

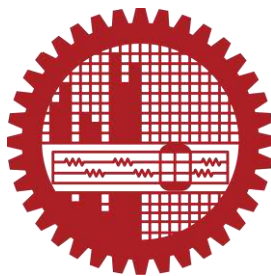
**BANGLA TEXT SENTIMENT ANALYSIS BASED ON EXTENDED LEXICON  
DICTIONARY USING SUPERVISED MACHINE LEARNING AND DEEP LEARNING  
ALGORITHMS**

by

Nitish Ranjan Bhowmik

1017312022



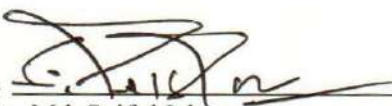

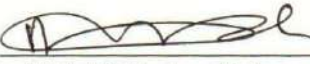
MASTER OF SCIENCE  
IN  
INFORMATION AND COMMUNICATION TECHNOLOGY



Institute of Information and Communication Technology  
BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY  
Dhaka, Bangladesh  
February, 2022

The thesis titled “**BANGLA TEXT SENTIMENT ANALYSIS BASED ON EXTENDED LEXICON DICTIONARY USING SUPERVISED MACHINE LEARNING AND DEEP LEARNING ALGORITHMS**” submitted by Nitish Ranjan Bhowmik, Roll No.: 1017312022, Session: OCTOBER-2017, has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Master of Science in Information and Communication Technology on 19 February, 2022.

### BOARD OF EXAMINERS

1.   
 Dr. Md. Rubaiyat Hossain Mondal  
 Professor  
 IICT, BUET, Dhaka  
 Chairman  
 (Supervisor)
2.   
 Dr. Md. Rubaiyat Hossain Mondal  
 Professor & Director  
 IICT, BUET, Dhaka  
 Member  
 (Ex- officio)
3.   
 Dr. Md. Saiful Islam  
 Professor  
 IICT, BUET, Dhaka  
 Member
4.   
 Dr. Md. Liakot Ali  
 Professor  
 IICT, BUET, Dhaka  
 Member
5.   
 Dr. Md. Mahbubur Rahman  
 (External)  
 Professor  
 Department of CSE, Military Institute of Science and  
 Technology, Dhaka-1216  
 Member

## CANDIDATE'S DECLARATION

This is to certify that the work presented in this thesis entitled, "**BANGLA TEXT SENTIMENT ANALYSIS BASED ON EXTENDED LEXICON DICTIONARY USING SUPERVISED MACHINE LEARNING AND DEEP LEARNING ALGORITHMS**", is the outcome of the research carried out by NITISH RANJAN BHOWMIK under the supervision of Dr. Md. Rubaiyat Hossain Mondal, Professor, Institute of Information and Communication Technology (IICT), Bangladesh University of Engineering and Technology (BUET), Dhaka-1000, Bangladesh.

It is hereby declared that this thesis/project or any part of it has not been submitted elsewhere for the award of any degree or diploma.



Signature of the Candidate

Nitish Ranjan Bhowmik  
1017312022

## **DEDICATION**

I dedicate my dissertation work to the God and my family. A special feeling of gratitude to my loving parents, Monika Bhowmik and Jagadish Chandra Bhowmik whose words of encouragement and push for tenacity ring in my ears. Especially, I am thankful to my wife Anuradha; nobody has been more supportive than you throughout this particularly challenging journey.

# Table of Contents

|  |             |
|--|-------------|
| <b>List of Tables.....</b>                           | <b>viii</b> |
| <b>List of Figures.....</b>                          | <b>xi</b>   |
| <b>List of Abbreviation.....</b>                     | <b>xiv</b>  |
| <b>Acknowledgments .....</b>                         | <b>xvi</b>  |
| <b>Abstract.....</b>                                 | <b>xvii</b> |
| <b>1 Introduction.....</b>                           | <b>1</b>    |
| 1.1 Overview.....                                    | 1           |
| 1.2 Natural language processing.....                 | 2           |
| 1.3 Sentiment Analysis .....                         | 3           |
| 1.3.1 Levels of SA.....                              | 4           |
| 1.3.1.1 Document level.....                          | 4           |
| 1.3.1.2 Sentence level .....                         | 5           |
| 1.3.1.3 Aspect-based SA(ABSA) .....                  | 5           |
| 1.3.2 Lexicon Based Approach .....                   | 5           |
| 1.3.3 SA on Machine Learning: .....                  | 5           |
| 1.3.4 SA on Deep Learning:.....                      | 6           |
| 1.4 Objectives and Possible Outcome.....             | 7           |
| 1.5 Outline of the Thesis:.....                      | 8           |
| <b>2 Literature Reviews.....</b>                     | <b>10</b>   |
| 2.1 Overview:.....                                   | 10          |
| 2.2 Related Works: .....                             | 10          |
| 2.2.1 ML Based Related Works: .....                  | 10          |
| 2.2.2 DL Based Related Works:.....                   | 12          |
| 2.3 Conclusion .....                                 | 14          |
| <b>3 Methodology .....</b>                           | <b>15</b>   |
| 3.1 Overview.....                                    | 15          |
| 3.2 Lexicon Data Dictionary(LDD).....                | 16          |
| 3.3 Construction of Extended LDD: .....              | 17          |
| 3.3.1 Creation of Sentimental Dictionary List: ..... | 17          |

|          |   |           |
|----------|---|-----------|
| 3.3.2    | Creation of Adjective, Adverb Quantifier & Conjunction Word Dictionary<br>Weighted List ..... | 18        |
| 3.4      | Generalized Dataset Preprocessing .....   | 19        |
| 3.4.1    | Tokenization & Normalization.....   | 19        |
| 3.4.2    | Stemming .....  | 20        |
| 3.4.3    | Parts of Speech (POS) Tagger.....   | 20        |
| 3.5      | Objectives & Methodology on DL.....   | 20        |
| 3.5.1    | DL Based Data Preprocessing:.....   | 21        |
| 3.5.1.1  | Neural Network Based Data Preprocessing: .....  | 21        |
| 3.5.1.2  | Data Preprocessing on Attention Based Mechanism: .....  | 22        |
| 3.5.1.3  | Data Preprocessing on Transformer Neural Network (BERT) Based<br>Mechanism:.....              | 23        |
| 3.5.2    | Word Embedding: .....   | 24        |
| 3.6      | BTSC Algorithm.....   | 26        |
| 3.6.1    | Discussion of Algorithmic Pseudocode.....   | 26        |
| 3.6.2    | Score Calculation .....   | 27        |
| 3.6.3    | Simulation of BTSC Algorithm .....  | 29        |
| 3.6.3.1  | Illustration of Example on BTSC Algorithm.....  | 30        |
| 3.7      | Data Augmentation .....   | 31        |
| 3.7.1    | Dataset Creation .....  | 32        |
| 3.7.2    | Creation of Augmented Dataset Algorithm.....  | 34        |
| 3.7.2.1  | Description of Augmented Dataset Algorithm.....   | 34        |
| 3.8      | Conclusion .....  | 35        |
| <b>4</b> | <b>Experiments.....</b>   | <b>36</b> |
| 4.1      | Overview.....   | 36        |
| 4.2      | Experiment on BTSC Algorithm .....  | 36        |
| 4.2.1    | Metrics Evaluation .....  | 38        |
| 4.3      | Experiment on ML Approach .....   | 40        |
| 4.3.1    | Term Frequency - Inverse document Frequency (TF-IDF).....                                     | 40        |
| 4.3.2    | Construction of TF-IDF Matrix.....  | 41        |
| 4.3.2.1  | TF-IDF Matrix Calculation from Restaurant Dataset .....                                       | 42        |
| 4.3.2.2  | Formation of Term Frequency and Inverse Document Frequency                                    | 43        |
| 4.4      | Experiment on DL approach.....  | 44        |
| 4.4.1    | Learning Curve: .....   | 45        |
| 4.4.2    | Convolutional Neural Network (CNN):.....  | 46        |
| 4.4.3    | Dynamic Convolutional Neural Network (DCNN):.....   | 47        |
| 4.4.4    | Multichannel Variable-Size Convolution Neural Network (MVCNN): .....                          | 49        |
| 4.4.5    | Very Deep Convolutional Neural Network (VDCNN): .....   | 50        |
| 4.4.6    | Recurrent Neural Network (RNN):.....  | 52        |
| 4.4.7    | Long Short Term Neural Network(LSTM) .....  | 54        |
| 4.4.8    | Bidirectional Long Short Term Neural Network (Bi-LSTM): .....                                 | 56        |
| 4.4.9    | Asymmetric Convolutional Bidirectional LSTM (AC_Bi-LSTM):.....                                | 57        |
| 4.4.10   | Recurrent Convolutional Neural Network (RCNN):.....   | 58        |
| 4.4.11   | Gated Recurrent Unit (GRU): .....   | 60        |
| 4.4.12   | Bi-directional Gated Recurrent Unit (Bi-GRU):.....  | 62        |
| 4.4.13   | Attention Based Neural Network: .....   | 63        |

|   |   |            |
|---|---|------------|
| 4.4.14                                  | Hierarchical Attention Based Neural Network:.....   | 63         |
| 4.4.15                                  | Capsule Neural Network (CapsNet):.....  | 66         |
| 4.4.16                                  | Bidirectional Encoder Representation From Transformer (BERT): .....                           | 69         |
| 4.4.17                                  | BERT-LSTM Architecture for Sentiment Classification:.....                                     | 69         |
| 4.4.18                                  | Experiment on Augmented Dataset in BERT-LSTM Architecture for Sentiment Classification: ..... | 71         |
| 4.5                                     | Conclusion .....  | 73         |
| <b>5</b>                                | <b>Results and Discussion.....</b>  | <b>74</b>  |
| 5.1                                     | Overview.....   | 74         |
| 5.2                                     | Experimental Results .....  | 74         |
| 5.2.1                                   | Experimental Result of ML on UniGram Model .....  | 75         |
| 5.2.1.1                                 | Support Vector Machine Classification on Tf-IDF Model.....                                    | 76         |
| 5.2.1.2                                 | Confusion Matrix on UniGram Model.....  | 76         |
| 5.2.2                                   | Performance on Different ML Classifier Approach on UniGram Model.....                         | 77         |
| 5.2.3                                   | Experiment on BiGram Model and Comparison Between UniGram Model Approach on SVM .....         | 78         |
| 5.2.3.1                                 | Comparison Between Existing ML Model and Proposed Approach .....                              | 78         |
| 5.2.4                                   | Result Discussion on ML Approach .....  | 80         |
| 5.3                                     | Experimental Result on Deep Neural Networks .....   | 80         |
| 5.3.1                                   | Deep Neural Network Model Training and Fitting: .....   | 81         |
| 5.3.2                                   | Results and Analysis: .....   | 81         |
| 5.3.3                                   | CNN Based Model: .....  | 83         |
| 5.3.4                                   | RNN Based Model: .....  | 85         |
| 5.3.5                                   | GRU Based Model: .....  | 86         |
| 5.3.6                                   | Attention and Capsule Based Model: .....  | 86         |
| 5.3.7                                   | Transformer Based Model.....  | 87         |
| 5.3.8                                   | Result Analysis for Augmented Dataset in Transformer Based Model.....                         | 87         |
| 5.3.9                                   | Comparison Between Existing DL model with our Proposed Hybrid Neural Network Approach .....   | 88         |
| 5.3.9.1                                 | Comparison of ML vs DL Approach in Terms of Accuracy.....                                     | 88         |
| 5.3.10                                  | Result Discussion of Our Proposed Hybrid DL Approach.....                                     | 89         |
| 5.3.11                                  | Complexity Factors of DL Model: .....   | 90         |
| 5.4                                     | Conclusion .....  | 90         |
| <b>6</b>                                | <b>Conclusion .....</b>   | <b>92</b>  |
| 6.1                                     | Journal Publications: .....   | 94         |
| <b>Appendix A Appendix Section.....</b> |   | <b>95</b>  |
| <b>Bibliography .....</b>               |   | <b>111</b> |

# List of Tables

|      |   |    |
|------|---|----|
| 3.1  | Statistical polarity of cricket and restaurant datasets with individual and total comments .....                      | 17 |
| 3.2  | Sentimental word list cricket and restaurant datasets with individual and total.....                                  | 17 |
| 3.3  | Weighted list of adjective, adverb word dictionary.....   | 18 |
| 3.4  | Weighted list of conjunction word dictionary .....  | 18 |
| 3.5  | Demonstration of neural network based data preprocessing.....   | 22 |
| 3.6  | Demonstration on Attention Based Neural Network Data Preprocessing.....   | 23 |
| 3.7  | Classification of tokens in transformer neural network .....  | 24 |
| 3.8  | Demonstration of transformer learning neural network based data preprocessing .....                                   | 24 |
| 3.9  | Score calculation of Ex: 01.....  | 28 |
| 3.10 | Score calculation of Ex: 02.....  | 28 |
| 3.11 | Score calculation of Ex: 03.....  | 28 |
| 3.12 | Score calculation of Ex: 04.....  | 28 |
| 3.13 | Score calculation of Ex: 05.....  | 29 |
| 3.14 | Statistical polarity of data augmentation in cricket and restaurant datasets with individual and total comments ..... | 32 |
| 3.15 | DA process in contextualized word embedding (Bangla-Bert) with BTSC polarity .  | 33 |
| 4.1  | Polarity detection by BTSC on restaurant data .....   | 37 |



|      |   |    |
|------|---|----|
| 4.2  | Polarity Detection by BTSC on Cricket Data .....  | 37 |
| 4.3  | Neutral data detection problem .....  | 37 |
| 4.4  | $C(x, y)$ notation for indicating the parameters of CM.....   | 39 |
| 4.5  | TF-IDF matrix format .....  | 41 |
| 4.6  | Sample data from restaurant dataset .....   | 42 |
| 4.7  | Sample stop word list from restaurant dataset .....   | 42 |
| 4.8  | Representation of word by document matrix .....   | 43 |
| 4.9  | Dataset preprocessing for TF-IDF matrix construction .....  | 43 |
| 4.10 | Calculation of terms in each document.....  | 44 |
| 4.11 | Calculation of term frequency [X], inverse document frequency [Y] and TF-IDF [X*Y] value .....                        | 44 |
| 5.1  | Weighted average of precision, recall, f1-score & accuracy in unigram model for both dataset. ....                    | 75 |
| 5.2  | Hyperparameter dependency on each model .....   | 84 |
| 5.3  | Accuracy, precision, recall, f1-score measures of the model of bangla dataset cricket reviews. ....                   | 85 |
| 5.4  | Accuracy, precision, recall, f1-score measures of the model of augmented dataset cricket and restaturant reviews..... | 88 |
| 5.5  | Comparison of major sentiment classifiers in both ML and DL regarding accuracy. ....                                  | 89 |
| A.1  | Word frequency and inverse document frequency list in restaurant dataset .....  | 96 |
| A.2  | Calculation of term frequency matrix .....  | 97 |
| A.3  | Word by document matrix in restaurant dataset.....  | 98 |
| A.4  | Representation of TF-IDF feature matrix for restaurant dataset .....  | 99 |
| A.5  | Sample data from cricket dataset .....  | 99 |

|     |  |     |
|-----|--|-----|
| A.6 | Word frequency and inverse document frequency list in cricket dataset..... | 100 |
| A.7 | Term frequency matrix in cricket dataset.....                              | 101 |
| A.8 | Word by document table matrix in cricket dataset.....                      | 102 |
| A.9 | TF-IDF feature matrix for cricket dataset.....                             | 103 |

# List of Figures

|     |  |    |
|-----|--|----|
| 1.1 | Natural Language Processing (NLP) System.....  | 2  |
| 1.2 | Overview of Sentiment Analysis System.....   | 4  |
| 3.1 | Visualization of proposed system architecture in ML approach.....  | 16 |
| 3.2 | Visualization of DL Based Proposed System Architecture.....  | 21 |
| 3.3 | Word2Vec (Skip-Gram) Architecture Diagram.....   | 25 |
| 3.4 | Visualization of DL Based Proposed System Architecture.....  | 31 |
| 4.1 | Visualization performance of <b>BTSC</b> algorithm in restaurant and cricket dataset .....   | 40 |
| 4.2 | Convolutional neural network ( <b>CNN</b> ) architecture for sentiment classification .....  | 47 |
| 4.3 | (a) LC of <b>CNN</b> model training accuracy (TA) and validation accuracy (VA) and<br>(b) LC of <b>CNN</b> model training loss (TL), validation loss (VL).....     | 47 |
| 4.4 | Dynamic convolutional neural network ( <b>DCNN</b> ) architecture for sentiment classification .....   | 48 |
| 4.5 | (a) LC of <b>DCNN</b> model training accuracy (TA) and validation accuracy (VA) and<br>(b) LC of <b>DCNN</b> model training loss (TL), validation loss (VL).....   | 49 |
| 4.6 | Multichannel variable-size convolutional neural network ( <b>MVCNN</b> ) architecture for sentiment classification .....   | 50 |
| 4.7 | (a) LC of <b>MVCNN</b> model training accuracy (TA) and validation accuracy (VA) and<br>(b) LC of <b>MVCNN</b> model training loss (TL), validation loss (VL)..... | 50 |
| 4.8 | Very deep convolutional neural network ( <b>VDCNN</b> ) architecture for sentiment classification.....   | 51 |

|      |   |    |
|------|---|----|
| 4.9  | (a) LC of <b>VDCNN</b> model training accuracy (TA) and validation accuracy (VA) and<br>(b) LC of <b>VDCNN</b> model training loss (TL), validation loss (VL) .....       | 52 |
| 4.10 | RNN block diagram [26] .....  | 52 |
| 4.11 | Recurrent neural network ( <b>RNN</b> ) architecture for sentiment classification.....  | 53 |
| 4.12 | (a) LC of <b>RNN</b> model training accuracy (TA) and validation accuracy (VA) and<br>(b) LC of <b>RNN</b> model training loss (TL), validation loss (VL) .....           | 53 |
| 4.13 | <b>LSTM</b> block diagram [27].....   | 54 |
| 4.14 | Long term short term neural network ( <b>LSTM</b> ) architecture for sentiment classification   | 55 |
| 4.15 | (a) LC of <b>LSTM</b> model training accuracy (TA) and validation accuracy (VA) and<br>(b) LC of <b>LSTM</b> model training loss (TL), validation loss (VL) .....         | 55 |
| 4.16 | Bidirectional long term short term neural network ( <b>Bi-LSTM</b> ) architecture for sentiment classification .....  | 56 |
| 4.17 | (a) LC of <b>Bi-LSTM</b> model training accuracy (TA) and validation accuracy (VA) and<br>(b) LC of <b>Bi-LSTM</b> model training loss (TL), validation loss (VL).....    | 57 |
| 4.18 | Asymmetric convolutional bidirectional LSTM ( <b>AC_Bi-LSTM</b> ) neural network architecture for sentiment classification .....  | 58 |
| 4.19 | (a) LC of <b>AC_Bi-LSTM</b> model training accuracy (TA) and validation accuracy (VA) and<br>(b) LC of <b>AC_Bi-LSTM</b> model training loss (TL), validation loss (VL) . | 58 |
| 4.20 | Recurrent convolutional neural network ( <b>RCNN</b> ) architecture for sentiment classification.....   | 59 |
| 4.21 | (a) LC of <b>RCCNN</b> model training accuracy (TA) and validation accuracy (VA) and<br>(b) LC of <b>RCCNN</b> model training loss (TL), validation loss (VL) .....       | 59 |
| 4.22 | <b>GRU</b> block diagram [31] .....   | 60 |
| 4.23 | Gated recurrent unit ( <b>GRU</b> ) neural network architecture for sentiment classification  | 61 |
| 4.24 | (a) LC of <b>GRU</b> model training accuracy (TA), validation accuracy (VA) and (b)<br>LC of <b>GRU</b> model training loss (TL), validation loss (VL).....               | 61 |
| 4.25 | Bi-Directional gated recurrent unit ( <b>Bi-GRU</b> ) neural network architecture for sentiment classification .....  | 62 |

|      |   |    |
|------|---|----|
| 4.26 | (a) LC of <b>Bi-GRU</b> model training accuracy (TA), validation accuracy (VA) and (b) LC of <b>Bi-GRU</b> model training loss (TL), validation loss (VL).....                          | 63 |
| 4.27 | Hierarchical attention based neural network ( <b>HAN</b> ) architecture for sentiment classification.....   | 64 |
| 4.28 | (a) LC of <b>HAN-LSTM</b> model training accuracy (TA), validation accuracy (VA) and (b) LC of <b>HAN-LSTM</b> model training loss (TL), validation loss (VL) .....                     | 66 |
| 4.29 | Dynamic Routing Based Capsule Neural Network ( <b>D-CapsNet-Bi-LSTM</b> ) architecture for sentiment classification .....   | 67 |
| 4.30 | (a) LC of <b>D-CapsNet-Bi-LSTM</b> model training accuracy (TA), validation accuracy (VA) and (b) LC of <b>D-CapsNet-Bi-LSTM</b> model training loss (TL), validation loss (VL).....    | 68 |
| 4.31 | Bidirectional encoder representation from transformer with LSTM neural network ( <b>BERT-LSTM</b> ) architecture for sentiment classification .....                                     | 70 |
| 4.32 | (a) LC of <b>BERT-LSTM</b> model training accuracy (TA), validation accuracy (VA) and (b) LC of <b>BERT-LSTM</b> model training loss (TL), validation loss (VL).....                    | 71 |
| 4.33 | (a) DA of LC of <b>BERT-LSTM</b> model training accuracy (TA), validation accuracy (VA) and (b) DA of LC of <b>BERT-LSTM</b> model training loss (TL), validation loss (VL) .....       | 72 |
| 4.34 | (a) DA of LC of <b>BERT-LSTM</b> model training precision (TP), validation precision (VP) and (b) DA of LC of <b>BERT-LSTM</b> model training recall (TR), validation recall (VR) ..... | 72 |
| 4.35 | DA of LC of <b>BERT-LSTM</b> model training f1-score (Tf), validation f1-score (Tf) .   | 73 |
| 5.1  | <b>BTSC</b> algorithm polarity prediction on both dataset.....  | 77 |
| 5.2  | Visualization performance of different classifier in restaurant and cricket dataset .....   | 78 |
| 5.3  | Visualization performance of UniGram & BiGram model on SVM classifier .....   | 79 |
| 5.4  | Comparing accuracy between the existing and proposed systems.....   | 79 |
| 5.6  | LC of each model training accuracy (TA), validation accuracy (VA) vs. Epoch and LC of each model training loss (TL), validation loss (VL) vs. epoch .....                               | 83 |

# List of Abbreviations

**NLP:** Natural Language Processing

**LDD:** Lexicon Data Dictionary

**DD:** Data Dictionary

**BTSC:** Bangla Text Sentiment Score

**TF-IDF:** term frequency-inverse document frequency

**Word2Vec:** Word To Vector

**DL:** Deep Learning

**ML:** Machine Learning

**SA:** Sentiment Analysis

**POS:** Parts of Speech

**JJ:** Adjective Quantifier

**RB:** Adverb Quantifier

**CC:** Conjunction Parts of Speech

**VB:** Verb Parts of Speech

**DA:** Data Augmentation

**DNN:** Deep Neural Network

**CM:** Confusion Matrix

**TPR:** True Positive Rate

**TNR:** True Negative Rate

**FPR:** False Positive Rate

**TP:** True Positive

**FN:** False Positive

**TN:** True Negative

**FP:** False Positive

**SVM:** Support Vector Machine

**lr:** Learning Rate

**CNN:** Convolutional Neural Network

**DCNN:** Dynamic Convolutional Neural Network

**LC:** Learning Curve

**TA:** Training Accuracy

**TL:** Training Loss

**VA:** Validation Accuracy

**VL:** Validation Loss

**MVCNN:** Multichannel Variable-Size Convolution Neural Network

**VDCNN:** Very Deep Convolutional Neural Network

**RNN:** Recurrent Neural Network

**LSTM:** Long Short Term Neural Network

**Bi-LSTM:** Bidirectional Long Short Term Neural Network

**AC\_Bi-LSTM:** Asymmetric Convolutional Bidirectional LSTM

**RCNN:** Recurrent Convolutional Neural Network

**GRU:** Gated Recurrent Unit

**Bi-GRU:** Bi-directional Gated Recurrent Unit

**HAN:** Hierarchical Attention Based Neural Network

**CapsNet:** Capsule Neural Network

**D-CapsNet-Bi-LSTM:** Dynamic Routing Based Capsule Neural Network

**HAN-LSTM:** Hierarchical Attention-based LSTM

**BERT:** Bidirectional Encoder Representation from Transformer

# Acknowledgments

First and foremost, I express my sincere gratitude to the Almighty for bestowing His blessings upon me and enabling me to complete this work successfully.

I would like to express sincere gratitude to my esteemed supervisor Dr. Md. Rubaiyat Hossain Mondal, Professor, Institute of Information and Communication Technology (IICT), Bangladesh University of Engineering and Technology (BUET), Dhaka, for his valuable guidance throughout the time. Without his encouragement and trust in me, I would not be able to pursue my M.S.c degree. My heartiest indebted to Dr. Mohammad Arifuzzaman, Associate Professor, Department of Electronics and Communications Engineering (ECE), East West University (EWU), Dhaka, who was patient with me and motivated and guided me toward a quantitative methodology in shaping my experiment. His immense knowledge and plentiful experience have encouraged me in all the time of my academic research and daily life. I am thankful to Dr. Md. Saiful Islam, Professor, Institute of Information and Communication Technology (IICT), Bangladesh University of Engineering and Technology (BUET), Dhaka, for his treasured support which was influential. I would like to thank my other committee members for their continuous support and feedback.

Finally, I would like to dedicate my gratitude to my parents, wife, child, sister and to my colleagues. It would not have been feasible for me to finish my work without their great support and understanding over the past several years. Thank you.



# Abstract

With the Internet's social digital content proliferation, sentiment analysis (SA) has gained a wide research interest in natural language processing (NLP). A little significant research has been done in the Bangla language domain because of having intricate grammatical structures in the text. This paper focuses on SA in the context of the Bangla language. Firstly, a specific domain-based categorical weighted lexicon data dictionary (LDD) is developed to analyze Bangla text sentiments. This LDD is developed by applying the concepts of normalization, tokenization, and stemming to two Bangla datasets available in the GitHub repository. Secondly, a novel rule-based algorithm termed as Bangla Text Sentiment Score (BTSC) is developed to detect sentence polarity. This algorithm considers parts of speech tagger words and special characters to generate a word score and extract polarity from a sentence and a blog. The BTSC algorithm, with the help of LDD is applied to extract sentiments by generating scores of the two Bangla datasets. Thirdly, two feature matrices are developed by applying the term frequency-inverse document frequency (tf-idf) to the two datasets and the corresponding BTSC scores. Next, supervised machine learning classifiers are applied to the feature matrices. In the deep learning part, these polarities are then fed into the hybrid neural network and the preprocessed text as training samples. The preprocessed texts are formatted as a vectorization of words of unique numbers of pre-trained word embedding models. Word2Vec matrix with the top highest probability word is applied on the embedding layer as a weighted matrix to fit the DL models. This paper also presents a remarkably detailed analysis of selective DL models with fine-tuning. The fine-tuning includes the use of drop out, optimizer regularization, learning rate, multiple layers, filters, attention mechanism, capsule layers, transformer

with progressive training along with validation and testing accuracy, precision, recall and F1-score. Experimental results indicate that the proposed new long short-term memory (LSTM) models are highly accurate in performing SA tasks. Experimental results corroborate our theoretical claim and show the efficiency of our proposed approach in both machine learning and deep learning approach. Results show that for the case of BiGram feature, support vector machine (SVM) achieves the best classification accuracy of 82.21%. For our proposed hierarchical attention-based LSTM (HAN-LSTM), Dynamic routing based capsule neural network with Bi-LSTM (D-CAPSNET-Bi-LSTM) and bidirectional encoder representations from Transformers (BERT) with LSTM (BERT-LSTM) model we achieved accuracy values of 78.52%, 80.82% and 84.18% respectively.

# Chapter 1

## Introduction

### 1.1 Overview

With the augmentation of modern web technologies, a large scale of data is being stored across different platforms on the Internet. In the age of globalization of the Internet, these data are being used as a repository of resources. These resources are constantly mined as individual or organizational data for discovering knowledge and information. By gathering information from all these platforms, new patterns can be discovered in the data by emphasizing public opinion. Public opinion refers to what people are thinking, their views on current affairs, and what they think about the flow of contemporary events, or an ongoing situation. Public sentiment can be analyzed by collecting public opinion in text or speech. Nowadays, the public expresses their views or opinions through the eruption of various web-based social platforms and microblogging sites. In recent years, social media systems have provided a prominent platform for opinion mining. It allows them to communicate efficiently and cooperate in exchanging information. However, Social media systems on the web have provided excellent platforms for facilitating and enabling audience participation, engagement, and community, due to our new participatory culture.

With the phenomenal growth of Internet social media services, such as microblogging and social networking, offered by platforms such as Twitter, Facebook, etc., interactions among people are

increasing rapidly. People share their different aspects of daily life on these microblogging sites. They also seek needful information on these sites, which varies from person to person. Therefore, finding information on these users text is an interesting idea that helps predict their behaviors in real-life scenarios. Information extraction is an assignment of discovering structured information from unstructured or semi-structured content that means retrieval of information. Information extraction tasks might include named entities recognition, relation within the text, summarization, question answering, etc. It is often performed as a preliminary processing step for text mining applications.

## 1.2 Natural language processing

Natural language processing (NLP) is a section of artificial intelligence (AI) that deals with training a considerable corpus of data in a computer to perceive, process, and generate language for it. efficientlyThe machine can explore, presume, and invent meaningful information from human language intelligently and efficiently in the NLP mechanism. Technologies based on NLP are rapidly increasing. By taking

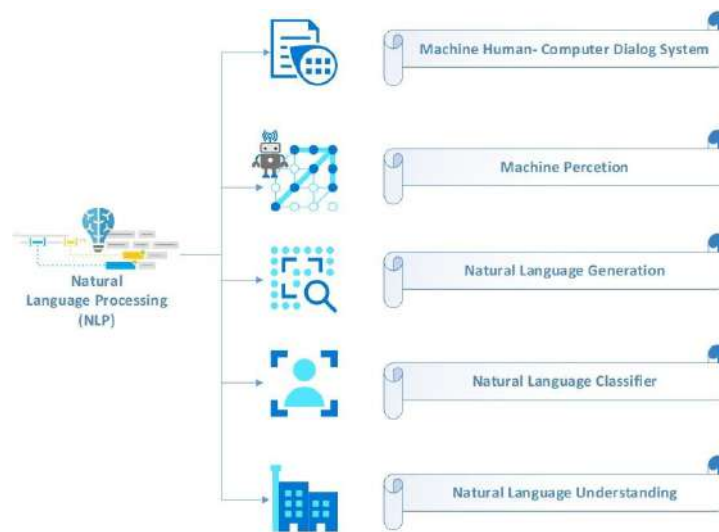


Figure 1.1: Natural Language Processing (NLP) System

advantage of NLP, developers can incorporate and structure knowledge to carry out tasks such as automatic summarization, translation, named entity recognition, relationship extraction, sentiment

analysis (SA), speech recognition, Google search engines, voice assistants and topic segmentation, etc., demonstration in Figure 1.1. NLP is distinguished as a difficult task in computer science. Human language usually refers to the one who utters the word. The language uttered by human beings is not always correct or rarely precise. Understanding human language means for a machine that understanding not just words but concepts and how they relate to meaning. Although language is one of the easiest things for people to learn, its ambiguity makes it difficult for computers to master NLP. If the machine can train with complex things like language, it is possible to do everything with it. The most complex work in the world is contract analysis conducted by multinational companies' legal and financial institutions. While NLP is helping companies out there, people need to find new jobs. However, this automation opens up new job opportunities for people that one did not think. However, people who will not lose jobs will be re-skilling across the sectors.

### **1.3 Sentiment Analysis**

Sentiment analysis (SA), also called opinion mining [1], is a field of study that predicts polarity in public opinion or textual data from microblogging sites [2] on a well-publicized topic by extracting people's attitudes, emotions, etc. However, SA is becoming a relevant subject for NLP in machine learning (ML), researchers are gradually finding interest in this topic because of the large scale of opinionated data on the Internet. Nowadays, people on social media sites, newspapers, blogs, etc., express their opinions on specific products or items, posts, comments, forum discussions, emotions towards an individual or organizations, etc. There may arise many obstructive in detecting binary or ternary class sentiment such as subjectivity or opinion-based identification, if a phrase or text does not have any core opinion word. So, the lexicon-based [3] data dictionary approach is jointed with their semantic tendency with polarity and word strength. To determine these data with sentiment as a polarity i.e., positive, negative or neutral class describe in Figure 1.2, the ML framework has acquired more interest because of building model in many linguistic domains with versatile feature extraction, alternating input easily, predicting with probabilistic theory and computing valuable

feature matrix representations. Various types feature have been observed for this type of work such as bag of words (BoW) model, lexical analysis and semantic feature [4]. This matrix feature is language dependent. Bangla, an ancient Indo-European language, spoken by over 250 million people [5]. So, extracting sentiment in the Bangla language will be significant for NLP researchers to make substantive progress in ML.

