**Aspect Based Sentiment Analysis on Romanized Nepali Restaurant Text Reviews**

**A Major Project Report Submitted in the Partial Fulfillment**

**of**

**the Requirements for the Degree of Bachelor in Information Technology Engineering**

**at**

**Everest Engineering College Sanepa, Lalitpur**

**Affiliated to Pokhara University**

**by**

**Adarsha Wagle (20120001)**

**Amir Poudel (20120003)**

**Dikshya Bhujel (20120017)**

**Kamal Lamichhane (20120027)**



**2024**

**DECLARATION**

We hereby declare that the report of the project entitled “Aspect Based Sentiment Analysis on Romanized Nepali Restaurant Text Reviews” which is being submitted to the Department of Computer and Information Technology Engineering, Everest Engineering College, Sanepa, in the partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in Information Technology Engineering, is a bonafide report of the work carried out by us. The materials contained in this report have not been submitted to any University or Institution for the award of any degree and we are the only author of this complete work and no sources other than the listed here have been used in this word.

Adarsha Wagle (20120001) signature

Amir Poudel (20120003) signature

Dikshya Bhujel (20120017) signature

Kamal Lamichhane (20120027) signature

# CERTIFICATE OF APPROVAL

The   project   report   entitled   **“Aspect Based Sentiment Analysis on Romanized Nepali Restaurant Text Reviews”**,   submitted   by **Adarsha Wagle, Amir Poudel,  Dikshya Bhujel  and  Kamal Lamichhane** in partial fulfillment of the requirement for the Bachelor’s degree in Information Technology Engineering has been accepted as a bonafide record of work independently carried out by the group in the department.

|  |  |
| --- | --- |
| …………………………….. Name of the external examiner Designation  Affiliation | …………………………….. Name of the Supervisor Designation  Affiliation |
| …………………………….. Nischal Regmi  Head of Department  Department of Computer and IT Engineering | |

# COPYRIGHT

The author has agreed that the library, **Everest Engineering College (EEC)**, Sanepa, Lalitpur may make this report freely available for inspection. Moreover, the author has agreed that permission for extensive copying of this project report for scholarly purpose may be granted by the lecturers, who supervised the project works recorded herein or, in their absence, by the Head of Department wherein the project report was done. It is understood that the recognition will be given to the author of the report and to the Department of Information Technology, EEC in any use of the material of this project report. Copying or publication or other use of this report for financial gain without approval of the Department and author’s written permission is prohibited. Request for permission to copy or to make any other use of the material in this report in whole or in part should be addressed to the Head of Department, Department of Computer and Information Technology Engineering.

ACKNOWLEDGEMENT

We would like to express our deep and sincere gratitude to our final year project supervisor Er. Birod Rijal sir for all the guidance and help.

# ABSTRACT

For our final year project, we focused on **Aspect-Based Sentiment Analysis (ABSA)** for restaurant reviews. ABSA is a more granular form of sentiment analysis, which aims to extract specific aspects or features from textual data and analyze the sentiment associated with each of them. In the context of restaurant reviews, this means identifying aspects like **food, service, ambiance, pricing, and staff behavior**, and determining whether customers express positive, negative, or neutral sentiments towards these aspects.

To achieve this, we first built a **custom dataset** comprising restaurant reviews, where each review was annotated with the relevant aspects and corresponding sentiment labels. This dataset was created using a combination of manual annotation and automated techniques, ensuring a diverse set of reviews that cover multiple aspects of restaurant experiences.

The core of our project involved training and evaluating multiple machine learning and deep learning models. We implemented both traditional machine learning algorithms like **Multinomial Naive Bayes, Random Forest, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN)** as well as advanced deep learning architectures, including **Recurrent Neural Networks (RNN), Long Short-Term Memory networks (LSTM), Bidirectional LSTM (BiLSTM), and Convolutional Neural Networks (CNN)**. Additionally, we leveraged the **BERT-base** model for more sophisticated natural language understanding, given its ability to capture context-dependent meaning in text through its transformer architecture. The inclusion of BERT was motivated by its state-of-the-art performance on a wide range of NLP tasks.

Each of these models was trained on our custom dataset and optimized for sentiment classification and BERT-base was optimized for aspect extraction too. The traditional machine learning models rely on hand-engineered features such as **TF-IDF** and **bag-of-words** representations of the text, while the deep learning models automatically learn feature representations through embeddings and sequential processing. BERT, with its pre-trained contextualized embeddings, was fine-tuned on our specific dataset to better understand the domain of restaurant reviews.

We carried out extensive experiments to compare the performance of these models, using metrics like **accuracy, precision, recall, and F1-score**. One of the key goals of the project was to determine which model is most effective at handling aspect-based sentiment analysis in a real-world restaurant review setting. In particular, we wanted to see whether complex architectures like **LSTM** and **CNN** outperform simpler models like **SVM** and **Random Forest** on this task.

The results of our study provided insights into how different models capture the nuances of restaurant reviews. We found that while traditional machine learning models performed reasonably well, **deep learning models**, particularly **BERT** and **BiLSTM**, achieved higher accuracy and better generalization due to their ability to learn context and dependencies in the text. These models were particularly effective in cases where sentiment was implicit or context-dependent, such as when reviews mentioned aspects indirectly.

Our project also highlighted the practical importance of ABSA for the restaurant industry. By automatically identifying what customers like or dislike about specific aspects of a restaurant, business owners can gain actionable insights to improve their services. For example, if a restaurant consistently receives negative sentiment towards its service but positive sentiment for food quality, it can focus on improving staff training or customer service.

In conclusion, our work on ABSA using multiple machine learning and deep learning models demonstrates the potential of these technologies to provide detailed and actionable insights from customer reviews. Our experiments revealed that **transformer-based models like BERT** are highly effective at extracting aspect-level sentiments, outperforming traditional methods, and opening up new possibilities for businesses to make data-driven decisions based on customer feedback.

# Contents

[**Declaration**](#_bookmark0) **i**

[Approval](#_bookmark1) ii

[Ethics Statement](#_bookmark2) iv

[Abstract](#_bookmark3) v

[Dedication](#_bookmark4) vi

[Acknowledgement](#_bookmark5) vii

[Table of Contents](#_bookmark6) vii

[List of Figures](#_bookmark7) x

[List of Tables](#_bookmark8) xi

1. [Introduction](#_bookmark9) 1
   1. [Background](#_bookmark10) 1
   2. [Problem Statement](#_bookmark11) 1
   3. [Research Objective](#_bookmark12) 2
   4. [Research Outline](#_bookmark13) 2
2. [Related Work](#_bookmark14) 3
3. [Methodology](#_bookmark15) 8
   1. [Dataset Collection & EDA](#_bookmark17) 8
      1. [Data Annotation](#_bookmark18) 9
   2. [Data Preprocessing](#_bookmark23) 10
      1. [Removing StopWords](#_bookmark25) 10
      2. [Removing Punctuations](#_bookmark31) 13
      3. [Removing Low Length Data](#_bookmark34) 13
      4. [Word Distribution Statistics](#_bookmark38) 15
      5. [Train-Test Split](#_bookmark42) 16
      6. [Label Encoding](#_bookmark43) 16
      7. [Tokenization](#_bookmark45) 16
      8. [Vectorization](#_bookmark48) 17
   3. [Model Architectures](#_bookmark50) 19
      1. [Multinomial Naïve Bayes (MNB)](#_bookmark51) 19
      2. [K-Nearest Neighbor (KNN)](#_bookmark52) 19
      3. [Random Forest Classifier](#_bookmark53) 19
      4. [Kernel Support Vector Machine (KSVM)](#_bookmark55) 20
      5. [Recurrent Neural Network (RNN)](#_bookmark56) 21
      6. [Long Short-Term Memory Network (LSTM)](#_bookmark58) 22
      7. [Convolutional Neural Network (CNN)](#_bookmark60) 22
      8. [Gradient Recurrent Unit (GRU)](#_bookmark62) 23
      9. [BERT-base](#_bookmark63) 23
      10. [RoBERTA](#_bookmark65) 24
      11. [Custom Model Architecture](#_bookmark68) 25
   4. [Model Training](#_bookmark69) 26
      1. [Parameters of Different Models](#_bookmark70) 26
      2. [Large Language Models (LLMs)](#_bookmark78) 28
4. [Result & Analysis](#_bookmark81) 29
   1. [Evaluation Metrics](#_bookmark82) 29
   2. [Performance Evaluation](#_bookmark84) 30
   3. [Visualization](#_bookmark91) 32
   4. [Testing Best-Performing Model via Prompt](#_bookmark97) 34
   5. [Explainable AI (XAI)](#_bookmark99) 35
5. [Conclusion](#_bookmark101) 36

[Bibliography](#_bookmark102) 38

# List of Figures

* 1. [Research Concept Map](#_bookmark16) 8
  2. [Sentiment Frequency Bar Chart](#_bookmark22) 10
  3. [Data Preprocessing Concept Map](#_bookmark24) 10
  4. [An overview of the stopwords](#_bookmark26) 11
  5. [Word Cloud of Positive labels.](#_bookmark28) 12
  6. [Word Cloud of Negative labels.](#_bookmark29) 12
  7. [Word Cloud of Neutral labels.](#_bookmark30) 13
  8. [Regular Expression for removing punctuations](#_bookmark32) 13
  9. [Word Statistics of each class](#_bookmark35) 14
  10. [Length Frequency Distribution](#_bookmark36) 14
  11. [Statistics of Positive labeled classes](#_bookmark39) 15
  12. [Statistics of Negative labeled classes](#_bookmark40) 15
  13. [Statistics of Neutral labeled classes](#_bookmark41) 16
  14. [Tokenization Example](#_bookmark46) 17
  15. [TF-IDF Statistics of a Review](#_bookmark49) 18
  16. [Random Forest Classifier Workflow](#_bookmark54) 20
  17. [RNN Architecture](#_bookmark57) 21
  18. [LSTM Architecture](#_bookmark59) 22
  19. [CNN Architecture](#_bookmark61) 23
  20. [BERT Architecture](#_bookmark64) 24
  21. [RoBERTA Tokenization](#_bookmark66) 24
  22. [RoBERTA Architecture](#_bookmark67) 25
  23. [Confusion Matrix for Random Forest Classifier](#_bookmark92) 32
  24. [Confusion Matrix for Multinomial Naive Bayes](#_bookmark93) 32
  25. [Confusion Matrix for K-Nearest Neighbor](#_bookmark94) 33
  26. [Confusion Matrix for Kernel Support Vector Machine](#_bookmark95) 33
  27. [ROC Curves for ML models](#_bookmark96) 34
  28. [Prompt Results](#_bookmark98) 34
  29. [Sentiment Visualization of a review in waterfall plot](#_bookmark100) 35

# List of Tables

* 1. [Data Annotation Threshold](#_bookmark19) 9
  2. [Overview of the Dataset](#_bookmark20) 9
  3. [Dataset Description](#_bookmark21) 10
  4. [Overview of Stopwords length](#_bookmark27) 11
  5. [Original vs Cleaned data](#_bookmark33) 13
  6. [Max, Min & Average Length](#_bookmark37) 15
  7. [Label Mapping](#_bookmark44) 16
  8. [Tokenization Parameter for Deep learning models](#_bookmark47) 17
  9. [Random Forest Classifier Parameters](#_bookmark71) 26
  10. [Multinomial Naïve Bayes Parameters](#_bookmark72) 26
  11. [K-Nearest Neighbors Parameters](#_bookmark73) 26
  12. [Kernel Support Vector Machine Parameters](#_bookmark74) 27
  13. [LSTM and GRU Parameters](#_bookmark75) 27
  14. [1D-CNN Parameters](#_bookmark76) 27
  15. [BERT Parameters](#_bookmark77) 28
  16. [RoBERTa Specifications](#_bookmark79) 28
  17. [Custom Model Specifications](#_bookmark80) 28
  18. [Evaluation weights](#_bookmark83) 29
  19. [Accuracy of ML Models](#_bookmark85) 30
  20. [Precision of ML Models](#_bookmark86) 30
  21. [Recall of ML Models](#_bookmark87) 30
  22. [F1 Score of ML Models](#_bookmark88) 31
  23. [Accuracy of DL Models on Test Dataset](#_bookmark89) 31
  24. [Loss of DL Models on Test Dataset](#_bookmark90) 31

**Chapter 1 Introduction**

## Background

In the modern digital landscape, **customer reviews** have become a vital source of information for consumers and businesses alike. For restaurants, these reviews offer crucial feedback on various elements of the dining experience, including the quality of food, level of service, ambiance, and pricing. With the increasing volume of reviews available online, it is essential to have efficient methods to analyze and extract meaningful insights from this data. In Nepal, a significant number of reviews are written in **Nepali Romanized Text** —where Nepali speakers use the Latin alphabet to write their native language. This form of Romanized text presents unique challenges for traditional sentiment analysis tools, which are often designed for standard languages and scripts.

