY8UBRXwYXRRbRSEPI0. [Accessed: 02-Oct-2021]

[22] https://medium.com/analytics-vidhya/bi-directional-rnn-basics-of-lstm-and-gru-e114aa4779bb

[23] https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

[24] https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/

Generated using Undegraduate Thesis LaTeX Template, Version 1.4. Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh.

This project report was generated on Sunday 3rd October, 2021 at 5:40pm.