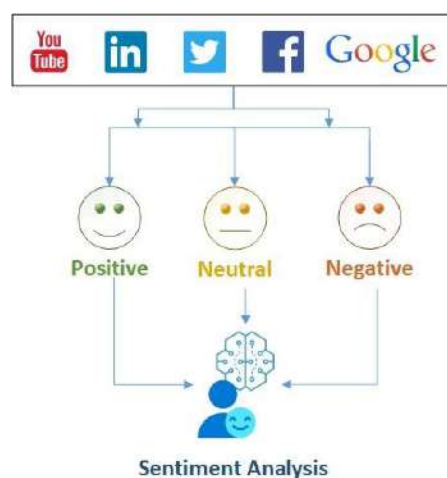


Figure 1.2: Overview of Sentiment Analysis System

### 1.3.1 Levels of SA

SA is accomplished at various levels of entities such as document, sentence, and aspect-based. These levels have been discussed in this sub-section.

#### 1.3.1.1 Document level

This level of SA denotes the sentiment of a complete paragraph or a document. The SA model adopts that document holds opinionated unstructured text about the single entity. It does not justify documents by balancing the multiple entities. The problem of document level SA is determining the positive or negative polarity as a binary classification problem. However, it can manage multi-classification problems as regression type problems such as five type movie review classification problems.

### **1.3.1.2 Sentence level**

This level of SA has a target to extract the sentiment from a single sentence entity. Two types of classification are handled here, one is subjectivity classification and another is polarity classification. These can be used for deriving the sentiment from a single sentence. Subjectivity classification concentrates on discovering whether a sentence is subjective or objective. On the contrary, the polarity classification denotes whether a given subjective sentence is positive or negative.

### **1.3.1.3 Aspect-based SA(ABSA)**

In this level of SA, the sentiments are determined by two aspects: feature-based and object-based entities. It means a single entity is a current per document. The ABSA method aims to detect polarity and aspect pairs from a given sentence. Four types of ABSA can be classified: aspect term, aspect polarity, aspect category and aspect category polarity.

## **1.3.2 Lexicon Based Approach**

The lexicon-based approach is used for detecting sentiment lexicon or core opinionated data to analyze the sentiments from a review. This approach is conducted by building a dictionary or corpus to classify the sentiment words. Due to the shortage of labeled data, a single classifier can be designed to classify reviews from different domains. However, a classifier designed to classify data from one domain may not work efficiently on another domain. This is due to domain-specific words which are different for every domain.

### **1.3.3 SA on Machine Learning:**

SA categorizes the text whether the knowledge about the product is satisfactory or not before customer decide to purchase it. Social networking sites distribute their data conveniently and freely on the web. This availability of information tempts young researchers into their immersive interest in SA. Based on this analysis, marketers and companies understand their product or service so that

it can be tailored to the needs of the user. There are two types of SA methods in ML approach: unsupervised and supervised. Supervised learning is worked on the labeled dataset. This labeled dataset is given to the model during the training process and produces a satisfactory output. Unsupervised learning does not work on category-based datasets, and a clustering technique follows this procedure. A fractional part of robust data is trained to classify the sentiment from unstructured text. ML models such as Unigrams and Bigrams models are applied in classification algorithms such as Naive Bayes, maximum entropy, or support vector machines. ML techniques have been developed in terms of SA and lenient automatic data evaluation.

#### **1.3.4 SA on Deep Learning:**

Social media has a wealth of information in the user-generated text that cannot be processed or classified in real-time extraction even by humans. In the modern web, particularly an outstretched of big data mining from social networks, a massive label of opinionated corpus continuously emerges with the evocation of data classification and scalability. In that case, NLP has required purifying out the noisy word and discovering pertinent insights from this flourishing data. In recent year, many NLP researchers have developed to find out the properties of the text, including emotion, polarity or subjectivity detection and document or context classification. SA has fulfilled this demand for researchers to predict a positive, negative or neutral context. SA or opinion extraction is narrated as collecting information from public content to generate people's attitudes, expressions, and views of customer products, news, topics, or forum discussion (i.e., political, cricket, economic, environmental, etc.) [1]. For example, real-time traffic monitoring systems such as location-based traffic jams, road accidents, and best route policy such as feedback on every situation can be analyzed by people's opinions from social media sites. Again, the level of national military defense or law enforcement organization(i.e., police cyber-crime unit [6]) paid observation and attention to the public opinions on what are doing or saying activities on the electronic media net. The orientation and proactivity of a particular text are argued based upon the polarity and context extracted from the text classification. Classification in Bangla sentences is a complex task, as modern hardware



is enhancing portable and powerful, Deep Learning (DL) is promising in significant performances for NLP operation including SA [7]. DL is a subset of multiple layers of neurons that perceive from a nonlinear neural network with matrix representations and convert the output at one level into an intense and abstract peak. A few ML techniques are driven out on the Bangla text to predict opinions. However, SA can be categorized into two approaches: the corpus-based approach and the other is a dictionary-based approach. However, in this research, we combine rule-based with lexicon dictionary approach and DL models to predict the text sentiment from Bangla text. We will implement a rule-based algorithm termed BTSC for automatically generating scores from the text with the help of categorical weighted LDD. Then we aggregate our BTSC polarity with our input text corpus and build different DL models found in the literature. We conduct multiple experiments on those DL baseline model's to show each model classification performance.

## 1.4 Objectives and Possible Outcome

The main objectives of this research are to analyse the sentiment from Bangla text in ML approach and DL by an unique rule-based algorithm and build a Lexicon Data Dictionary (LDD). To detect polarity from raw text, we have divided our whole work into five parts. To meet the goal, the following objectives have been identified:

- (a) To construct a specific domain-based categorical weighted LDD for analyzing sentiment classification from the Bangla dataset.
- (b) To develop a novel and effective rule-based algorithm for detecting sentence polarity classification by extracting scores from a chunk of Bangla text.
- (c) To investigate the feature matrix with target dataset on ML classifier algorithm.
- (d) To investigate the proposed of our hybrid DL classification algorithm in pretrained word embedding (Word2Vec) model.
- (e) To evaluate our approach and compare the circumference of our work with some existing research paper in both ML and DL algorithm.

The possible outcome will be as follows:

- (i) An LDD, and a rule-based BTSC algorithm will be developed for Bangla language.
- (ii) The effectiveness of ML classifiers and deep neural network will be evaluated in predicting sentiments in Bangla text.

## 1.5 Outline of the Thesis:

This thesis consists of five major sections by which the effect of BTSC algorithm efficiency in both ML and DL is explored. The first part inaugurates the basic concepts of NLP systems and SA techniques. The second part of the thesis will show the previous research works related to NLP, SA in ML classification algorithms, and DL-based architectures. The third part is responsible for exploring the theoretical approach to effect of the BTSC rule-based algorithm with the help of developing LDD, ML, and DL-based data preprocessing techniques. The fourth part will elaborate the experimental procedure in supervised ML and DL neural networks for SA approaches and validate those performances. This part also signifies the efficiency of the BTSC algorithm. The final part will deal with the experimental result discussion and compare it with other research on the accuracy, precision, and recall.

The descriptions of the chapters are the followings which are given below.

**Chapter 2** will show the review of previous research work related to ML and DL-based SA. Existing methods discussed in previous research work will also be presented here by highlighting the limitations of those studies.

**Chapter 3** demonstrates system methodology by developing the proposed rule-based algorithm BTSC, which identifies the text polarity with the help of building a LDD. ML and DL approaches are introduced to conduct our SA.

**In Chapter 4**, the demonstration of the ML classification system and construction of deep neural networks are focused. Support vector machine (SVM), Random Forest (RF), Naive Base Classifier, Logistic Regression, KNN, etc., are used to classify our documents. Convolutional, Recurrent, Long Short Term, Gated Recurrent Unit, Attention, Transformer, Capsule based high configuration-based hybrid model are introduced to conduct our SA experiment.

**Chapter 5** is for practical demonstration of our proposed SA on the BTSC algorithm in both ML and DL procedures. It will explain the confusion matrix of classifying text in the Unigram and Bigram ML model. It will describe the efficacy of the neural network approach in training and testing accuracy, precision, recall, and time estimated graphs with the number of epochs.

Finally, **Chapter 7** will present the concluding remarks and future research direction of the thesis.

# Chapter 2

## Literature Reviews

### 2.1 Overview:

This section exhibits a summary of existing ML and DL-based SA studies. We conclude this section by identifying existing research gaps and providing a study rationale for this chapter.

### 2.2 Related Works:

SA has become an exciting topic among researchers in expanding social media and microblogging sites. Immense research has been done on SA on many linguistic corpora. Researchers working on SA are tempting different approaches to dig up methods that deliver the best result.

#### 2.2.1 ML Based Related Works:

SA is done in many linguistic domains like English, French, Chinese, Arabic, etc. However, the depth of its progress in the Bengali language is insignificant due to some technical and empirical constraints [8]. Our work is highly inspired by this research [9]. To the best of our knowledge, SA in Bengali using an extended dictionary has not been done in any research. Experiment results using Lexicon based Data dictionaries in Arabic language have been obtained so far [10]. In [11]

authors described SentiWordNet(SW) as a curse of dimensionality. They used a sentimental lexicon dictionary based on word2vec to perform SA. Besides, in the Bangla text, authors [12] preprocessed data to carry through a SA by taking tf-idf vectorizer and classified the data with a support vector machine (SVM) algorithm. However, they did not measure the polarity by calculating the score of a text; hence it is required to detect the polarity of each sentence by a specific rule-based [13] algorithm. In [14], the authors proposed a semi-supervised bootstrapping approach in SVM and maximum entropy(MaxEnt) classifier to perform a SA using SW by translating Bengali words to English. Their rule-based bootstrapping approach only counted positive; and negative word polarity by SW, which only worked for a low-limited length text. In [15], the authors proposed using XML-based POS tagger and SW to identify the sentiment from Bangla text by adopting valency analysis. They used SW and WordNet(WN), designed for only the English language. So, a lexicon weighted word dictionary for Bangla is necessary to identify the text's word score or polarity. Besides, in [16], authors extracted positive, negative(bi-polar) polarity from Facebook text by tokenizing adjective words using POS tagger, doing valence shifting negative words at the right side of a sentence, and replacing it with antonym words using SW. SW has a weakness in giving proper polarity in Bangla text. In [17], the authors discussed an automated system for emotion detection by mapping each text to an emotion class, their accuracy was 90%. However, it was more time-consuming to label the data, and their phrase patterns were formed for only three subcategories of sentiment not used in complex sentences. In [18], the authors designed a framework for SA by counting only positive and negative words from their feature word list dictionary. In[19], the authors constructed an extended sentiment dictionary and a rule-based classifier was employed to classify the field of the text polarity by attaining the score of a sentence. In [20], the authors described a lexicon-based dictionary model by checking the occurrences of a sentimental feature word in tagging each sentence.

### 2.2.2 DL Based Related Works:

In this section, we sketch out the available methods with a taxonomy that explore the influences on the several DL architectures and discuss how those methods enhance operating in SA. In the interdisciplinary domain of NLP, sentiment classification was portrayed in [21], describing a connection between subjectivity detection and polarity classification. In [22], the authors showed a probabilistic neural model for learning a consecutive representation of words and a probabilistic function to the word sequences simultaneously. A simple one-layer-based convolutional neural network (CNN) approach was given in [22] to conduct a sensitivity analysis of the text. An artificial neural network (ANN) does not work on a large scale of inputs. However, CNN or Hybrid based CNN, i.e., Dynamic CNN (DCNN) [23], Very Deep CNN (VDCNN) [24], variable-size convolutional filters, i.e., (MVCNN) [25] model can do much better. DCNN uses a dynamic K-max pooling and a global pooling operation over the text sequence. In contrast, VDCNN and MVCNN use different dimensions of word embeddings on multiple filter sizes, respectively, in character and text levels. A recurrent Neural Network (RNN) is efficient in doing words or sentences as an unseen input on the network by propagating weight matrices over the time steps [26]. As RNN has a problem of vanishing gradient descent, gradient explosion and lack of backpropagation, those are mitigated in a modified version of RNN such as termed as Long Short Term Memory Network (LSTM) [27], Bi-Directional LSTM [28], Asymmetric Convolutional Bidirectional LSTM (AC-BLSTM) [29], Recurrent Convolutional Neural Network (RCNN) [30], Gated Recurrent Unit (GRU) [31]. Hierarchical Attention Network (HAN) based mechanism [32] on Bi-GRU [33] and LSTM [34] are also applied for document text classification because it works between the hidden (encoder and decoder) layer to give a weighted sum of all features fed as an input. The Google researchers published a recent NLP task, Transfer Neural Network BERT [35], which learns contextual relations from words or text and is also applied for SA [36].

A more significant portion of SA based on DL is conducted on many high resource language domains (i.e., English, Chinese); however, a few studies on Bangla language is on the primary stage.

In [37] the authors performed SA on 4000 positive and negative movie reviews, which was manually translated into Bangla and obtained accuracy on LSTM at 82.42%. Another LSTM-based approach was conducted on 9337 reviews for classifying polarity positive and negative sentiments, and they achieved an accuracy of 78% [38]. In [39], the authors extracted six types of emotions from different types of Bangla Youtube video comments using a CNN and LSTM-based approach. They showed 65.97% and 54.24% accuracy on three and five labels sentiment. Another CNN-based single-channel approach [40] was implemented on different domains from the Bangla dataset. However, it can not maintain proper tuning in layers. An RNN type of network Bi-LSTM approach was applied on a manual hand label dataset of 10000 comments from Facebook, and they obtained an accuracy of 85.67%; however, it has many notable drawbacks in data preprocessing [41]. In [42], the authors obtained an accuracy of 75.5% on the word2vec model by tuning the word co-occurrence score in word vector similarity. In [43], the authors experimented on Bangla Romanized dataset and tested on a deep recurrent model LSTM and achieved accuracy of 55% for three categories. In [44], the authors examined aspect-based SA data with 95% accuracy. However, global common words rephrased the common and proper noun of Bangla words. This was a hindrance to extracting sentiment in a lexicon-based dictionary approach. In a recent research, the authors [45] implemented an attention-based CNN model, and the authors [46] combined CNN with the LSTM model to analyze sentiment from Bangla text. However, in a lexicon-based approach, a word may have different meanings in different domains; so, a lexicon sentiment dictionary is a needed resource for conducting SA. However, it classifies the core word as annotated polarity with sentence or phrase sentiment strength. In [47, 48] the authors built a sentiment detection mechanism from tweets using a sentiment lexicon and a rule-based linguistic approach [49]. To the best of our knowledge, SA using categorical weighted LDD and rule-based algorithm BTSC in Bangla text with comprehensive DL approaches is not used yet.

## 2.3 Conclusion

Sentiment analysis using ML and DL-based approaches has drawn the attention of a large number of researchers. In this chapter, a series of state-of-the-art literature's has been reviewed. We have discussed the main issues and techniques related to the SA based on existing ML and DL research works. The findings of experimental works are also mentioned in Bangla domain. Various sentiment analysis methods with their performance parameters have been explored. Some new SA based on SW, WN, or manually annotated dataset techniques that show better results in ML and DL models, are also addressed in this study. However, we highlighted the pros and cons of each approach and dataset. From the discussion of impairments-related works it is clear that there is no work done on LDD in the Bangla dataset that can explain the effect of rule-based SA and experimentally in the ML and DL approaches.



# Chapter 3

## Methodology

### 3.1 Overview

The methodology of our SA in machine learning and deep learning approaches is to analyze the sentiment from Bangla text with a unique rule-based algorithm and build an LDD. We have divided our work into two parts, one is ML, and the other is DL. First, we will describe the ancillary mechanism for working in ML. Among the three levels of SA, we worked on the sentence level polarity classification by using the extended Bangla sentiment dictionary. These sentimental dictionary words are implied as opinion words, which is an impetus for identifying polarity from a text by implementing a set of rule-based automatic classifier algorithms [50]. In this thesis, an effective and unique rule-based algorithm, Bangla Text Sentiment Score (BTSC) is developed to detect sentence polarity that provides better sentiment extraction by giving a score from a chunk of Bangla text. We build an automated system that can extract opinions from Bangla dataset reviews with the help of an extended Bangla sentimental dictionary with weighted value. That automated system will be classified by a supervised ML algorithm [51] with the help of N-gram (Unigram, Bigram) models because this model performs better in text classification [52]. The overall SA structure in the ML process is described in Figure 3.1.

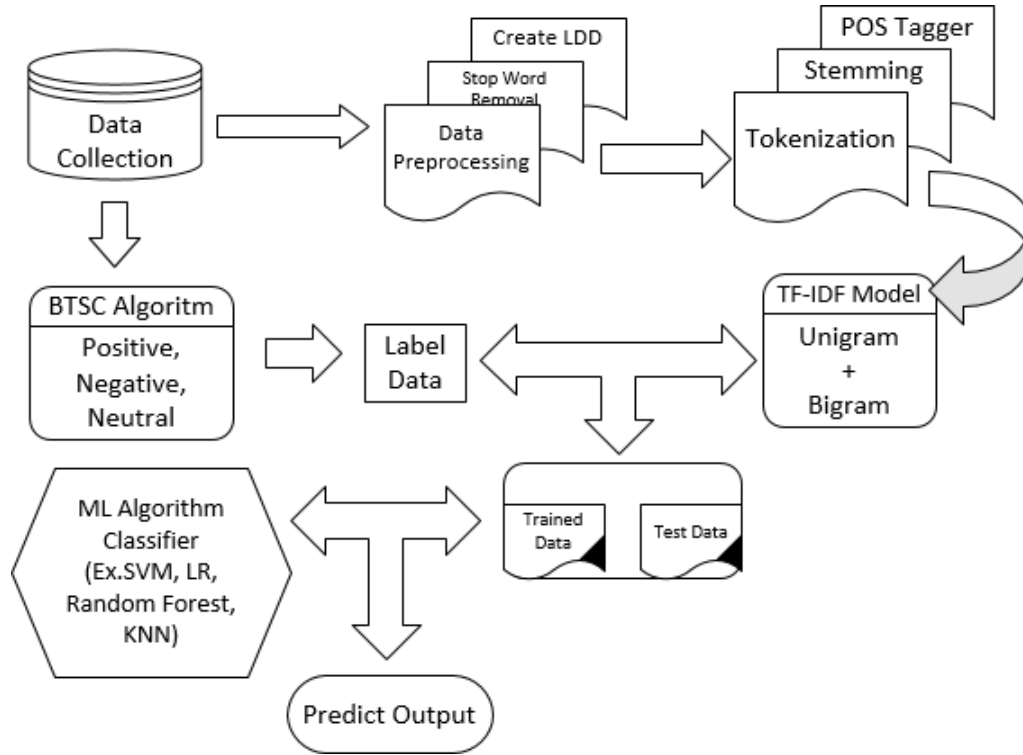


Figure 3.1: Visualization of proposed system architecture in ML approach

## 3.2 Lexicon Data Dictionary(LDD)

To communicate in a language and express thoughts and views in a society, a person needs vocabulary. If one says something outside of vocabulary, they may not understand it. When they speak in Bengali, we usually do not express our views on words outside the Bangla dictionary. It means that our vocabulary is almost specific in every language. People gradually grow up learning this vocabulary from their childhood. When we grow up, we do not understand the meaning of the word. However, we use a lexicon or dictionary as a reference to use those words in the future. The computer language is numbers. However, the way we speak or write, we have to change the numbers before putting them on the computer. Since machines are good at modeling numbers, our work is only reduced if we teach the machine the idea of converting this language into mechanical language, that is, numbers. To learn a language for a machine, even if it is the smallest unit letter of a language, the interpretation comes from where the words are in a sentence. In that case, we need to develop a lexicon data dictionary (LDD) to detect the sentence scores from the text.

### 3.3 Construction of Extended LDD:

An extended lexicon dictionary means the alphabetical list of words or phrases. I manually created a sentimental dictionary word list as a Bengali sentence containing many words. These words are applied for calculating the score from a sentence or phrase. We have collected data from [53], where there are two datasets based on two domains: cricket and restaurant. The construction of the Bangla dataset is described in [54] extensively.

#### 3.3.1 Creation of Sentimental Dictionary List:

Table 3.1: Statistical polarity of cricket and restaurant datasets with individual and total comments

| Dataset    | Polarity |          |         | Total |
|------------|----------|----------|---------|-------|
|            | Positive | Negative | Neutral |       |
| Restaurant | 1216     | 478      | 365     | 2059  |
| Cricket    | 620      | 2108     | 251     | 2979  |

Table 3.2: Sentimental word list cricket and restaurant datasets with individual and total

| Data       | Sentimental Dictionary |            | Total Words |
|------------|------------------------|------------|-------------|
|            | Active                 | Contradict |             |
| Restaurant | 1056                   | 970        | 2026        |
| Cricket    | 1115                   | 2190       | 3035        |

Table 3.1 shows the statistical polarity for both datasets. For performing SA, we have used those datasets to build up our extended sentimental dictionary, such as, 'যোগ্যতা' [Competence], 'অযোগ্যতা' [Inefficiency], 'পরিষেবা' [Service], 'বাধা' [Hindrance] and so on. In sentimental dictionary, a word can be intersecting in both datasets, like 'কল্পনাশ্রুত' [Imaginary] word is an active word list in restaurant dataset; however, a contradict word list in cricket dataset. While SentiWordNet works on the global domain data however to do SA in different domain data, a sentimental weighted dictionary has to be created. The number of sentimental word list is composed of active (weight = +1) and contradict word (weight = -1), which is extracted by manually, represented in Table 3.2. Besides a negative word (weight = -1) list like, 'না', 'নেই',

‘নাই’, ‘নয়’ [not] vocabulary has been created so that negative words can be counted during score calculation from the text.

### 3.3.2 Creation of Adjective, Adverb Quantifier & Conjunction Word Dictionary Weighted List

Since a major difference in English and Bangla grammar, we need to create our own weighted dictionary word list of adjective 'নাম বিশেষণ' [55] and adverb quantifier which is showed in Table 3.3. In Bangla grammar 'বিশেষণের অতিশায়ন' [Exaggeration of adjectives] and degree of adverb 'ক্রিয়া বিশেষণ' [56] is a segment of Adjective POS tagger. We partitioned the whole word set into 3 types: high, medium, low.

Table 3.3: Weighted list of adjective, adverb word dictionary

| Types  | Example  | Weight | Total Word |
|--------|--|--------|------------|
| High   | ‘সবচাইতে’ [Most of all], ‘সর্বাধিক’ [greatest], ‘যথেষ্ট’ [enough], ‘অতিশয়’ [too much] ... } | 3      | 18         |
| Medium | ‘অধিক’ [more than], ‘বেশী’ [more], ‘অনেক’ [lots of] ... }                                    | 2      | 15         |
| Low    | ‘অতিশয়’ [at least], ‘সামান্য’ [a little] ‘প্রায়’ [nearly] ... }                            | 0.5    | 20         |

Table 3.4: Weighted list of conjunction word dictionary

| Categories                 | Example  | Weight | Total Word |
|----------------------------|--|--------|------------|
| Coordinating Conjunction   | { ‘কিন্তু’ [but], ‘আদপে’ [in fact], ‘এবং’ [and], ‘অথবা’ [or], ‘বরং’ [or] ... }                       | 2      | 25         |
| Subordinating conjunctions | { ‘অধিকন্তু’ [Furthermore], ‘বিশেষত’ [in particularly], ‘এমনকি’ [even], ‘এসত্তেও’ [despite of] ... } | 1.5    | 12         |

Although these words do not affect determining the polarity of a sentence, however, sitting before a few words in a sentence can impact the score of the whole sentence. These words can quantify the sentence score. For example, ‘ব্যাটসম্যানদের মধ্যে সাকিব সবচাইতে ভালো’ [Shakib is the best among the batsmen] in that sentence ‘সবচাইতে’ [most of all] word quantify the word

‘ভালো’ [good] which produce the word [best] in the translated English sentence. A Bangla sentence can have a conjunction POS, which is used to joint words, phrases and clauses. These types of words sit in the middle, beginning, or at the end of a sentence, and connect one or more sentences together. This further increases the score of two sentences without effecting on polarity. As there are four main types of conjunction and many sub parts conjunctions in Bangla grammar, for simplification of our work we generalize them into two categories named as coordinating conjunctions [‘সমুচ্চয়ী’] and subordinating or progressive conjunctions [‘অনুগামী’]. However, in our research work, we simplify those words and assign appropriate weight values. It can be noted that the weight values of adjectives and adverbs in Table 3.3, and that of conjunctions in Table 3.4 are assigned carefully to make it particularly suitable for the Bangla language context. For this assignment, the GitHub Bangla dataset available in [53] is taken into consideration. Because of the difference in language structure, the weight values of Table 3.3 and Table 3.4 for Bangla are different from the values mentioned for the Chinese language in [9].

## 3.4 Generalized Dataset Preprocessing

Our SA is a document sentiment classification based on supervised ML. After collecting corpus data, we need to preprocess the data for creating training and testing data. Because data preprocessing is an important part in the NLP domain. We use BLTK [57] version 1.2 in open source python PyPI package OSI approved, MIT License to preprocess our data. Dataset pre-annotation or preprocessing step is described below. This step will be applied to removing ambiguity and redundancy from the whole dataset.

### 3.4.1 Tokenization & Normalization

Splitting the sentence into a word list is called a tokenization process. Each token is called a word. For example: "আচারের সংযোজন খুব ভালো ছিল।" [The addition of the pickle was very

good], after tokenize this sentence it will create a list, as like [ ‘আচারের [pickle], ‘সংযোজন’ [addition], ‘খুব’ [very], ‘ভালো’ [good], ‘ছিল’ [was] ]. While doing tokenization process we have also finished normalizing the data. Normalizing means removing characters [',', '!', '!', '@', '#', '%'], etc. these and stop words [58] from the sentence. The characters and stop word will no impact on creating training, test data and ML model construction.

### 3.4.2 Stemming

Stemming means originating the root word from the given word list after doing the tokenization process. During the stemming process, we remove ‘র’, ‘এর’, ‘গুলি’, ‘গুলো’, ‘টার’, ‘টি’, etc. these unnecessary words from the sentence. For example: ‘স্বাধীনতার [Independent], ‘বাংলাদেশের’ [Bangladesh], ‘দুর্বলতাগুলির’ [Weaknesses] words convert the root word into respectively ‘স্বাধীনতা’, ‘বাংলাদেশ’, ‘দুর্বলতা’ by stemming process.

### 3.4.3 Parts of Speech (POS) Tagger

Detecting the word pos tagger in a sentence have a great significance calculating the score. Our Bangla text sentiment algorithm requires pos tagger to find out word weighted value from LDD. For example, ‘এটি খুব বেশি চিত্তাকর্ষক এবং খুব সুস্বাদু নয়।’ [It’s not too impressive and not too tasty]. After generating in python pos tagger, we will get a list of word with POS [এটি\_TP, খুব\_RB, বেশি\_JJ, চিত্তাকর্ষক\_NN, এবং\_CC, খুব\_RB, সুস্বাদু\_VB, নয়\_NA}. Here, বেশি\_JJ] word quantify the word [চিত্তাকর্ষক\_NN], therefore it will amplify the score of the text and [এবং\_CC] word connects two sentences which will be tracked by our BTSC algorithm.

## 3.5 Objectives & Methodology on DL

Figure 3.2 shows the pictorial representation of our full approach. We will implement LDD and BTSC algorithms. Although we will not deep dive into the core details like mathematical description and construction of the neural network architecture, we will try to summarize

our approach to the details construction of the neural network model that we have used in our experiment.

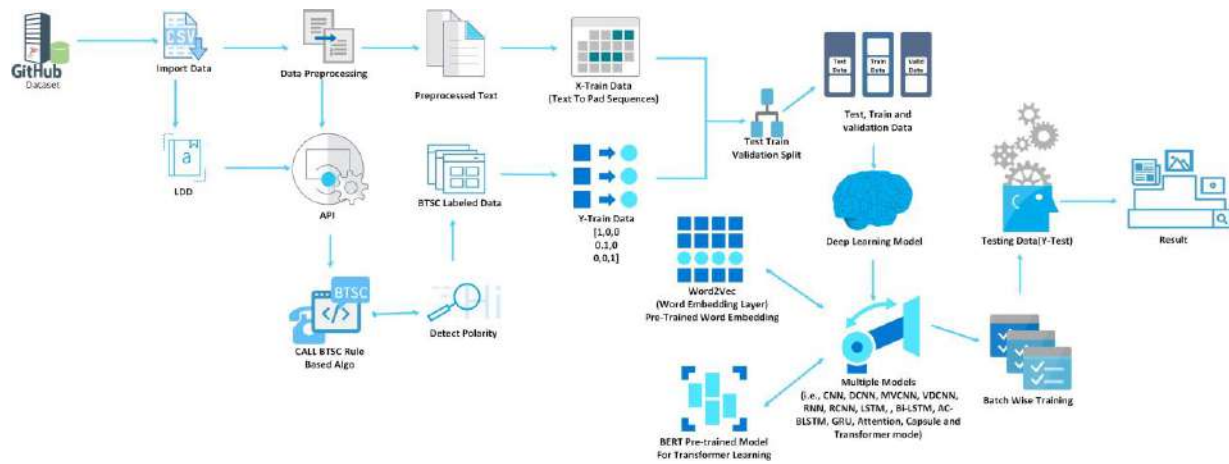


Figure 3.2: Visualization of DL Based Proposed System Architecture

### 3.5.1 DL Based Data Preprocessing:

According to our previous ML approach, we preprocess our data by removing stop words, unnecessary characters, performing tokenization, stemming and POS tagging operation. We have collected data from GitHub repository [53] and used cricket dataset for conducting our experiment. However, to represent the text into a neural network, we used a tensor- based matrix representation of the corpus (review) with its polarity. Compared to other text representation mechanisms, this sparse, dense matrix takes less memory for fitting in a neural network. We employ the Tensorflow neural network libraries simultaneously with Keras [59] for preprocessing our data. We use Keras tools such as tokenizer, text\_to\_sequences [60] and pad\_sequences [61].

#### 3.5.1.1 Neural Network Based Data Preprocessing:

In this work, we tokenize the words of our training data to keep a maximum number of words, text\_to\_sequences method to map the tokenized words in our vocabulary in a numeric representation. Then we find the maximum length (maxlen) of the text over encoded

sequences. Finally, the resulting encoded sequences need to be the same length (maxlen value) following the pad\_sequences approach. Extra 0's will be padded if the sequence is longer than the encoded sequence. Finally, the output of tensor data shape is  $[i_{\text{conpusLength}}, j_{\text{maxlen}}]$ , where indexes  $i$  and  $j$  denote, respectively, row and column. For example, Table 3.5 shows the full demonstration of data preprocessing for neural network training.

Table 3.5: Demonstration of neural network based data preprocessing

| Method                         | Data  | Comment   |
|--------------------------------|---|---|
| Text                           | জয় বাংলা কাপ! তাও আবার স্বাধীনতার মাস মার্চে।<br>যার মাথা থেকে এমন চমৎকার আইডিয়া এসেছে<br>তাকে স্যালুট। | Raw Text  |
| Tokenizer                      | ['জয়', 'বাংলা', 'কাপ', 'স্বাধীনতার', 'মাস',<br>'মার্চের', 'মাথা', 'চমৎকার', 'আইডিয়া', 'স্যালুট']        | Sentences are tokenized                                 |
| Stemming                       | ['জয়', 'বাংলা', 'কাপ', 'স্বাধীনতা', 'মাস', 'মার্চ',<br>'মাথা', 'চমৎকার', 'আইডিয়া', 'স্যালুট']           | Stemming word   |
| text_to_sequences              | [16, 170, 504, 81, 105, 450, 188, 64, 206,<br>4161, 788]  | Encoded each word as a numeric number representation    |
| pad_sequences<br>(maxlen = 40) | [16, 170, 504, 81, 105, 450, 188, 64, 206,<br>4161, 788, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, .....]             | padding the sequence as 40 length followed by extra 0's |

### 3.5.1.2 Data Preprocessing on Attention Based Mechanism:

The difference in our attention-based neural network data preprocessing [59] is dividing each sentence letting as a sequence followed by sentence piece tokenization method. This sequence is encoded as a numerical vector representation. We find the maximum length (maxSentLen) of each raw text sentence piece tokenization [60] for specifying out tensor data array length for training purposes. We count the maximum sequence length (maxSeqLen) in every sequence for padding over data. We padded over the sequence into extra 0's if the sequence is longer rather than the encoded sequence [61]. Finally, the output of tensor data shape is three dimensional  $[i_{\text{conpusLength}}, j_{\text{maxSentLen}}, k_{\text{maxSeqLen}}]$ , where indexes  $i$ ,  $j$  and  $k$  denote, respectively, row, column and height. Here Table 3.6 shows the whole process for attention-based mechanism data preprocessing in training on neural network approach.