**Aspect-Based Sentiment Analysis (ABSA)** is a sophisticated technique that dissects reviews into specific aspects and assesses the sentiment associated with each. Unlike conventional sentiment analysis, which provides a general sentiment score, ABSA offers a more granular view by focusing on individual components of the review. This method allows for a comprehensive understanding of customer opinions, helping restaurant owners identify specific areas for improvement.

## Problem Statement

The characteristics and grammar have been ignored by most of the papers that conducted sentiment analysis in other languages. These existing models may or may not perform well in the context of Bangla language. The existing models will most probably fail to identify the unique features and patterns of Bangla language. The reason we need a specific model to analyze and predict the sentiments of consumers is to improve the food quality and market research. This can make a huge impact on the businesses running inside the food and restaurant industry. As the public’s point of interest is steered by the policymakers, it is important to understand their

expression and sentiment toward valued goods. However, in the broader context, sentiment analysis research on Bangla language can help other researchers and for further works conducted in Bangla language in the future.

In this study, we aim to create a real time system that will be able to predict a customer’s sentimental expression from Bangla text in the context of the food and restaurant industry by using deep learning models. This can benefit the local businesses to better understand the demands of the consumers and expand their business territory. As a large number of consumers are not satisfied with the service that they receive from the vendors, with our machine learning and deep learning approach, we intend to benefit the food industry from our conducted research.

## Research Objective

This study aims to predict the emotion of an individual from Bangla text using machine learning and deep learning methods. The research objectives are following:

* + - To develop a comprehensive, high-quality annotated dataset on Bangla text.
    - Desigining machine learning and deep learning models that understands the unique characteristics of the Bangla language corpus, evaluates the sentimental nuances of Bangla words in context. These deep learning models can be used in market research and analytics.
    - This study specifically focuses on Ml and DL algorithms to analyze the sen- timents in Bangla text from customer reviews in social media food blogs and reviews.
    - Optimizing the models which includes hyperparameter tuning and other re- quired adjustments.
    - Providing practical implications such as market research, product quality im- provement etc.

## Research Outline

Our study’s significant goal is to provide actionable insights for enhancing product quality and refining market research strategies in the culinary domain.

**Chapter 1:** This chapter elucidates the objectives of our research and motivating factors that drive the continuation of our work. Additionally, we elaborate on the procedural measures adopted to carry out an investigative apporach to problem- solving.

**Chapter 2:** This chapter includes the related work part which consists of the reviews of the previous work that are related to our study.

**Chapter 3:** This chapter includes the steps of our research work and the details about it.

**Chapter 4:** This chapter shows the result and the analytics behind the models and their outcomes.

**Chapter 5:** This chapter concludes our research and opens up new doorways to further improvement of our study.

# Chapter 2

# Literature Review

The section on Related Work critically reviews prior research in the field, offering a comprehensive overview of existing methodologies and findings. This contextual analysis serves to identify gaps and set the stage for the unique contributions of our study.

In the research described in another paper by D. Strezoski et al. [[1],](#_bookmark103) The author employs a deep convolutional neural network to conduct tests on sentiment analysis in Twitter conversations. The network is trained using word embeddings that have already undergone unsupervised learning on big text corpora. The author uses CNN with many filters with different window widths, and then adds two fully connected layers with dropout and a softmax layer on top of those layers. This study shows that utilizing pre-trained word vectors and Twitter corpora for unsupervised learning is effective and beneficial. The experimental evaluation is based on the standard datasets for the Sentiment analysis in the social network challenge from the 2015 competition of SemVal.

In this paper, M. Hassan et al. [[2]](#_bookmark104) went on to a different route to prove the sufficiency of classical ML algorithms using NLP in sentiment classification of textual data and attempted to demonstrate classical ML algorithms are alone sufficient for the tasks highlighted. Dataset was created rather than selected by a post-processed source and the main source of the data were from Bangla movie and telefilm scripts that contained dialogues and conversations. Words were first preprocessed and trans- formed to be model-ready, categorized into tokens of word streams with two main labels positive and negative which further sub-classified into six sentiment emo- tions. Naive-Bayes, Random-Forest, Support-Vector machines were implemented and trained on the trained samples and tested for accuracy and Random Forest was observed to work best for the task. This paper addressed the limitations of classical ML and referenced a deep learning implementation for an improvement and future prospect.

In this paper by Corcuera-Platas, I. et al [[3],](#_bookmark105) the authors contend that deep learn- ing techniques outperform conventional approaches in terms of automated feature extraction and performance. They propose merging deep learning techniques with standard surface approaches since they recognize that these can offer robust base- lines (enhancing deep learning). Furthermore, the stages to our goal involve the introduction of two ensemble algorithms that integrate the baseline classifier with other surface classifiers commonly employed in sentiment research. Last but not least, statistical research shows that the suggested models perform better than the

initial baseline in case of F1-Score.

In this study, M. Heikal et al. [[4]](#_bookmark106) have developed an ensemble model that combines LSTM and CNN models that read Arabic tweets and predicts sentiment. This re- search focuses on the sentence-level sentiment analysis of Arabic tweets to identify the direction of the tweet, such as if it is positive, negative, or neutral. It doesn’t need any further feature engineering because it employs pre-trained word embed- dings.Firstly, the tweets go through a preprocessing and cleaning stage first to get rid of extraneous symbols and tokens. After that, the tweets are ready for the train- ing phase. A CNN model and an LSTM model are two deep learning models that they have trained. They chose the top two models that have the greatest F1-score after training both models with various hyper-parameters and created an ensemble model using those models. To forecast the ultimate emotion of the incoming tweets, they have followed the ensemble model.

In this study, A. Soumeur et al. [[5]](#_bookmark107) intended to create an automated analyzer of the feelings of Facebook users of Algeria, or Facebook page users who communicate using the Algerian Dialect (AlgD). They chose to test both shallow and deep learning, the latter of which needed a lot of annotated data, which meant they had to go through a collecting and annotation phase, followed by a preprocessing phase, before they could go on to the learning phase.Besides, from more than 25000 sentiment-annotated comments, researchers created a corpus. In order to assess each pre-processing stage, they have decided to construct the Naive Bayes classifier. By adjusting the number of layers, the number of neurons on each layer, and the activation functions, researchers explored a wide range of topologies for MLP neural networks.

In this study, W. Souma et al. [[6]](#_bookmark108) have used the 2014 articles of Gigaword and Wikipedia five corpus, word representation technique for global vectors, is applied to this large text in intention to generate word vectors that are then fed into the deep learning network library called Tensorflow. They look into the news archive called Thomas Reuters News’ intraday high-frequency and the historical high-frequency price tick data of individual stocks in the Dow Jones Industrial Average (DJIA 30) Index throughout that period of time. A permutation of RNNs and LSTMs’ units were employed, two deep learning techniques, to train their model from 2003 to 2012, using the same news archive data. Next, they evaluated their method’s predictive ability using data from the 2013 News Archive. When they move from randomly picking good and the news with the highest positive ratings being classified as positive news and the news with the highest negative scores being classified as negative news in order to create the training data set, the forecasting accuracy of their approach increases.

In this study, Tanvirul et al. [[7]](#_bookmark109) conducted a study to classify Bangla text with the use of Transformers using the data collected from multiple datasets including YouTube Sentiment, YouTube Emotion, News Comment Sentiment, Authorship At- tribution, News Classification. It showed a comparison between different algorithms and concluded that XLM-RoBERTa-large works better than other common machine learning algorithms. It demonstrated that using transformer models for fine-tuning can result in superior performance when compared to both deep learning models like CNN and LSTM trained on scattered word representation and conventional techniques that employ hand-crafted features.

In another paper by K. Zheng et al [[8],](#_bookmark110) the main area of the study that it focused on is the sentiment analysis of IMDB movie reviews using three deep learning net-

works. The dataset utilized in the study is equally split between reviews that are 50 percent favorable and 50 percent negative. While CNN is frequently used for image recognition, RNN and LSTM are frequently employed in NLP applications. The outcomes show that when used for sentiment analysis of movie reviews, the CNN network model produces good classification results. However, the accuracy rates for the RNN and LSTM models are 68.64 percent and 85.32 percent, respectively.

In another paper done by K. Chakraborty et al. [[9],](#_bookmark111) it focuses on analyzing how tweets about COVID-19 and the World Health Organization (WHO) helped the general people throughout the epidemic by offering advice and information. In the study, two different types of twitter datasets are examined. In comparison to the initial dataset, this study finds more supportive and neutral tweets published by internet users. To back up these claims, the research recommends a deep learning- based strategy using classifiers that achieve accuracy rates of up to 81 percent. The research also discusses the usage of a fuzzy rule base with a gaussian membership function, which has an accuracy rate of up to 79 percent, to precisely identify sen- timent from tweets.

This study conducted by N. Cach Dang et al [[10],](#_bookmark112) here it talks about the significance of analyzing public opinion and how sentiment analysis, particularly on social media platforms like Twitter and Facebook, may offer insightful data. Natural language processing (NLP) presents difficulties for sentiment analysis in terms of accuracy and efficiency. Moreover, it focuses on analyzing recent research that has used deep learning methods to challenge sentiment analysis, particularly polarity. Word embeddings and TF-IDF models were used to a range of datasets in these studies. In this paper, O. Habinaman et al. [[11]](#_bookmark113) used BERT model, cognition focused atten- tion models and sentiment domained word embedding models, models of common sense, reinforcement learning, and GANs are a few models of the paper.Unsuper- vised pre-trained UPNs, CNNs, RNNs, RvNNs, DRL, and neural networks of other hybrid structures are the six categories into which this study divides deep learning techniques. They draw the readers’ attention to the fact that in this review they focus primarily on two novel deep-learning methodologies: GANs and DRLs. Re- searchers have used UPNs, which is the subset of deep neural networks that allows them to work with the unsupervised algorithm in order to train the layers of unla- beled data. In short, they have shown some demanding issues that are trending and provided some solutions using the above models.

In a different paper, M.R Amin et al. [[12],](#_bookmark114) a comparative study was performed on Sentimental classification on Bangla textual content to compare and contrast the performances of both classical and deep learning algorithms. Dataset was taken from ABSA and BengFastext; It was analyzed, and further tokenized into sets, to identify patterns of lexical content for positive and negative sentiment and experi- ments were carried out in 75:25 train-test-split ratio and trained on CNN, FastText, Transformer Models and RF. Across all the experiments, deep learning models per- formed significantly better than classical counterparts with an average 76 percent accuracy which therefore created a clearly visible distinction between the six differ- ent classified sentiment labels.

Another study done by A. Daudpota et al [[13].,](#_bookmark115) aspires to investigate how people from five different cultural backgrounds reacted to the novel coronavirus and their opinions on the subsequent actions taken by different countries in response. Deep LSTM neural network models—a replacement for the RNN—have been trained to

achieve state-of-the-art accuracy on the ”sentiment140” dataset. These models are applied to analyze the polarity of sentiment and sentiments from extracted tweets. The supervised deep learning models on the recovered Twitter tweets were evaluated in a unique and creative way with the aid of emoticons.

In this paper, Moqsadur et al. [[14]](#_bookmark116) conducted research using Deep Learning tech- niques which identifies and categorizes opinions expressed in Bangla Sentences. The data was collected from a news portal containing more than 24 newspapers where Prothom Alo had a huge collection of visitor’s comments. CNN outplayed the other models. The models were also applied on a dataset of Hindi language and it was found that it performed exceptionally well on the Hindi language dataset.