Table 3.6: Demonstration on Attention Based Neural Network Data Preprocessing

| Method   | Data   | Comments   |
|--|--|--|
| Text   | জয় বাংলা কাপ! তাও আবার স্বাধীনতার মাস মার্চে। যার মাথা থেকে এমন চমৎকার আইডিয়া এসেছে তাকে স্যালুট।                                  | Raw Text   |
| Sentence Piece Tokenizer                               | [[ 'জয় বাংলা কাপ!',<br>'তাও আবার স্বাধীনতার মাস মার্চে।',<br>'যার মাথা থেকে এমন চমৎকার আইডিয়া এসেছে তাকে স্যালুট।' ]]              | 3 Sentence are tokenized                                 |
| Tokenization + Stemming                                | [[ 'জয়', 'বাংলা', 'কাপ',<br>'স্বাধীনতা', 'মাস', 'মার্চ',<br>'মাথা', 'চমৎকার', 'আইডিয়া', 'স্যালুট' ]]                               | Tokenize, Stemming word in every sentence                |
| text_to_sequences                                      | [[ [16, 170, 504],<br>[81, 105, 450],<br>[188, 64, 206, 788] ]]  | Encoded each sentence as a numeric number representation |
| pad_sequences<br>(maxSeqLen= 25<br>maxSentenceLen = 3) | [[ [16, 170, 504, 0, 0, 0, 0, 0, . . . ],<br>[81, 105, 450, 0, 0, 0, 0, 0, . . . ],<br>[188, 64, 206, 788, 0, 0, 0, 0, 0, . . . ] ]] | padding each sequence as 25 length followed by extra 0's |

### 3.5.1.3 Data Preprocessing on Transformer Neural Network (BERT) Based Mechanism:

There are some special tokens in the pre-trained language model for preprocessing in Transformer neural network BERT. In our experiment, we used the Bangla bert base from hugging-face library [62, 63], a PyTorch version to preprocess our text for learning in a transformer encoder network. The special tokens are shown in Table 3.7. [CLS] tokens are at the beginning of the sentence, [SEP] tokens are at the end of the sentences and [PAD] tokens are to pad and truncate the sentence in the maximum length of sentence in the corpus. First, we tokenize the sentence text using the transformer package BertTokenizer [64]. We use the encode\_plus [65] function to generate token\_ids, then convert\_ids\_to\_token and attention\_mask. The attention\_mask is used to identify which tokens are used (represented as 1) or not (represented as 0's). Finally, the input matrix is encoded as ['input\_ids', 'attention\_mask']. The whole demonstration is shown in Table 3.8.

Table 3.7: Classification of tokens in transformer neural network

| Token Name             | Identification | Id representation |
|------------------------|----------------|-------------------|
| Ending Sentence marker | [SEP]          | 102               |
| Classification Token   | [CLS]          | 101               |
| Padding Token          | [PAD]          | 0                 |

Table 3.8: Demonstration of transformer learning neural network based data preprocessing

| Method                           | Data   | Comment  |
|----------------------------------|--|--|
| Text                             | জয় বাংলা কাপ! তাও আবার স্বাধীনতার মাস মার্চে।<br>যার মাথা থেকে এমন চমৎকার আইডিয়া এসেছে<br>তাকে স্যালুট।  | Raw Text   |
| Preprocessed Text +<br>Stemming  | জয় বাংলা কাপ স্বাধীনতা মাস মার্চ মাথা চমৎকার<br>আইডিয়া স্যালুট   | Preprocessed text along with<br>stemming                   |
| tokens                           | ['জ', '##য', 'বাংলা', 'কাপ', 'সব', '##াধীন',<br>'##তা', 'মাস', 'মার', '##চ', 'মাথা', 'চমৎকার',<br>'আইডি', '##যা', 'স', '##যা', '##লট']   | Sentence are tokenized                                     |
| token_ids                        | tensor([101, 7360, 9294, 2492, 2991,<br>2132, 24484, 3274, 2416, 6723, 7464,<br>3755, 6162, 9709, 7724, 3091, 7724,<br>40654, 102, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,<br>0])  | Encoded each word as a nu-<br>meric number representation  |
| convert_ids_to_tokens            | ['[CLS]', 'জ', '##য', 'বাংলা', 'কাপ', 'সব',<br>'##াধীন', '##তা', 'মাস', 'মার', '##চ', 'মাথা',<br>'চমৎকার', 'আইডি', '##যা', 'স', '##যা',<br>'##লট', '[SEP]', '[PAD]', '[PAD]', '[PAD]',<br>[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',<br>[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]'] | Covert ids into token                                      |
| encoding ['atten-<br>tion_mask'] | tensor([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,<br>1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,<br>0])   | padding the sequence as 40<br>length followed by extra 0's |

### 3.5.2 Word Embedding:

In a neural network, word embedding is a measurement of language modelling and feature learning which maps the textual word into low dimensionality dense vectors. As a word embedding system, Word2Vec [66] research by Google is computationally efficient and practicable in a deep neural network model, which captures the semantic relations between

words by calculating the co-occurrence of words in a given corpus. It contains two models: a continuous bag of word (CBoW) [41] and Skip-Gram (SG) [67]. The procedure of CBoW model is to portend the current or target word from the neighbouring co-occurrence word, whereas the SG model portends the entire context word from the target word shown in Figure 3.3. The basic difference between these two models is that, in each target-context pair, a newly annotation is considered in the SG model, whereas the entire context as one annotation is considered in CBoW model. As our training data is relatively small, we work on SG algorithm to represent words in n-dimensional vector space. In our neural network model, we build three dimensional(D) vector space [128D, 200D, 300D] with a window size of 5 (window=5) which means the distance between the current and predicted word in a sentence, and the minimum length is 1 (min\_count=1) for our neural network models.

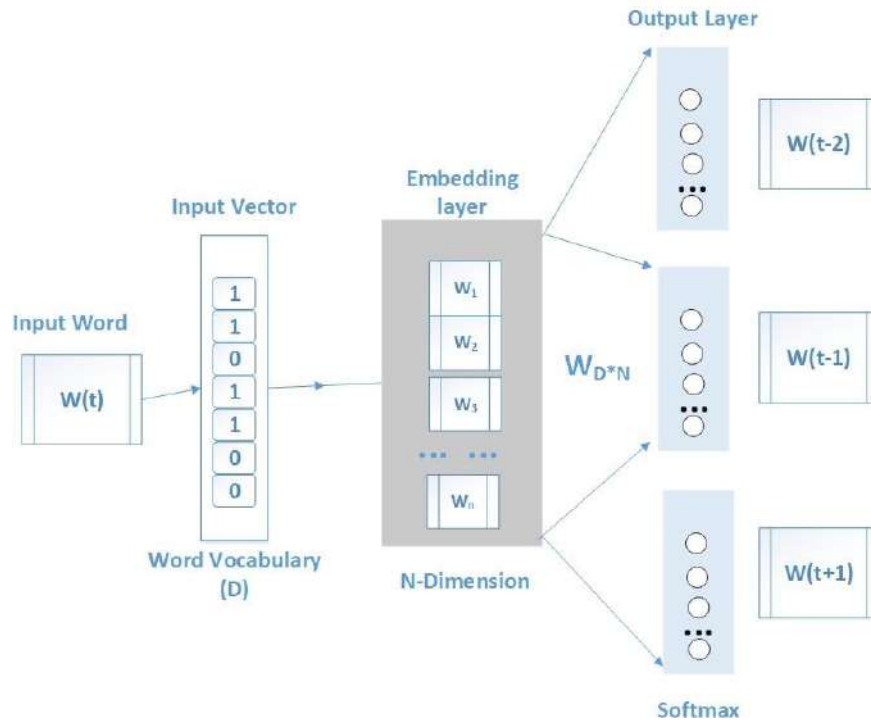


Figure 3.3: Word2Vec (Skip-Gram) Architecture Diagram

## 3.6 BTSC Algorithm

### 3.6.1 Discussion of Algorithmic Pseudocode

This section discusses the proposed novel BTSC algorithm which has a total of 30 steps. This BTSC algorithm is unique to Bangla and any other language. Out of the 30 steps, the unique steps are from steps 11 to 26 that manages the POS conjunction, adjective, adverb, punctuation and question marks. This Algorithm 1 termed as *BTSC* which is used for generating score from the sentence. The inputs, notations, output and pseudocode are described in below:

#### Inputs & Notations :

*DD*: Dataset Dictionary

*LDD*: Lexicon Dataset Dictionary[Active (Score = +1) & Contradict word (Score = -1)].

*JJ/RB*: Adjective or Adverb Word Quantifier Dictionary(3 types in dataset)[HIGH=3, MID=2, LOW=0.5]

*CC*: Conjunction Type Pos Tagger, *CD*: Co-ordinating & *CS*: Sub-ordinating Conjunction Word

*POS*: Parts of Speech, *PR*: Pronoun, *VB*: Verb, *NN*: Noun, *RB*: Adverb Type POS Tagger Word

*PU*: Punctuation (!) Character, *QM*: Question Mark (?) Character

*TP*: Transitional Preposition Word, *k*: count of negative word (initial, k=0), *SC[Word]*: Score of a Word

#### Output :

*SCS*: Score(SC) calculation form a Sentence (per sentence by sentence)

1. If *SCS* is  $> 0$ , Sentence polarity is Positive.
2. If *SCS* is  $= 0$ , Sentence polarity is Neutral.
3. If *SCS* is  $< 0$ , Sentence polarity is Negative.

---

**Algorithm 1** Bangla Text Sentiment Score Calculation (BTSC)

---

```

1: for each Sentence[i] in Dataset do
2:   for each Tokenize(word[j]) in Sentence do
3:     Remove(TP , PR)
4:     Scanning List of Word[j] from LDD
5:     if word[j] is Active word in LDD then
6:        $SC[Word[j]] = +1$ 
7:     else if word[j] is Contradict word in LDD then
8:        $SC[Word[j]] = -1$ 
9:     else if word[j] is a negative word in LDD then
10:       $k = k+1$ 
11:    else if word[j] is a CC type of POS tagger then
12:      if CD type word[j] occurs in a sentence then
13:         $SC[Word[j]] = +2$ 
14:      if CS type word[j] at the beginning of a sentence then
15:         $SC[Word[j]] = +1.5$ 
16:    else if word[j] is a JJ/RB POS tagger then
17:       $SC[Word[j]] =$  explore in JJ/RB type of POS tagger in DD to get word[j] score
18:    else if PU occurs at the Sentence[i] then
19:      if word[j-1] of PU is a Contradict word in LDD then
20:         $SC[Word[j]] = -2$ 
21:      else if word[j-1] of PU is a VB type of POS tagger then
22:         $SC[Word[j]] = -1$ 
23:    else if QM towards the end Sentence[i] and word[j-1] of QM is a VM type POS then
24:       $SCS[Sentence[i]] = -1$ 
25:    break
26:  else
27:     $SCS[Sentence[i]] = (-1)^k * [ SC[Word[j]] * SC[Word[j+1]] ... * SC[Word[j_n]] ]$ 
28:  end for
29:   $SCS = SCS[Sentence[i]] + SCS[Sentence[i+1]] + ... + SCS[Sentence[i_n]]$ 
30: end for

```

---

### 3.6.2 Score Calculation

To demonstrate our Algorithm 1 : *BTSC*, we have considered five examples from the cricket and restaurant datasets. We consider five tables for simulating our example scores, besides showing each word score, English translations, POS tagger and total final score. These tables are formatted below according to algorithm 1.

Ex 01: "জয় বাংলা কাপ স্বাধীনতা মাস মার্চ মাথা চমৎকার আইডিয়া স্যালুট "[ Not only is cooking great, the service has always been attentive and good.]

Table 3.9: Score calculation of Ex: 01

| List of Word        | রান্না | সেরা  | সেবা    | মনোযোগী   | এবং  | ভাল  | Final Score |
|---------------------|--------|-------|---------|-----------|------|------|-------------|
| English Translation | cook   | great | service | attentive | and  | good | (2)         |
| Word Score Value    | (+1)   | (+1)  | (+1)    | (+1)      | (+2) | (+1) |             |

Ex 02: "বাংলাদেশের ব্যাটিং বিপর্যয়।। ভাল লক্ষণ নয়।।" [Bangladesh's batting disaster. Not a good sign.]

Table 3.10: Score calculation of Ex: 02

| List of Word        | বাংলাদেশ   | ব্যাটিং | বিপর্যয় | ভাল  | লক্ষণ | নয়  | Final Score |
|---------------------|------------|---------|----------|------|-------|------|-------------|
| English Translation | bangladesh | batting | disaster | good | sign  | not  | (-2)        |
| Word Score Value    | (+1)       | (+1)    | (-1)     | (+1) | (+1)  | (-1) |             |

Ex 03: "সময় বাংলাদেশের ভাগ্যে ড্র রেখেছে, নিশ্চয়ই হার ছাড়া উপায় ছিলোনা!!"[Time has left a draw for the fate of Bangladesh, of course there wasn't a way without a defeat!!]

Table 3.11: Score calculation of Ex: 03

| List of Word        | বাংলাদেশ   | ভাগ্য | ড্র  | নিশ্চয়ই  | হার    | উপায় | ছিলোনা  | Final Score |
|---------------------|------------|-------|------|-----------|--------|-------|---------|-------------|
| POS Tagger          | NN         | NN    | NN   | CC        | NN     | VB    | VB      | (-3)        |
| English Translation | bangladesh | fate  | draw | of course | defeat | way   | was not |             |
| Word Score Value    | (+1)       | (+1)  | (-1) | (+1.5)    | (-1)   | (+1)  | (-2)    |             |

Ex 04: "খুব সীমিত আসন আছে এবং ঠিক সময়ে খাদ্য পাওয়ার জন্য যথেষ্ট অপেক্ষা করতে হবে।"[There are very limited seats and you have to wait long enough to get food on right time.]

Table 3.12: Score calculation of Ex: 04

| List of Word        | খুব  | সীমিত | আসন  | এবং  | ঠিক   | খাদ্য | পাওয়া | যথেষ্ট | অপেক্ষা | Final Score |
|---------------------|------|-------|------|------|-------|-------|--------|--------|---------|-------------|
| POS Tagger          | RB   | NN    | NN   | CC   | VB    | NN    | VB     | JJ     | VB      | (-18)       |
| English Translation | very | limit | seat | and  | right | food  | get    | enough | Wait    |             |
| Word Score Value    | (+3) | (-1)  | (+1) | (+2) | (+1)  | (+1)  | (+1)   | (+3)   | (+1)    |             |

Ex 05: "টেস্টে মশরাফির কি দোষটা ছিল? সে তো টেস্ট খেলেই না । এখানেও তাকে টেনে আনতে হবে?"

[What was Mashrafe's fault in the test? He doesn't even play Tests. Should he be dragged here too?]

Table 3.13: Score calculation of Ex: 05

| List Of Word        | টেস্টে | মশরাফি     | দোষ   | ছিল  | খেলা | না   | টেনে    | আনা    | হবে | Final Score |
|---------------------|--------|------------|-------|------|------|------|---------|--------|-----|-------------|
| POS Tagger          | NN     | NN         | NN    | VB   | NN   | NA   | VB      | VB     | VB  | (-3)        |
| English Translation | test   | Mashrafe's | fault | was  | play | not  | dragged | Should |     |             |
| Word Score Value    | (+1)   | (+1)       | (-1)  | (+1) | (+1) | (-1) | (-1)    | (+1)   |     |             |

### 3.6.3 Simulation of BTSC Algorithm

The input for our Algorithm 1 is a list of sentences considered in the dataset. Line 1 in the algorithm considers each text or sentence which score will be calculated. At lines 2 and 3, tokenizing sentences along with stemming, removing transitional preposition (TP) word, parts of speech (POS) tagging processes are performed. Here TP word i.e., 'শুধুমাত্র' [only], 'না হয়' [or else], 'না তো' [not at all], 'তা নয়' [not that], 'সেইজন্য' [that's why], 'তবুও কেন' [yet why] have not any significance in Bangla sentence for calculating score. At line 4, we scan every preprocessed word in each sentence from the LDD. With the help of LDD, we have found the weight score values of each active and contradict words at lines 5 to 8. As 'না', 'নেই', 'নাই' [not] are negative dictionary words in the LDD, the k counter is automatically incremented at line 9 to 10. At lines 11 to 17, the POS conjunction, adjective and adverb are managed. The rules for punctuation and question mark characters are set at lines 18 to 24. The sentiment of a single sentence is calculated by multiplying each word score at line 27, and the total polarity of a whole paragraph score is calculated at line no 29 by adding each sentence score.

### 3.6.3.1 Illustration of Example on BTSC Algorithm

A number of examples are shown to demonstrate the BTSC algorithm. In the first example shown in Table 3.9, ‘মনোযোগী’ [attentive], ‘সেবা’ [service] are active words and ‘বিপর্যয়’ [disaster], ‘হার’ [defeat] are negative words in LDD, these word scores are calculated at lines 6 and 8, respectively. In this case, there is one sentence and the score value is the multiplication of individual scores resulting in (+2). Now, example 2 is demonstrated in Table 3.10 from the cricket dataset. Here are two sentence, first (i = 1) sentence [‘বাংলাদেশের ব্যাটিং বিপর্যয়’ [Bangladesh's batting disaster]] score is (-1) and the second (i = 2) sentence [‘ভালো লক্ষণ নয়’ [Not a good sign]] score is (-1) and final total score of this phrase is (-1)+(-1)=(-2) which is calculated at line 29. In the third example as shown in Table 3.11, one word ‘নিশ্চয়ই’ [of course], is a CC (CS type word) POS tagger and this score value is obtained from lines 14 to 15. There is a contradict word i.e., ‘ছিলোনা’ [was not] before the punctuation(!) character, the score of this word is calculated at line 21 to 22. In this case there is only one sentence and the score is (-3).

In example 4, shows in Table 3.12, there is one sentence, ‘এবং’ [and] is a CC (CD type word). This is calculated at lines 11 to 13. There is ‘যথেষ্ট’ [enough] as adjective (JJ) and ‘খুব’ [very] as adverb (RB) quantifier POS tagger word. The score of the words is calculated from line 16 to 17. The final score is calculated as (-18) after multiplying the individual scores.

In example 5, shows in Table 3.13, there are three sentences. A question mark (QM) occurs at the end of the first (i=1) [‘টেস্টে মশরাফির কি দোষটা ছিল?’ [What was Mashrafe's fault in the test?]], and there is a ‘ছিল’ [was] VB type POS tagger before a QM. So, this sentence has a negative meaning due to the presence of a QM after VB POS tagger. The score of the first sentence is (-1). The score of the second sentence is (-1) executed at lines 5 to 10. In the third (i=3) sentence [‘এখানেও তাকে টেনে আনতে হবে?’ (Should he be dragged here too?)] sentences there is a ‘হবে’ [Should] VB type POS tagger before a QM, So, this sentence has negative meaning due to the presence of a QM after VB POS tagger. The score of this sentence is (-1). The score of the first sentence and the third sentences are (-1) executed



as lines 23 to 24. Finally, the total calculated score of the three sentences are summed as  $(-1)+(-1)+(-1)=(-3)$ .

### 3.7 Data Augmentation

Data augmentation(DA) refers to a technique used to expand the quantity of data by manipulating or adding slightly changes to existing data. To influence in training of the DNN model, DA is a practical use case for preventing noisy data or overfitting. DA can explore the advancements of supervised learning without labeled data. Without learning from domain-based labeled data, DA enables the interface of label-based data transformation means self-supervised learning.

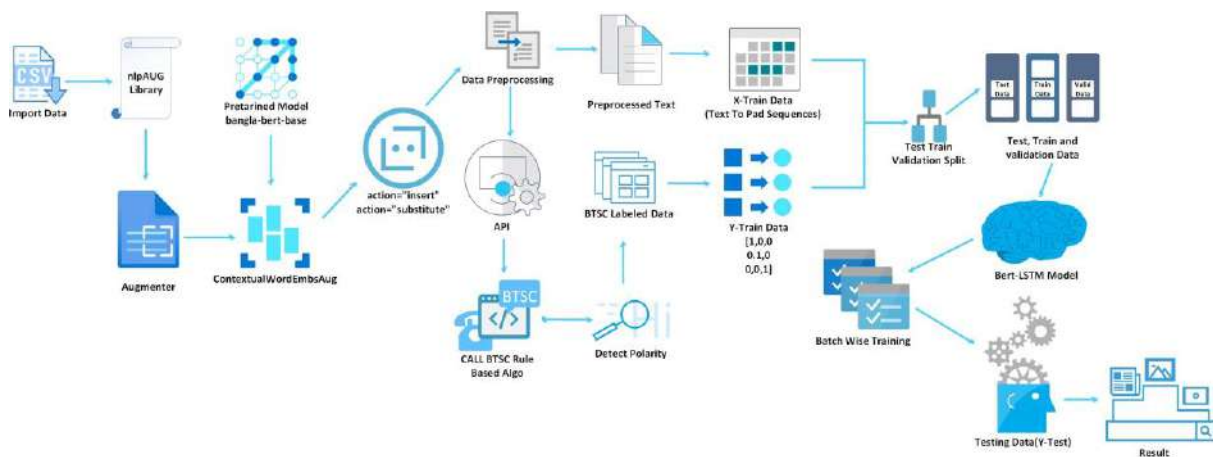


Figure 3.4: Visualization of DL Based Proposed System Architecture

In classification of text or images DA technique is used for rising the performance of DL models. In our experiment, we have used lexicon data dictionary (LDD) from our previous work. We have done prerequisite preprocessing part in building LDD from dataset. It includes noise reduction, substitute words in lexicon, words shuffling mainly used for short text which is mostly related to data sampling analysis. BTSC is used for detecting score from large text that is why we do not need any hybrid data augmentation method for generalization our text. By doing word shifting in sentence increases the data in training samples. We have

used a categorical aspect-based dataset (cricket) that means a comment has a positive on x category and negative on y category or neutral on z category with specific polarity but BTSC algorithm detects polarity only extracts sentiments according to our global extended lexicon dictionary not to use in categorical sentimental dictionary. The process of DA is depicted on Figure 3.4. As there is not enough data [53] to produce high-quality classifiers, we apply NLP a data augmentation technique to solve this issue. We apply contextualized word embedding techniques [68] to extend our dataset. We modify the data by inserting and substituting a word by Bangla-bert base [62] contextualized word embedding process. Table no 3.15 shows the DA process.

### 3.7.1 Dataset Creation

After applying data augmentation technique, we have created a merged dataset. The total merged dataset is of 15,114 samples. The total construction of the merged dataset is shown in in Table 3.14.

Table 3.14: Statistical polarity of data augmentation in cricket and restaurant datasets with individual and total comments

| No                            | Dataset           | Total Dataset |
|-------------------------------|-------------------|---------------|
| Cricket Main Data             | Raw Text          | 2059          |
| Augmented Data                | Insert Method     | 4118          |
|                               | Substitute Method |               |
| Restaurant Main Data          | Raw Text          | 2979          |
| Augmented Data                | Insert Method     | 5958          |
|                               | Substitute Method |               |
| Augmented Dataset (summation) |                   | 15114         |

For example, in Table 3.15, main data-3 raw text “বাংলাদেশের পরে ভারতের সাপোর্ট ই করি?।” [Support India after Bangladesh?], when applying insert method “স্বাধীনতার” [independent] word is inserted in the raw text. Then this text will be as “বাংলাদেশের স্বাধীনতার পরে করি ভারতের সাপোর্ট?।” [Support India after the independence of Bangladesh?]. Again, applying substitute method in the main data-3 raw text, “ভারতের” [India] word is substituted by

“জনগণ” [people] “কতটা” [much] words and finally this will be emerged as “বাংলাদেশের জনগণ কতটা সাপোর্ট ই করি? বিবিসি।” [How much do the people of Bangladesh support? BBC]. There is a polarity difference in augmented data. In Table 3.15, the insert method augmented text polarity is positive, and the substitute-based augmented text polarity is negative.

Table 3.15: DA process in contextualized word embedding (Bangla-Bert) with BTSC polarity

| No             | Dataset           | Comments  | BTSC Polarity |
|----------------|-------------------|---|---------------|
| Main Data-1    | Raw Text          | পরিমিত থাই খাদ্য - যদিও একটি নরম টুকরা - সামান্য ঘুরা ফিরা, কিন্তু সেবা ভাল।  | Positive      |
| Augmented Data | Insert Method     | পরিমিত <b>শুকনো</b> থাই <b>প্রকার</b> খাদ্য - যদিও একটি <b>ছোট</b> নরম টুকরা - <b>যেমন</b> সামান্য ঘুরা <b>সাথে</b> ফিরা, কিন্তু বাড়িতে সেবা নিলে ভাল হবে। | Positive      |
|                | Substitute Method | <b>কম পরিমাণে</b> খাদ্য আছে - যদিও <b>কম শক্ত</b> টুকরা - ঘুরা ফিরা থেকে কিন্তু ভাল।  | Negative      |
| Main Data-2    | Raw Text          | জয় বাংলা কাপ! তাও আবার স্বাধীনতার মাস মার্চে। যার মাথা থেকে এমন চমৎকার আইডিয়া এসেছে তাকে স্যালুট।   | Positive      |
| Augmented Data | Insert Method     | জয় হলো বাংলা <b>ফুটবল</b> কাপ! তাও তার আবার স্বাধীনতার মাস মার্চে আছে। যার মতে মাথা থেকে এমন <b>বহু</b> চমৎকার আইডিয়া আর এসেছে তাকে এক স্যালুট।           | Positive      |
|                | Substitute Method | আমার বাংলা <b>নববর্ষ!</b> শুরু আবার <b>সেপ্টেম্বর</b> মাস থেকে। যার মাথা থেকে কোনো চমৎকার আইডিয়া এসেছে সেই <b>বলুক</b> ।                                   | Positive      |
| Main Data-3    | Raw Text          | বাংলাদেশের পরে ভারতের সাপোর্ট ই করি?।   | Positive      |
| Augmented Data | Insert Method     | বাংলাদেশের <b>স্বাধীনতার</b> পরে করি ভারতের সাপোর্ট?।   | Positive      |
|                | Substitute Method | বাংলাদেশের <b>জনগণ কতটা</b> সাপোর্ট ই করি? <b>বিবিসি</b> ।  | Negative      |

After creating this augmented dataset, we merged the cricket and restaurant datasets and applied the BTSC algorithm to detect sentiment. Then we check the grammatical spell using by spell checker method.

### 3.7.2 Creation of Augmented Dataset Algorithm

#### Inputs & Notations :

- (i) *Pre\_trained\_model\_path* = "sagorsarker/bangla-bert-base"
- (ii) *ContextualWordEmbsAug*: Contextualized Word Embedding Function Augmented Dataset
- (iii) *INSERT* = *ContextualWordEmbsAug*(*Pre\_trained\_model\_path*, action="insert")
- (iv) *SUBSTITUTE* = *ContextualWordEmbsAug*(*Pre\_trained\_model\_path*, action="substitute")
- (v) *SpellWordFrequencyList* = Load Words of Frequency to Check the spell of a Word

#### Output :

*AugmentedDatasetList*: Cricket and Restaurant Merged Augmented Dataset

---

#### Algorithm 2 Augmented Dataset Algorithm

---

```

1: for each corpus[i] in Dataset do
2:   augmented_text = INSERT.augment(corpus[i])
3:   augmented_text = SUBSTITUTE.augment(corpus[i])
4:   augmentedDataList.append(augmented_text)
5: end for
6: SPELL = SpellChecker()
7: for each Sentence[i] in augmentedDataList do
8:   SpellWordFrequencyList = SPELL.word_frequency.load_words()
9:   for each Word[j] in SpellWordFrequencyList do 10:
10:    check_spell = SPELL.correction(Word[j])
11:    if check_spell is True then:
12:      Sentence[i] = append the spell-checked Word[j] in i – th Sentence
13:    else if check_spell is False then:
14:      Sentence[i] = no changes in Word[j]
15:    end for
16:   augmentedDataList = Sentence[i]
17: end for

```

---

#### 3.7.2.1 Description of Augmented Dataset Algorithm

The algorithmic process of DA is described below. In the inputs section, at line number (i), pretrained model path is set. In lines (ii), the Contextualized Word Embedding function

for augmented dataset is declared. In lines (iii) and (iv), the insert and substitute actions are called through contextualized word embedding function. In lines (v), the spell-checking word frequency is loaded to check the spell of a word. In lines 1 to 5, we iterate through every text of both datasets (restaurant and cricket) to produce augmented text by insert and substitute method and, finally, merge it into an augmented list. Then we check our grammatical font errors in lines no 6 to 17. For example, 'স্বাধীনতার' [independent], 'সাপোর্ট' [support], 'স্যালুট' [salute] fonts are misspelled as 'সবাধীনতার' [independent], 'সাপরট' [support] 'সযালট' [salute] during augmenting text. In line no 11, we load every possible combination of j-th word vectors, then it is corrected and spelled in line no 13. The corrected spelling word is appended in the i-th sentence if the word is suitable for the misspelled word, and checking activities are performed in lines 11 to 14.

### 3.8 Conclusion

In this section, we discuss the methodology of ML and DL-based approaches by following a proper dataset preprocessing mechanism. We describe building a LDD by appointing a specific weight on each POS category. We develop our BTSC as a rule-based algorithm for extracting our polarity from text and simulate our algorithm using some examples. We merged our cricket and restaurant dataset by applying augmentation technique and develop a augmented dataset algorithm. We will experiment with our newly developed rule-based algorithm in the next chapter.

# Chapter 4

## Experiments

### 4.1 Overview

This section describes the classical ML and DL approaches explored for SA from Bangla text, and the model overview and experimental setup in each case. We discuss the experiments based on the word embedding pretrained model. In ML approach, we trained a number of text classification algorithms so that we could compare them and draw further inferences. In DL approach, many layers, neural networks, frameworks and approaches have been proposed for SA from text. We applied our proposed BTSC algorithm polarity in training data in ML, and DL approaches. The rest of this section discusses the details of the steps used for construction, discusses the features used followed by in-depth view of the aspect model.

### 4.2 Experiment on BTSC Algorithm

After applying the BTSC algorithm on both datasets, we construct a confusion matrix (CM) based on positive, negative and neutral polarity labels shown on Table 4.1 and 4.2. From Table 4.1, a total of 1067 and 398 comments is identified out of 1216 positive and negative comments in restaurant dataset. Similarly, from Table 4.2, 547 and 1905 comments are identified out of 620 positive and negative comments in the cricket dataset. From these both

Table 4.1: Polarity detection by BTSC on restaurant data

| True\Predicted |    | Predicted Label |     |    | Total |
|----------------|----|-----------------|-----|----|-------|
|                |    | +1              | -1  | 0  |       |
| True Label     | +1 | 1067            | 140 | 9  | 1216  |
|                | -1 | 74              | 398 | 6  | 478   |
|                | 0  | 242             | 63  | 60 | 365   |
| Total          |    | 1383            | 601 | 75 | 2059  |

Table 4.2: Polarity Detection by BTSC on Cricket Data

| True\Predicted |    | Predicted Label |      |     | Total |
|----------------|----|-----------------|------|-----|-------|
|                |    | +1              | -1   | 0   |       |
| True Label     | +1 | 547             | 67   | 6   | 620   |
|                | -1 | 186             | 1905 | 17  | 2108  |
|                | 0  | 61              | 39   | 151 | 251   |
| Total          |    | 794             | 2011 | 174 | 2979  |

CM, it can be inferred that the BTSC rule-based algorithm has been able to detect sentiment fairly accurately except the neutral sentiments. Because the total dataset comments polarity are voted based on category based. The maximum neutral data is manually generated above the aspect based category. It means a comment has a positive on x category and negative on y category or neutral on z category.

Table 4.3: Neutral data detection problem

| No. | Comments  | Category | Polarity | BTSC Algorithm Polarity |
|-----|---|----------|----------|-------------------------|
| 1   | "পরিমিত থাই খাদ্য - যদিও একটি নরম টুকরা - সামান্য ঘুরা ফিরা, কিন্তু সেবা ভাল"।<br>Moderate Thai food - although a soft piece - turns slightly, however the service is good. | Food     | Positive | Positive                |
|     |   | Service  | Negative |                         |
|     |   | Ambience | Neutral  |                         |
| 2   | "বোলিং পিচ তবে আমাদের ব্যাটসম্যানদের আউটগুলোই আত্মহত্যা ছাড়া আর কিছুই নয়।"Bowling pitch, but the batsmen outs are nothing but suicide.                                    | Batting  | Negative | Negative                |
|     |   | Bowling  | Neutral  |                         |

Consider that example from Table 4.3: row 1 is taken from restaurant dataset that has three categories on three polarities, and row 2 is taken from cricket dataset that has two categories on two polarities. However, the BTSC algorithm only extracts sentiments according to a global extended lexicon dictionary not used in a categorical sentimental dictionary. For this reason, the calculation of neutral sentiments will be challenging to check.

### 4.2.1 Metrics Evaluation

From Table 4.1 and 4.2, we calculate some parameter measurements named as true positive rate ( $TPR$ ), true negative rate ( $TNR$ ) and false positive rate ( $FPR$ ). In order to keep consistency with the relevant literature of natural language processing [69, 70], we have used true positive rate ( $TPR$ ), which is also known as recall or sensitivity indicated as equation (4.1) below. It is measured by the ratio of the true positive ( $TP$ ) of a particular label to the sum of its true positive ( $TP$ ) and false negative ( $FN$ ).  $TNR$  is also known as specificity, which is measured as the ratio of true negative ( $TN$ ) of a particular label to the sum of the true negative ( $TN$ ) and false positive ( $FP$ ), shows in equation (4.2).  $FPR$  is known as type II error which is calculated by the ratio of the false positive ( $FP$ ) of a particular label to the sum of the false positive ( $FP$ ) and true negative ( $TN$ ), shows in equation (4.3).

$$TPR(label) = \frac{TP}{TP + FN} \quad (4.1)$$

$$TNR(label) = \frac{TN}{TN + FP} \quad (4.2)$$

$$FPR(label) = \frac{FP}{FP + TN} \quad (4.3)$$

These formulas are extracted by the concept of  $TP$ ,  $TN$ ,  $FP$ ,  $FN$ . Basically,  $TP$  is a correctly predicted class,  $TN$  is a correctly predicted non-class,  $FP$  is an incorrectly predicted class and  $FN$  is an incorrectly predicted non-class. Before calculating equation (4.1), (4.2), and (4.3), we need to consider Table 4.4 for stepping out these formulas. Here,  $C(x, y)$  notation for each box is introduced to measure the parameters for CM. In  $C(x, y)$ ,  $x$  is a predicted label or class and  $y$  is a true label or class. Calculation of  $TPR$ ,  $TNR$  and  $FPR$  for negative label (-1) is shown below at equation (4.4), (4.5) and (4.6).



Table 4.4:  $C(x, y)$  notation for indicating the parameters of CM

| <b>True\Predicted</b> |    | <b>Predicted Label</b> |            |           |
|-----------------------|----|------------------------|------------|-----------|
|                       |    | +1                     | -1         | 0         |
| <b>True Label</b>     | +1 | $C(1,1)$               | $C(-1,1)$  | $C(0,1)$  |
|                       | -1 | $C(1,-1)$              | $C(-1,-1)$ | $C(0,-1)$ |
|                       | 0  | $C(1,0)$               | $C(-1,0)$  | $C(0,0)$  |

Here Considering for negative labels,

$$TP = C(-1, -1),$$

$$TN = C(1, 1) + C(0, 1) + C(1, 0) + C(0, 0),$$

$$FP = C(-1, 1) + C(-1, 0),$$

$$FN = C(1, -1) + C(0, -1).$$

Finally we get  $TPR$ ,  $TNR$  and  $FPR$  from equation number (4.4), (4.5) and (4.6), respectively.

$$TPR(-1) = \frac{C(-1, -1)}{C(-1, -1) + C(1, -1) + C(0, -1)} \quad (4.4)$$

$$TPR(-1) = \frac{C(1, 1) + C(0, 1) + C(1, 0) + C(0, 0)}{C(1, 1) + C(0, 1) + C(1, 0) + C(0, 0) + C(-1, 1) + C(-1, 0)} \quad (4.5)$$

$$TPR(-1) = \frac{C(-1, 1) + C(-1, 0)}{C(-1, 1) + C(-1, 0) + C(1, 1) + C(0, 1) + C(1, 0) + C(0, 0)} \quad (4.6)$$

Similarly, other labels will be calculated in this way. A full summary of the calculation is shown in Figure 4.1. In these measurements,  $TPR$  is above average at 85%, which signifies our dictionary and BTSC algorithm efficacy. At most, 90%  $TPR$  at a negative label is obtained in the restaurant dataset, and 87%  $TPR$  at a positive label is obtained in the cricket dataset. As the high rate of  $TPR$  has a low rate of  $TNR$ , both will be preferable in better performance.  $TNR$  and  $TPR$  are better on positive and negative labels however not

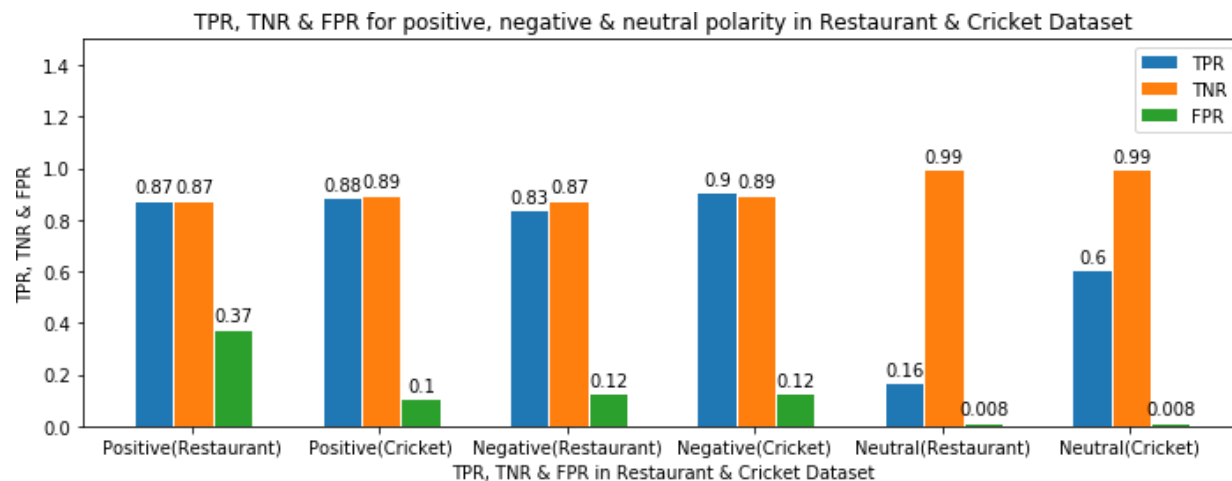


Figure 4.1: Visualization performance of **BTSC** algorithm in restaurant and cricket dataset

in neutral datasets because of the categorical identification polarity. In neutral comments, 60% and 16% *TPR* are carried in the cricket and restaurant dataset, respectively.

### 4.3 Experiment on ML Approach

To evaluate our experiment, we used supervised ML classification algorithm to classify our data. At least 20% of the dataset have been randomly chosen for testing dataset and rest of the data is trained for classifying the polarity. The evaluation of our result is measured through a CM including the classifier metrics called accuracy, precision, recall and f1-Score with the help of using Spyder, python IDE environment. Among the classifier, SVM with linear kernel trick ( $c=1$ ) is the best for giving proper result in new observations because SVM has found better accuracy in finding text classification.

#### 4.3.1 Term Frequency - Inverse document Frequency (TF-IDF)

A standard feature matrix called term frequency - inverse document frequency (TF-IDF) vectorizer is used to calculate the feature matrix. It maps text or word into a significant representation number. Tf-Idf is an algorithm that inspects every core word in a document

and find out the most necessary keywords from the document. It is developed for the documented analysis and retrieval of information from text.

Lets, define some notations, given a corpus  $D$ , a term  $t_i$  and a document  $d_j \in D$ , we denote the number of occurrences of  $t_i$  in  $d_j$  by  $tf_{ij}$ . This is referred as the term frequency.

$$tf_{ij} = \left( \frac{t_i}{d_j} \right) \quad (4.7)$$

The inverse document frequency for a term  $t_i$ , denote as  $idf_i$ , is defined on Equation 4.8

$$idf_i = \log \left[ \left( \frac{|D|}{|d : t_i \in d|} \right) + 1 \right] \quad (4.8)$$

where  $|D|$  is the number of documents in corpus, and  $|d : t_i \in d|$  is the number of documents in which the term appears [71].

### 4.3.2 Construction of TF-IDF Matrix

The construction of TF-IDF matrix is followed by Table 4.5. In row wise, the number of documents containing words are placed. The number of documents in which word appears are placed in the column wise. This is one kind of sparse matrix which contains zeros in several indexes.

Table 4.5: TF-IDF matrix format

| <b>Document/Word</b> | <b>Word-1</b> | <b>Word-2</b> | <b>Word-3</b> | . | . | . | <b>Word-30</b> |
|----------------------|---------------|---------------|---------------|---|---|---|----------------|
| <b>DOC-1</b>         | D1/W1         | D1/W2         | D1/W3         | . | . | . | D1/W30         |
| <b>DOC-2</b>         | D2/W1         | D2/W2         | D2/W3         | . | . | . | D2/W30         |
| <b>DOC-3</b>         | D3/W1         | D3/W2         | D3/W3         | . | . | . | D3/W30         |
| <b>DOC-4</b>         | D4/W1         | D4/W2         | D4/W3         | . | . | . | D4/W30         |
| .                    | .             | .             | .             | . | . | . | .              |
| .                    | .             | .             | .             | . | . | . | .              |
| .                    | .             | .             | .             | . | . | . | .              |
| <b>DOC-10</b>        | D10/W1        | D10/W2        | D10/W3        | . | . | . | D10/W30        |

### 4.3.2.1 TF-IDF Matrix Calculation from Restaurant Dataset

Data shown in Table 4.6 is from restaurant dataset for calculating tf-idf matrix. At first, we need to preprocess our text according to text preprocessing section followed by tokenization, stemming and stop word removal procedures. Here Table 4.5 shows the demonstration of DOC-10 and DOC-5 texts preprocessing mechanisms. Then we calculate word or term frequency (TF) according to equation (4.7) and calculate word inverse document frequency (IDF) according to equation (4.8).

Table 4.6: Sample data from restaurant dataset

| DOC No. | SENTENCE   |
|---------|--|
| DOC-1   | এটি খুব বেশি চিত্তাকর্ষক এবং খুব সুস্বাদু অনুভূতি নয়।   |
| DOC-2   | যাইহোক, খাদ্য গুলো খুব ভাল।  |
| DOC-3   | চমৎকার ভাল ওয়াইন তালিকা।  |
| DOC-4   | মরুভূমি পরিষেবা খুব আনন্দদায়ক, চমৎকার বায়ুমণ্ডল ছিল।   |
| DOC-5   | যদিও অর্ডার এবং খাদ্য পেতে অপেক্ষা ছিল, পরিষেবা ধীরেধীরে ভিড় না বাড়ে।                          |
| DOC-6   | অবিশ্বাস্য স্থান এত শীতল এবং পরিষেবা যে প্রম্পট এবং বিনয়ী।                                      |
| DOC-7   | আপনি একটি সম্পূর্ণ অভিজ্ঞতা অনুভূতি দিয়ে মেনু থেকে খাবার অর্ডার দেন।                            |
| DOC-8   | অনেক অর্ডার তাই দ্রুত খাবার খাও,।  |
| DOC-9   | পরিষেবা অবিশ্বাস্য এবং গ্রেট ভারতীয় খাদ্য।  |
| DOC-10  | আনন্দদায়ক না হলেও - আনন্দময় খাবার এবং স্থান একটি রঙিন রঙ আনন্দময় পরিবেশ - মরুভূমি বায়ুমণ্ডল। |

Table 4.7: Sample stop word list from restaurant dataset

| Stop Word List  |
|---|
| এটি, যাইহোক, গুলো, ছিল, না, যাতে, এবং, যদিও, এত, দিয়ে, যে, একটি, আপনি, সম্পূর্ণ, থেকে, দেন, তাই, অনেক, হলেও, হয় |

Here in Table 4.10, the word “আনন্দময়” occurs two times in the 10-th document (DOC-10), and the word “পরিষেবা” occurs one time in the 5-th document (DOC-5) represented as (a). At table 4.10, DOC-10 has a length of word 14, and DOC-5 has a length of 12 represented as (b). At Table 4.10, the number of documents length (d) is ten as there are ten documents (DOC-1 to DOC-10) in the Table 4.6. Here, the word “আনন্দময়” occurs in every document (DOC-1 to DOC-10) only 1 time and word “পরিষেবা” occurs in every document (DOC-1 to DOC-10) 4 times represented as (c).

### 4.3.2.2 Formation of Term Frequency and Inverse Document Frequency

The term frequency (TF) will be calculated as X, and inverse document frequency (IDF) will be calculated as Y, shown at Table 4.11. Then we calculate the term frequency matrix (TF) represented by word by document matrix, shows as Table 4.8 which is stated as the frequency of i-th document word divides the length of that i-th document tokens.

Table 4.8: Representation of word by document matrix

| Word/ Document | DOC-1  | DOC-2  | DOC-3  | . | . | . | DOC-10  |
|----------------|--------|--------|--------|---|---|---|---------|
| Word-1         | W1/D1  | W1/D2  | W1/D3  | . | . | . | W1/D10  |
| Word-2         | W2/D1  | W2/D2  | W2/D3  | . | . | . | W2/D10  |
| Word-3         | W3/D1  | W3/D2  | W3/D3  | . | . | . | W3/D10  |
| Word-4         | W4/D1  | W4/D2  | W4/D3  | . | . | . | W4/D10  |
| .              | .      | .      | .      | . | . | . | .       |
| .              | .      | .      | .      | . | . | . | .       |
| .              | .      | .      | .      | . | . | . | .       |
| Word-30        | W30/D1 | W30/D2 | W30/D3 | . | . | . | W30/D10 |

Table 4.9: Dataset preprocessing for TF-IDF matrix construction

| Document | No.    | Process                        | Output  |
|----------|--------|--------------------------------|---|
| DOC-10   | EX: 01 | Tokenization and Normalization | ['আনন্দদায়ক', 'না', 'হলেও', 'আনন্দময়', 'খাবার', 'এবং', 'একটি', 'রঙিন', 'রঙের', 'আনন্দময়', 'পরিবেশে', 'মরুভূমির', 'বায়ুমণ্ডল'] |
|          |        | Stemming                       | ['আনন্দদায়ক', 'না', 'হলেও', 'আনন্দময়', 'খাবার', 'এবং', 'একটি', 'রঙিন', 'রঙ', 'আনন্দময়', 'পরিবেশ', 'মরুভূমি', 'বায়ুমণ্ডল']     |
|          |        | Stop Word Removal              | ['আনন্দদায়ক', 'আনন্দময়', 'খাবার', 'রঙিন', 'রঙ', 'আনন্দময়', 'পরিবেশ', 'মরুভূমি', 'বায়ুমণ্ডল']                                  |
| DOC-5    | EX: 02 | Tokenization and Normalization | ['যদিও', 'অর্ডার', 'এবং', 'খাদ্য', 'পেতে', 'অপেক্ষা', 'ছিল', 'পরিষেবা', 'ধীরেধীরে', 'ভিড়', 'না', 'বাড়়ে']                       |
|          |        | Stemming                       | ['যদিও', 'অর্ডার', 'এবং', 'খাদ্য', 'পেত', 'অপেক্ষা', 'ছিল', 'পরিষেবা', 'ধীরেধীরে', 'ভিড়', 'না', 'বাড়়']                         |
|          |        | Stop Word Removal              | ['অর্ডার', 'খাদ্য', 'পেত', 'পরিষেবা', 'ধীরেধীরে', 'ভিড়', 'বাড়়']  |

The full representation of TF-IDF matrix calculation is shown in the appendix section. The calculation of word frequency and word inverse document frequency is shown at Table A.1(a) and A.1(b) respectively. The TF matrix transposes the IDF matrix from Table A.2. Table A.3 is produced by the transpose of Tf matrix from Table A.2. Finally, X and Y are multiplied

Table 4.10: Calculation of terms in each document

| Document | Word     | No. of occurrences of a word in a doc (a) | No. of words in that doc (b) | No. of Documents containing word (c) | No. of Documents (d) |
|----------|----------|---|------------------------------|--------------------------------------|----------------------|
| DOC-10   | আনন্দময় | 2   | 14                           | 1                                    | 10                   |
| DOC-5    | পরিষেবা  | 1   | 12                           | 4                                    | 10                   |
| DOC-5    | খাদ্য    | 1   | 12                           | 3                                    | 10                   |

Table 4.11: Calculation of term frequency [X], inverse document frequency [Y] and TF-IDF [X\*Y] value

| Document | Word     | TF Value (a/b) = [X] | IDF Value $\log((c/d)+1) = [Y]$ | TF-IDF value [X*Y] |
|----------|----------|----------------------|---------------------------------|--------------------|
| DOC-10   | আনন্দময় | 0.14285              | 2.39789                         | 0.342538           |
| DOC-5    | পরিষেবা  | 0.08333              | 1.2527                          | 0.104387           |
| DOC-5    | খাদ্য    | 0.08333              | 1.46633                         | 0.122189           |

to produce our term frequency - inverse document frequency matrix (TF-IDF) shown in Table A.4. Similarly, a sample form cricket dataset Table A.5, the overall construction of b word frequency and inverse document frequency matrix shows at Table A.6, term frequency matrix shows at table A.7 inverse frequency matrix at Table A.8 and TF-IDF feature matrix shows at Table A.9.

## 4.4 Experiment on DL approach

Our main aim of rule-based DL sentiment extraction experiments is to analyze the effectiveness of the well-known DL models. We used *Tensorflow* == 2.4.1, *Keras* == 2.4.3 and *Transformer* == 3.0 for developing our DL model. The cricket dataset [53] with a total of 2978 entries was divided into three parts, namely training, validation and testing, distributed as follows: 80% (2412 entries) for model training and 20% (566 entries) distributed for validation (268 entries) and testing (298 entries). The model is not trained on the validation dataset. The validation dataset accuracy determines how well the model will perform on test data. The definition of training, test, validation set from [72] are defined below:

- (i) Training set: A set of examples used for learning, that is to fit the parameters of the classifier.
- (ii) Validation set: A set of examples used to tune the hyper-parameters of a classifier.
- (iii) Test set: A set of examples used only to assess the performance of a fully-specified classifier.

To summarize, the training set is used to train the model while validation samples help to tune the hyper-parameters (i.e., learning rate, batch size, filter size, kernel size, activation function, dropout rate, number of hidden units etc.) of the model. However, training data is a subset of the primary dataset used to fit the model. The validation set determines the model performance and finds the optimal network layer size. Finally, the trained model is evaluated with the test set.

We trained a model on different hyperparameter settings such as embedding dimension, dropout rate, kernel, filter and batch size, learning rate (lr), epoch number etc. We iterate our model on this hyperparameter until we find its optimum value for training, avoiding overfitting on the dataset. In our experiment, we set the epoch number to 50 and batch size to 256. Except for the transformer learning training mechanism, we use a batch size of 16.

#### **4.4.1 Learning Curve:**

A learning curve (LC) plots model learning performances over the epoch rate or time. LC performances on the train and validation datasets are applied to determine an underfit, overfit or well-fit model [73]. It can be used whether the train or validation datasets are not relatively representative of the problem domain [74]. LC plot the training and validation accuracy and loss of training and validation data over time. Our experiment showed each model's learning performance by plotting LC. For a better understanding of our experiment, we have showed each model training accuracy (TA) vs. validation accuracy (VA) with respect to epoch and training loss (TL) vs. validation loss (VL) with respect to epoch during training

the model, similar approach are shown in [75, 76, 77, 78, 79]. The X-axis indicates the epoch number, while the Y-axis indicates the training, validation accuracy, and training, validation loss. TA and VA determine whether the model was overfitting or not and TL and VL determine whether the model was overfitting or underfitting. The VA and VL for dataset how well the model will perform for the unseen future data.

#### **4.4.2 Convolutional Neural Network (CNN):**

CNN is a type of feed-forward neural network in the field of computer vision that consists of convolutional, pooling and fully connected layers. For text classification, raw text must be represented as a vector in the input layer, represented at Table 3.5. After a series of convolutional stacked with multiple filters and pooling operations, the model has an activation function in the neural network. Our experiment uses a simple CNN for classifying text because it can extract the features from global information with the help of a convolutional layer. We add an embedding layer with vocabulary size, maximum text input length, and embedding size and weight of embedded matrix with 128d. Then we apply a learning sequence to our vocabulary by using a convolutional 1D layer with 300 filters, kernel(k) size of 5(k=5) value and Relu activation unit. The convolution layer can shift the window over the sentence and the weighted matrix and let the C learn the weights for applying in the neighboring words in tensor input data. For effectively operating in the learning rate, we use a spatial dropout one dimensional (SpatialDropout1D) parameter of 0.5, which drops the 1D feature from the embedding layer. To eliminate the overfitting problem, we use a Dropout regularization technique with 0.5. As our C model is sequential, we add a batch normalization layer for learning efficiently from previous output layers. Finally, we add a dense layer with the Sigmoid activation function because we perform a trinary-based classification. Figure 4.2 shows the entire demonstration process of our sentiment classification in CNN.

After constructing our CNN model, we compile it with our 128D word embedding matrix



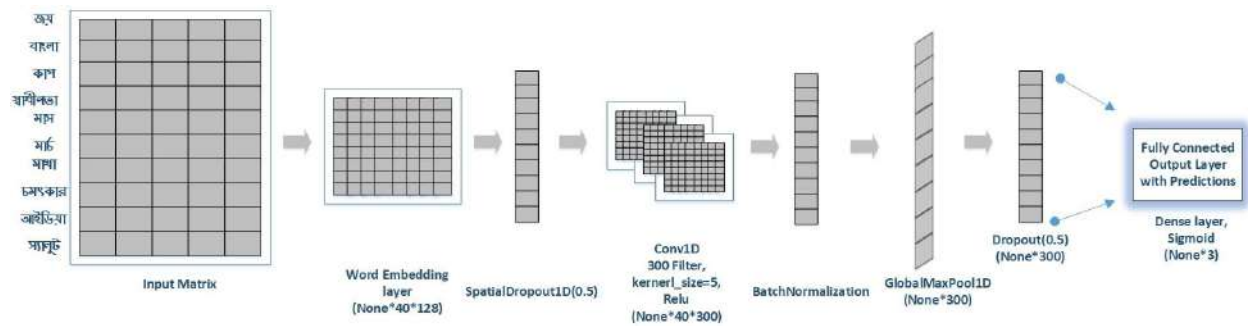
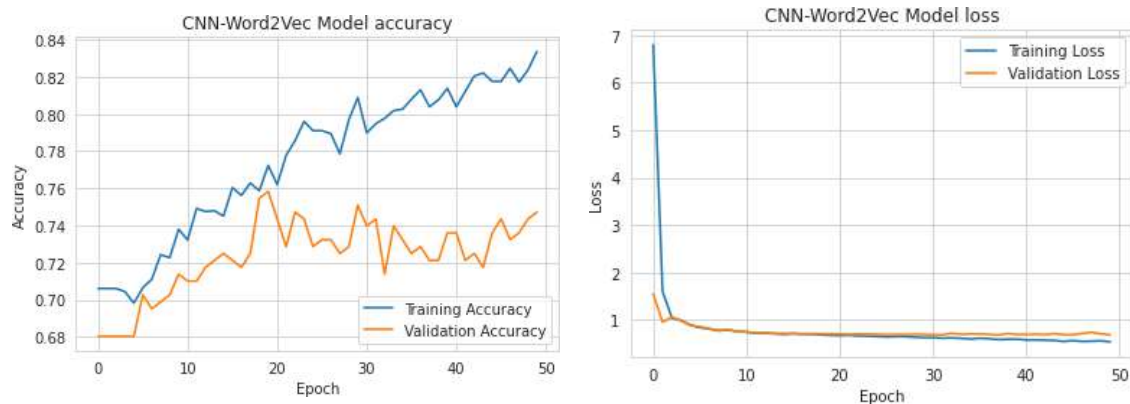


Figure 4.2: Convolutional neural network (CNN) architecture for sentiment classification



(a) TA, VA on CNN

(b) TL, VL on CNN

Figure 4.3: **(a)** LC of CNN model training accuracy (TA) and validation accuracy (VA) and **(b)** LC of CNN model training loss (TL), validation loss (VL)

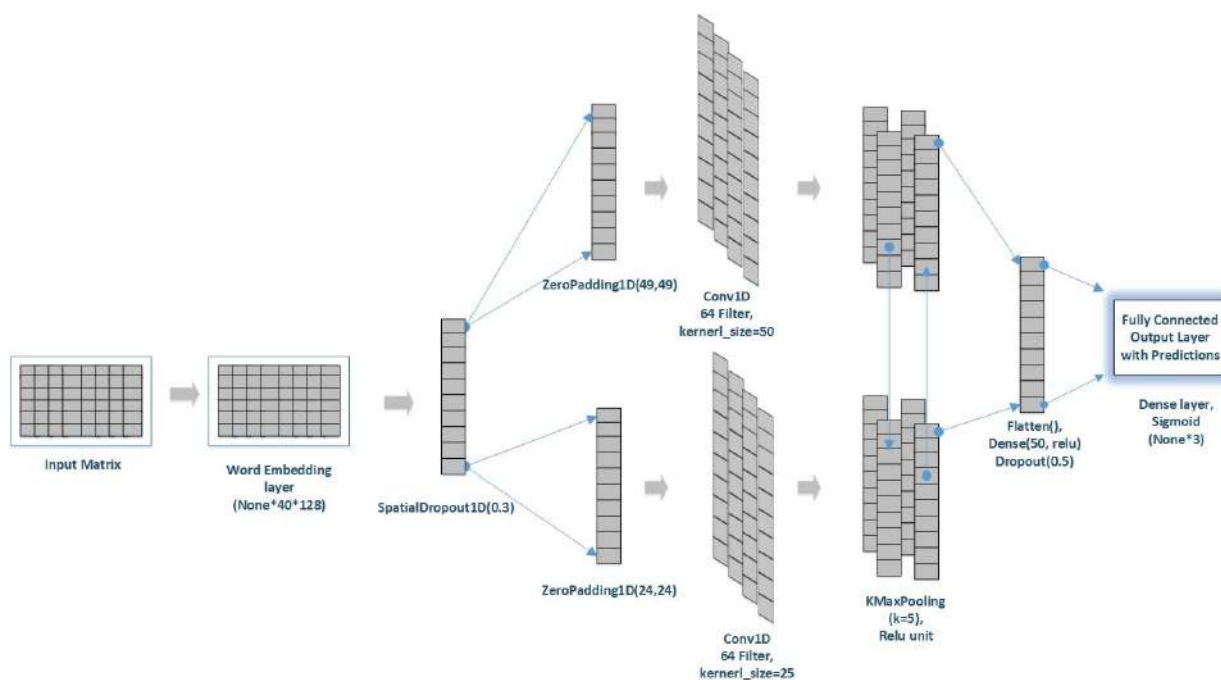
by setting the parameter loss function as categorical cross entropy (categorical\_crossentropy) and optimizer as adam at a learning rate of 0.01. After compiling the model, we fit the model with our training and validation data having a batch size of 256 with 50 epochs; we achieved 87.58% training accuracy (TA), 73.49% testing accuracy, 78.56% testing precision and 79.66% and testing recall value. Figure 4.3 shows the training, validation accuracy (VA), validation loss (VL) vs epoch during training the model.

#### 4.4.3 Dynamic Convolutional Neural Network (DCNN):

DCNN uses convolutional layer with dynamic k-max pooling layers to extract a sentence feature map. K-max pooling layer is used to identify the short and long contextual relations in

the word embedding text. The altitude of convolutional size and corpus text size determine the  $k$  value dynamically that is why it is called the dynamic  $k$ -max pooling layer in the network. In our experiment, we have used five  $k$ -max ( $k=5$ ) pooling layers two times followed by zeropadding 1D with 49 filter size and convolutional 1D with  $(64*50)$  size. A flatten fully-connected layer is added with a hidden layer. Dropout layer is used before the independent weights with 50 neurons having ReLU activation layer. Finally, each neuron from the fully connected dense layer is fed as output to the sigmoid layer with three neurons. Here in Figure 4.4 shows the whole demonstration process of our sentiment classification in DC .

Figure 4.4: Dynamic convolutional neural network (DCNN) architecture for sentiment classification



tion

After constructing our DCNN model, we compile it with our 128D word embedding matrix by setting the parameter loss function as `categorical_crossentropy` and optimizer as `adam` at a learning rate of 0.0001. After compiling the model, we fit the model with our training and validation data having a batch size of 256 with 50 epochs; we achieved 87.58% training accuracy and 73.49% testing accuracy with 78.56% testing precision and 79.66% testing recall

value. Figure 4.5 shows the training, validation accuracy, loss vs epoch during training the model. At point 30, epoch validation accuracy increases with respect to epoch, which means that the training procedure should be stopped on 30 epoch point.

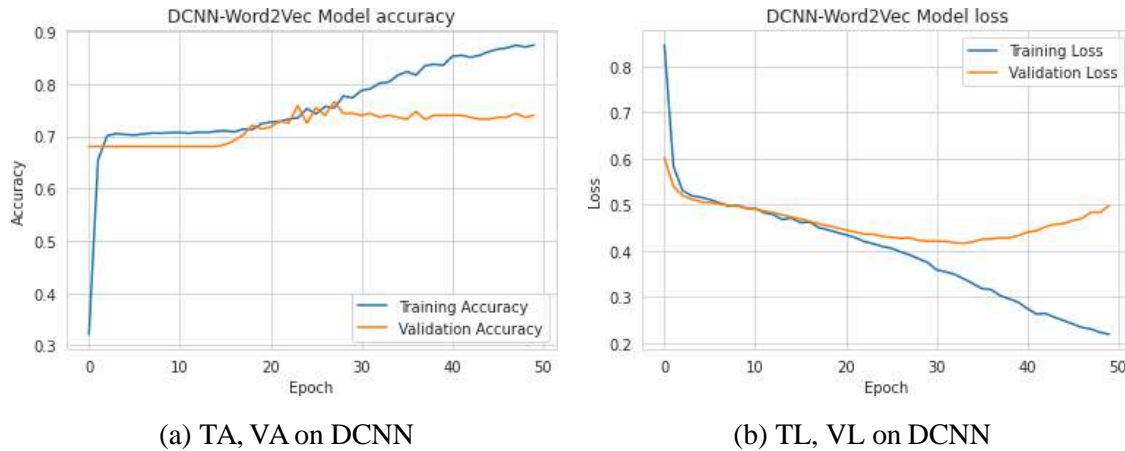


Figure 4.5: **(a)** LC of **DCNN** model training accuracy (TA) and validation accuracy (VA) and **(b)** LC of **DCNN** model training loss (TL), validation loss (VL)

#### 4.4.4 Multichannel Variable-Size Convolution Neural Network (MVCNN):

MVCNN is similar to CNN and DCNN, except it has variable size filter mechanisms with different sizes of word embedding layers. In our experiment, we used two embedding matrices dimensions (D) i.e., 128D and 200D. The two embedding layers are iterated over 3, 4, 5 filter sizes followed by zeropadding1D (2, 3, 4), convolutional layer with 100 filter and k-max pooling layer with 10. The output of this layer is iterated again according to the first layer mechanism. Finally, these three layer (layer\_1 and layer\_2, layer\_3) are concatenated and flatten output is followed as same as before DCNN and CNN , process demonstration in Figure 4.6

In the MVCNN model, we compile it with a two-dimensional word embedding matrix: 128D and 200D and set the parameter loss function as binary crossentropy (binary\_crossentropy), use a adam optimizer at a learning rate of 0.001. After compiling the model, we fit the model with our training and validation data having a batch size of 256 with

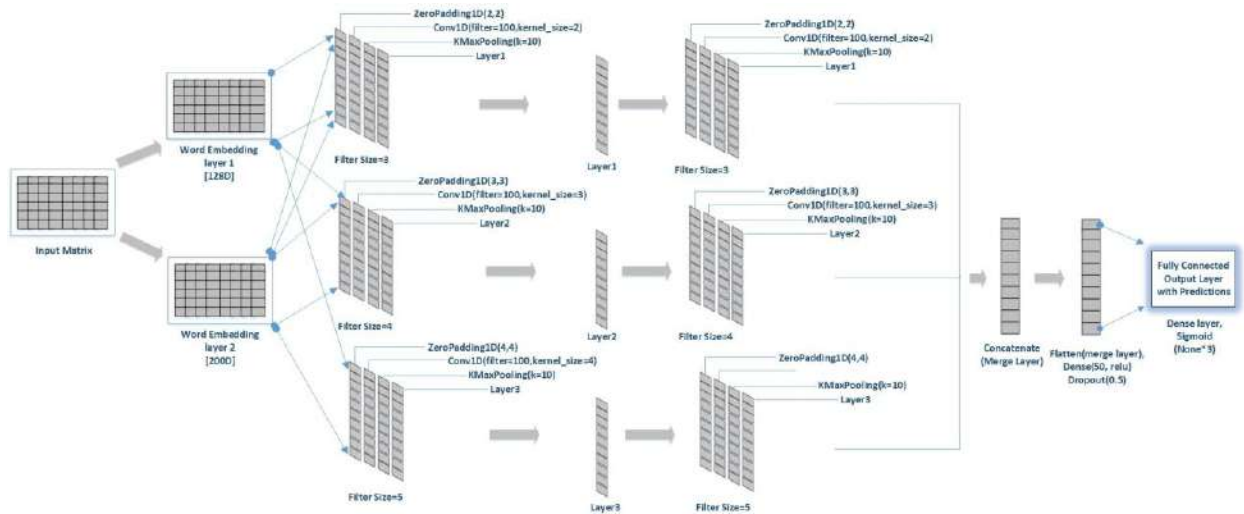


Figure 4.6: Multichannel variable-size convolutional neural network (MVCNN) architecture for sentiment classification

50 epochs. We achieved 96.42% training accuracy and 76.51% testing accuracy with 78.56% testing precision and 79.66% testing recall value. Figure 4.7 shows the training, validation accuracy, loss vs epoch during training the model.

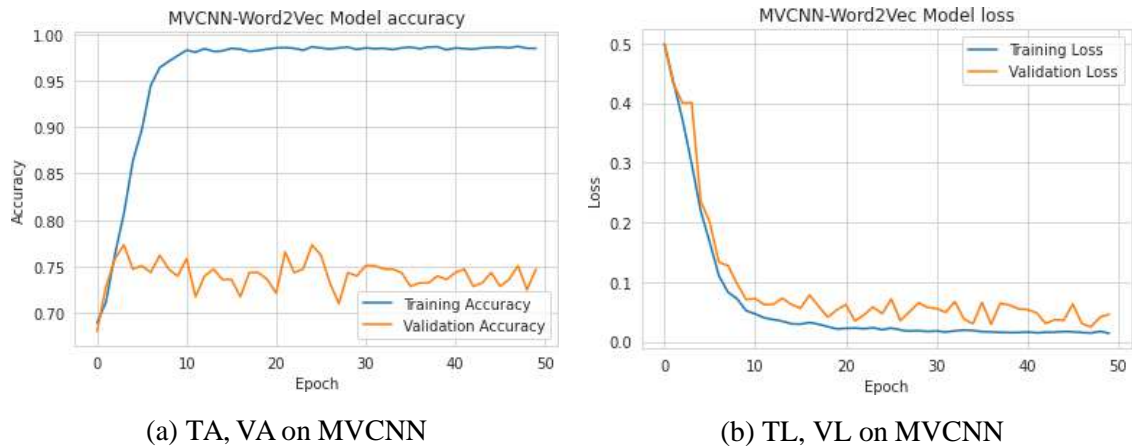


Figure 4.7: (a) LC of MVCNN model training accuracy (TA) and validation accuracy (VA) and (b) LC of MVCNN model training loss (TL), validation loss (VL)

#### 4.4.5 Very Deep Convolutional Neural Network (VDCNN):

Difference from DCNN: We have used a three-dimensional word embedding layer [128D, 200D, 300D] with ZeroPadding1D three filter sizes (filter\_size -1, filter\_size -1) are added on

three convolution1D layer with iteration of 3, 4, 5 sizes and GlobalMaxPool1D with k-max pooling layer of 4. After three iterations, we get three layers (Layer\_1, Layer\_3, Layer\_3) that are concatenated and flatten the merged layer with l2(0.01) regularization, dropout of 0.5 and finally, attach three dense neurons of fully connected output with sigmoid activation. The full architecture is shown in Figure 4.8 GlobalMaxPool1D minimizes the shape of the vector-matrix with the help of pool length.