In this paper, A. Yadav et al. [[15]](#_bookmark117) showed a review of deep learning models broadly. They presented a taxonomy for sentiment analysis and discussed how common deep learning architectures affect it. The prominents deep learning classifiers architec- tures were covered. Additionally, researchers have given a brief overview of three current research trends: capsule networks, bi-directional RNNs, and attention-based networks. Additionally, they have highlighted tasks requiring sentiment analysis and spoken about the deep learning models used to do them.

In this paper, N. R. Bhowmick et al. [[16]](#_bookmark118) conducted a study to analyze sentiments on Bangla texts using supervised machine learning with Extended Lexicon Dictionary. The authors made a sentimental dictionary list from a dataset manually created from collected cricket and restaurant data. A few deep learning and machine learning approaches were considered for this study and among them, BiGram features matrix achieved an accuracy of 82.21 percent which was far better than the other models on both the datasets. The study can be continued further if a high volume of sentimental dictionary is constructed.

In this paper by H. Gelbukh et al. [[17],](#_bookmark119) it made an outline of two main goals as follows: Create a benchmark dataset for the resource-constrained Urdu language for sentiment analysis, and then test out several ML and deep learning techniques. The author compares two text representation methods in order to determine which is the most effective: count-based, which uses word n-gram feature vectors to represent the text, and pre-trained Urdu word embeddings in fastText. The author highly appre- ciated a collection of deep learning models and ML models. The study demonstrates that the word n-gram feature combination with LR outperformed other classifiers for the sentiment analysis task.

In this paper S. Shereen et al. [[18],](#_bookmark120) the author suggests categorizing a large number of tweets according to their emotion. Here, the author applies deep learning algorithms to categorize expressing feelings as either positive or negative. In order to get better accuracy in sentiment classification, the author experimented with and assessed the strategy employing RNN and LSTM on three separate datasets. A report shows that the system excels at its positive or negative classification in the LSTM model taking an accuracy of 86.56 and 90.20 and 89.23 percent accuracy for the subclasses. In the paper by N.R Bhowmick et al. [[19],](#_bookmark121) it was discussed that in the case of “rule-based sentimental score generation” and nominal based weighted dictionary, works on sentimental analysis was still an unexplored paradigm especially using Bengali text and deep-learning approaches. This paper approaches this issue by proposing an extended lexicon data dictionary to create deep learning models for Bengali SA. To extract polarity from a large set of texts, BTSC algorithm was used to filter out the polarity and then those are fed to NN models along with the training

samples, finally vectorized into chunks of unique numbers. This paper walks through a detailed analysis of selected deep learning models, finetuned using capsule layers, optimizer regularization etc . Through the analysis of these models, it came to the conclusion that LSTM networks were highly accurate in performing SA operations. In another paper A Hassan et al. [[20],](#_bookmark122) sentiment analysis was carried out using deep recurrent models on not only Bangla text but with an addition of romanized/la- tinized transliteration of Bangla text, also known as BRBT. There is a scarcity of available datasets that are in Bangla or romanized Bangla and to help bridge the challenge gap, this paper brought upon a very large dataset, made by them, of not only Romanized Bangla text but also ten thousands Bangla and samples with each sample annotated to a native reviewer. In addition to the dataset creation, this pa- per applies deep recurrent models such as the RNN and LSTM on the given Bangla and the romanized text corpus. Loss function was applied with few modifications to the textual data during using LSTM and RNN for the training samples. Results were recorded and it was observed that LSTM in Bangla dataset.

# Chapter 3 Methodology

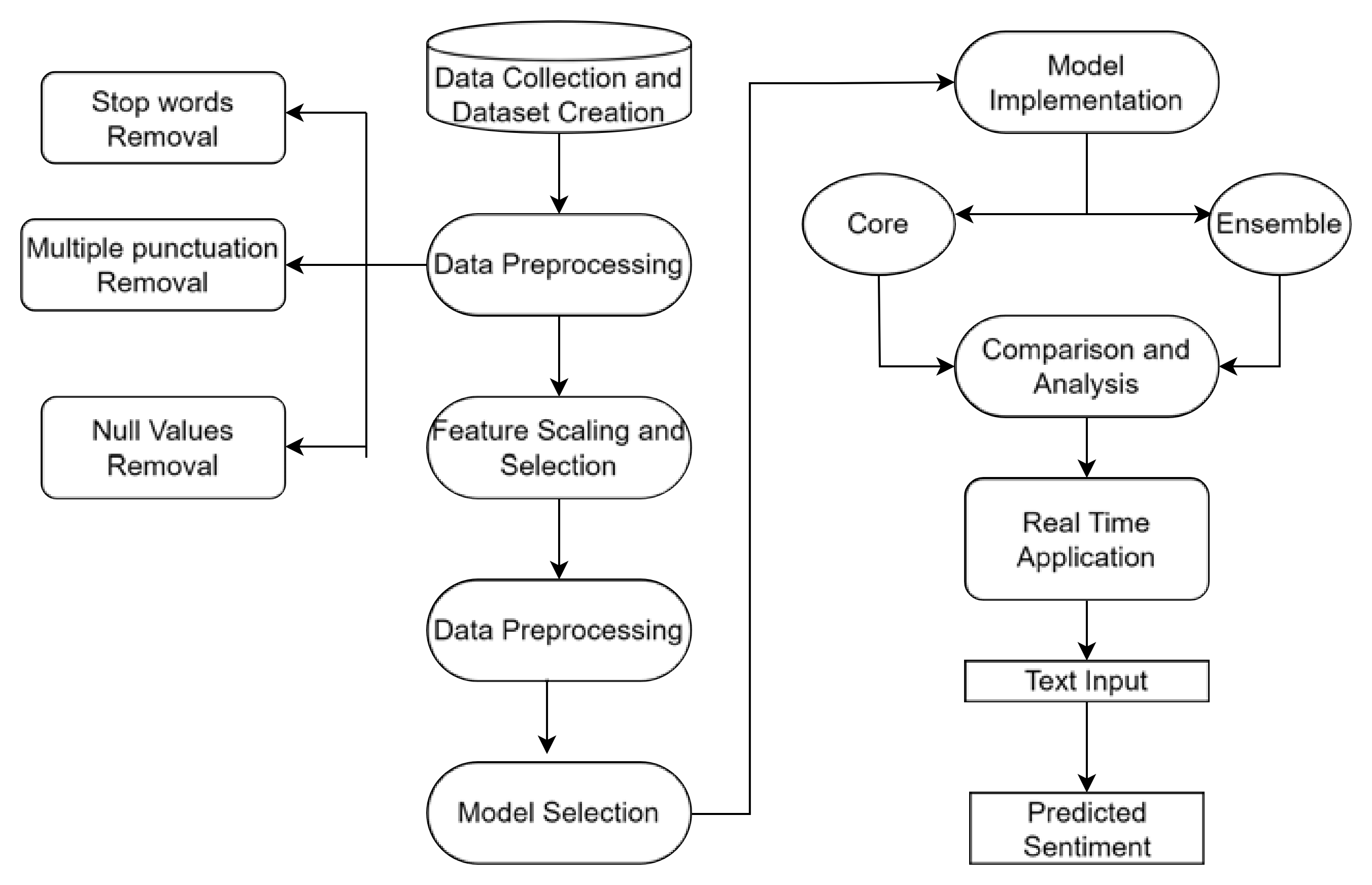


Figure 3.1: Research Concept Map

We began by gathering data from various sources, then carefully cleaned and orga- nized it to ensure its quality and relevance. Next, we selected the most important features from the data and chose an appropriate machine learning model to fit the task. After training the model on the prepared data, we rigorously evaluated its per- formance to guarantee its accuracy and ability to generalize to new data. Finally, we successfully integrated the trained model into a real-time application, enabling it to make immediate predictions on new data as it becomes available.

## Dataset Collection & EDA

For this study, various social media platforms, websites were used to collect the data and compile into a single dataset. The dataset was gathered based on the evalua- tions of food and restaurant reviews from restaurant websites, blogs and facebook comments. It contains a large collection of food and restaurant reviews accompanied by a scale of 1 to 10 and 1 to 5 which varied in different sources. This dataset in- cludes almost 21000 food reviews covering a wide range of information.The dataset

primarily consists of textual reviews, comments and captions related to food. These textual contents differ in length and reflect the diversity of opinions of reviewers. Typically, it is divided into specific subsets, with a sizable chunk given aside for training to speed up the learning process. Additionally, test and validation sets are established to evaluate how well the model performs on untested data and to avoid overfitting. The dataset’s evenly distributed positive and negative attitudes make sure that the performance of the models is not biased toward any one class. The overviews of the dataset are given below.

### Data Annotation

While collecting the review data from different social media sites and other related resources, we also gathered a score along with the review. In the social media platforms, the score was recalled as stars. There were two different scales for the scores, one was from 1 to 5 and another was from 1 to 10. We converted the 1 to 10 scale into 1 to 5. Then we began the annotation process, where we had put some thresholds to annotate our data. The threshold can be seen in the below figure:

Table 3.1: Data Annotation Threshold

|  |  |
| --- | --- |
| **Score** | **Sentiment** |
| 5, 4 | Positive |
| 3 | Neutral |
| 1, 2 | Negative |

So after that, we noticed that, there were around 8000 Positive labeled texts, 6500 Negative labeled texts and around 4000 Neutral labeled texts. The texts where no score was assigned, or also referred as null values, were moved to a different dataframe and later manually annotated in order to create a validation set.

The following figure represents all of our dataset which is evenly categorized into negative and positive sentiments based on respective food reviews. After that, we moved forward to data preprocessing in order to make the dataset more useful to work.

Table 3.2: Overview of the Dataset

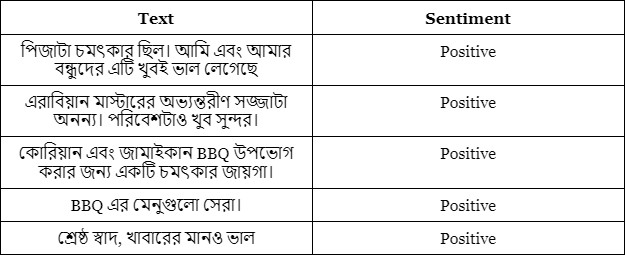


Table 3.3: Dataset Description

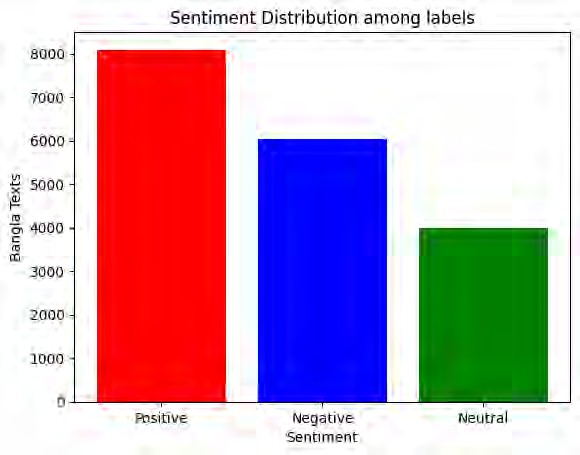
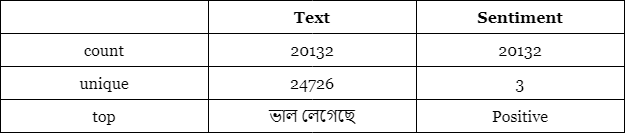


Figure 3.2: Sentiment Frequency Bar Chart

## Data Preprocessing

Preparing data for analysis or machine learning comprises a number of processes to make sure the data you’re dealing with is in the optimum condition. It involves activities such as handling missing values, getting rid of duplicates, transforming categorical data into numerical forms, scaling or normalizing numerical characteris- tics, and frequently lowering dimensionality. The null instances were transferred into another dataframe as a validation set. The goal is to arrange the data in a manner that improves algorithm performance and enables algorithms to deliver more precise and insightful answers during analysis or modeling.