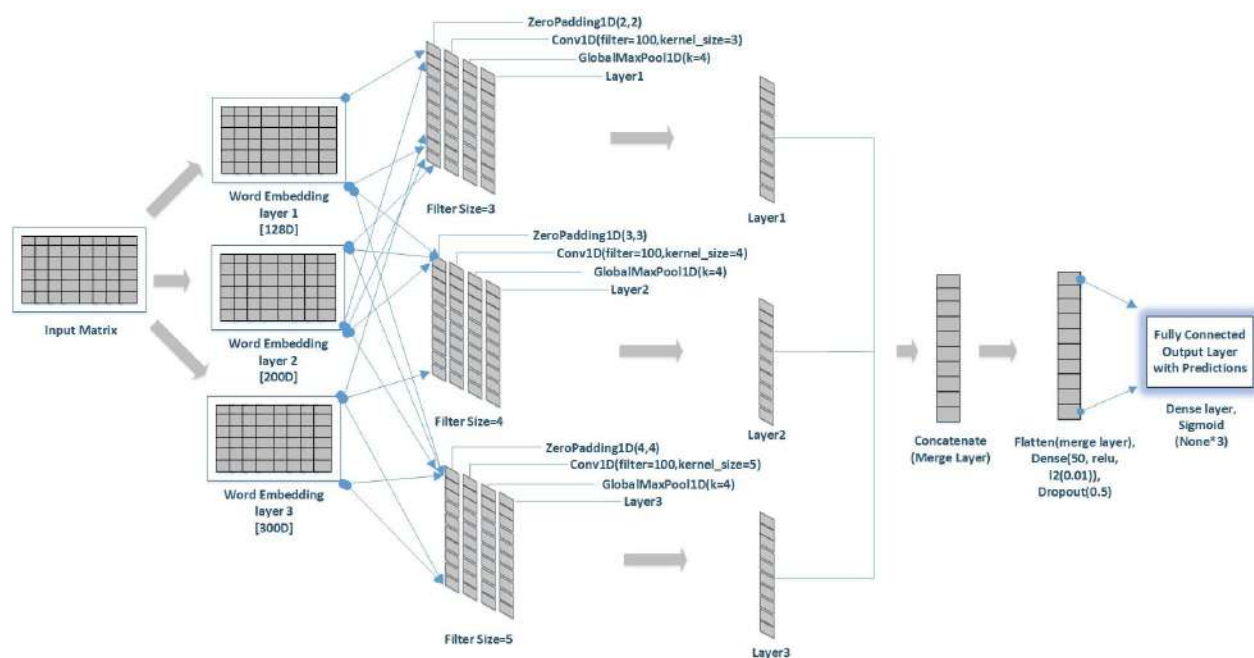


Figure 4.8: Very deep convolutional neural network (VDCNN) architecture for sentiment classification

In the VDCNN model, we compile it with a three-dimensional word embedding matrix: 128D, 200D, and 300D and set the parameter loss function as binary cross entropy (binary\_crossentropy), use as adam as a optimizer at a learning rate of 0.01. After compiling the model, we fit the model with our training and validation data having a batch size of 256 with 50 epochs. We achieved 96.16% training accuracy and 77.85% testing accuracy, with 80.16% testing precision and 79.56% testing recall value. Figure 4.9 shows the training, validation accuracy and loss vs epoch during training the model where validation accuracy slightly differs from the training accuracy. However, this model achieved higher accuracy

than CNN, DCNN and MVCNN models

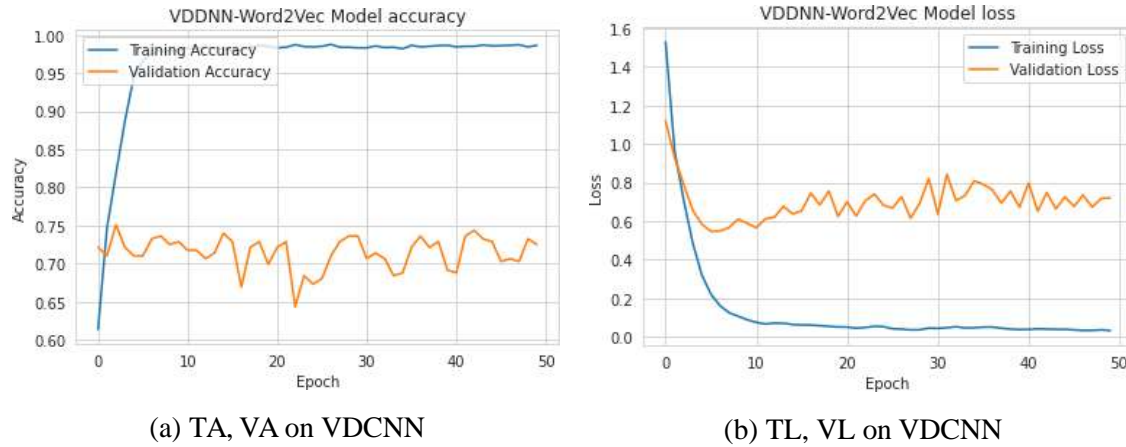


Figure 4.9: (a) LC of **VDCNN** model training accuracy (TA) and validation accuracy (VA) and (b) LC of **VDCNN** model training loss (TL), validation loss (VL)

#### 4.4.6 Recurrent Neural Network (RNN):

RNN is a feed-forward neural network for sequence modelling and data in which the output depends on the previous state. It maintains new state information during iteration over the sequence of elements and feedback to the previous layer to capture the correlation between current and previous time steps. A single hidden layer (h) is shown in Figure 4.10

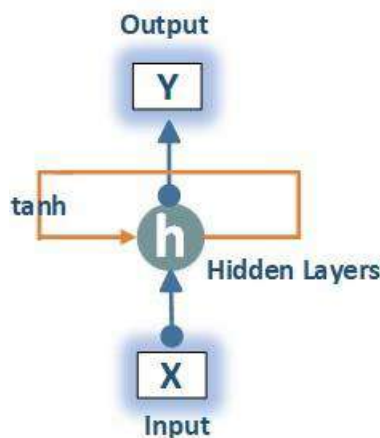


Figure 4.10: RNN block diagram [26]

In our experiment, we use a simple RNN with 32 layers of unit shown in Figure 4.11



Here at timestamp  $t$ ,  $x$  is the input layer, where the previous  $x_{t-1}$  is fetched from the hidden layer  $h_{t-1}$  and feedback to the current  $x_t$  input at the current hidden layer ( $h_t$ ). A spatial dropout (SpatialDropout) parameter of 0.4 is plugged on the previous RNN layer; after the RNN layer, we add a batch normalization (BatchNormalization) layer having dropout of 0.4 and global max pool 1D (GlobalMaxPool1D) as a sequentially and finally a three connected layer with sigmoid activation function ( $\sigma$ ) is put through into the Dense layer.

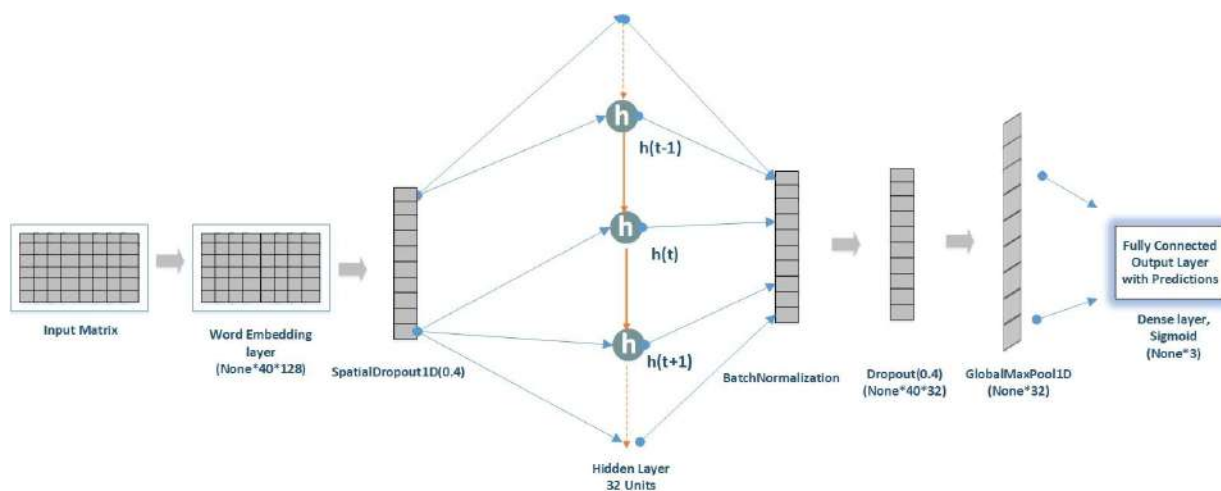


Figure 4.11: Recurrent neural network (RNN) architecture for sentiment classification

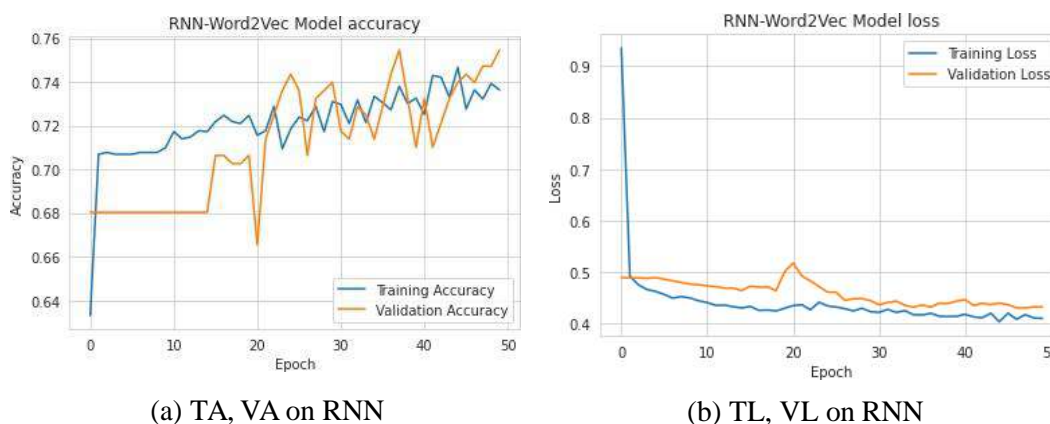


Figure 4.12: (a) LC of RNN model training accuracy (TA) and validation accuracy (VA) and (b) LC of RNN model training loss (TL), validation loss (VL)

We compile the RNN model with our 128D word embedding matrix with binary\_crossentropy loss and adam optimizer at a learning rate of 0.01. After compiling the model, we fit the

model with our training and validation data, having a batch size of 256 with 50 epochs. We achieved 76.57% training accuracy and 73.82% testing accuracy. Figure 4.12 shows the models training and validation accuracy loss.

#### 4.4.7 Long Short Term Neural Network(LSTM)

LSTM is designed to reduce the vanishing gradient descent problem and remember the data as a long-term period in a left to the right context. Unlike RNN, LSTM also has a recurrent structure of interacting layer called Input gate ( $x$ ), Forget gate ( $f_t$ ) and output gate. At the timestamp  $t$ , input gate ( $x_t$ ) with  $\tanh$  layer generates a vector of all possible values which is triggered by sigmoid activation function ( $\sigma$ ) and produces a new cell state ( $c_t$ ). Input gate ( $x_t$ ) decides how much information needs to be updated or ignored. The forget gate ( $f_t$ ) decides what part of the information needs to be removed from the previous cell state ( $c_{t-1}$ ) of the previous hidden layer ( $h_{t-1}$ ). The output gate concatenates the input with sigmoid layer and decides what part of the current cell state through a  $\tanh$  function and multiplies it, full block architecture shows in Figure 4.13.

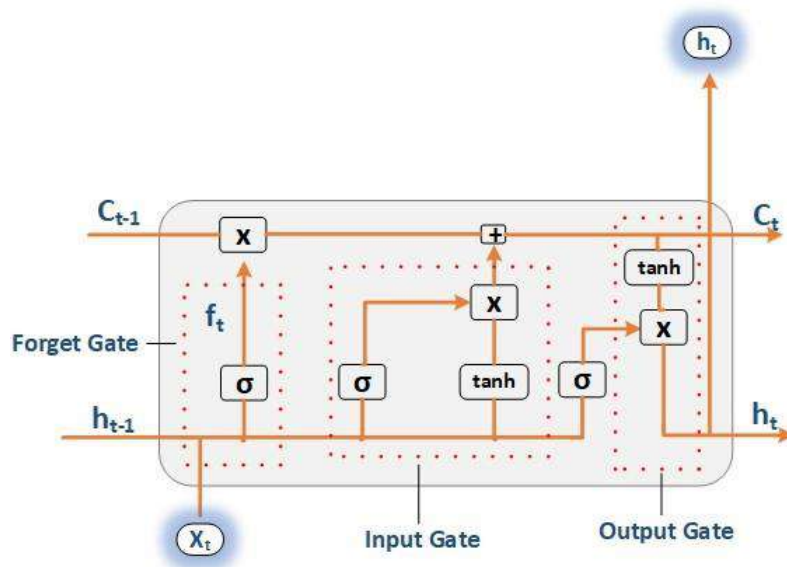


Figure 4.13: LSTM block diagram [27]

In Figure 4.14, our LSTM model consists of 32 unit hidden layer and Unlike RNN, at



timestamp  $t$ , the previous input layer ( $x_{t-1}$ ) is fetched along with previous cell state ( $c_{t-1}$ ) from the hidden layer ( $h_{t-1}$ ) and feed back to the current input ( $x_t$ ) at the current hidden layer ( $h_t$ ) and produce a new cell state ( $c_t$ ). After LSTM layer, we add a batch normalization layer and the rest of this architecture is followed by our RNN layer above shows in Figure 4.11

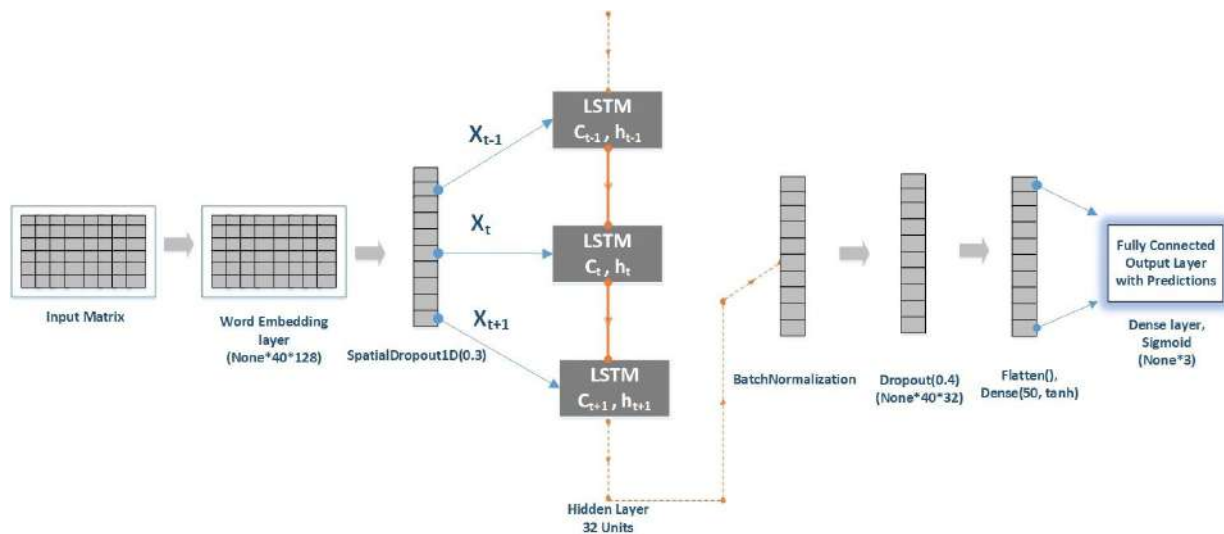


Figure 4.14: Long term short term neural network (**LSTM**) architecture for sentiment classification

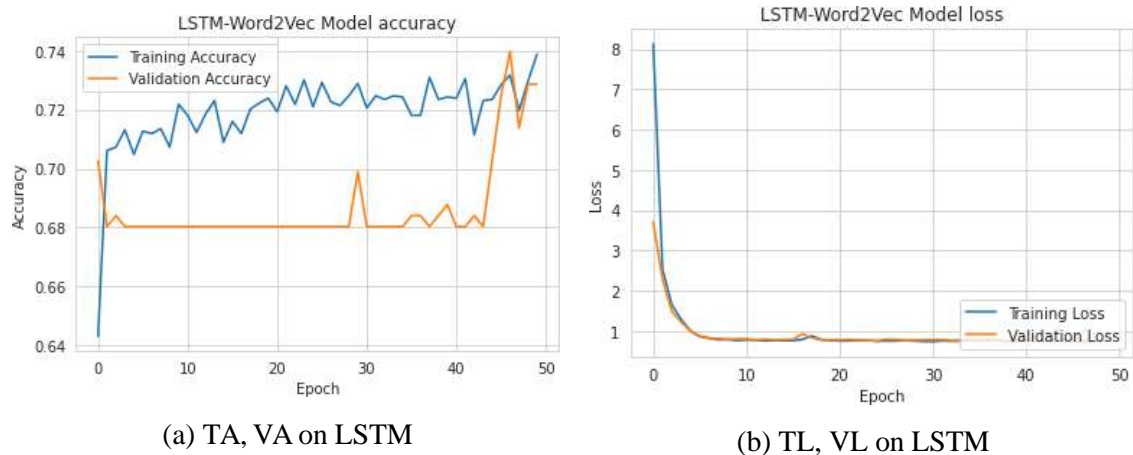


Figure 4.15: **(a)** LC of **LSTM** model training accuracy (TA) and validation accuracy (VA) and **(b)** LC of **LSTM** model training loss (TL), validation loss (VL)

We compile the LSTM model with our 128D word embedding matrix with binary cross entropy loss and adam optimizer at a learning rate of 0.01. After compiling the model, we

fit the model with our training and validation data, with a batch size of 256 with 50 epochs. We achieved 76.57% training accuracy and 73.82% testing accuracy. Figure 4.15 shows the training and validation accuracy loss during fit the model.

#### 4.4.8 Bidirectional Long Short Term Neural Network (Bi-LSTM):

Bi-LSTM follows LSTM architecture, except it works on inputs in two ways: left to right (capturing forward context) and right to left (capturing backward context). It detects the feature from both past and future contexts in sequential modelling.

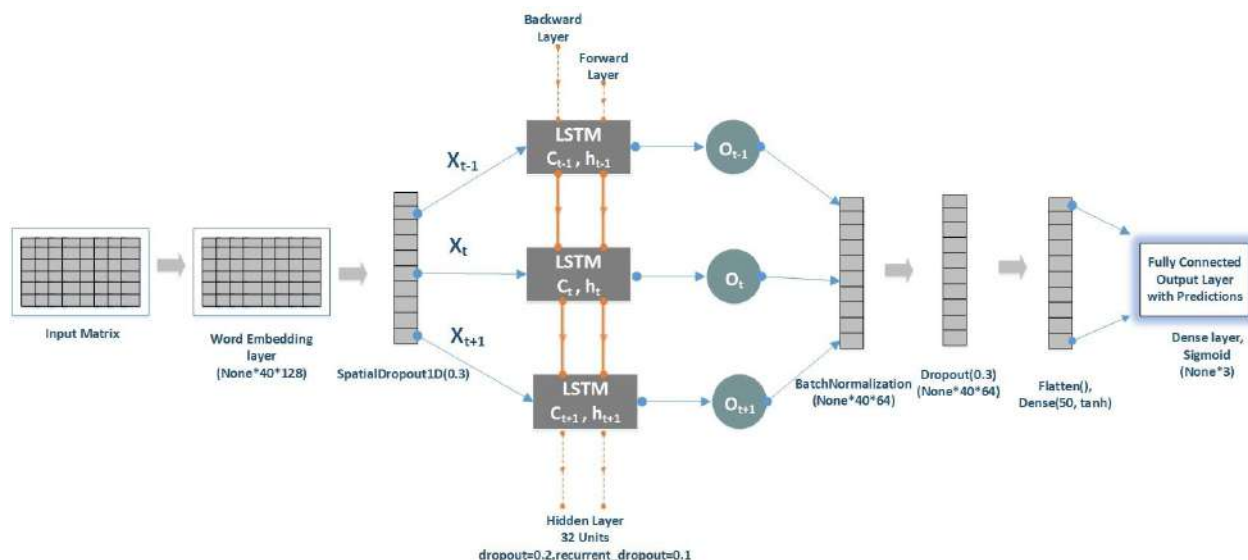


Figure 4.16: Bidirectional long term short term neural network (**Bi-LSTM**) architecture for sentiment classification

In Figure 4.16, same as our previous LSTM network, we use LSTM of 32 units in a bidirectional way having a dropout of 0.2 and recurrent\_dropout of 0.1. In the Bi-LSTM network, there are two states to resolve both contextual relations: left to right (Forward) and right to left (Backward). At timestamp  $t$ , each hidden layer ( $h_t$ ) output ( $O_t$ ) is produced along with memory cell state ( $c_t$ ) and forwards to a convolutional1d layer of 64 filters, kernel\_size of 4. The rest of the network is followed by the previous LSTM network.

We compile our Bi-LSTM model with our 128D word embedding matrix with binary cross entropy loss and adam optimizer at a learning rate of 0.001. We achieved 84.33% training

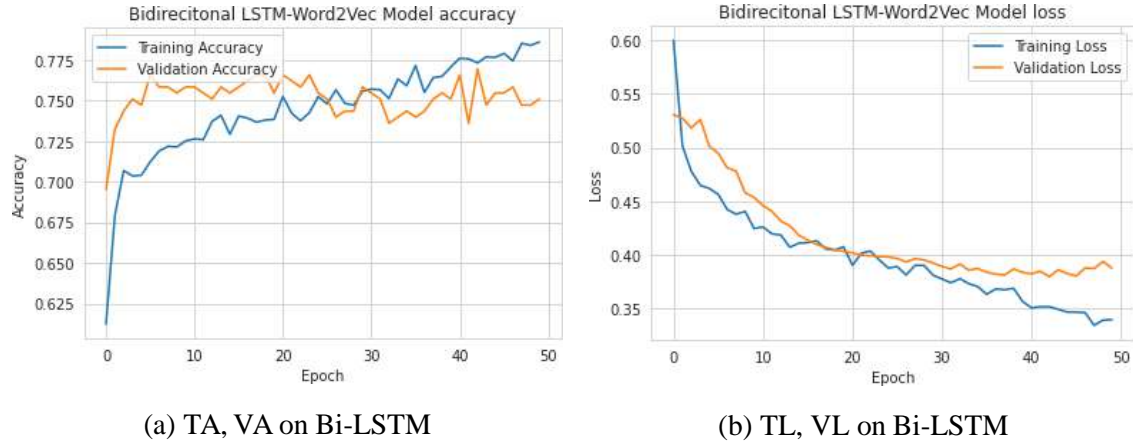


Figure 4.17: (a) LC of **Bi-LSTM** model training accuracy (TA) and validation accuracy (VA) and (b) LC of **Bi-LSTM** model training loss (TL), validation loss (VL)

accuracy and 78.14% testing accuracy during training and validating the model having a batch size of 256 with 50 epochs. Figure 4.17 shows the training and validation accuracy loss during fit the model.

#### 4.4.9 Asymmetric Convolutional Bidirectional LSTM (AC\_Bi-LSTM):

AC\_Bi-LSTM layer is a hybrid model combination of CNN and LSTM approaches. In our sentiment classification, we applied that hybrid model. In our experiment, at Figure 4.18, we uses a 128D word embedding layer, which iterated over with convolution1D layer with 100 filter size, kernel\_size of 2 with ReLU activation layer along with another convolution1D layer of 100 filters and 30,40,50,60 sizes of the kernel with ReLU activation layer. This is called asymmetric because there are different kernel sizes on the same filter and activation layer unit. Then these two different layers of convolutional1d are merged, and the input  $(x_t, x_{t+1}, x_{t+2} \dots, x_{t+n})$  is passed to the LSTM layer of 32 units and the rest is followed by the previous LSTM network.

We compile our AC\_Bi-LSTM model with our 128D word embedding matrix with binary\_crossentropy loss and adam optimizer at a learning rate of 0.001. We achieved 95.71% training accuracy and 75.50% testing accuracy during training and validating the model,

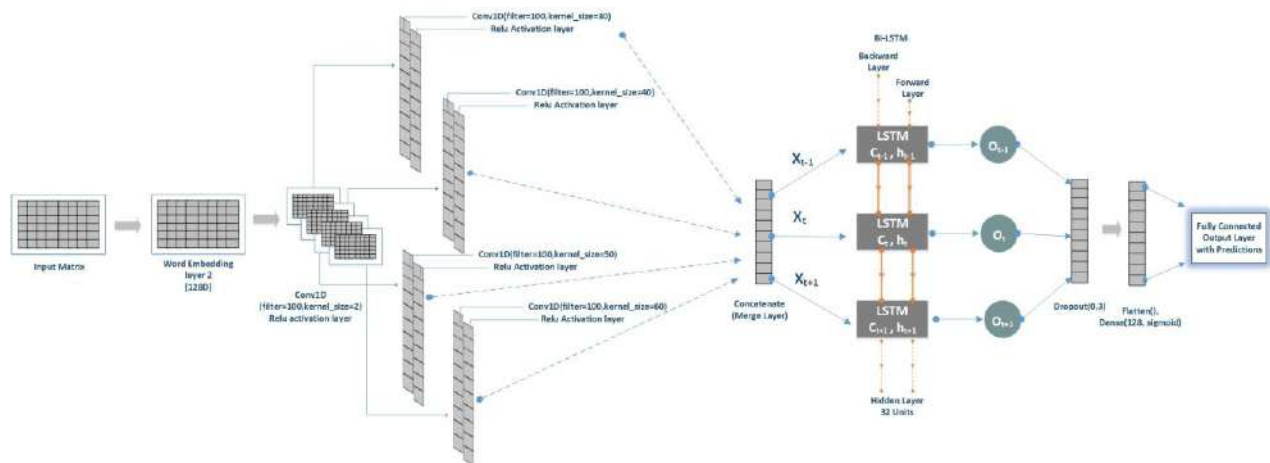
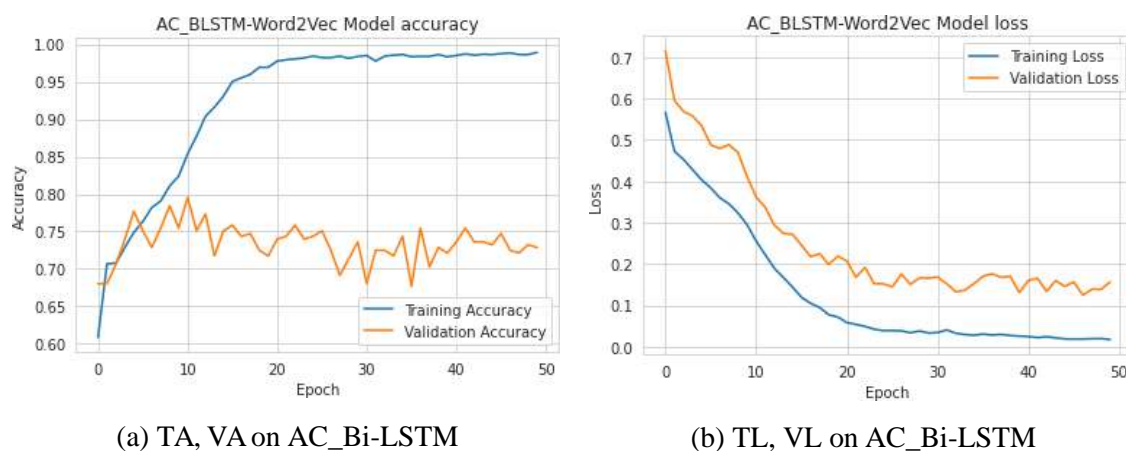


Figure 4.18: Asymmetric convolutional bidirectional LSTM (**AC\_Bi-LSTM**) neural network architecture for sentiment classification

having a batch size of 256 with 50 epochs. Figure 4.19 shows the training and validation accuracy loss during fit the model.



(a) TA, VA on AC\_Bi-LSTM

(b) TL, VL on AC\_Bi-LSTM

Figure 4.19: (a) LC of **AC\_Bi-LSTM** model training accuracy (TA) and validation accuracy (VA) and (b) LC of **AC\_Bi-LSTM** model training loss (TL), validation loss (VL)

#### 4.4.10 Recurrent Convolutional Neural Network (RCNN):

Another hybrid model we used for sentiment classification is composed of four blocks with 32 units of LSTM layer with multiple recurrent convolutional units (conv1D). We add four conv1D having with filter size of 100, kernel size of 2 and activated with tanh layer. Each conv1D layers is connected with each LSTM layer and this layer blocks are sequentially

connected with another block layer, shows in Figure 4.20. A flatten with 50 neurons, relu activation layer is forwarded from LSTM block. Then, it is finally attached with three dense neurons of fully connected output with sigmoid ( $\sigma$ ) activation function.

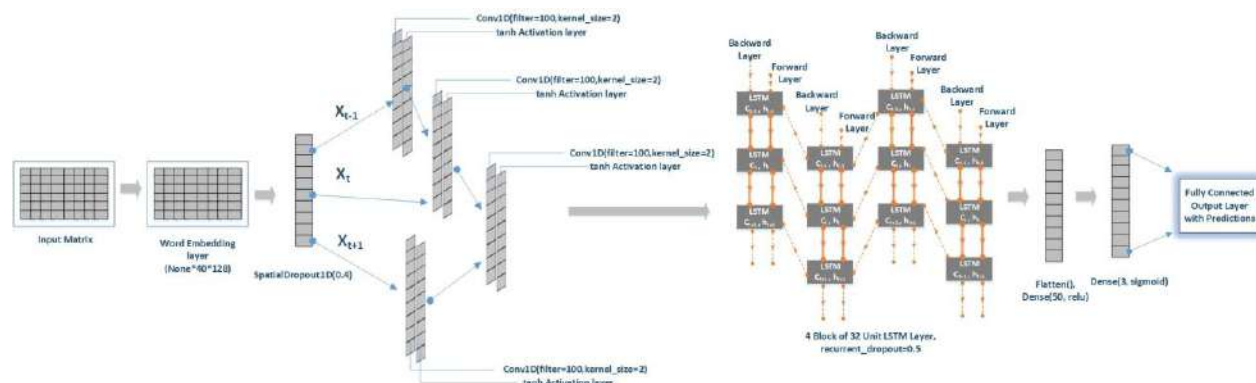


Figure 4.20: Recurrent convolutional neural network (**RCNN**) architecture for sentiment classification

In the **RCCNN** model, we compile it with 128D dimensional word embedding matrix and set the parameter loss function as `binary_crossentropy` and optimizer as `adam` at a learning rate of 0.001. After compiling the model, we fit the model with our training and validation data, with a batch size of 256 with 50 epochs. We achieved 70.69% training accuracy and 73.83% testing accuracy, 77.80% testing precision and 77.80% testing recall value. Figure 4.21 shows the training, validation accuracy, loss vs epoch during training the model.

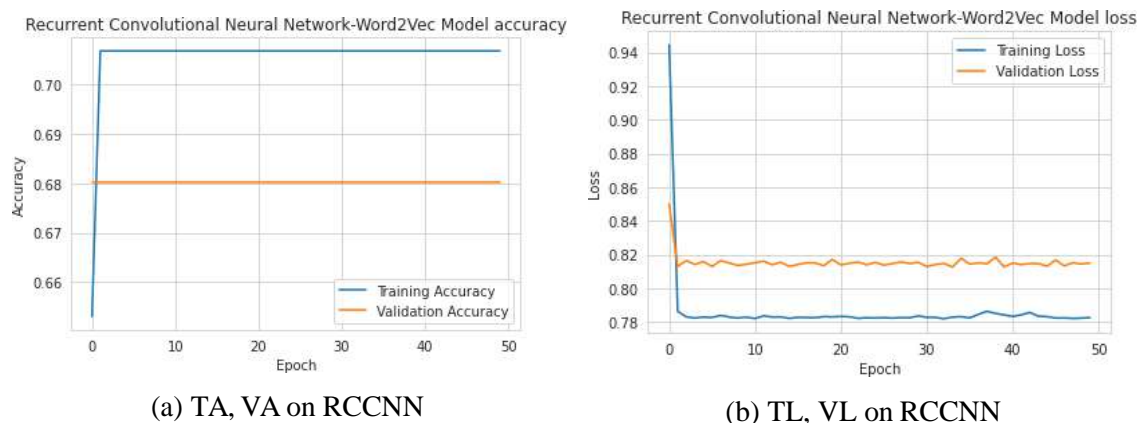


Figure 4.21: (a) LC of **RCCNN** model training accuracy (TA) and validation accuracy (VA) and (b) LC of **RCCNN** model training loss (TL), validation loss (VL)

#### 4.4.11 Gated Recurrent Unit (GRU):

GRU is similar architecture of LSTM model, where it consists of only two gates: update ( $z_t$ ) and reset ( $r_t$ ) gate. It has a memory of only one state, termed as hidden state ( $h_t$ ) that considers a separate cell  $c_t$ ,  $c_{t-1}$  like LSTM model. From a series of sequential input, update gate ( $z_t$ ) helps to learn long term dependencies and to determine what amount of information from previous hidden state ( $h_{t-1}$ ) needs to be forward. Whereas reset gate ( $r_t$ ) is supervised to learn short term dependencies and to generate how much information needs to be forgotten. In Figure 4.22, Update gate, at time step  $t$ ,  $t-1$  respectively, current input ( $x_t$ ) and the previous hidden layer ( $h_{t-1}$ ) information is multiplied by their own weights. These are added together and triggered by sigmoid activation function ( $\sigma$ ). Similarly, reset gate ( $r_t$ ) works by multiplication of  $x_t$  and  $h_{t-1}$ . Finally, an element wise product (Hadamard Product, ( $H^*P$ )) is performed on what information needs to collect from the current memory and previous state.

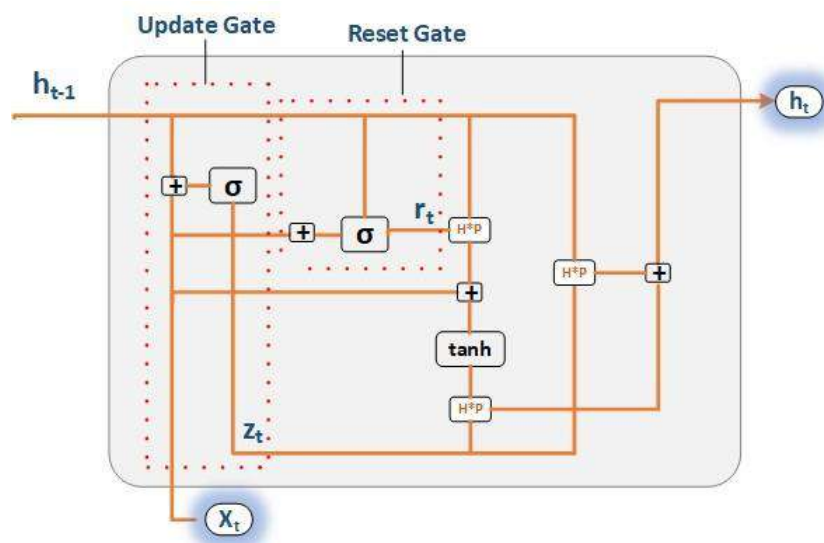


Figure 4.22: GRU block diagram [31]

In Figure 4.23, same from our LSTM network we use GRU layer of 32 units. After GRU memory we add a convolutional1D layer with 65 filter size, kernel size of 5 and a golort uniform (golort\_uniform) regularization of kernel initializer. Then sequentially add a



GlobalAveragePooling1D and GlobalMaxPooling1D layer. Finally these are concatenated with three neurons of dense sigmoid layer activation function.

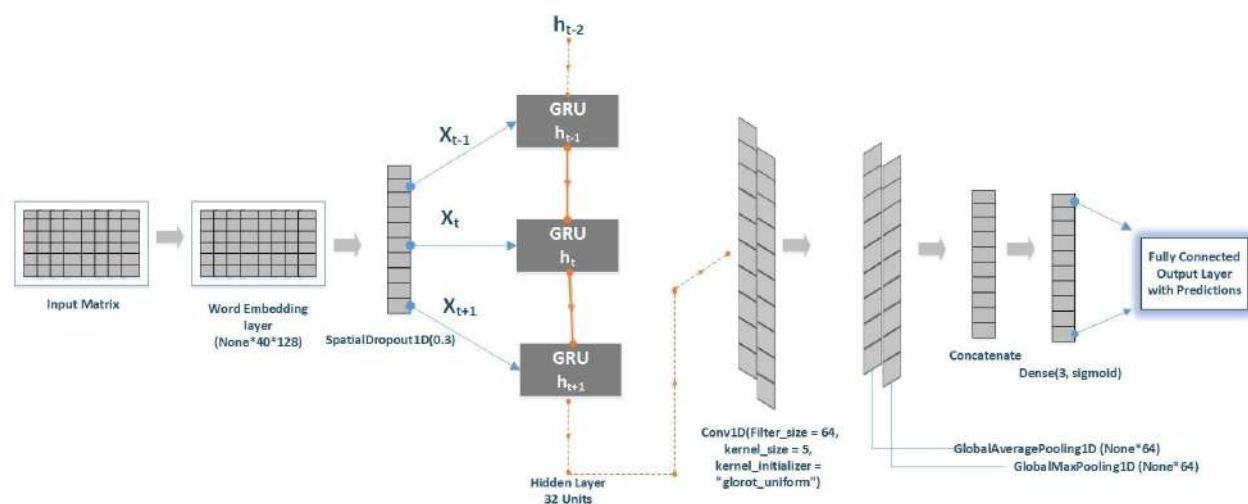


Figure 4.23: Gated recurrent unit (**GRU**) neural network architecture for sentiment classification

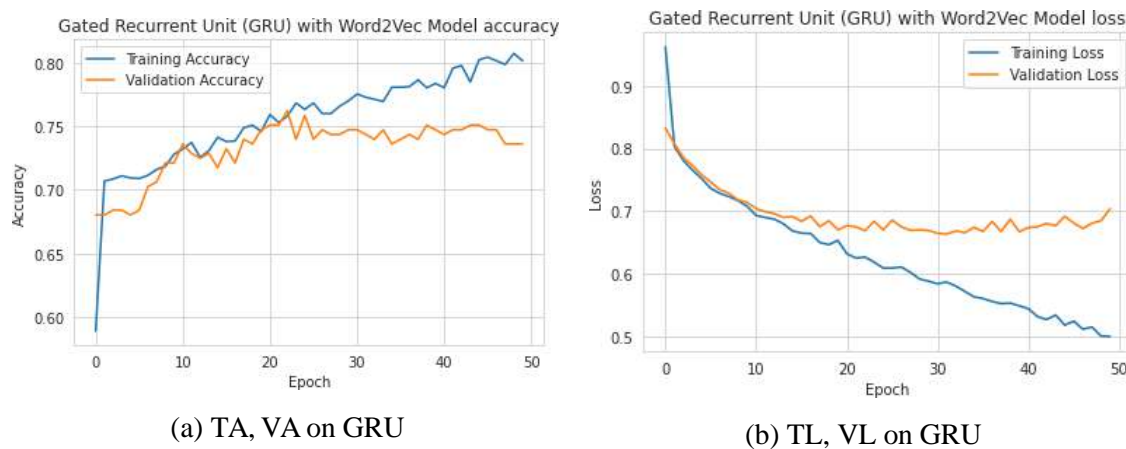


Figure 4.24: **(a)** LC of **GRU** model training accuracy (TA), validation accuracy (VA) and **(b)** LC of **GRU** model training loss (TL), validation loss (VL)

We compile our GRU model with our 128D word embedding matrix with binary cross entropy loss and adam optimizer at a learning rate 1e-3. We achieved 84.41% training accuracy and 75.84% testing accuracy during training and validating the model, having a batch size of 256 with 50 epochs. Figure 4.24 shows the training and validation accuracy, loss during fitting the model.

#### 4.4.12 Bi-directional Gated Recurrent Unit (Bi-GRU):

Similar neural network form Bi-LSTM and updated version form GRU, Bi-GRU works on both forward and backward layers without having to use a cell memory unit, as shown in Figure 4.25. Similar architecture form LSTM model, Bi-GRU works on resolving vanishing gradient descent problem. However, GRU capture and remember longer range of correlation and train faster more efficiently than LSTM [80]. Same as LSTM network we use a GRU of 32 units in a bidirectional way, having a dropout of 0.2 and recurrent dropout of 0.1. As similar form Bi-LSTM, Bi-GRU networks have two states to resolve both contextual relations in left to right (forward) and right to left (backward) window except maintaining no cell state mechanisms. At time stamp  $t$ , each hidden layer ( $h_t$ ) output ( $o_t$ ) is produced and forwarded to a convolutional1d layer of 64 filters of kernel size of 4 and the rest of the network is followed by the previous GRU network.

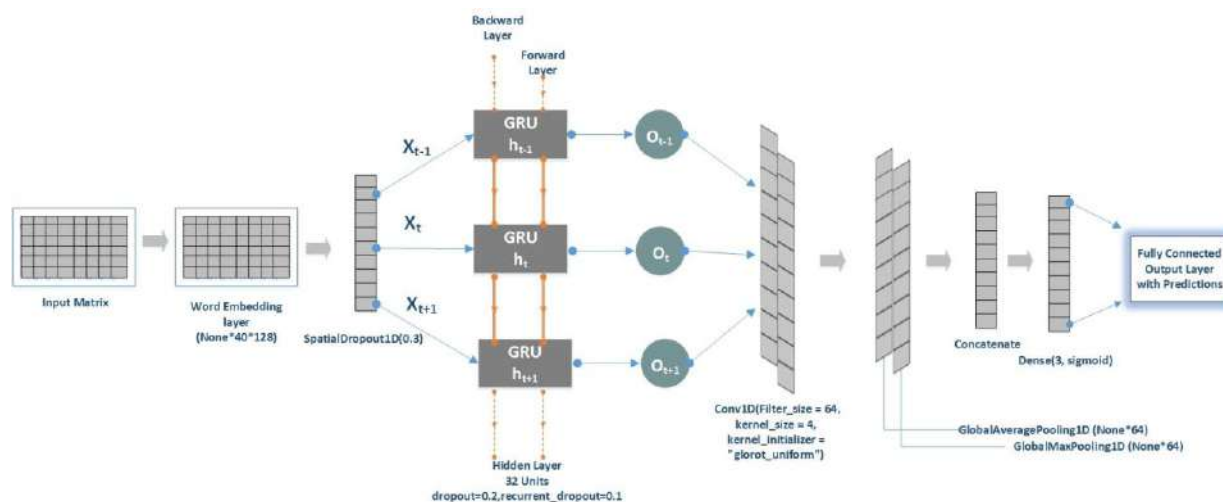


Figure 4.25: Bi-Directional gated recurrent unit (**Bi-GRU**) neural network architecture for sentiment classification

We compile our Bi-GRU model with our 128D word embedding matrix with binary cross-entropy loss and adam optimizer at a learning rate of  $1e-3$ . We achieved 83.07% training accuracy and 75.17% testing accuracy during training and validating the model, with a batch size of 256 with 50 epochs. Figure 4.26 shows the training and validation accuracy loss



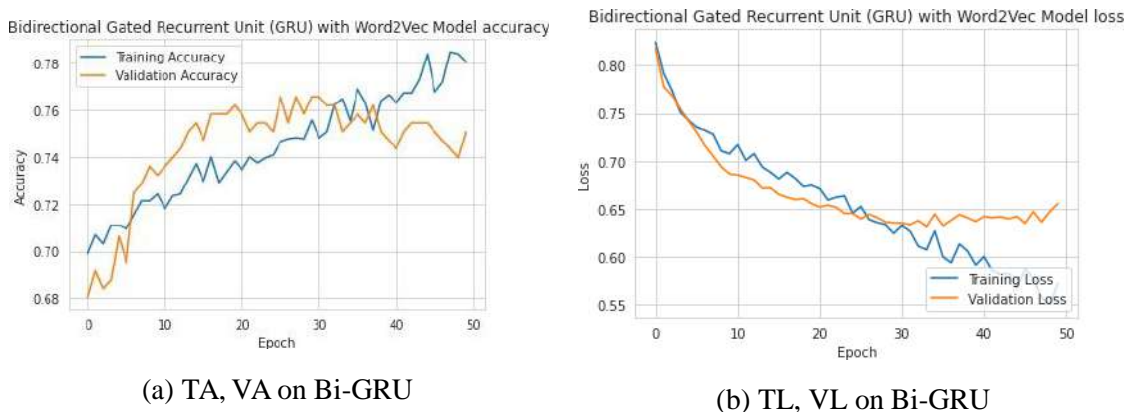


Figure 4.26: (a) LC of **Bi-GRU** model training accuracy (TA), validation accuracy (VA) and (b) LC of **Bi-GRU** model training loss (TL), validation loss (VL)

during fitting of the model.

#### 4.4.13 Attention Based Neural Network:

Attention mechanism has been designed to increase the RNN model's ability to produce better representations of a corpus and capture long-term dependencies at a low computational cost. This mechanism is applied to deploy the model to focus on the important part of a text rather than encoding the full sentence length. The main objective of the attention mechanism is to identify each hidden state's significance and provide a weighted sum of all the features matrix fed as input. Our experiment uses a hierarchical attention neural network (HAN) to conduct our SA in Bangla text.

#### 4.4.14 Hierarchical Attention Based Neural Network:

The previous model in this methodology works on only sentence-level encoding; however, HAN works on two-level encoder networks, i.e., word and sentence encoders. It formulates the text as a hierarchical structure on word and sentence level attention to capturing compositional features hierarchical dependencies from a sequence of input and contributes to the polarity of the text. Several sentence splits a document, and each sentence word are

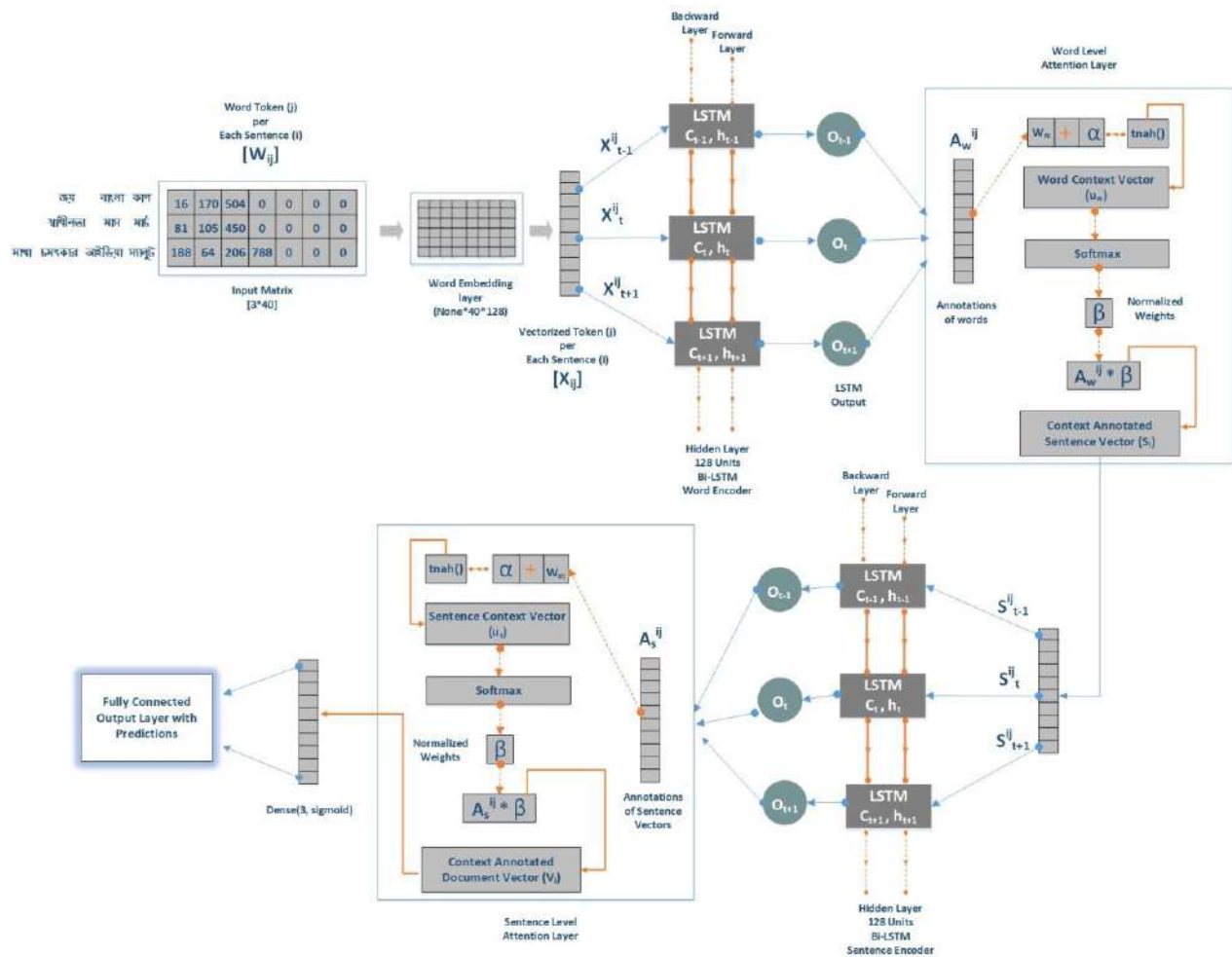


Figure 4.27: Hierarchical attention based neural network (**HAN**) architecture for sentiment classification

tokenized to transform into a vector, and then these vectors are used as an input matrix in the neural network. Authors [28] proposed a hierarchical attention-based structure for word and sentence encoders. The word encoder propagates the information from the hidden layers on the word level attention and forwards it to the sentence encoder. Then this information is processed by the sentence encoder hidden layers, and the output probabilities are predicted at the final layer through the sentence attention layer. Here, the sentence structure is formulated by the word attention layer by adding appropriate weights with the help of individual linear hidden layers. The sentence attention layer summarized the alignment of the sentence by extracting the relevant context of each sentence that classifies

the document. The preprocessing of our text encoding sequence is followed by Table 3.6 A bidirectional RNN model can achieve the context. We use a Bi-LSTM in our HAN mechanism, shown in Figure 4.27.

In Figure 4.27, we demonstrated our HAN mechanism on a bidirectional LSTM network. From input matrix, each word token (i) from each sentence (j) is placed on the word embedding layer noted as  $(W_{ij})$  on 128-dimensional (128D) matrix layer. Then it generates a vectorized token (i) for each sentence (j) noted as  $(X_{ij})$  which is projected on Bi-LSTM with 128 units as a word encoder layer. At time step t, the input  $X_{t-1}^{ij}$  from previous hidden state  $(h_{t-1})$  with previous current memory cell  $(C_{t-1})$  is sequentially forwarded to current hidden state  $(h_t)$  with output  $(O_t)$  to the HAN word level attention layer. Similarly backward channel resolves the contextual relation between current hidden layer  $(h_t)$  having current memory cell  $(C_t)$  to the previous hidden layer  $(h_{t-1})$ . The word level attention layer projects the output from Bi-LSTM word encoding layer. The annotation of word matrix  $(A_w^{ij})$  denote as a continuous vector space that makes the base for attention mechanism. This one hidden layer operates as a multilayer perceptron to do the model learn through a randomly initialized weights  $(W_m)$  by adding with biases  $(a)$  and puts it through a *tanh* activation functions to create a more improved annotation as a context vector of word  $(u_w)$ . Then this context word vector  $(u_w)$  is normalized by a softmax function by adding normalize weights  $(\beta)$ . Then finally the normalized context vector with weights  $(\beta)$  is concatenated with the previously calculated context annotation matrix  $(A_w^{ij})$  which produces the sentence vector  $(s_i)$ .

After getting the sentence vector  $(s_i)$ , a similar mechanism is followed for the sentence-based attention layer except without using an embedding layer. The context annotation sentence vector  $(A_s^{ij})$  which is projected from another Bi-LSTM with 128-dimensional (128D) units noted as a sentence encoding layer and forwards it for calculating an improved context annotated document vector  $(v_i)$ . Finally, these are concatenated with three dense sigmoid layer activation function neurons.

We compile our HAN-LSTM model and 128D word embedding matrix, `binary_crossentropy`

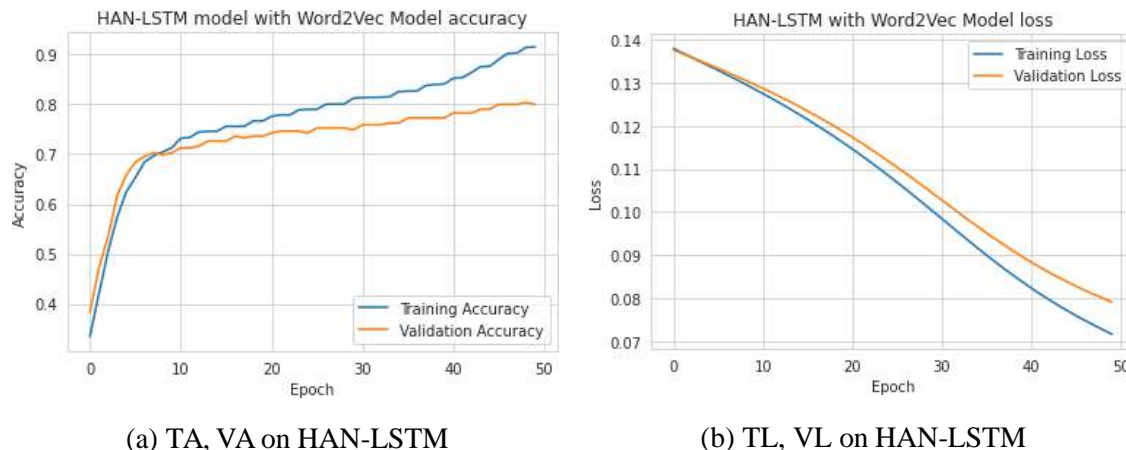


Figure 4.28: (a) LC of **HAN-LSTM** model training accuracy (TA), validation accuracy (VA) and (b) LC of **HAN-LSTM** model training loss (TL), validation loss (VL)

loss, and adam optimizer at a learning rate of 0.00001. We achieved 98.84% training accuracy and 78.52% testing accuracy during training and validating the model, with a batch size of 256 with 50 epochs. Figure 4.28 shows the training and validation accuracy loss during fitting the model.

#### 4.4.15 Capsule Neural Network (CapsNet):

A capsule neural network is a group of neural networks that solve the local feature problem of CNN pooling or max-pooling operations by providing vector output capsules, especially in dynamic routing algorithms. The computational complexity, i.e., reducing the matrix dimension, intercepts the various features and is captured by the pooling operation while losing data based on spatial relationships, however, without changing each feature. Again, CNN does not capture the hierarchical relationship between the local and global features. With the help of dynamic routing, it establishes the connection on spatial relationships between entities by mapping nonlinear vectors. This mapping transmits the capsule from lower level to upper level by iterating many routing loops based on a weight coupling coefficient. The weights coupling coefficient determines the leaning representation of which lower-level capsule will be forwarded to the upper-level capsule layer. It also detects the

similarity between vectors, predicting the lower and upper-level layer capsule. Our sentiment classification uses dynamic routing for capsule neural network that decides how much text or information is altered from each word to the encoding text sequences. The preprocessing of our text encoding sequence is followed by Table 3.5.

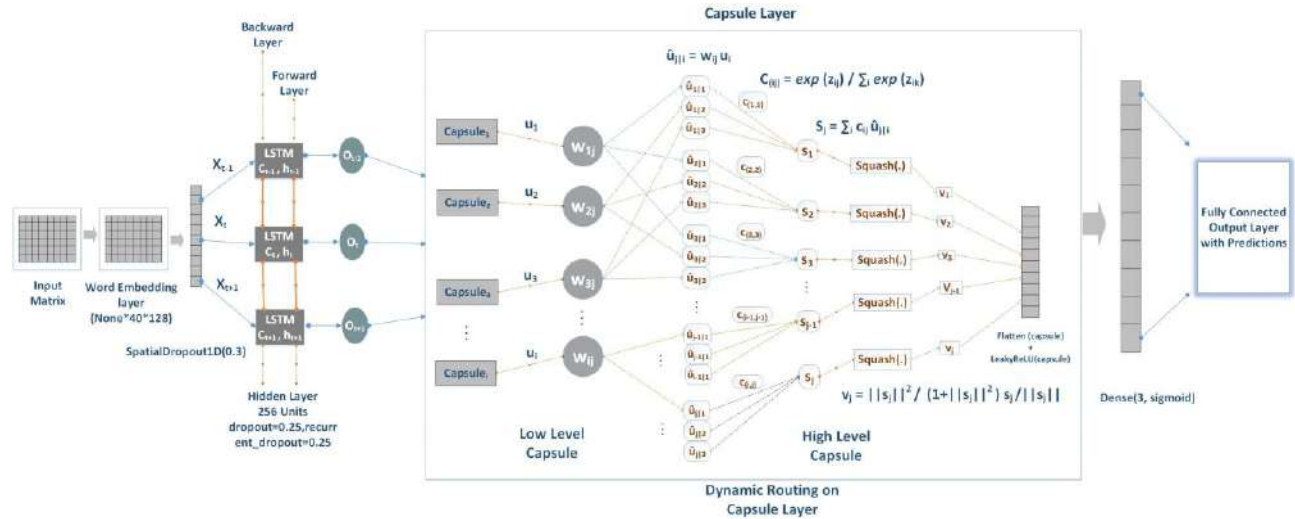


Figure 4.29: Dynamic Routing Based Capsule Neural Network (**D-CapsNet-Bi-LSTM**) architecture for sentiment classification

LSTM forwards its output into each primary layer capsule denote as Capsule<sub>*i*</sub>. In our experiment, we used a number of 16 capsules in our neural network. This is a lower level capsule (LC) that identifies more features from text and transforms the scalar output (receives form Bi-LSTM layer) into a vector output ( $u_1, u_2, u_3, \dots u_i$ ) to be the input of the next capsule layer. This vector has two core elements: length and direction. Using this length, the lower level capsule identifies the corresponding feature text probabilities. The direction parameter of the vectors determines the next path of the higher level capsule to confirm. Then the spatial relationship between higher and lower features on capsule is constructed by the affine transformation ( $\hat{u}_{j|i}$ ) that is the multiplication of corresponding weight matrices ( $w_{ij}$ ) with these vectors ( $u_1, u_2, u_3, \dots u_i$ ). We use the number of three iterations in our capsule network for calculating this linear or affine transformation. The affine transformation ( $\hat{u}_{j|i}$ ) value represents the predicted position of feature matrices of each sentence word which is the higher-level features. Here,  $\hat{u}_{j|i}$  indicates as a prediction vectors that what  $i^{\text{th}}$  features

should predict the correct position for the  $j^{\text{th}}$  sentence. That means if all 16 capsules detect the same features as the lower-level capsule, it will be that target feature value for that specific sentence. The affine transformation output value ( $\hat{u}_{j|i}$ ) is multiplied (dot product) in a weighted sum by a coupling coefficient value noted as ( $c_{(i,j)}$ ). This output value ( $\hat{u}_{j|i}$ ) is formed as next (higher) capsule level ( $s_j$ ), that determines the number of routing iteration process. The dot product differentiates the lower-level capsule  $i$  and higher level capsule  $j$ , although  $i$  capsule sees its output in the  $j$  capsule. In our D-Capsnet-Bi-LSTM network, we set the number of routing iterations is 3. This coupling coefficient is calculated by a routing softmax function where the exponential coefficient  $\exp(z_{ij})$  indicates some prior probabilities in which  $i^{\text{th}}$  capsule layer will be coupled to  $j^{\text{th}}$  capsule layer. Then the next level capsule ( $s_j$ ) is forwarded to the squash(.), a nonlinear activation function that is used to scalar (with additional and also unit scaling) the output vector ( $v_j$ ). By using this activation function, the output vector ( $v_j$ ) direction will not fluctuate if this vector length has more than 1. Then this higher level capsule vector ( $v_j$ ) is activated by the LeakyRelu function and finally densed by three neuron output classification. The full process is demonstrated in Figure no. 4.30

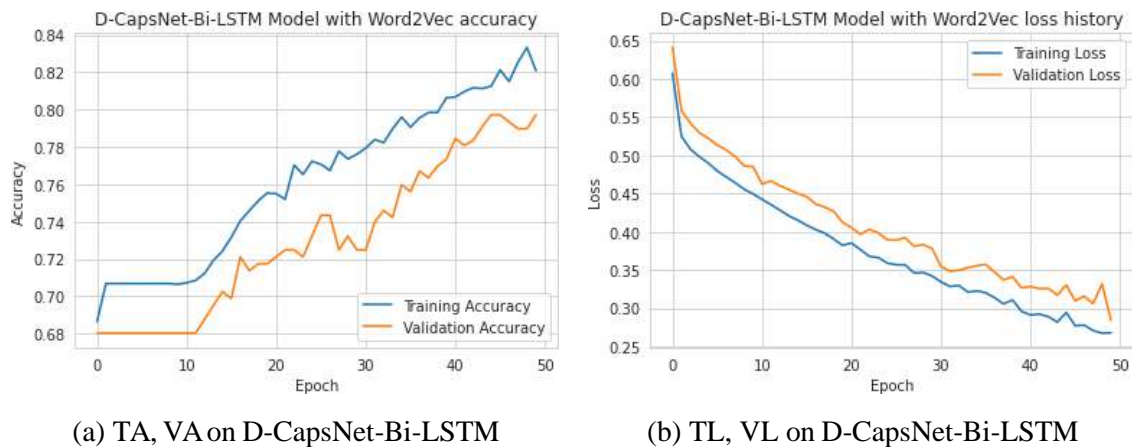


Figure 4.30: (a) LC of **D-CapsNet-Bi-LSTM** model training accuracy (TA), validation accuracy (VA) and (b) LC of **D-CapsNet-Bi-LSTM** model training loss (TL), validation loss (VL)

We compile our D-CapsNet-Bi-LSTM model on 128D word embedding matrix, with binary crossentropy loss function and adam optimizer at a learning rate of 0.0001. We achieved 82.97% training accuracy and 80.82% testing accuracy during training and validating the

model, with a batch size of 256 with 50 epochs. Figure 4.30 shows the model's training and validation accuracy loss during fitting the model.

#### **4.4.16 Bidirectional Encoder Representation From Transformer (BERT):**

Transformer belongs to an encoder-decoder architecture model having attention mechanisms [81] that are used for transfer learning in the field of machine translation as well as in NLP task and also leverages with long term dependencies finer than as a replacement of other conventional sequential models, i.e., RNN, LSTM, GRU, etc. In transfer learning, a model is trained on massive unlabeled content corpora utilizing self-supervised learning, and this pre-trained model is negligibly balanced during fine-tuning on a particular NLP task [82]. It is also more potential in training the model by removing the sequential dependencies of the past tokens. BERT was recently developed by Google [35], an encoder based transformer architecture for language modelling that is used for dynamic embedding in NLP tasks, which considers both current and previous tokens on both left and right in a bidirectional way. As a contextual model, Bert generates a representation of each word based on every other sentence. However, in static embedding, i.e., Word2Vec model generates a single word embedding representation for each word in the vocabulary.

#### **4.4.17 BERT-LSTM Architecture for Sentiment Classification:**

In our sentiment classification, we use a BERT-BASE model with a number of 12 transformer blocks, 768 hidden layers and 12 attention heads to generate contextualized embeddings. The input layer of BERT is a vector of sequence tokens along with special tokens shown in Table 3.7. LSTM reads text input sequentially, whereas BERT takes the entire tokenization of words at once. In our experiment, we build a sentiment classifier using huggingface library to fine-tune with the pre-trained BERT model [7] on the upper layer of LSTM, shown in Figure 4.31. We install the transformer version as 3.0 and load the BERT classifier and tokenizer for input processing.