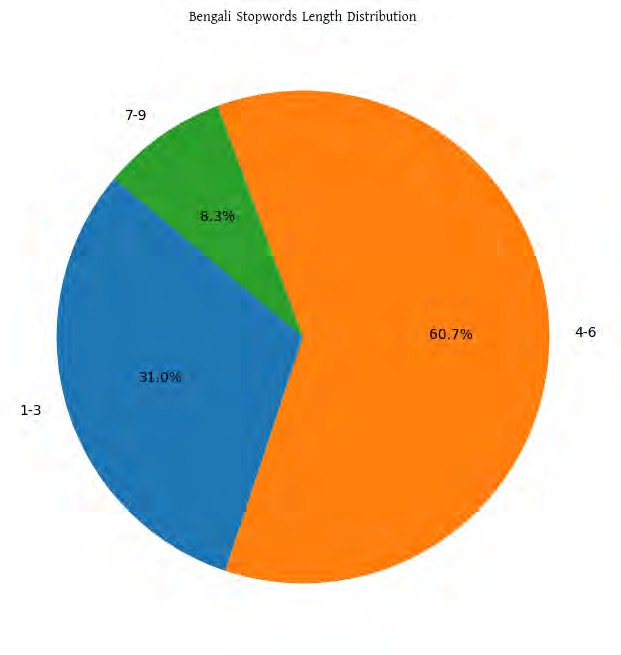


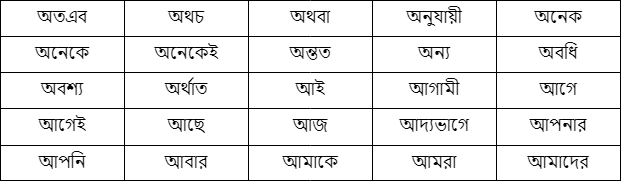
Figure 3.3: Data Preprocessing Concept Map

### Removing StopWords

We particularly focused on cleaning and possibly removing stopwords from Bengali text data. Firstly, we establish a stopwords variable, which stores the name of a file containing a list of Bengali stopwords. Common words are known as stopwords and

are frequently eliminated from text data during text processing since they might not have a meaningful significance.

Figure 3.4: An overview of the stopwords Table 3.4: Overview of Stopwords length



Cleaned reviews function takes a text review as input and performs the following operations:

* + - * Splits the input review into words
      * Iterates over every word and tests if it exists in the list of stopwords.
      * If a word is not in the list of stopwords, it is kept; otherwise, it is removed.
      * The filtered words are then joined back together into a single string.
      * If removing stopwords is False, it only cleans the review using the cleaned reviews function.
      * If removing stopwords is True, it first cleans the review and then removes stopwords using the stopword removal function.



Figure 3.5: Word Cloud of Positive labels.



Figure 3.6: Word Cloud of Negative labels.



Figure 3.7: Word Cloud of Neutral labels.

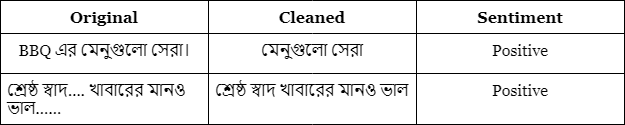
### Removing Punctuations

In order to extract the actual context of a word, we removed the punctuations from the review texts. There may be sentences that include multiple punctuations, and in order to make the prediction more accurate while keeping the context of the word same as it is we applied a regular expression that removed the unnecessary numbers, punctuations and other unnecessary characters.

[ ^\ **u**0980 −\**u**09FF ]

Figure 3.8: Regular Expression for removing punctuations

Table 3.5: Original vs Cleaned data



### Removing Low Length Data

In order to remove the low length data, we applied the following steps:

* + - * The code calculates the length (word count) of each review in a DataFrame (df) and stores it in a new column called ’length’.

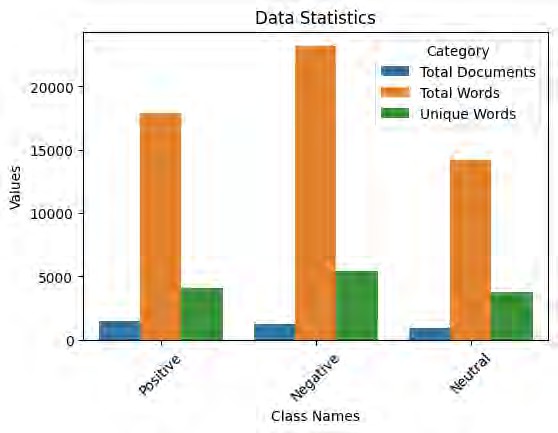


Figure 3.9: Word Statistics of each class

* + - * It then filters out reviews with very few words (less than or equal to and creates a new DataFrame called dataset containing only the longer reviews. Here the threshold of the data length was 2 meaning that texts that had a length less than or equal to 2 were removed from the dataset.
      * Finally, it prints statistics about the cleaned dataset, including the number of reviews, the count of positive reviews, and the count of negative reviews. This step provides an overview of the dataset after the cleaning and filtering process.

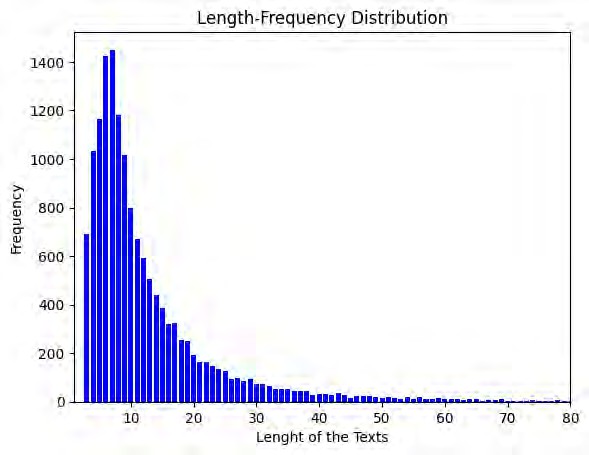


Figure 3.10: Length Frequency Distribution

Table 3.6: Max, Min & Average Length

|  |  |
| --- | --- |
| Criteria | Length |
| Maximum length of a review | 249 |
| Minimum length of a review | 3 |
| Average length of a review | 14.0 |

### Word Distribution Statistics

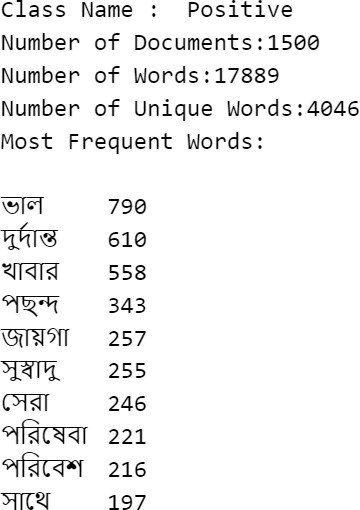


Figure 3.11: Statistics of Positive labeled classes

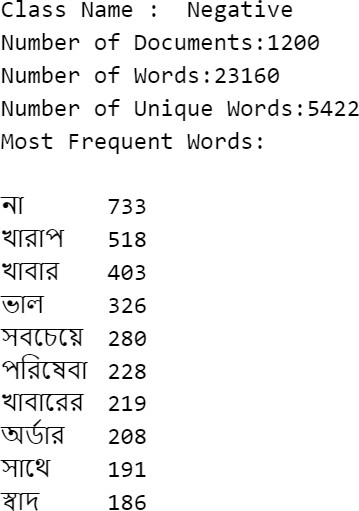


Figure 3.12: Statistics of Negative labeled classes

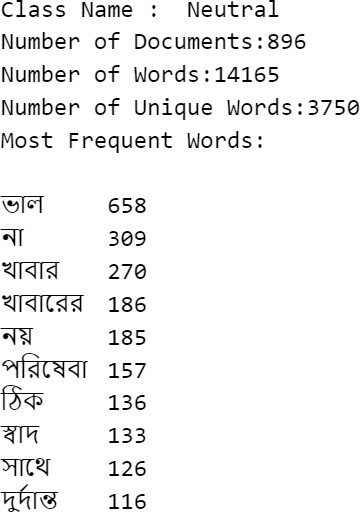


Figure 3.13: Statistics of Neutral labeled classes

### Train-Test Split

In order to perform a train-test split, we used the cleaned data. First we applied different train-test split ratios including 70:30, 80:20 and 90:10. Next, we came to notice that if a 70:30 train-test split ratio is used then the training data for some labels become unbalanced and if a ratio of 90:10 is applied then the same happens for the test ratio. Therefore, we chose the in between 80:20 ratio where the training sets and the testing sets looked quite balanced.

### Label Encoding

As we applied different models for our research, we performed different label encod- ing approaches accordingly to the models. For some models, we manually mapped the labels in the following method:

Table 3.7: Label Mapping

|  |  |
| --- | --- |
| **Sentiment** | **Mapped Label** |
| Negative | 0 |
| Neutral | 1 |
| Positive | 2 |

And the other label encoding approach that we used was one-hot encoding where multiple columns are created according to the labels. In the correct label column, the cell is filled with the value 1 and other labeled cells are filled with 0s according to the review texts.

### Tokenization

In order to apply and fit the data into the models, we had to tokenize the textual data. The process first took a whole review as a sentence and divided the whole

sentence in smaller chunks while removing the spaces. So, each word in a sentence got stored into an array. An example is given below:



Figure 3.14: Tokenization Example

However, this approach was followed for machine learning models but for other deep learning models, different tokenization method was applied. For the deep learning models, the textual data after splitting was converted to strings in order to ensure uniformity. And later the tokenizer was applied using the folllowing parameters:

Table 3.8: Tokenization Parameter for Deep learning models

|  |  |
| --- | --- |
| Parameter | Value |
| max words | 10,000 |
| max sequence length | 100 |
| oov token | <OOV> |

Finally, when LLM (Large Language Models) were applied, they used their own tokenizer package in order to tokenize the texts.

### Vectorization

As the models didn’t understand the textual data directly, we had to vectorize these texts so that the models could process the data and make predictions. For the ml models, TF-IDF vectorizer was used. TF means Term Frequency, which can be denoted by the formula:

Total number of terms in document *d* Number of occurrences of term *t* in document *d*

TF(*t, d*) =

(3.1)

The term IDF stands for Inverse Document Frequency, the below formula denotes IDF:

Number of documents containing term *t* Total number of documents in the collection *D*

IDF(*t, D*) = log (3.2)

Finally, the TF-IDF is calculated using the following formula:

TF-IDF(*t, d, D*) = TF(*t, d*) *×* IDF(*t, D*) (3.3)

When we applied the TF-IDF into our text reviews, the texts were converted into a vector where each dimenstion represented a unique term. An example is given below:

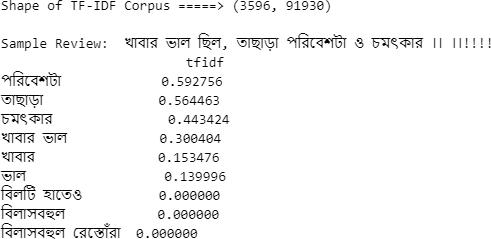


Figure 3.15: TF-IDF Statistics of a Review

## Model Architectures

### Multinomial Naïve Bayes (MNB)

It is employed to forecast the likelihood that a term will fall into a specific class. Its simplicity of usage throughout training and classification processes makes it popular. The Naive Bayes classifier is used to apply pre-processed data as input to the training input set. The trained model is then used on tests to provide either positive or negative sentiment. The Bayes theorem is as follows:

*P* (*Ck*) *·* Q*n*

*P* (*C*

*|*x) = *i*=1

*P* (*xi|Ck*)*count*(*xi*)

(3.4)

*k* Σ*K*

*j*=1

*j*

*P* (*C* ) *·* Q*n*

*P* (*x |C* )*count*(*xi*)

where *Ck* represents the class, x represents the input features, *P* (*Ck|*x) represents the posterior probability of the class conditioned on the input features (i.e., the probability that the class *Ck* holds true given the values of x), *P* (*Ck*) represents the prior probability of the class (i.e., the probability that class *Ck* holds true irrespective of the input feature values), *P* (*xi|Ck*) represents the posterior probability of the feature *xi* conditioned on the class *Ck* (i.e., the probability that *xi* will have a certain value for a given class *Ck*), and count(*xi*) represents the count of occurrences of the feature *xi* in the input features. According to dictionary techniques of score, the stated system determines if the tweet is favorable or negative.

*i*=1

*i*

*j*

### K-Nearest Neighbor (KNN)

KNN, supervised machine learning technique, is used to make predictions on the basis of majority class (classification) or average class (regression) in the feature space using closes k nearest points. Initially labeled dataset with feature instances and matching class labels (for classification is provided. It measures the similarity between instances in the feature space using a distance metric, Minkowski distance was used to determine the distance between each test instance and every other instance in the training dataset.

*D*Minkowski(*P, Q*) =

*n*

*i*=1

Σ

1

*p*

!

*|x*1*i − x*2*i|p*

(3.5)

Decide on a value for K, the number of closest neighbors to take into account while predicting. The hyperparameter K must be set before the algorithm may be used and determine which K training dataset instances are closest to the test instance in terms of distance. These are those that are ”nearest neighbors.” Feature scaling was applied Expected class label for the test instance is found by using the majority voting/averaging rules.

### Random Forest Classifier

The random forest classifier was selected since it ranked highest on a single deci- sion tree in terms of efficiency and reliability. This ensemble technique is based on bulging. A forest appears more sturdy in general the more trees it has. In a similar vein, greater forest tree counts yield higher accuracy outcomes in the random for- est classifier. We will handle the missing data using random forest classifiers. An

example of an ensemble machine learning method is Random Forest, also known as Bootstrap Aggregation or bagging. The Random Forest method and the Bagging ensemble technique are utilized for predictive modeling. The bootstrap technique is utilized to estimate statistical quantities from samples, and the bootstrap aggrega- tion process is employed to generate numerous distinct models from a single training dataset. Random forest algorithm has accuracy, it runs on large dataset , it gener- ates accurate data when large proportion data is missing , generated forest is used for future use to provide accurate results. We strategically assembled a forest of 100 individual decision trees, each independently trained on distinct subsets of the data. This ensemble approach safeguards against the biases and inconsistencies inherent in solitary trees. To meticulously guide the growth of each decision tree, we selected entropy as our information gain criterion. This metric meticulously measures the level of uncertainty or impurity within a dataset, meticulously selecting features that maximize information gain and foster optimal splits. To ensure reproducibility and consistency across multiple model runs, we fixed the random seed to 0. This guarantees that the random processes underpinning model training yield identical results each time, fostering reliable model evaluation and comparison.

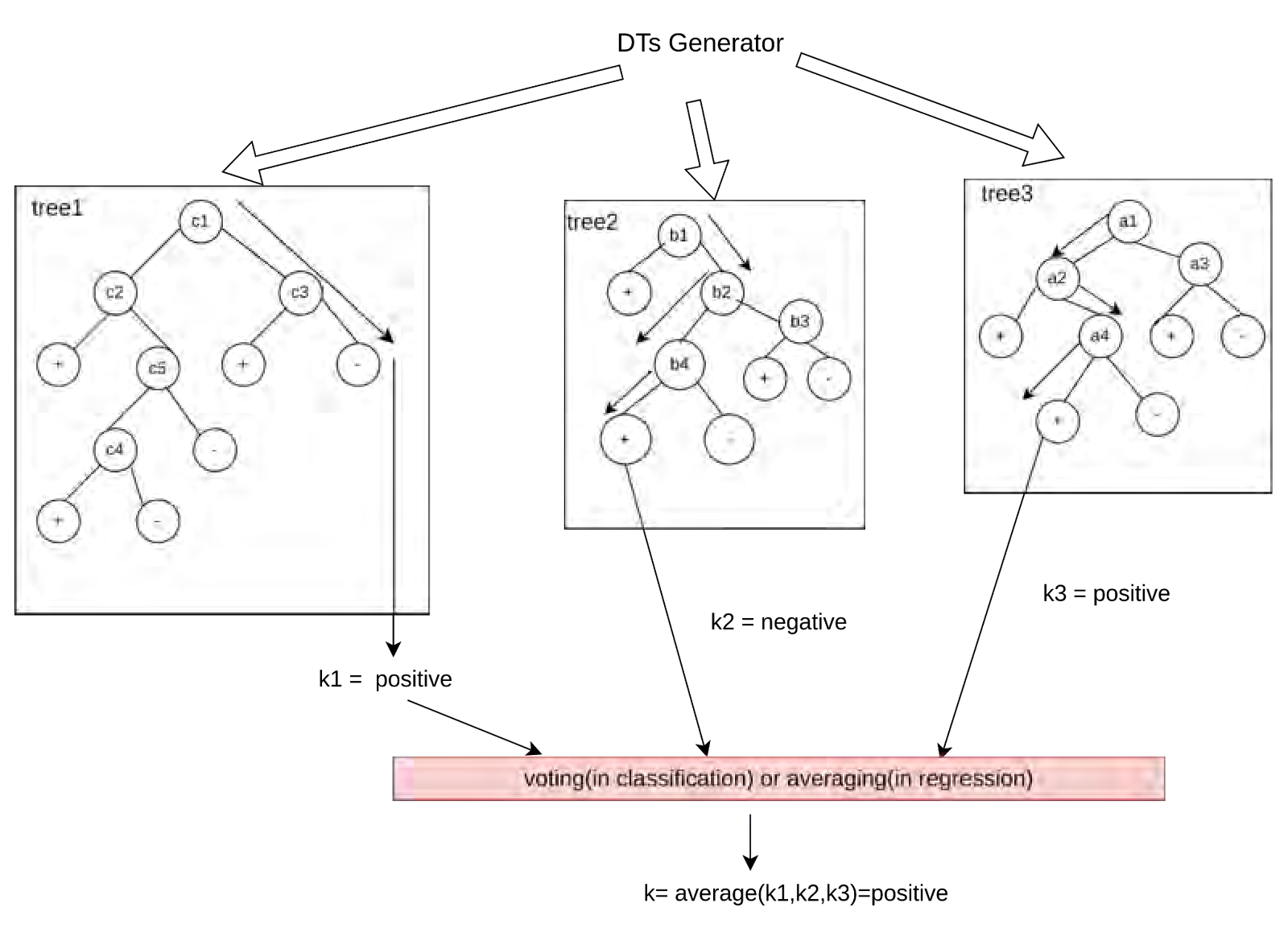


Figure 3.16: Random Forest Classifier Workflow

### Kernel Support Vector Machine (KSVM)

Kernel Support Vector Machine (SVM) accomplishes classification by implicitly transferring the input data to a higher-dimensional feature space using a kernel function; to discover a non-linear decision boundary in the converted space through a kernel function that represents the intended non-linear mapping of the input data

- translates the input features to a higher-dimensional space without explicitly cal- culating the transformed feature vectors.

*K*(x*i,* x*j*) = exp

x*i −* x*j* 2

*—* 2*σ*2



(3.6)

To depict the pairwise similarity of all data points, the kernel/gram matrix is used. *TN* represents the number of support vectors, *αi* represents the Lagrange multi- pliers, *yi* represents the class labels, *K*(x*i,* x) represents the kernel function, and *b* represents the bias. The regularization parameter *C* balances maximizing margin and minimizing classification error.

### Recurrent Neural Network (RNN)

RNN, made to handle sequential data by accounting for the information’s sequential character to store information about earlier inputs and generates an output and updates its internal state at each processing stage by combining an input with previously stored data in its hidden state. At each step of t, the network receives an input xt and the previous hidden state ht-1. The current state (ht) is computed using the input and the previous hidden state is also taken into account.

*yt* = *f* (*Wyh · ht* + *by*) (3.7)

*ht* = *f* (*Whx · xt* + *Whh · ht−*1 + *bh*) (3.8)

*Whx* and *Whh* are weight matrices. *bh* the bias term. f non-linear activation function. Secondly, it updates the hidden state (*ht*) and the network generated output (*yt*). Wyh weight matrix connecting the hidden state to the output. by, output bias term and the output (*yt*) for the next time step *xt*+1. Maximum words used were 1000 and converted text data into sequences of integers, prepared by the tokenizer and padding as post-type, max-sequence-length 100. Embedding layer was created with output dimension 128; dense layer size 1, recurrent dropout of 0.2 and Relu activation.

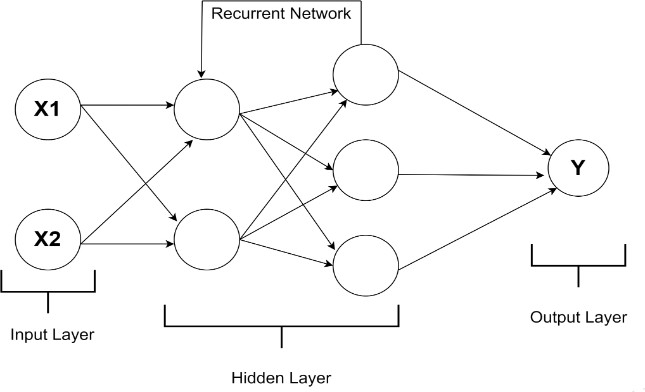


Figure 3.17: RNN Architecture

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

Long Short-Term Memory Networks (LSTMs), created to an improved capture long- term dependencies and eradicate the vanishing gradient problem present in based- traditional RNNs. In contrast to RNN, LSTM consists of cells that can retain information for longer periods of time. These cells enable the model to constantly manage and update information, forget and overcome the vanishing gradient prob- lem that is found in RNNs. BPTT is used in case to train LSTMs in order to

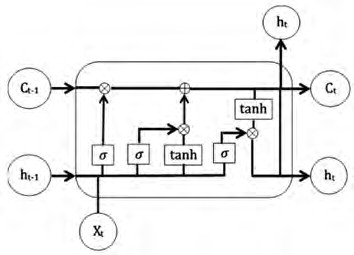


Figure 3.18: LSTM Architecture

minimize a loss function that enhances the network’s prediction capabilities over se- quential data by updating weights and biases. LSTMs have demonstrated superior performance in tasks involving sequential data, such as natural language processing, speech recognition, and time series prediction.

### Convolutional Neural Network (CNN)

The CNN model is made up of a linear form of layers that are passed into a con- structor format by various lists of layers. The layer that receives the bangla text reviews of meals. All of the inputs were later padded, and the review duration was increased to 100. The vocabulary size of 8392 and the input length of 100 are con- tained in the Embedding layer. It selects an embedding space with 300 dimensions in a 100\*300 matrix form, which is helpful for determining the text characteristics from the in-build length of a large quantity of data. This 1D network flow fits the number of output filters employed in the convolution network and contains the filter of 128 feature vectors. The size of the convolutional window in a 1D convolution layer is determined by applying a kernel size to the kernel weight matrix of 5, which consists of 5 feature vectors. Next, for 1D temporary data, the GlobalMaxpool- ing1D layer was used, which maximizes vector space in comparison to the neural network’s step-by-step dimensions. From the meal review dataset, it will gather the maximum vector value of the phrases that contain the most often used vector. ReLu and sigmoid activation were employed, with a 0.2 dropout rate, to prevent the data from being overfitted.

Pooling layers reduce the dimensionality and the final layer produces the classifi- cation and categorizes into labels. It is useful in our study because it can detect sentiment bearing compound words that indicate positive and negative sentiments. This allows the model to learn and identify sentiment indicators and makes it valu- able for our research.