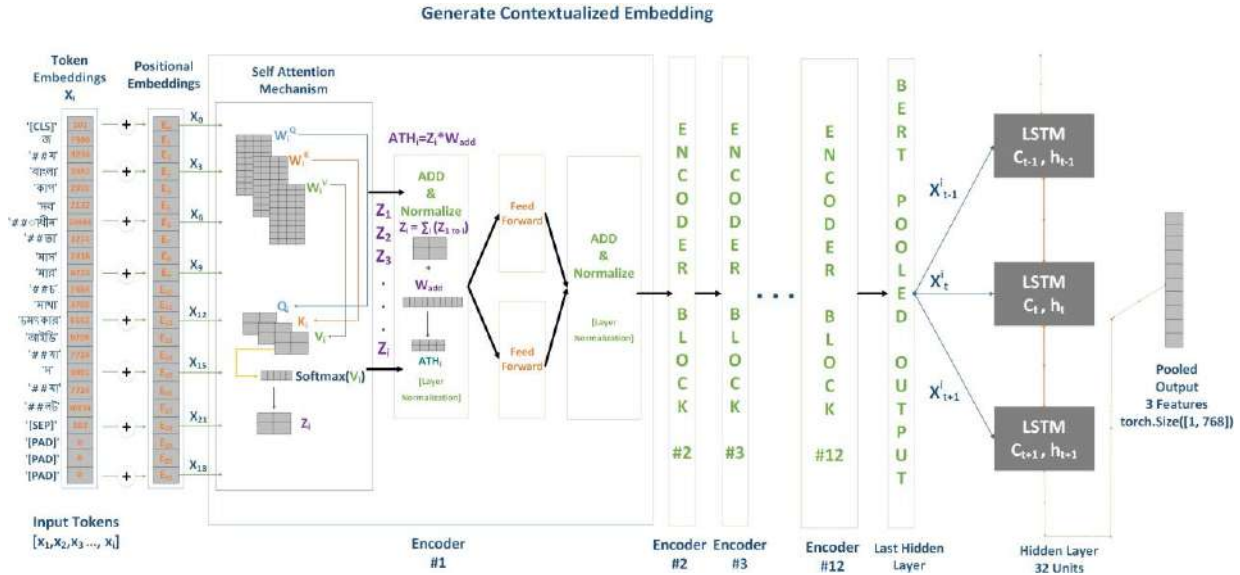


Figure 4.31: Bidirectional encoder representation from transformer with LSTM neural network (**BERT-LSTM**) architecture for sentiment classification

The input sequences are a raw sentences which is splitted by the BERT tokenizer and that tokens converted into token id and with attention mask showed in Table 3.8. BERT uses a self-attention mechanism over the input sequences, showed in Figure 4.31 which predefines the transformer for keeping pace with the relevant words from the inputs. In attention block of BERT transformer, it generates each attention head ( $ATH_i$ ) as a multi-headed self-attention from the input sequences  $(x_1, x_2, x_3, \dots, x_i)$ . This sequence  $(x_i)$  is multiplied with three weight matrixes ( $W_i^Q, W_i^K, W_i^V$ ) in scalar dot matrix way to generate three vectors termed respectively as query ( $Q_i$ ), keys ( $K_i$ ) and values ( $V_i$ ). These weight matrixes ( $W_i^Q, W_i^K, W_i^V$ ) are produced during the training process on BERT. The main mechanism for self-attention layer is to calculate each word score from input sequences and this score indicates how a word concentrates on other words to place in the correct position. For example a word  $(x_1)$  score value ( $V_1$ ) is calculated by the product of query ( $Q_1$ ) with keys ( $K_1, K_2, K_3, \dots, K_i$ ) matrices. Then the score value ( $V_i$ ) is divided by the dimension of key vector then pass to the softmax function and finally summed up the all values ( $V_1, V_2, V_3, \dots, V_i$ ) to produce another matrixes ( $Z_i$ ). In ADD and Normalize layer step[83] which is then multiply



with the additional weight matrices ( $W_{add}$ ) to produce attention head ( $ATH_i$ ) which captures all the information from all attention heads, then it is forwarded in the feed-forward layer. Similarly, other encoder will follow this mechanism [84] for processing pooled output from BERT. Finally, the last hidden layer from BERT is encoded in LSTM layer with 32 units and predicts the three features pooled output.

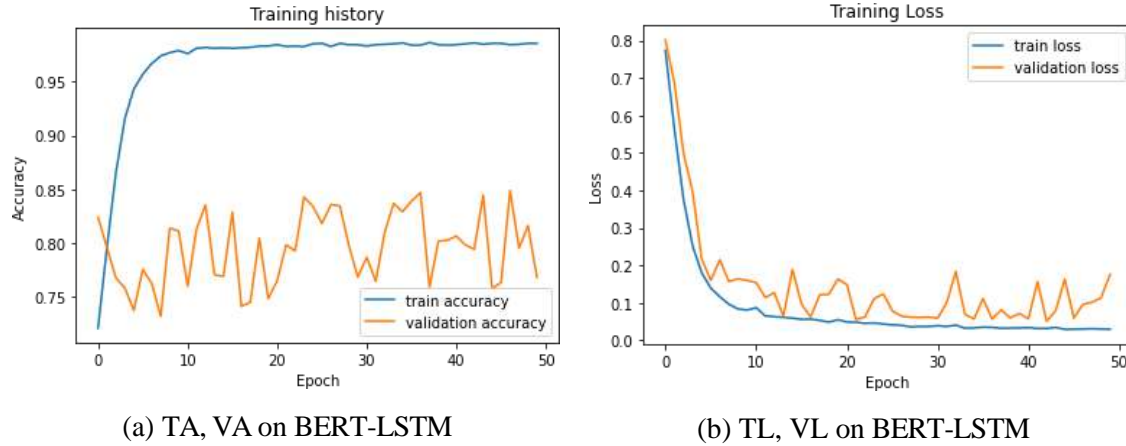


Figure 4.32: (a) LC of **BERT-LSTM** model training accuracy (TA), validation accuracy (VA) and (b) LC of **BERT-LSTM** model training loss (TL), validation loss (VL)

The challenge in the BERT-LSTM training process is that the memory was not released after training was done. In that case, we use a 16 size batch size because of prohibiting to crash the GPU. Because when the process is finished, Tensorflow releases the GPU memory. We compile our BERT-LSTM model with bert embedding layer, cross-entropy loss and adam optimizer at a learning rate  $1e-5$ . We achieved 98.68% training accuracy and 84.18% testing accuracy during training and validating the model, having a batch size of 16 with 50 epochs. Figure 4.32 shows the training and validation accuracy loss during fitting the model.

#### 4.4.18 Experiment on Augmented Dataset in BERT-LSTM Architecture for Sentiment Classification:

We preprocess the texts and then we divide the augmented dataset as 80% for training, 10% for validation and 10% for testing dataset. We compile our BERT-LSTM model with

bert embedding layer, cross-entropy loss and adam optimizer at a learning rate  $2e-5$ . We achieved 84.35% training accuracy and 81.11% testing accuracy during training and validating the model, having a batch size of 32 with 50 epochs.

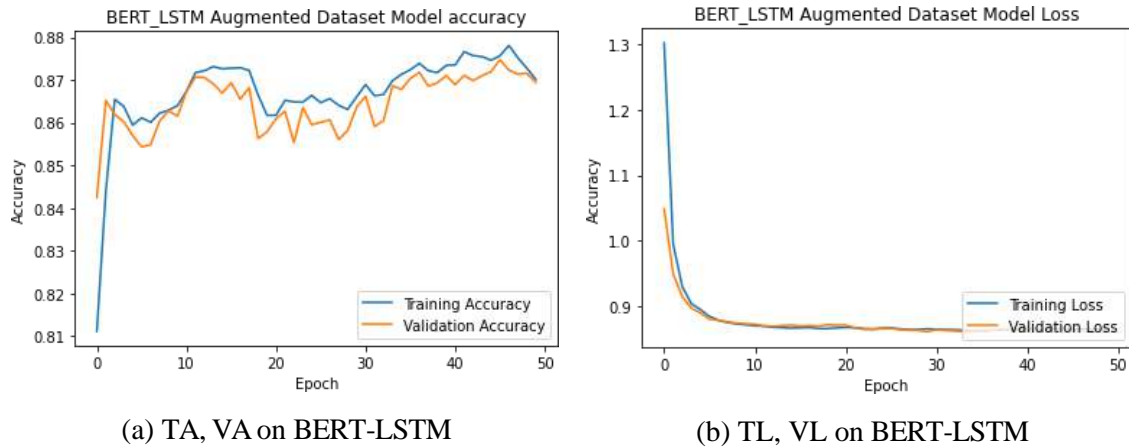


Figure 4.33: **(a)** DA of LC of **BERT-LSTM** model training accuracy (TA), validation accuracy (VA) and **(b)** DA of LC of **BERT-LSTM** model training loss (TL), validation loss (VL)

Figure 4.33 shows the training and validation accuracy and loss during fitting the model. The validation loss curve is both decreased and stabilized during the training loss curve. This LC curve indicates that our DA method in BERT-LSTM model is optimally fitted and in this case, this model can able to predict accurately on the dataset.

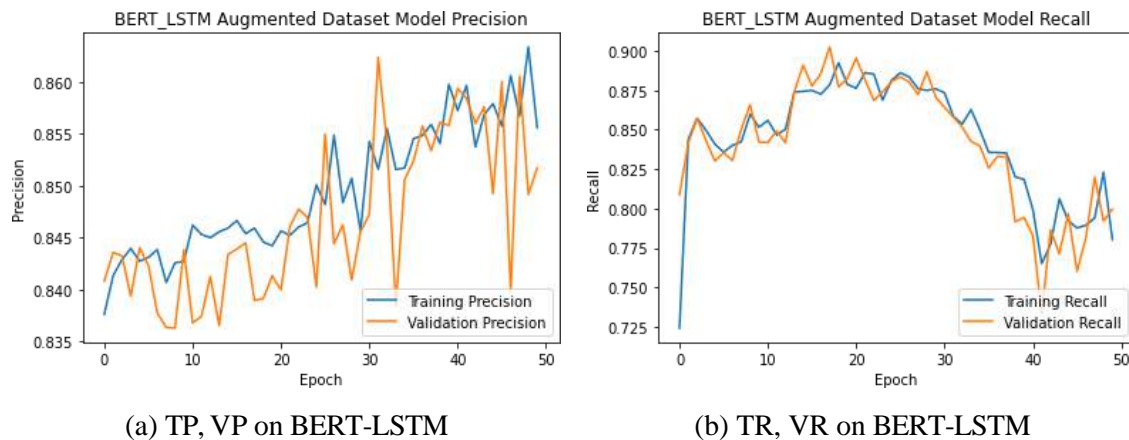


Figure 4.34: **(a)** DA of LC of **BERT-LSTM** model training precision (TP), validation precision (VP) and **(b)** DA of LC of **BERT-LSTM** model training recall (TR), validation recall (VR)

In LC of Figure 4.34, precision the validation precision is scattered and provides some noisy movements in an augmented neutral dataset. The recall LC indicates that the validation dataset may detect accuracy on the testing dataset.

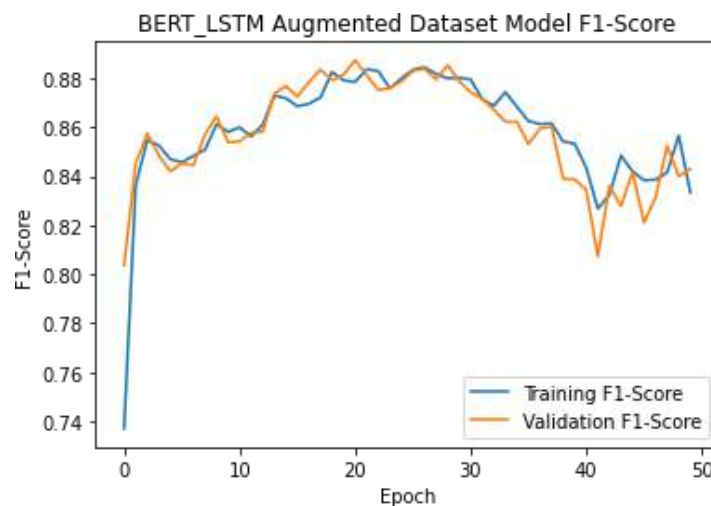


Figure 4.35: DA of LC of **BERT-LSTM** model training f1-score (Tf), validation f1-score (Tf)

We achieved 83.47% testing f1-score during training the model, Figure 4.35 shows this summary of LC in f1-score curve.

## 4.5 Conclusion

This chapter presents a summary of our experiment on BTSC algorithm and shows the effectiveness along with the given dataset polarity. We evaluated our BTSC algorithm performance matrices in both cricket and restaurant datasets. We construct the Tf-Idf matrix on both datasets and build two models named UniGram and BiGram. Then we describe our hybrid neural network model to experiment on BTSC generated target data and to show each model experimental graph at epoch rate 50. In the next section, we will show our model performances.

# Chapter 5

## Results and Discussion

### 5.1 Overview

This chapter of this section demonstrates our experimental results with a brief discussion on ML and DL methodology. In ML technique, we have shown our accuracy on the BTSC algorithm in both UniGram and BiGram models with a notable text classification algorithm. In DL process, we have trained the dataset in different types of hybrid neural networks, and finally, the performance of the proposed architecture was evaluated on the test dataset. We have shown model evaluation matrices on graphical representation and delivered our model performance limitations. In addition, we provided a comparative analysis between ML and DL methods. We have concluded our task by giving a brief overview of the evaluation metrics and with a list of hyperparameters.

### 5.2 Experimental Results

The experiments are deployed on two corpora: cricket and restaurant of the original corpus polarity, aiming to detect the score by the BSTC algorithm. The training and test splits could bind all examples from all classes involved to keep data from being centralized. We observe a confusion matrix in the SVM classification algorithm in ML. It is used in extensively and

widely reported as the best classifier in the literature for SA. We intend to experiment on other classifiers like Logistic Regression (LR), K-Nearest Neighbors (KNN), Random Forest (RF) algorithm. In DL, we have experimented on a hybrid neural network on cricket datasets in terms of variation using fine-tuning such as dropout, optimizer regularization, learning rate, adding multi-layer, etc. In the first degree of SA, we aimed to determine the accuracy of the ML and DL approach experiment.

### 5.2.1 Experimental Result of ML on UiGram Model

We used a supervised ML classification algorithm to evaluate our experiment to classify our data. The evaluation of our result is measured through a confusion matrix including classifier metrics called accuracy, precision, recall, and f1-score with the help of using Spyder, python IDE environment. Among the classifiers, SVM with linear kernel trick ( $c=1$ ) is the best for giving good results in new observations because SVM has found better accuracy in finding text classification.

At least 20% of the dataset has been randomly chosen for the testing dataset, and the rest of the data is trained to classify the polarity. A standard feature matrix called Term Frequency - Inverse Document Frequency (Tf-Idf) vectorizer calculates the feature matrix. It maps text or words into a significant representation number.

Table 5.1: Weighted average of precision, recall, f1-score & accuracy in unigram model for both dataset.

| Dataset    | Polarity      | Precision | Recall | F1-score | Accuracy | Support | Feature Matrix | No. of Feature Word |
|------------|---------------|-----------|--------|----------|----------|---------|----------------|---------------------|
| Restaurant | -1            | 0.76      | 0.47   | 0.58     | 77.91%   | 123     | UniGram        | 3454                |
|            | 0             | 0.44      | 0.33   | 0.37     |          | 12      |                |                     |
|            | +1            | 0.72      | 0.90   | 0.80     |          | 259     | BiGram         | 6673                |
|            | Weighted Avg. | 0.78      | 0.77   | 0.76     | Total    | 412     |                |                     |
| Cricket    | -1            | 0.80      | 0.94   | 0.86     | 78.69%   | 389     | UniGram        | 3751                |
|            | 0             | 0.56      | 0.21   | 0.30     |          | 41      |                |                     |
|            | +1            | 0.74      | 0.56   | 0.63     |          | 166     | BiGram         | 12854               |
|            | Weighted Avg. | 0.80      | 0.78   | 0.74     | Total    | 596     |                |                     |

Since we have used the BTSC algorithm to calculate our sentence score, Table 5.1 shows algorithm results in classifying polarity with expected accuracy. Around 78% accuracy in both cricket and restaurant datasets is achieved on the UniGram model. In a multi-class confusion matrix, we use a weighted average to define our metrics because macro and micro averages can not give accurate results on the same number of instances. As weighted average precision in restaurant are 78% and 80% in cricket datasets, our extended Bangla sentiment dictionary construction is quietly proved in both score and polarity determination.

### 5.2.1.1 Support Vector Machine Classification on Tf-Idf Model

SVM algorithm is performed on destining boundaries through hyperplanes to discrete a class from the others. The primary purpose of the SVM algorithm is to construct hyperplanes among corpus samples so that the classification between classes expands as much possible. Around 78% accuracy in both cricket and restaurant datasets is achieved on the UniGram model. Having a multi-class confusion matrix, we use a weighted average to define our metrics because macro and micro averages can not give accurate results on the same number of instances. As weighted average precision in restaurant is 78% and 80% in cricket, our extended Bangla sentiment dictionary construction is quietly proved in both score and polarity determination.

### 5.2.1.2 Confusion Matrix on UniGram Model

Having a higher value of precision and recall vindicates a good model. Precision is a measurement of accuracy with respect to the prediction of a specific label or class. It is measured by the ratio of true positive ( $TP$ ) of a particular label or class in the sum of true positive ( $TP$ ) and false positive ( $FP$ ), indicated in equation (5.1). F1-score is a combined formula of precision and recall shown in equation (5.2). Here in Figure 5.1a and 5.1b show the percentage of classifying polarity during SVM classification. At Figure 5.1a, at most 62.86% positive, 14.08% negative and 0.97% neutral comments are identified as  $TP$  during

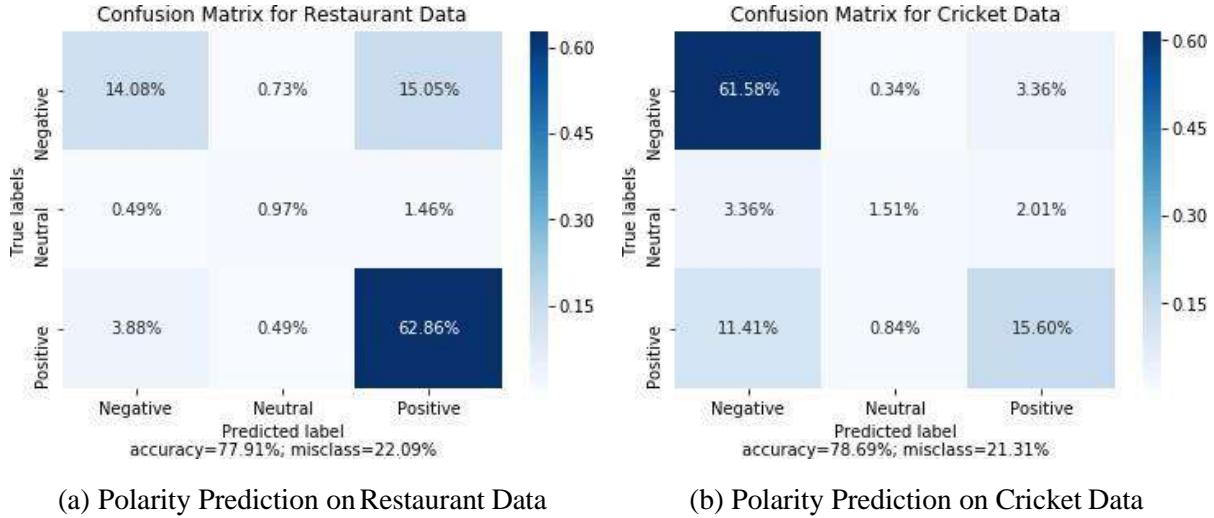


Figure 5.1: **BTSC** algorithm polarity prediction on both dataset

SVM classification. Here  $FP$  is much lower than the  $TP$ . Total 4.37% positive and 3.7% negative comments incorrectly identified those classes with lower  $FP$  than  $TP$ . At most 61.58% negative, 15.60% positive and 1.51% neutral comments are identified as  $TP$ , shows on Figure 5.1b. Total 4.37% positive and 3.7% negative comments have incorrectly identified those classes with much lower  $FP$  than  $TP$ .

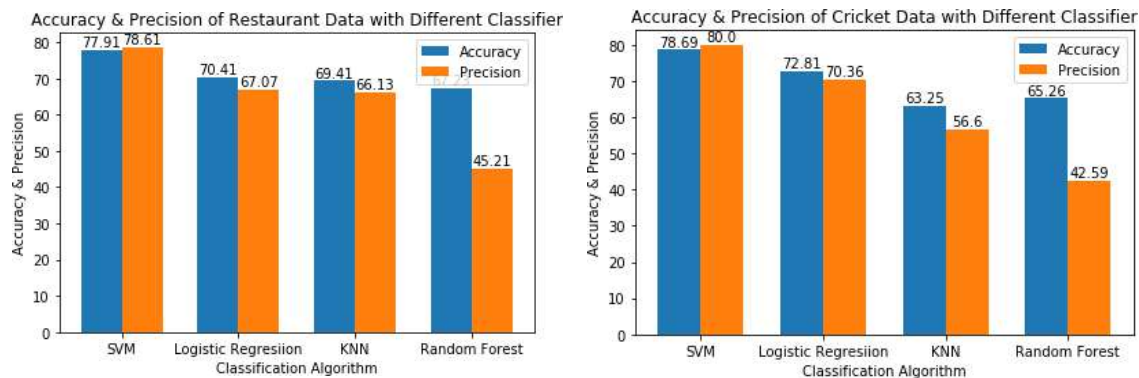
$$Precision(label) = \frac{TP}{TP + FP} \quad (5.1)$$

$$F1 - score(label) = \frac{2 * Precision * Recall}{Precision + Recall} \quad (5.2)$$

### 5.2.2 Performance on Different ML Classifier Approach on UniGram Model

Besides, other classifiers like logistic regression (LR), k-nearest neighbors (KNN), random forest (RF) algorithms are applied on our UniGram model. Among these classifiers, SVM shows better accuracy.

Figure 5.2a and 5.2b shows the performance of different classifiers. At Figure 5.2a, we have achieved best accuracy 77.91% and precision 78.61% at restaurant dataset. At Figure



(a) Performance of different classifier in Restaurant Data (b) Performance of different classifier in Cricket Data

Figure 5.2: Visualization performance of different classifier in restaurant and cricket dataset

5.2b, 78.69% accuracy and 80% precisions are achieved in cricket dataset in SVM classification. Both datasets have shown much better accuracy and precision rather than other classification.

### 5.2.3 Experiment on BiGram Model and Comparison Between UniGram

#### Model Approach on SVM

After finding quite improvement in the UniGram approach in the Tf-Idf model, we created another BiGram model in Tf-Idf word vectorization. In this model, we performed a Linear SVM classification algorithm; finally, accuracy is attained in both datasets 80.58% and 82.21% respectively, which is greater than the UniGram approach and also having precision 80.92 and 81.64 in both datasets. Figure 5.3 shows the performance and summary of the SA of the UniGram and Bigram models. This analysis shows that cricket data has higher accuracy than the restaurant dataset because the cricket dataset has trained more data than the restaurant data.

#### 5.2.3.1 Comparison Between Existing ML Model and Proposed Approach

Figure 5.4 shows a comparative summary of our results with previous studies, although the comparison is not fair because of the use of varying datasets. The dataset used in this study



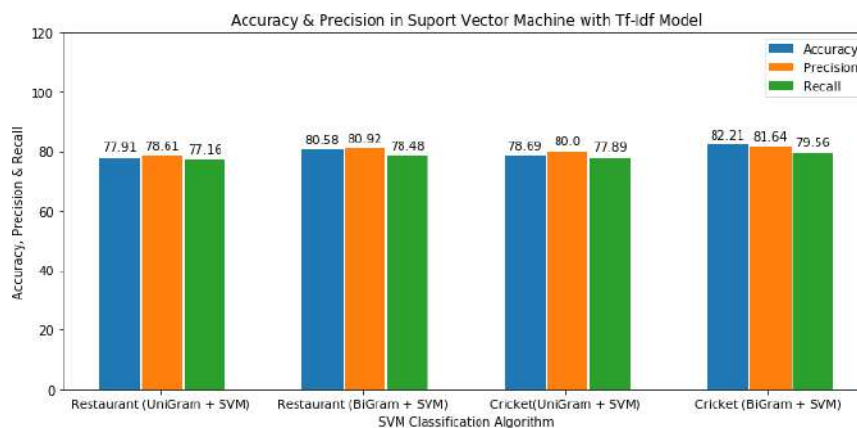


Figure 5.3: Visualization performance of UniGram & BiGram model on SVM classifier

is unique compared to other research works. The study in [14] achieved 69% accuracy when trained on 1000 tweets in UniGram with negation features. The study set only one rule to specify the sentiment from the text by counting only positive and negative words from tweets. The limitation of [14] is the use of only one rule, which cannot properly detect the real sentiment from the text.

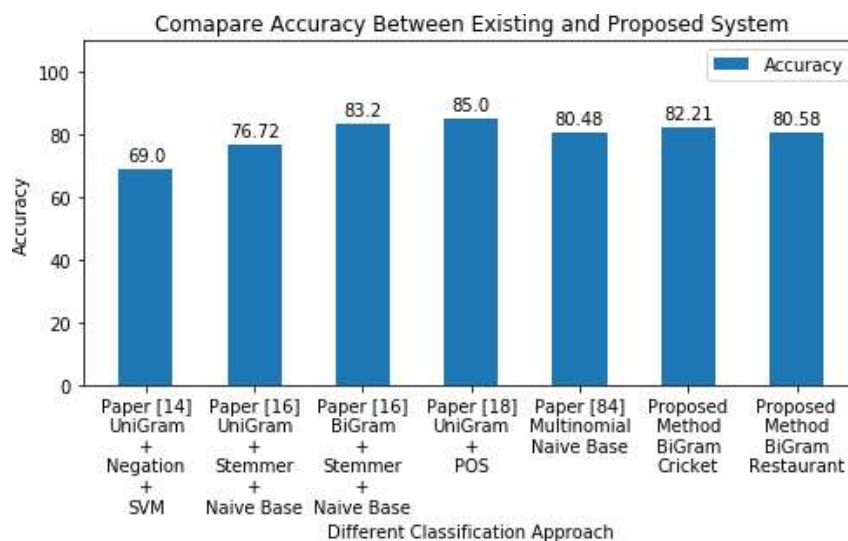


Figure 5.4: Comparing accuracy between the existing and proposed systems

In [16], a precision value of 77%, a recall/TPR value of 68% and a F1-score of 72% were achieved. The authors in [16] manually normalized the Bangla text with the help of valence shifting words by detecting one adjective in a sentence. However, the study did not consider

complex and compound sentences. The study in [18] trained only 850 and tested 200 texts in RF classification, achieving 85% accuracy for positive and negative data; however, the volume of the training dataset is small. The study determined sentiments by only assigning feature words to positive and negative tags without considering the POS tagger. In a recent research [85], 80.48% accuracy was attained during the 6-fold cross-validation approach in multinomial Naive Bayes classification. The authors used polarity from the given dataset as a target output without generating any text sentiment. This means the study did not apply any semantic connections between text and polarity.

#### **5.2.4 Result Discussion on ML Approach**

However, our UniGram and BiGram features have higher accuracy with precision, recall or TPR, and f1-score than previous works. Moreover, our Unigram and Bigram feature matrices have included stemming, normalization, and POS tagger processes. The dataset used in our study is much larger than the other studies shown in Figure 5.4. Still, our results are comparable with others and thus acceptable.

### **5.3 Experimental Result on Deep Neural Networks**

The overview of our proposed hybrid deep neural network architecture is followed by a set of inputs of reviews represented by a feature representation model Word2Vec, having multi-layer perception, different learning rate regularization, dropout parameter, block of neural code with multiple dependencies, etc. In this work, we have used TensorFlow at the backend mechanism developed by Google, which is actively maintained, robust and flexible enough to be competent in developing neural network models.

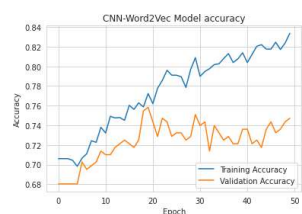
### 5.3.1 Deep Neural Network Model Training and Fitting:

After the construction of DL models from chapter 4, section 4.3, we compile each model with our word embedding matrix [128D, 200D, 300D] as per the requirement by setting the parameter loss function as categorical cross entropy and the optimizer as adam at a learning rate on different points shown at Table 5.2. Table 5.2 shows the summarization of our different parameters on each model. After compiling the model, we fits the model with our training and validation data, with a batch size of 256 with 50 epochs except using a batch size of 16 with 50 epochs in the BERT-LSTM model.

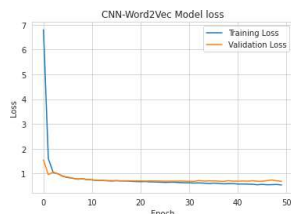
We have validated our testing datasets, and for a better understanding of every experiment, at Figure 5.6, we have shown each model training accuracy (TA) vs. validation accuracy (VA) with respect to epoch and training loss (TL) vs. validation loss (VL) with respect to epoch during training the model. The X-axis indicates the epoch number, while the Y-axis indicates the training, validation accuracy, and training validation loss. TA and VA determine whether the model was overfitting or not and TL and VL determine whether the model was overfitting or underfitting. The VA and VL for dataset how well the model will perform for the unseen future data.

### 5.3.2 Results and Analysis:

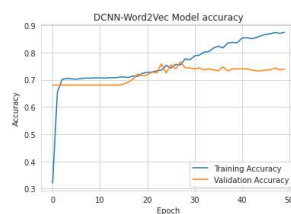
We have conducted fourteen experiments to offer a reasonable comparison between recent DL algorithms and traditional methods. Table 5.3 shows the experimental results of each model, which is obtained by setting the optimal values for each parameter in the model through trial and error. As for the sentiment classification, different models have outperformed on different learning rate(lr) parameters to achieve outstanding results. From these results, the researcher can identify which is perfect for their sentiment classification task in the Bangla low-resourced dataset.



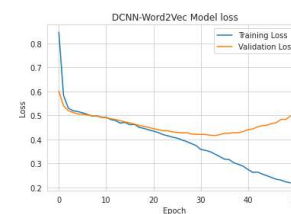
(i.a) TA, VA on CNN



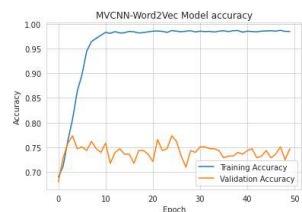
(i.b) TL, VL on CNN



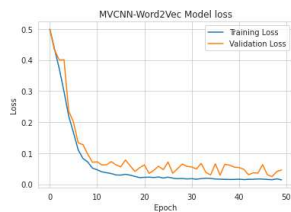
(i.c) TA, VA on DCNN



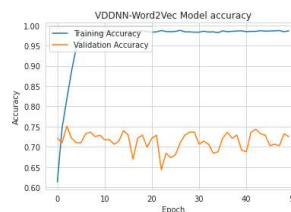
(i.d) TL, VL on DCNN



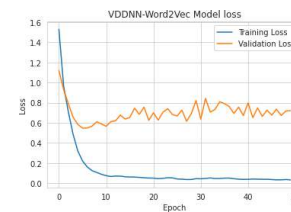
(ii.a) TA, VA on MVCNN



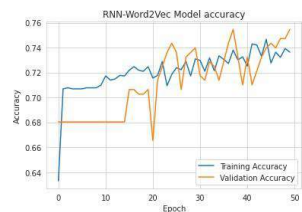
(ii.b) TL, VL on MVCNN



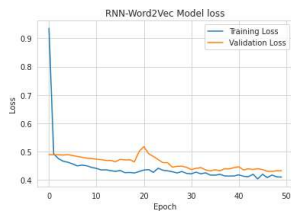
(ii.c) TA, VA on VDCNN



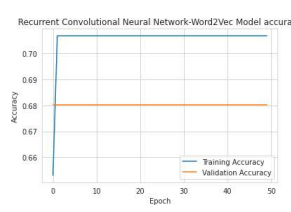
(ii.d) TL, VL on VDCNN



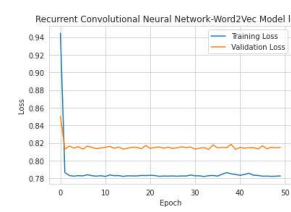
(iii.a) TA, VA on RNN



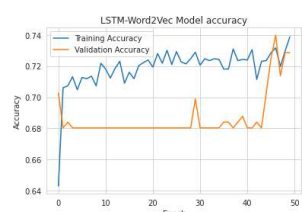
(iii.b) TL, VL on RNN



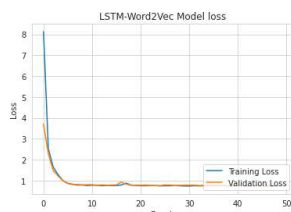
(iii.c) TA, VA on RCNN



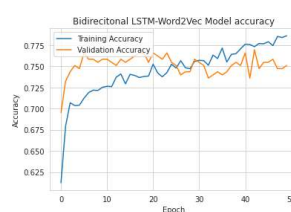
(iii.d) TL, VL on RCNN



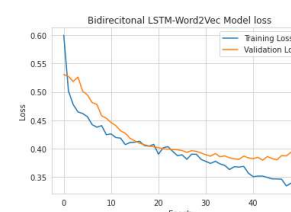
(iv.a) TA, VA on LSTM



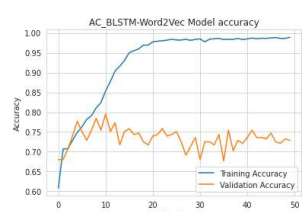
(iv.b) TL, VL on LSTM



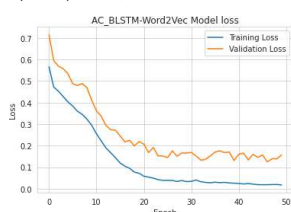
(iv.c) TA, VA on Bi-LSTM



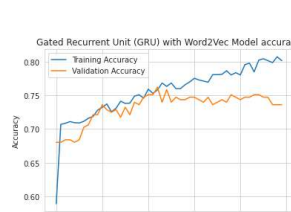
(iv.d) TL, VL on Bi-LSTM



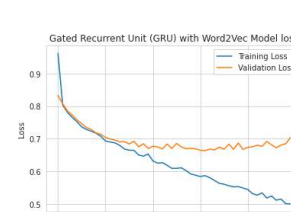
(v.a) TA, VA on AC\_Bi-LSTM



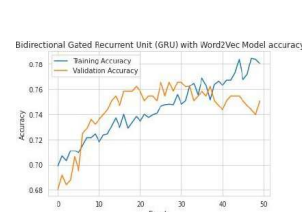
(v.b) TL, VL on AC\_Bi-LSTM



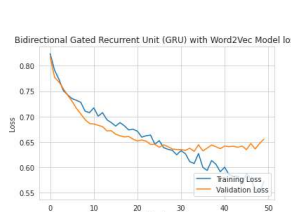
(v.c) TA, VA on GRU



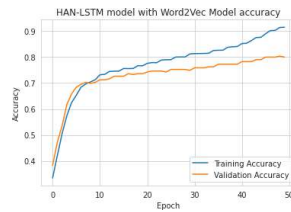
(v.d) TL, VL on GRU



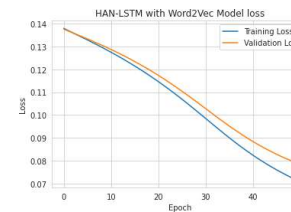
(vi.a) TA, VA on Bi-GRU



(vi.b) TL, VL on Bi-GRU



(vi.c) TA, VA on HAN-LSTM



(vi.d) TL, VL on HAN-LSTM

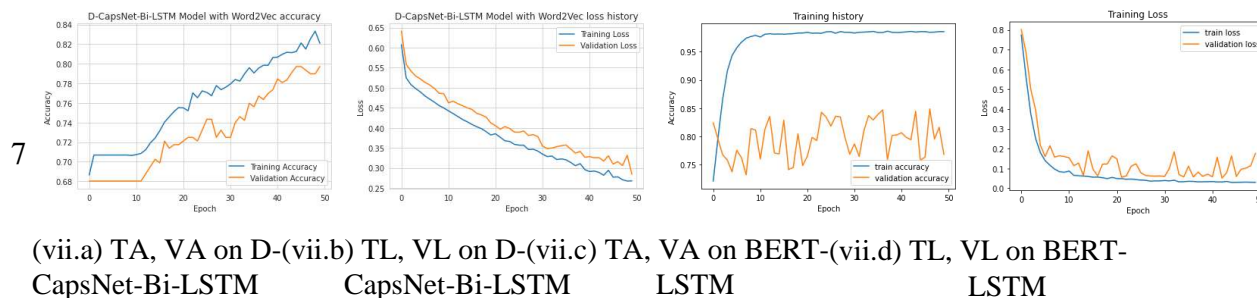


Figure 5.6: LC of each model training accuracy (TA), validation accuracy (VA) vs. Epoch and LC of each model training loss (TL), validation loss (VL) vs. epoch

### 5.3.3 CNN Based Model:

Through the comparison between our convolutional neural network types models (CNN, DCNN, MVCNN and VDCNN), it is noticed that VDC achieves the highest 77.85% accuracy, 80.16% precision, 79.56% recall, and 79.86% F1-score. VDCNN and MVCNN are more complex models than CNN and DCNN because of using three-dimensional (D) [128D, 200D, 300D] word embedding layers. CNN achieves 74.50% accuracy, which is better than the 73.49% accuracy of the CNN model.