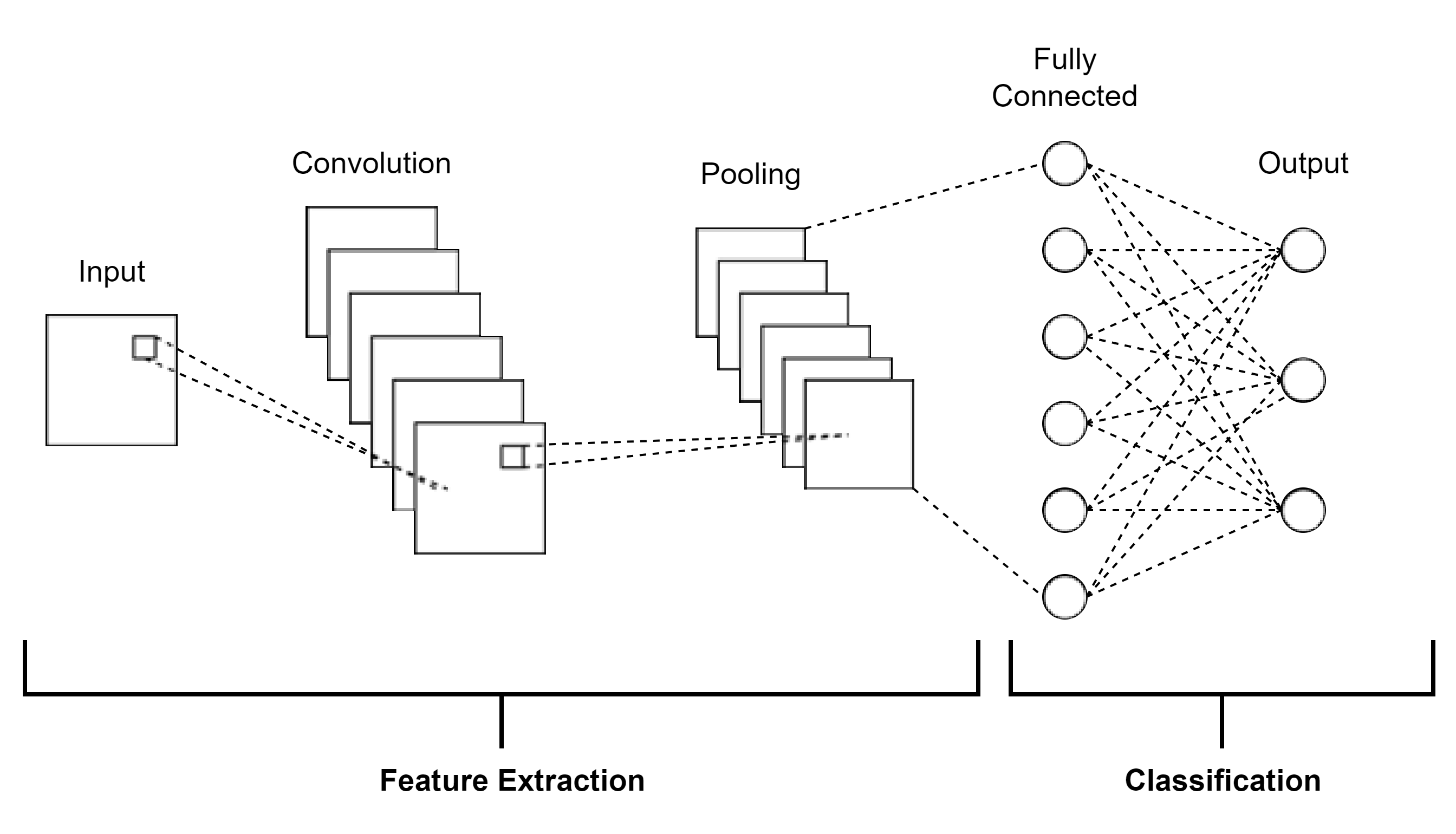


Figure 3.19: CNN Architecture

### Gradient Recurrent Unit (GRU)

GRUs have hidden states that preserve information from prior time steps, which helps grasp the context of the input sequence -preceded by an embedding layer which converts the Bangla text to a dense vector space, capturing the semantic links between words. Padding was applied using the max input to a fixed length of

100. It has a vocabulary size of 8392, an input length of 100, and an embedding space with 300 dimensions. ReLU activation function was used. To avoid overfit- ting, a dropout rate of 0.2 was used, similar to the CNN. Its output was reconfigured with a convolutional layer with Filter size 128 and Kernel size 5 and Convolution operation, *Ct* = Conv1D(*ht,* kernel\_size = 5*,* filters = 128). Applying global max pooling to the convolution layer’s output obtains the greatest value along the time dimension. GlobalMaxPooling1D(*Ct*) = Max*t*(*Ct*). It captures sequential relation- ships in Bangla text, and output is processed through convolutional and pooling layers to extract features for classification using appropriate regularization.

### BERT-base

This implementation employed the ”bert-base-uncased” variant of the BERT model for sequence classification. The tokenizer is initialized and configured accordingly. Text data, comprising both training and testing sets, undergoes tokenization and padding, with a vocabulary size set to 10,000 words. The resulting sequences are truncated or padded to a maximum length of 100 tokens. The neural network architecture takes the form of a Sequential model in TensorFlow. It encompasses an Input layer for integer sequences, the BERT model, and a Dense layer utilizing a softmax activation function. 3 nodes are involved in the output layer, aligning with the three classes in the classification task. BERT uses Adam optimizer while setting up 3e-5 as its learning rate.

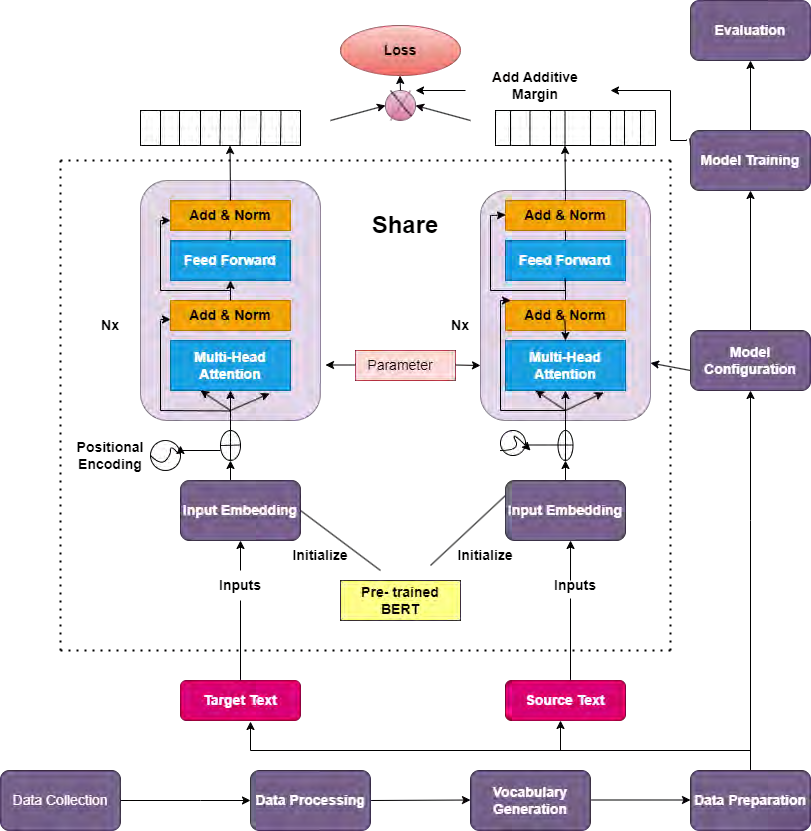


Figure 3.20: BERT Architecture

### RoBERTA

The RoBERTa model, variant of BERT and mainly used for sequence-to-sequence modeling, breaks down data into three parts that are gradually named tokenizer, transformers, and heads. It transformed the raw data into sparse index encodings with a tokenization and sparse content was shaped into contextual embedding by the transformers for deeper training using contextual embedding.

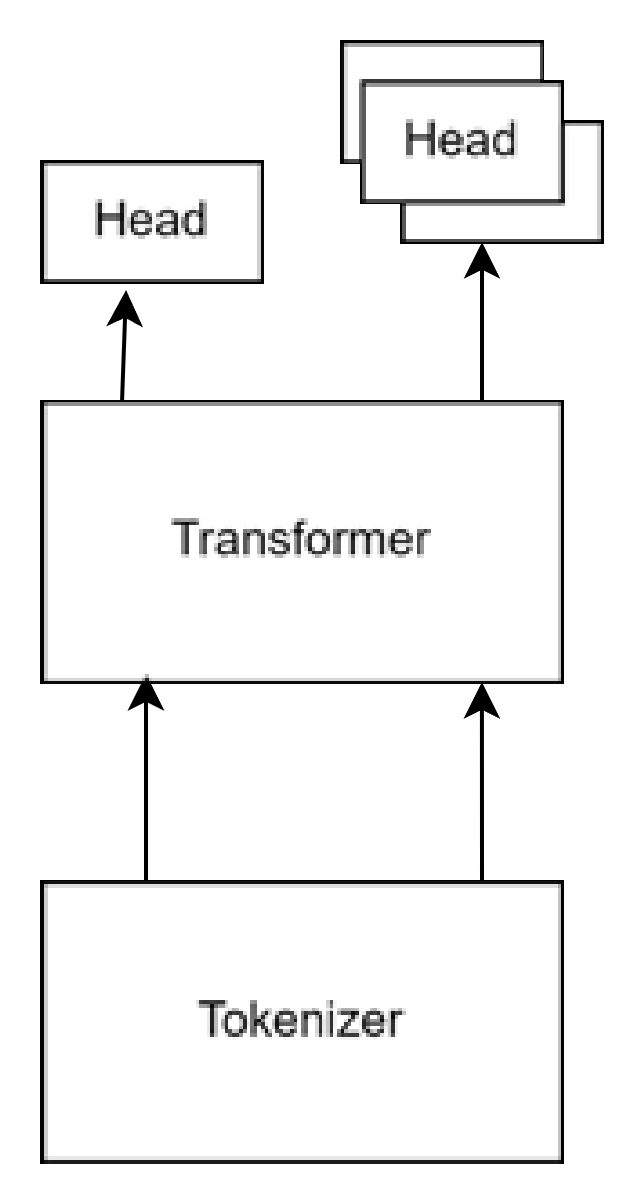


Figure 3.21: RoBERTA Tokenization

It implemented 60,000 vocabulary of the tokenizer which is made of Byte-Pair en- coding and was trained on 4 different corpora, as follows into word embeddings.

There are some special tokens in the R0BERTa tokenizer. The ‘<s>’ and ‘</s>’ is a special token that indicates the starting of the sentence and the closing part is stated by the <pad> token. RoBERTa tokenizer encodes our raw text by input ids and an attention mask. On the other hand, the attention mask ensemble the batch of our sequences. as an elective variable.Our RoBERTa model takes the input ids and attention mask into it. This model carries 12 basic layers, 768 secret conditioned vectors, and 125 million variables.

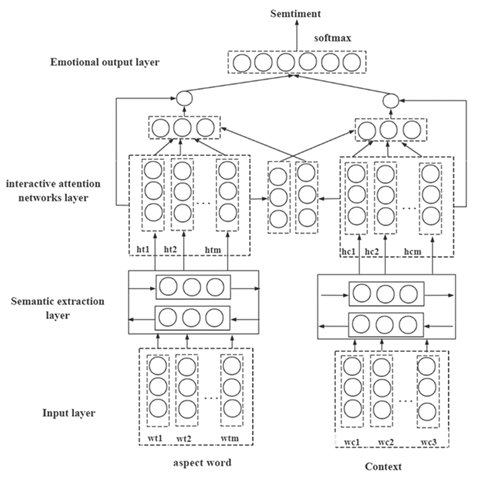


Figure 3.22: RoBERTA Architecture

### Custom Model Architecture

The custom model in use integrates a hybrid architecture, combining the RoBERTa- based transformer with a 1D-CNN for the purpose of text classification. In this design, the RoBERTa model operates as the initial feature extractor, capturing con- textual nuances from input sequences. Subsequently, the last hidden states derived from RoBERTa undergo processing through a 1D-CNN layer featuring 256 output channels and a kernel size of 3. This is followed by the application of a Recti- fied Linear Unit (ReLU) activation function and a max-pooling operation. The resultant features are then subject to global averaging, and a fully connected layer with an output unit count corresponding to the specified three classes facilitates the final classification. The architecture adeptly harnesses the strengths of both RoBERTa’s contextual comprehension and the 1D-CNN’s proficiency in discerning local patterns, thereby furnishing a holistic representation conducive to effective text classification. In numerical terms, the RoBERTa model transforms input sequences into 768-dimensional embeddings, while the 1D-CNN layer produces 256 features.

Notably, this architecture is intentionally crafted to accommodate diverse classifi- cation tasks, and the model’s configuration allows for fine-tuning to suit specific applications.

## Model Training

### Parameters of Different Models

In the Random Forest Classifier, the model used Bootstrap Aggregation (Bagging) technique and each tree was trained independently on subsets. Also, Information Gain criterion was applied. The following table shows the paramters for the Random Forest Classifier:

Table 3.9: Random Forest Classifier Parameters

|  |  |
| --- | --- |
| Parameter | Value |
| n estimators | 100 |
| criterion | entropy |
| random state | 0 |

In the Multinomial Naïve Bayes (MNB) model, an alpha value was set to train this model that worked as a smoothing factor and regularization parameter

Table 3.10: Multinomial Naïve Bayes Parameters

|  |  |
| --- | --- |
| Parameter | Value |
| alpha | 0.15 |

In the K-Nearest Neighbor Model, the the distance of nearest neighbors was calcu- lated to predict the class of the text. Minkowski distance was used, where if the value of p is 1 then it is converted to Manhattan distance and if it is 2 then gets converted to Euclidean distance. the parameters used in K-NN were:

Table 3.11: K-Nearest Neighbors Parameters

|  |  |
| --- | --- |
| Parameter | Value |
| n neighbors | 3 |
| metric | minkowski |

In the Kernel Support Vector Machine algorithm, the textual data was handled in a non-linear way and the kernel transformed the data into higher-dimensional space while maximizing the margin between the classes. The following parameters were used for this model:

Table 3.12: Kernel Support Vector Machine Parameters

|  |  |
| --- | --- |
| Parameter | Value |
| C | 1000 |
| kernel | rbf |
| probability | True |
| gamma | 0.0002 |
| random state | 0 |

Although, to set the bar at the same level, max words and max sequence length parameter was set equally for all the deep learning models. Although rest of the paramters for each model were slightly different.

For the Long-Short Term Memory (LSTM) model, we had to pad the text data and then train the model. Even GRU uses the same paramters but we just add a GRU layer insead of a LSTM layer. The following were used as LSTM and GRU parameters:

Table 3.13: LSTM and GRU Parameters

|  |  |
| --- | --- |
| Parameter | Value |
| input dim | max words |
| output dim | 128 |
| input length | max sequence length |
| dropout | 0.2 |
| dense metric | 1 |
| activation | sigmoid |
| loss | binary crossentropy |
| optimizer | adam |

The Convolutional Neural Network Model had the same paramters as the GRU and LSTM except it used a 1-Dimensional Convolutional Layer with an extra parameter called pool size.

Table 3.14: 1D-CNN Parameters

|  |  |
| --- | --- |
| Parameter | Value |
| input dim | max words |
| output dim | 128 |
| input length | max sequence length |
| dropout | 0.2 |
| dense metric | 1 |
| pool size | 5 |
| activation | sigmoid |
| loss | binary crossentropy |
| optimizer | adam |

In case of BERT, the batch size was changed to 32 from 64 so that the training process would be efficient. The parameter are as follows:

Table 3.15: BERT Parameters

|  |  |
| --- | --- |
| Parameter | Value |
| activation | softmax |
| optimizer | Adam |
| learning rate | 3e-5 |
| from logits | True |

### Large Language Models (LLMs)

RoBERTa, a variant of BERT, was also applied. The base RoBERTa model was loaded. The specifications of the used RoBERTa model are:

Table 3.16: RoBERTa Specifications

|  |  |
| --- | --- |
| Parameter | Value |
| Variant | RoBERTa 7b |
| Layers | 12 |
| Hidden State Vectors | 768 |
| Parameters | 125 million |

The model was already pre-trained with text data which we later had to fine tune for our specific task. We used the same parameters as BERT while fine tuning.

Table 3.17: Custom Model Specifications

|  |  |
| --- | --- |
| Parameter | Value |
| Batch Size | 16,32,64,128 |
| Epochs | 5 |
| Optimizer | Adam |
| Criterion | Binary Cross Entropy |
| Dropout | 0.2 |
| activation | sigmoid |
| input dim | 100 |
| output dim | 128 |
| input length | 10,000 |
| dense metric | 1 |
| pool size | 5 |

The custom model employs CrossEntropyLoss, Adam optimizer (lr=1e-4), and pro- cesses data in batches. The training loop iterates for five epochs, updating param- eters and displaying loss and accuracy metrics. Numerically, the RoBERTa model processes input sequences into 768-dimensional embeddings, and the 1D-CNN layer outputs 256 features.

# Chapter 4

# Result & Analysis

## Data Description

Table 4.1: Max, Min & Average Length

|  |  |
| --- | --- |
| Criteria | Length |
| Maximum length of a review | 79 |
| Minimum length of a review | 2 |
| Average length of a review | 9.27 |
| No. of unique words | 953 |
| No. of Reviews | 610 |
| Most repeated word | thiyo |

## Evaluation Metrics

We calculated accuracy, precision, recall and F1 score to evaluate the performances of the proposed system.

**Confusion Matrix**:

Confusion matrix gives matrix as output. In confusion matrix 4 important terms are there:

* True Positive(TP) : The model predicted YES and the actual output was also YES.
* True Negative(TN) : The model predicted NO and the actual output was NO.
* False Positive(FP) : The model predicted YES and the actual output was NO.
* False Negative(FN) : The model predicted NO and the actual output was YES.

Table 4.1: Evaluation weights

|  |  |
| --- | --- |
| **Abbreviation** | **Description** |
| TP | True Positive |
| FP | False Positive |
| TN | True Negative |
| FN | False Negative |

#### Accuracy

Accuracy refers to the correctly predicted instances out of the total number of in- stances.

#### Precision

Accuracy = *TP* + *TN*

*TP* + *TN* + *FP* + *FN*

(4.1)

Precision may be defined as the accuracy of positive forecasts divided by the sum of genuine positives and false positives.

#### Recall

Precision = *TP*

*TP* + *FP*

(4.2)

Recall is another name for true positive rate, or sensitivity, which evaluates how well models are able to identify pertinent cases. It determines the ratio of real positives to the total of false negatives and true positives.

l = *TP*

Recall = *TP*

*TP* + *FN*

(4.2)

#### F1 Score

The F1 score is the melodious means of precision and recall. It delivers a fair analysis that takes into account both false positives and false negatives. The formula for F1 score is given by:

F1 Score = 2 *·* (*Precision · Recall*)

*Precision* + *Recall*

(4.3)

## Performance Evaluation

The accuracy for our applied models are given below:

Table 4.2: Accuracy of ML Models

|  |  |
| --- | --- |
| Model | Accuracy |
| Random Forest | 80.49 |
| Multinomial Naive Bayes | 72.36 |
| K-Nearest Neighbors | 69.92 |
| Support Vector Machine | 82.93 |

The highest accuracy of 82.93% was achieved by Support Vector Machine while K-NN underperformed the most.

The precision for our applied models is given below:

Table 4.3: Precision of ML Models

|  |  |
| --- | --- |
| Model | Precision |
| Random Forest | 81.18 |
| Multinomial Naive Bayes | 73.51 |
| K-Nearest Neighbors | 71.41 |
| Support Vector Machine | 82.90 |

The highest precision of 82.90 was achieved by the Support Vector Machine, while K-NN exhibited the lowest precision among the models.

The recall for our applied models is given below:

Table 4.4: Recall of ML Models

|  |  |
| --- | --- |
| Model | Recall |
| Random Forest | 80.49 |
| Multinomial Naive Bayes | 72.36 |
| K-Nearest Neighbors | 69.92 |
| Support Vector Machine | 82.93 |

The highest recall of 82.93 was achieved by the Support Vector Machine model, while K-NN exhibited the lowest recall among the models.

The F1 score for our applied models is given below:

Table 4.5: F1 Score of ML Models

|  |  |
| --- | --- |
| Model | F1 Score |
| Random Forest | 80.21 |
| Multinomial Naive Bayes | 72.25 |
| K-Nearest Neighbors | 69.83 |
| Support Vector Machine | 82.73 |

The highest F1 score of 82.73 was achieved Support Vector Machine model, while K-NN exhibited a comparatively lower F1 score among the models.

For the deep learning models we got the following results:

Table 4.6: Accuracy of DL Models

|  |  |
| --- | --- |
| Model | Accuracy in Percent |
| Recurrent Neural Network | 81.30 |
| Long Short Term Memory Network | 78.05 |
| Convolution Neural Network | 80.49 |

Table 4.6: Precision of DL Models

|  |  |
| --- | --- |
| Model | Precision in Percent |
| Recurrent Neural Network | 81.55 |
| Long Short Term Memory Network | 78.16 |
| Convolution Neural Network | 80.77 |

Table 4.6: Recall of DL Models

|  |  |
| --- | --- |
| Model | Recall in Percent |
| Recurrent Neural Network | 81.30 |
| Long Short Term Memory Network | 78.05 |
| Convolution Neural Network | 80.49 |

Table 4.6: F1 Score of DL Models

|  |  |
| --- | --- |
| Model | F1 in Percent |
| Recurrent Neural Network | 81.37 |
| Long Short Term Memory Network | 77.75 |
| Convolution Neural Network | 80.53 |

## Visualization

#### Confusion Matrix

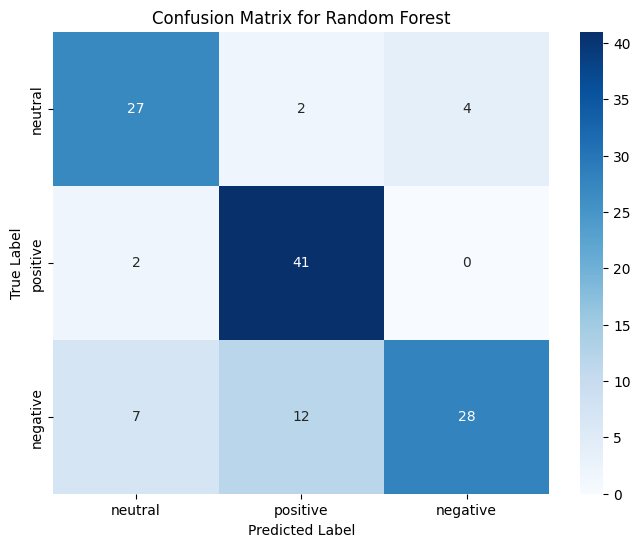


Figure 4.1: Confusion Matrix for Random Forest Classifier

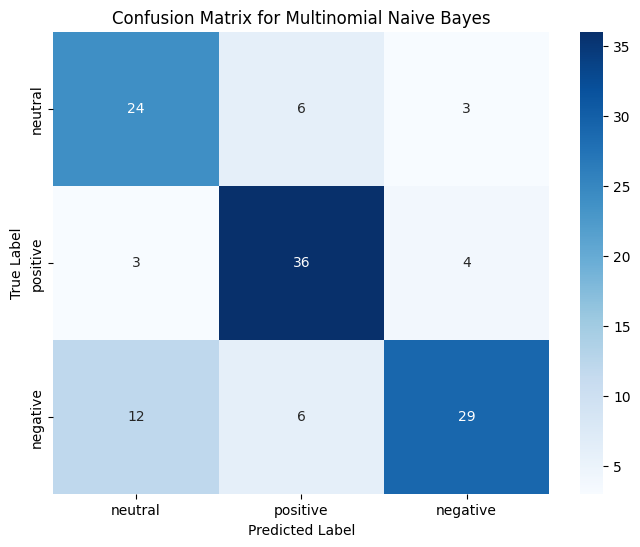


Figure 4.2: Confusion Matrix for Multinomial Naive Bayes

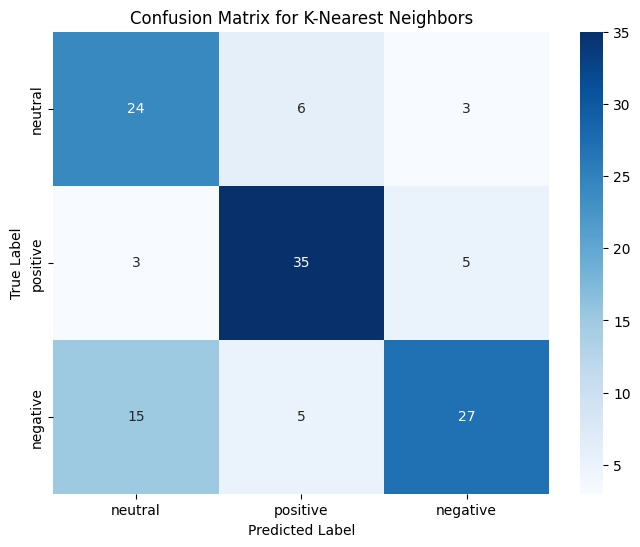


Figure 4.3: Confusion Matrix for K-Nearest Neighbor

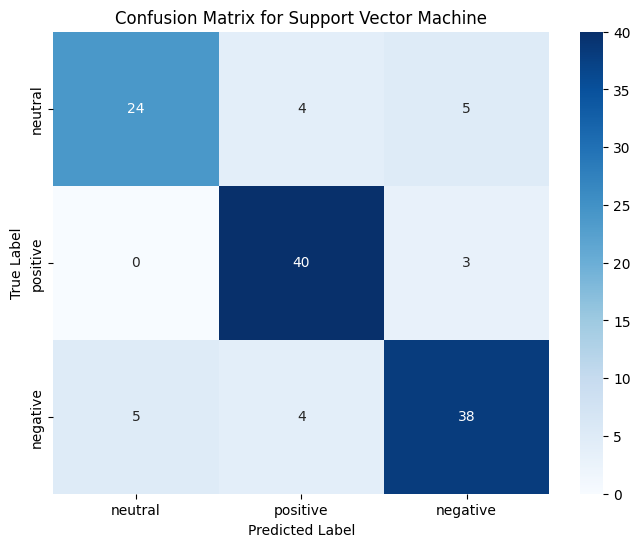


Figure 4.4: Confusion Matrix for Support Vector Machine

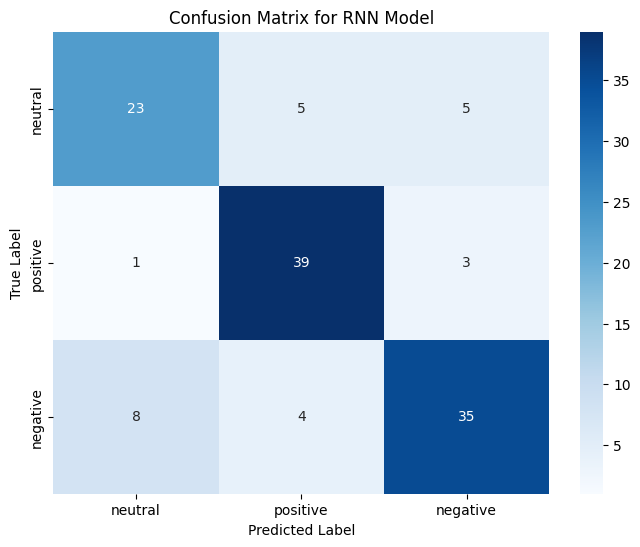


Figure 4.5: Confusion Matrix for Recurrent Neural Network

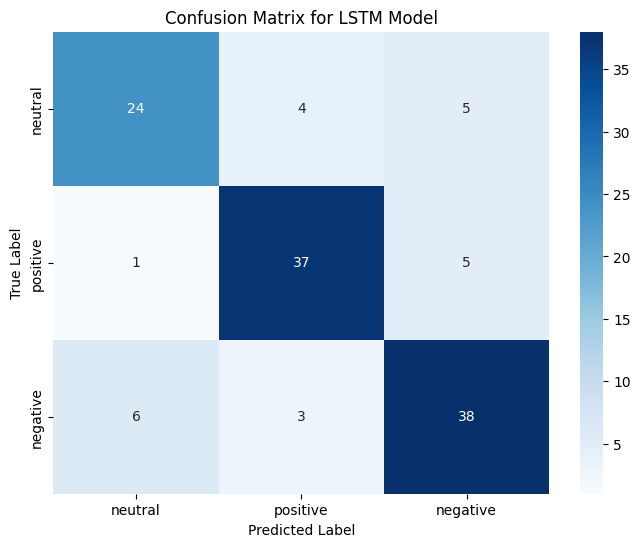


Figure 4.6: Confusion Matrix for LSTM

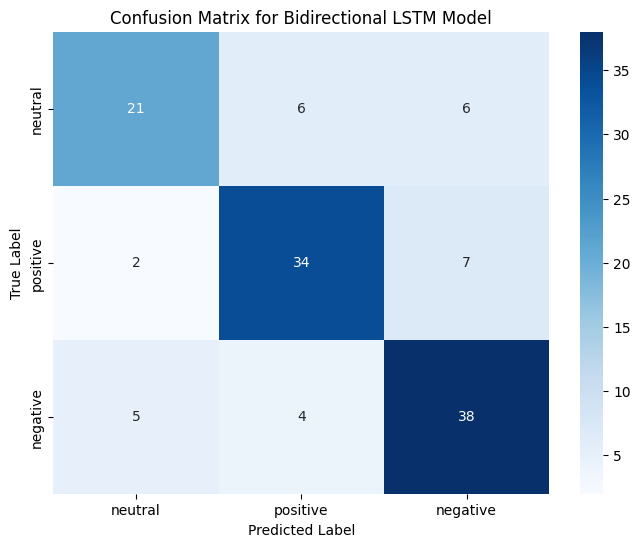


Figure 4.7: Confusion Matrix for Bidirectional LSTM

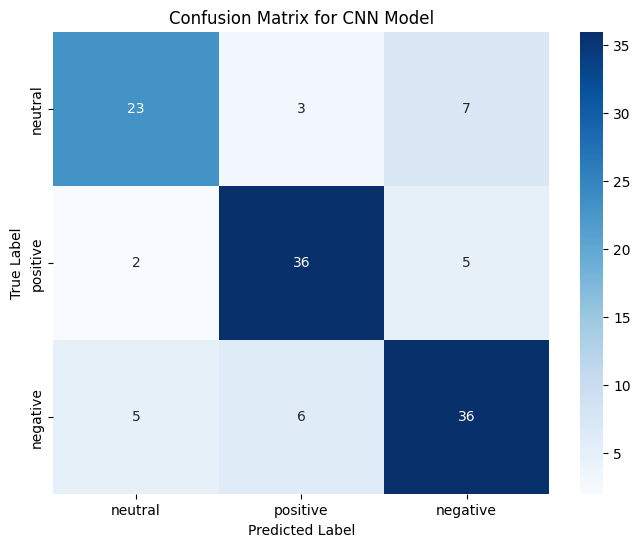


Figure 4.8: Confusion Matrix for CNN

# Chapter 5 Conclusion

In conclusion, our research endeavors revolved around the training and evaluation of five distinct models on a limited dataset, with CNN emerging as the standout performer in terms of accuracy in contrast with RNN, RoBERTA, BERT, LSTM, GRU. The reason for accuracy is focus on local features; it excels at extracting sentiment-bearing words and phrases from short sentences and since it has fewer parameters than LLMs, it requires less data for effective learning. ML algorithms - KNN, MNB, RF classifier and Kernel SVM were also applied and yielded compara- ble results, however they could not outperform CNN.

Future enhancements in Bangla food review sentiment classification involve explor- ing hybrid models, combining the contextual understanding of recurrent models like LSTMs and GRUs with feature extraction from convolutional models like CNNs. This approach aims to boost accuracy, especially in resource-constrained settings. Our ongoing work provides a foundation, emphasizing the potential for improved sentiment analysis in Bangla, contingent on larger datasets and strategic model fusion. Last but not the least, we are expecting to develop robust deep learning models that are capable of accurately discerning sentiment in Bangla text across various domains. Also, we anticipate contributing valuable insights and models to the growing field of Bangla natural language processing.

# Bibliography

1. A. Hossain, Sajin, Tanjim Hussain, M. H. Bhuiyan, F. A. Khan, and S. Anka, “Sentiment classification on Bengali food and restaurant reviews,” Bracu.ac.bd, 2024, doi: id: 20301115. ‌
2. A. Hassan, M. R. Amin, A. K. Al Azad, and N. Mohammed, “Sentiment analysis on bangla and romanized bangla text using deep recurrent models,” in *2016 International Workshop on Computational Intelligence (IWCI)*, IEEE, 2016, pp. 51–56.
3. O. Araque, I. Corcuera-Platas, J. F. Sánchez-Rada, and C. A. Iglesias, “En- hancing deep learning sentiment analysis with ensemble techniques in social applications,” *Expert Systems with Applications*, vol. 77, pp. 236–246, 2017.
4. M. Heikal, M. Torki, and N. El-Makky, “Sentiment analysis of arabic tweets using deep learning,” *Procedia Computer Science*, vol. 142, pp. 114–122, 2018.
5. A. Soumeur, M. Mokdadi, A. Guessoum, and A. Daoud, “Sentiment analysis of users on social networks: Overcoming the challenge of the loose usages of the algerian dialect,” *Procedia computer science*, vol. 142, pp. 26–37, 2018.
6. W. Souma, I. Vodenska, and H. Aoyama, “Enhanced news sentiment analysis using deep learning methods,” *Journal of Computational Social Science*, vol. 2, no. 1, pp. 33–46, 2019.
7. T. Alam, A. Khan, and F. Alam, “Bangla text classification using transform- ers,” *arXiv preprint arXiv:2011.04446*, 2020.
8. P. Cen, K. Zhang, and D. Zheng, “Sentiment analysis using deep learning approach,” *J. Artif. Intell*, vol. 2, no. 1, pp. 17–27, 2020.
9. K. Chakraborty, S. Bhatia, S. Bhattacharyya, J. Platos, R. Bag, and A. E. Has- sanien, “Sentiment analysis of covid-19 tweets by deep learning classifiers—a study to show how popularity is affecting accuracy in social media,” *Applied Soft Computing*, vol. 97, p. 106 754, 2020.
10. N. C. Dang, M. N. Moreno-Garcı́a, and F. De la Prieta, “Sentiment analysis based on deep learning: A comparative study,” *Electronics*, vol. 9, no. 3, p. 483, 2020.
11. O. Habimana, Y. Li, R. Li, X. Gu, and G. Yu, “Sentiment analysis using deep learning approaches: An overview,” *Science China Information Sciences*, vol. 63, pp. 1–36, 2020.
12. M. A. Hasan, J. Tajrin, S. A. Chowdhury, and F. Alam, “Sentiment classifica- tion in bangla textual content: A comparative study,” in *2020 23rd Interna- tional Conference on Computer and Information Technology (ICCIT)*, IEEE, 2020, pp. 1–6.
13. A. S. Imran, S. M. Daudpota, Z. Kastrati, and R. Batra, “Cross-cultural po- larity and emotion detection using sentiment analysis and deep learning on covid-19 related tweets,” *Ieee Access*, vol. 8, pp. 181 074–181 090, 2020.
14. M. Rahman, S. Haque, and Z. R. Saurav, “Identifying and categorizing opin- ions expressed in bangla sentences using deep learning technique,” *Interna- tional Journal of Computer Applications*, vol. 975, p. 8887, 2020.
15. A. Yadav and D. K. Vishwakarma, “Sentiment analysis using deep learning architectures: A review,” *Artificial Intelligence Review*, vol. 53, no. 6, pp. 4335–

4385, 2020.

1. N. R. Bhowmik, M. Arifuzzaman, M. R. H. Mondal, and M. Islam, “Bangla text sentiment analysis using supervised machine learning with extended lexi- con dictionary,” *Natural Language Processing Research*, vol. 1, no. 3-4, pp. 34– 45, 2021.
2. L. Khan, A. Amjad, N. Ashraf, H.-T. Chang, and A. Gelbukh, “Urdu sentiment analysis with deep learning methods,” *IEEE Access*, vol. 9, pp. 97 803–97 812,

2021.

1. P. Shilpa, R. Shereen, S. Jacob, and P. Vinod, “Sentiment analysis using deep learning,” in *2021 Third International Conference on Intelligent Communica- tion Technologies and Virtual Mobile Networks (ICICV)*, IEEE, 2021, pp. 930– 937.
2. N. R. Bhowmik, M. Arifuzzaman, and M. R. H. Mondal, “Sentiment analy- sis on bangla text using extended lexicon dictionary and deep learning algo- rithms,” *Array*, vol. 13, p. 100 123, 2022.
3. M. Hassan, S. Shakil, N. N. Moon, M. M. Islam, R. A. Hossain, A. Mariam, and

F. N. Nur, “Sentiment analysis on bangla conversation using machine learn- ing approach,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 5, pp. 5562–5572, 2022.