It is shown in Figure 5.6 (i.a) that at about 20 epochs, the CNN model achieves the highest testing accuracy. In contrast, as shown in Figure 5.6 (i.c), DCNN decreases testing accuracy from point 25 epochs. It is shown in Figure 5.6 (ii.a) and (ii.c) that MVCNN and VDCNN models have a huge deflection between training and testing accuracy because of having a high dependency on using a multichannel layer with different iteration filters [filter kernel size = 3, 4, 5]. MVCNN uses a two-dimensional [128D, 200D] word embedding layer, and VDCNN uses a three-dimensional [128D, 200D, 300D]. We keep the dropout rate at 0.5 on each layer; however, changing kernel size when a variable size of zero padding1D is added to perform over spatial dimension to the output.

When we add several nodes in the layer in our model, the capacity increases, which means accuracy, precision, and recall increase. Our training data is small, so our model is pretty

Table 5.2: Hyperparameter dependency on each model

| Model Name        | Learning Rate (lr) | Word Embedding Dimension                              | Layer  |   | Dropout                               |
|-------------------|--------------------|---|--|---|---------------------------------------|
| CNN               | (lr = 0.01)        | 128D  | Conv1D(filter = 300, Kernel_size = 5, Relu), GlobalMaxpool1D             |   | SpatialDropout1D = 0.5, Dropout = 0.5 |
| DCNN              | (lr = 0.0001)      | 128D  | ZeroPadding1D(49,49)   | Conv1D(64, Kernel_size = 50),   | SpatialDropout1D = 0.3, Dropout = 0.5 |
|                   |                    |   | ZeroPadding1D(24,24)   | Conv1D(64, Kernel_size = 25), Kmaxpooling(k=5)                        |                                       |
| MVCNN             | (lr = 0.001)       | (128D, 200D) iteration with filter size 3, 4, 5       | ZeroPadding1D(2,2)   | Conv1D(100, Kernel_size = 2),   | Dropout = 0.5, l2(0.0.1)              |
|                   |                    |   | ZeroPadding1D(3,3),  | Conv1D(100, Kernel_size = 3),   |                                       |
|                   |                    |   | ZeroPadding1D(4,4)   | Conv1D(100, Kernel_size = 4), Kmaxpooling(k=10)                       |                                       |
| VDCNN             | (lr = 0.01)        | (128D, 200D, 300D) iteration with filter size 3, 4, 5 | ZeroPadding1D(2,2)   | Conv1D(100, Kernel_size = 2), GlobalMaxpool1D(k=3),                   | Dropout = 0.5                         |
|                   |                    |   | ZeroPadding1D(3,3)   | Conv1D(100, Kernel_size = 3), GlobalMaxpool1D(k=4),                   |                                       |
|                   |                    |   | ZeroPadding1D(4,4)   | Conv1D(100, Kernel_size = 5), GlobalMaxpool1D(k=3), Kmaxpooling(k=10) | Dropout = 0.5, l2(0.0.1)              |
| RNN               | (lr = 0.01)        | 128D  | GlobalMaxpool1D  |   | SpatialDropout1D = 0.4, Dropout = 0.4 |
| RCNN              | (lr = 0.001)       | 128D  | Conv1D(100, Kernel_size = 2)   | 4 Block Bi-LSTM(unit = 32, recurrent_dropout = 0.1)                   | Dropout = 0.3                         |
| LSTM              | (lr = 0.01)        | 128D  | 1 Block LSTM(unit = 32)  |   | SpatialDropout1D = 0.3, Dropout = 0.4 |
| Bi-LSTM           | (lr = 0.001)       | 128D  | 1 Block of Bi-LSTM(unit = 32, dropout = 0.2, recurrent_dropout = 0.1)    |   | SpatialDropout1D = 0.3, Dropout = 0.3 |
| AC_Bi-LSTM        | (lr = 0.001)       | 128D  | Conv1D(100, Kernel_size = 2)   | Conv1D(100, Kernel_size = 30),  | Dropout = 0.3                         |
|                   |                    |   |  | Conv1D(100, Kernel_size = 40),  |                                       |
|                   |                    |   |  | Conv1D(100, Kernel_size = 50),  |                                       |
|                   |                    |   |  | Conv1D(100, Kernel_size = 60), LSTM(unit = 32)                        |                                       |
| GRU               | (lr = 1e-3)        | 128D  | 1 Block GRU (unit=32)  | Conv1D(64, Kernel_size = 5)   | SpatialDropout1D = 0.3, Dropout = 0.3 |
|                   |                    |   |  | GlobalAverageMaxpooling1D, GlobalMaxpool1D                            |                                       |
| Bi-GRU            | (lr = 1e-3)        | 128D  | Conv1D(64, Kernel_size = 4)  | 1 Block Bi-GRU (unit=32, dropout = 0.2, recurrent_dropout = 0.1)      | SpatialDropout1D = 0.3, Dropout = 0.3 |
|                   |                    |   | GlobalAverage Maxpooling1D, GlobalMaxpool1D                              |   |                                       |
| HAN-LSTM          | (lr = 1e-4)        | 128D  | 1 block of Bi-LSTM (units = 128), sentence attention layer               |   | Dropout = 0.3                         |
|                   |                    |   | 1 block of Bi-LSTM (units = 128), word attention layer                   |   |                                       |
| D-CAPSNET-Bi-LSTM | (lr = 0.0001)      | 128D  | 1 Block of Bi-LSTM(unit = 256, dropout = 0.25, recurrent_dropout = 0.25) |   | SpatialDropout1D = 0.3, Dropout = 0.3 |
|                   |                    |   | Capsule Layer (Low Level, Higher Level Capsule)                          |   |                                       |
| BERT-LSTM         | (lr = 1e-5)        | Pretrained BERT                                       | 12 Block Bert Encoder  | 1 block of Bi-LSTM (units = 32)                                       | None                                  |

small; however, increasing model layers can drive a more precise model. The model should be larger if more training data are given in the model. In our experiment, multilayer perception of CNN is applied in the DCNN, VDCNN, and MVCNN model as there is no bound to use a specific number of layers, so this stacked layer is susceptible to generalizing our

model better. Adding continuous layers of convolution and pooling operation in CNN might lose spatial information on classifying data.

Table 5.3: Accuracy, precision, recall, f1-score measures of the model of bangla dataset cricket reviews.

| MODEL NAME        | Train Accuracy | Test Accuracy | Train Precision | Test Precision | Train Recall | Test Recall | Train F1-score | Test F1-score |
|-------------------|----------------|---------------|-----------------|----------------|--------------|-------------|----------------|---------------|
| CNN               | 0.8758         | 0.7349        | 0.8936          | 0.7856         | 0.8947       | 0.7966      | 0.8422         | 0.7123        |
| DCNN              | 0.8765         | 0.7450        | 0.8945          | 0.7565         | 0.8661       | 0.6849      | 0.8801         | 0.7188        |
| MVCNN             | 0.9642         | 0.7651        | 0.9668          | 0.7627         | 0.9633       | 0.7300      | 0.9650         | 0.7458        |
| VDCNN             | 0.9616         | 0.7785        | 0.9625          | 0.8016         | 0.9615       | 0.7956      | 0.9620         | 0.7986        |
| RNN               | 0.7658         | 0.7383        | 0.8040          | 0.7888         | 0.7126       | 0.6695      | 0.7554         | 0.7242        |
| RCNN              | 0.7069         | 0.7383        | 0.7094          | 0.7780         | 0.7094       | 0.7780      | 0.7094         | 0.7780        |
| LSTM              | 0.7281         | 0.7416        | 0.7406          | 0.7853         | 0.7103       | 0.7267      | 0.7251         | 0.7548        |
| Bi-LSTM           | 0.8433         | 0.7814        | 0.8697          | 0.7795         | 0.8197       | 0.7352      | 0.8439         | 0.7703        |
| AC_Bi-LSTM        | 0.9571         | 0.7550        | 0.9581          | 0.7395         | 0.9558       | 0.7380      | 0.9570         | 0.7387        |
| GRU               | 0.8441         | 0.7584        | 0.8286          | 0.7029         | 0.8476       | 0.7183      | 0.8379         | 0.7104        |
| Bi-GRU            | 0.8307         | 0.7517        | 0.7133          | 0.7127         | 0.9101       | 0.7321      | 0.8064         | 0.7416        |
| HAN-LSTM          | 0.9884         | 0.7852        | 0.9908          | 0.7343         | 0.9872       | 0.8012      | 0.9890         | 0.7659        |
| D-CAPSNET-Bi-LSTM | 0.8297         | 0.8082        | 0.7629          | 0.8100         | 0.7332       | 0.7305      | 0.7477         | 0.7544        |
| BERT-LSTM         | 0.9568         | 0.8418        | 0.8584          | 0.8645         | 0.8672       | 0.7849      | 0.8979         | 0.8227        |

### 5.3.4 RNN Based Model:

Sequential models such as RNN and RCNN have similar test accuracy of 73.83%; however, RNN achieved 78.88% precision, greater than RCNN. Besides preserving LSTM and Bi-LSTM, the Bi-LSTM model performs well on cricket datasets with an optimal accuracy of 78.14%. However, the LSTM model performs well with a testing precision of 78.53% that is greater than the Bi-LSTM model.

At arbitrary time intervals, the three gates of RNN remember the propagation of information into the cell. It is shown in Figure 5.6 (iii.a) that RNN TA and VA curves are overlapped because of facing difficulties in capturing long-term dependencies and co-relation between words. However, in Figure 5.6 (iv.a) and (iv.c), LSTM and Bi-LSTM models capture long span word relations in text. As RCNN and AC\_Bi-LSTM models have Conv1d layer [kernel size = 2] however AC\_Bi-LSTM has a channeling Conv1D layer [kernel\_size = 30, 40, 50, 60] for that

reason, there is a huge gap in TA, VA in Figure 5.6 (v.a) and has the highest testing recall of 95.58% and testing F1-score of 95.70%.

### **5.3.5 GRU Based Model:**

As GRU intends in the last hidden state to represent the sentences which means modeling the the whole sentence causes neglecting main key parts of words, this might result in an incorrect prediction. From Table 5.3, while the learning rate is so high in GRU and Bi-GRU models, training and testing accuracy, precision is not getting higher than LSTM, Bi-LSTM model. At most 75.84% accuracy is achieved on the LSTM model with an average of 71% precision, recall, and f1-score value. As GRU limits the stream of data just like the LSTM units, however, except for utilizing a memory unit, the LSTM model performs well on this dataset. For this reason, we have applied the LSTM model as a hybrid layer on attention, capsule, and BERT-based model.

In Figure 5.6 (v.c) and (vi.a), both GRU-based models have similar curves regarding testing accuracy of 75.84% and 75.17%. The Bi-GRU model intersects at epoch points 5 and 35, which means it has a bi-directional dependency to co-relate words in a text.

### **5.3.6 Attention and Capsule Based Model:**

All models are relatively smooth with respect to learning rate (lr), and also variation in hidden size, hyperparameters (i.e., filters, kernel size, dropout) causes oscillation in curves. From Figure no. 6 of (vi.c), (vi.d), (vii.a), and (vii.b), the training and testing accuracy are increasing with respect to epoch while training and testing loss are decreasing with exponentially that shows an ideal state of the model. At most, 78.52% and 80.12% of testing accuracy and recalls are produced in the HAN-LSTM model, and 80.82% and 81.00% testing accuracy, precisions are produced in D-CAPSET-Bi-LSTM.

The attention level in words and sentences has increased model accuracy regarding other



CNN, RNN type models. The calculation of attention vectors co-related to word and sentence level determines the less content for constructing the document representation. The main advantage of the capsule modules is to resolve the max pooling level conversion for feature extractions, which means the improvement of CNN and RNN type models. Because losing the information in polling layers might cause less accuracy. Again, augmenting the compositional capsule network with a k-clustering mechanism will improve the classification accuracy of our HA -LSTM model.

### **5.3.7 Transformer Based Model**

The results of the BERT-LSTM model have a high learning rate ( $lr = 1e-5$ ), producing maximum accuracy of 84.18% precision of 86.45% compared to other models. Adding an LSTM layer on the pre-trained BERT model amplifies the core advancement in classification accuracy over embedding models. That means BERT is more able to represent semantic and syntactic features. This language model representation has substantially improved over other models at a state-of-the-art result compared to the Word2Vec model.

### **5.3.8 Result Analysis for Augmented Dataset in Transformer Based Model**

From practical engineering perceptions, text data enhancement can significantly expand the amount of data and improve the effectiveness of the deep learning model. When compared, it can be seen that the accuracy of the BERT-LSTM model is slightly lower for the augmented dataset than the stand-alone datasets. For future work, we will generate new samples by importing new augmented methods for text classification and apply hybrid DL models. Some of those hybrid models can be attention-based and dynamic routing based. We hope that researchers can develop a more acceptable model performance by introducing these strategies. The results of augmented dataset in BERT-LSTM model is depicted at Table no.

Table 5.4: Accuracy, precision, recall, f1-score measures of the model of augmented dataset cricket and restaurant reviews.

| MODEL NAME            | Train Accuracy | Test Accuracy | Train Precision | Test Precision | Train Recall | Test Recall | Train F1-score | Test F1-score |
|-----------------------|----------------|---------------|-----------------|----------------|--------------|-------------|----------------|---------------|
| BERT-LSTM (lr = 2e-5) | 0.8435         | 0.8111        | 0.8935          | 0.8911         | 0.7967       | 0.786       | 0.8419         | 0.8347        |

### 5.3.9 Comparison Between Existing DL model with our Proposed Hybrid

#### Neural Network Approach

At last, we compare our work with other existing approaches in both DL and ML approaches. Table 5.5 compares our work with respect to accuracy measurements. In [38], authors achieved 55% accuracy with three category sentiment on the above nine thousand social comments and in [37], authors showed 82.42% accuracy in four thousand movie reviews dataset in two category sentiment in LSTM network where our LSTM model achieves 74.16% accuracy. Similarly, in [39, 86], LSTM network has achieved 65.97% and 46.80% accuracy, respectively, in [45] attention-based LSTM (A-LSTM) achieved 65.97% accuracy. However, our hierarchical attention-based LSTM (HAN-LSTM) and combined C -LSTM [46] achieved 75.01% accuracy. AC\_Bi-LSTM and D-CAPS ET-Bi-LSTM hybrid models achieve greater accuracy than those research. Since there are drawbacks in preprocessing data on pronoun type word replacing, however, they still manage to achieve 85.67% accuracy in the Bi-LSTM model on ten thousand comments. Our Bi-LSTM gained much satisfactory results of 78.14% accuracy on the cricket ABSA dataset. Our CNN type model with multi-channel, i.e., DCNN, VDCNN, MVCNN model achieves higher accuracy than [45, 39] type CNN model.

#### 5.3.9.1 Comparison of ML vs DL Approach in Terms of Accuracy

Our supervised ML-based approach on SA with rule-based achieved satisfactory accuracy on the SVM classifier with 82.21% accuracy; however, the long-term dependencies between words are not considered on the bi-gram approach. The traditional ML-based bag of words (BOW) approach does not capture semantic relation between words where hidden layers in

Table 5.5: Comparison of major sentiment classifiers in both ML and DL regarding accuracy.

| Research Work   | Context                             | Dataset | Methods                     | Accuracy                               | Our Approach Accuracy  |
|-----------------|-------------------------------------|---------|-----------------------------|--|--|
| Our ML Approach | Cricket ABSA Dataset                | 2979    | Bi-Gram, SVM                | 82.21%                                 | Our Proposed ML Approach   |
| [37]            | Movie Reviews                       | 4000    | LSTM                        | 82.42% for 2 category                  | 74.16% for 3 category  |
| [38]            | Social media, news, product reviews | 9337    | RNN (LSTM)                  | 78% for 2 category, 55% for 3 category | 74.16% for 3 category  |
| [41]            | Facebook                            | 10000   | Bi-LSTM                     | 85.67%                                 | 78.14% for 3 category  |
| [45]            | Cricket ABSA Dataset                | 2979    | A-LSTM<br>A-CNN             | 66.06%<br>72.06%                       | 78.52% HAN-LSTM Model  |
| [46]            | Web Site                            | 1000    | CNN-LSTM                    | 75.01%                                 | 75.50% AC_Bi-LSTM Model  |
| [39]            | Social Media                        | 8910    | LSTM<br>CNN<br>NB<br>SVM    | 65.97%<br>60.89%<br>60.79%<br>59.18%   | 74.16% LSTM,<br>77.85% VDCNN,<br>80.82% D-CAPSNET-Bi-LSTM,<br>84.18% BERT-LSTM |
| [87]            | Microblogging sites                 | 3000    | Word2Vec with Hellinger PCA | 70.00%                                 |  |
| [86]            | Social Media                        | 1000    | LSTM                        | 46.80%                                 |  |

neural network boosts the model with the help of contextual relationship between words are retained by the word embedding. ML classifiers, i.e., SVM, Naive Base, etc., do not perform well on unstructured noisy datasets. For example, target categories overlap. Both approaches require a lot of large labeled corpus for training to deliberate best prediction; however, the semantic understanding between the sequences of data is preserved in the DL approach. Although the DL-based model requires more initial tuning parameters and each model provides different results, one does not require considering feature engineering.

### 5.3.10 Result Discussion of Our Proposed Hybrid DL Approach

From the experimental results, our DL mechanism of SA has the highest accuracy, precision, and recall against the ML approach. It is noticed that DL has an expressive improvement while containing high unbalancing between positive-negative classes. The core advantage in DL is that it learns high-level features from data in an exponential perspective through multi hidden layers with hyper-tuned parameters and provides better results than others.

### 5.3.11 Complexity Factors of DL Model:

DL model complexity implies investigating neural network generalization capability and limitation on the training process. Since our hybrid DL models are hyper-parameterized, however, it has very little effect on model complexity [88]. Model complexity indicates how the neural network model express its behavior on distribution function or activation function [89], prevents overfitting by adding L1, L2 regularization to the loss function [90] and the amplification coefficient ( $w_{ij}$ ) which is defined by the multilayer perception of hidden neurons. By selecting an activation function, i.e., ReLU, Sigmoid, tanh in-network hidden layers, an active module for learning and computing complex tasks takes the piecewise nonlinear transformation to the input. Pooling functions such GlobalAverage Maxpooling1D, GlobalMaxpool1D with a variety of filter sizes on feature maps are needed to reduce fixed size vectors. However, the DL model's size impacts model complexity, i.e., number of filters, kernel size, hidden neurons, dropout rate, depth and width efficiency of model, training, and testing time. This is partially noted in Table 5.2. In our work, we have limited our experiment in detecting the accuracy of SA in Bangla text with the help of LDD and BTSC. In the future, we will conduct the complexity of our DL algorithm on a broad scale.

## 5.4 Conclusion

This section is structured around the details of experiments and discusses the results in both ML and DL methodologies. The central hindrance aspect of conducting SA is related to the fact that opinions are intended too subjective. The proposed chapter leads to overcoming this task by providing a method of ML and DL-based blocks of a SA system. We examined the BiGram model as a multi-word feature vector and the unigram features. Our initial experiments noticed that the aggregation of only multi-word features like Unigram and BiGram dependency features slightly performs well using our proposed rule-based BTSC algorithm due to the dataset sparseness. Word2Vec is a learning representation generated

from an unstructured text that enables the model to capture more dependencies from the target context on the DL approach. BERT-LSTM, D-CAPSNET-Bi-LSTM, and HAN-LSTM models have shown state-of-the-art performance on our SA task.

# Chapter 6

## Conclusion

With the rapidly growing of Internet users, SA depends on the dataset of particular content. A lexicon-based extended data dictionary is developed based on a specific domain, restaurant, and cricket. Manual construction of positive and negative dictionaries with weighted values is complex while mining data from the Bangla dataset. However, precise observation of these data will give more accurate results in classifying polarity. In this thesis, the BTSC algorithm detects the three types of polarity from the sentences using the Bangla extended dictionary approach. Since a document belongs to more than one category, any rule-based algorithm is required to detect text category and classification for categorical specific domain-based data. We achieved the highest 82.21% accuracy in cricket on the BiGram feature matrix. In the confusion matrix, identifying neutral data has performed less than the other two polarities. Every dataset has its variabilities. If we use, i.e., fifty (50) thousand datasets in our ML process, our result will predict more accuracy than the obtained accuracy with the current dataset. For this, we need to construct a huge volume of the sentimental dictionary. In the future, we will apply more datasets to our method. Approximately five thousand data is used as a sentimental dictionary in our approach. Moreover, there is still a scope to redefine the weights of the dictionary. To make this approach even more significant, we introduce a categorical-based data dictionary that will play a pioneering role in further research.

This research investigated the most noteworthy work on SA on cricket reviews as a low-resourced Bangla dataset using various DL architectures. This empirical study is an initial dive into the lexicon dictionary-based approach on neural network mechanisms. We measure the performance of our work based on accuracy, precision, recall, and f1 score. Firstly, we developed a lexicon-based approach and used the BTSC algorithm to detect polarity from preprocessed text from our previous work. Then we implemented the popular DL models, i.e., CNN, DCNN, MVCNN, VDCNN, RNN, RCNN, LSTM, Bi-LSTM, AC-Bi-LSTM, GRU, Bi-GRU, HAN-LSTM, D-CAPSNET-Bi-LSTM, BERT-LSTM with setting as customized and tuned with hyperparameters on individual models. We found that LSTM had better results than CNN and RNN type models. Then we used attention, capsule, Bert based mechanism by adding LSTM layer, and the result showed significant improvement, which is indeed effective in the sentiment classification effect. This hybrid model, HAN-LSTM and D-CAPSNET-Bi-LSTM, and semantic learning representations (word2vec) have an excellent performance in accuracy, precision, recall, and f1-score. Finally, Transformer based as a pretrained BERT with LSTM as a hybrid model were used for this classification task, and it surpasses other results having satisfactory accuracy and precision.

However, researchers do not publish their datasets; this LDD dataset will be published for research since it can be further enriched. Our Bangla cricket review dataset is relatively small concerning the benchmark dataset. The lack of enough large training corpus in the Bangla domain is the drawback of our SA task. We have identified trinary polarity in cricket reviews since this dataset is not properly balanced, so using balanced data for each polarity in the training process might enhance the accuracy of prediction results. Because increasing the amount of training data can assist in promoting the accuracy of prediction results. We could not use a well-pretrained model due to lack of hardware resources, so in the future, we developed a large corpus and trained it with various parameters or layers with a tuned model.

Although we gained many satisfactory results, it still has scope for further improvement in our approach. In the future, we will conduct our research using other transformer models, i.e., ALBERT, ELECTRA, RoBERT, and multilayer and hybrid capsule as well as attention-based models.

## 6.1 Journal Publications:

Publication resulting from this thesis:

- Bhowmik, Nitish Ranjan, Mohammad Arifuzzaman, M. Rubaiyat Hossain Mondal, and M. S. Islam. "Bangla text sentiment analysis using supervised machine learning with extended lexicon dictionary." *Natural Language Processing Research* 1, no. 3-4 (2021): 34-45. [DOI <https://doi.org/10.2991/nlpr.d.210316.001>]
- Bhowmik, Nitish Ranjan, Mohammad Arifuzzaman, and M. Rubaiyat Hossain Mondal. "Sentiment analysis on Bangla text using extended lexicon dictionary and deep learning algorithms." *Array* 13 (2022): 100123. [DOI <https://doi.org/10.1016/j.array.2021.100123>]



# **Appendix A**

## **Appendix Section**

Table A.1: Word frequency and inverse document frequency list in restaurant dataset

| Word        | Frequency |
|-------------|-----------|
| পরিষেবা     | 4         |
| খাদ্য       | 3         |
| অর্ডার      | 3         |
| খাবার       | 3         |
| অনুভূতি     | 2         |
| চমৎকার      | 2         |
| মরুভূমি     | 2         |
| আনন্দদায়ক  | 2         |
| বায়ুগুণ    | 2         |
| অবিশ্বাস্য  | 2         |
| স্থান       | 2         |
| আনন্দময়    | 2         |
| চিত্তাকর্ষক | 1         |
| সুস্বাদু    | 1         |
| ওয়াইন      | 1         |
| তালিকা      | 1         |
| ধীরেধীরে    | 1         |
| ভিড়        | 1         |
| বাড়ে       | 1         |
| শীতল        | 1         |
| প্রম্পট     | 1         |
| বিনয়ী      | 1         |
| অভিজ্ঞতা    | 1         |
| মেনু        | 1         |
| খাও         | 1         |
| গ্রেট       | 1         |
| ভারতীয়     | 1         |
| রঙিন        | 1         |
| রঙ          | 1         |
| পরিবেশ      | 1         |

(a) Word frequency

| Word        | Inverse Document Frequency |
|-------------|----------------------------|
| পরিষেবা     | 1.252762968                |
| খাদ্য       | 1.466337069                |
| অর্ডার      | 1.466337069                |
| খাবার       | 1.466337069                |
| অনুভূতি     | 1.791759469                |
| চমৎকার      | 1.791759469                |
| মরুভূমি     | 1.791759469                |
| আনন্দদায়ক  | 1.791759469                |
| বায়ুগুণ    | 1.791759469                |
| অবিশ্বাস্য  | 1.791759469                |
| স্থান       | 1.791759469                |
| আনন্দময়    | 2.397895273                |
| চিত্তাকর্ষক | 2.397895273                |
| সুস্বাদু    | 2.397895273                |
| ওয়াইন      | 2.397895273                |
| তালিকা      | 2.397895273                |
| ধীরেধীরে    | 2.397895273                |
| ভিড়        | 2.397895273                |
| বাড়ে       | 2.397895273                |
| শীতল        | 2.397895273                |
| প্রম্পট     | 2.397895273                |
| বিনয়ী      | 2.397895273                |
| অভিজ্ঞতা    | 2.397895273                |
| মেনু        | 2.397895273                |
| খাও         | 2.397895273                |
| গ্রেট       | 2.397895273                |
| ভারতীয়     | 2.397895273                |
| রঙিন        | 2.397895273                |
| রঙ          | 2.397895273                |
| পরিবেশ      | 2.397895273                |

(b) WORD IDF





Table A.4: Representation of TF-IDF feature matrix for restaurant dataset

| DOC\Word | পরিষেবা  | পরিষেবা  | পরিষেবা  | পরিষেবা  | পরিষেবা  | পরিষেবা  | পরিষেবা   | পরিষেবা   | পরিষেবা  | পরিষেবা  |
|----------|----------|----------|----------|----------|----------|----------|-----------|-----------|----------|----------|
|          | 0        | 1        | 2        | 3        | 4        | 5        | 6         | 7         | 8        | 9        |
| DOC-1    | 0        | 0        | 0        | 0        | 0.199084 | 0        | 0         | 0         | 0        | 0        |
| DOC-2    | 0        | 0.293267 | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| DOC-3    | 0        | 0        | 0        | 0        | 0        | 0.44794  | 0         | 0         | 0        | 0        |
| DOC-4    | 0.178966 | 0        | 0        | 0        | 0        | 0.255966 | 0.255966  | 0.2559656 | 0.255966 | 0        |
| DOC-5    | 0.104397 | 0.122195 | 0.122195 | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| DOC-6    | 0.125276 | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0.179176 |
| DOC-7    | 0        | 0        | 0.133303 | 0.133303 | 0.162887 | 0        | 0         | 0         | 0        | 0        |
| DOC-8    | 0        | 0        | 0.24439  | 0.24439  | 0        | 0        | 0         | 0         | 0        | 0        |
| DOC-9    | 0.208794 | 0.24439  | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0.298627 |
| DOC-10   | 0        | 0        | 0        | 0.104738 | 0        | 0        | 0.127983  | 0.1279828 | 0.127983 | 0        |
| স্থান    | স্থান    | স্থান    | স্থান    | স্থান    | স্থান    | স্থান    | স্থান     | স্থান     | স্থান    | স্থান    |
| 10       | 11       | 12       | 13       | 14       | 15       | 16       | 17        | 18        | 19       |          |
| 0        | 0        | 0.266433 | 0.266433 | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0.599474 | 0.599474 | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0.199825 | 0.1998246 | 0.199825  | 0        | 0        |
| 0.179176 | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0.23979  |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0.127983 | 0.342556 | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| প্রম্পট  | প্রম্পট  | প্রম্পট  | প্রম্পট  | প্রম্পট  | প্রম্পট  | প্রম্পট  | প্রম্পট   | প্রম্পট   | প্রম্পট  | প্রম্পট  |
| 20       | 21       | 22       | 23       | 24       | 25       | 26       | 27        | 28        | 29       |          |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0.23979  | 0.23979  | 0        | 0        | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0.21799  | 0.21799  | 0        | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0.399649 | 0        | 0        | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0.399649 | 0.399649 | 0         | 0         | 0        | 0        |
| 0        | 0        | 0        | 0        | 0        | 0        | 0        | 0.1712782 | 0.171278  | 0.171278 | 0.171278 |

Table A.5: Sample data from cricket dataset

| DOC No. | Sentence   |
|---------|--|
| DOC-1   | বাংলাদেশ জিতবে ইনশাআল্লাহ। শুভ কামনা রইলো টাইগার জন্য।                             |
| DOC-2   | জয় বাংলা, হারলেও বাংলাদেশ, জিতলেও বাংলাদেশ। শুভ কামনা।                            |
| DOC-3   | সাকিব ছাড়া বাংলাদেশ স্পিন নির্ভর বোলিং লাইন খুব সাধারণ মানের পারফরম্যান্স।        |
| DOC-4   | অধিনায়ক রিয়াদ আরো পরিপক্ব আর দায়িত্বশীল হবেন, বাজে অধিনায়ক                     |
| DOC-5   | বাংলাদেশ জিতার কথা, এই রান শ্রীলংকা পারা সম্ভাবনা খুব কম।                          |
| DOC-6   | বাংলাদেশ দল পারফরম্যান্স এর কোনো দাম নেই।  |
| DOC-7   | তামিম, সাকিব ও মুশফিক উপর সব সময় নির্ভর হয়ে খেললে এই রকম বাজে অবস্থা হতেই থাকবে। |
| DOC-8   | ক্রিকেট টিম নিয়ে এই ধরনের নাটক গ্রহণযোগ্য নয়!।                                   |
| DOC-9   | স্পিন সামনে নয়, সাকিব সামনে ধরাশায়ী টিম শ্রীলংকা।                                |
| DOC-10  | মাশরাফি তুমি তোমার সেরা প্রমাণ করে দাও। টাইগার ইজ টাইগার।                          |

Table A.6: Word frequency and inverse document frequency list in cricket dataset

| Word Frequency |           | Inverse Document Frequency |                            |
|----------------|-----------|----------------------------|----------------------------|
| WORD           | Frequency | WORD                       | Inverse Document Frequency |
| বাংলাদেশ       | 6         | বাংলাদেশ                   | 1.098612289                |
| জিত            | 3         | জিত                        | 1.466337069                |
| টাইগার         | 3         | টাইগার                     | 1.791759469                |
| সাকিব          | 3         | সাকিব                      | 1.466337069                |
| শুভ            | 2         | শুভ                        | 1.791759469                |
| কামনা          | 2         | কামনা                      | 1.791759469                |
| স্পিন          | 2         | স্পিন                      | 1.791759469                |
| নির্ভর         | 2         | নির্ভর                     | 1.791759469                |
| পারফরম্যান্স   | 2         | পারফরম্যান্স               | 1.791759469                |
| অধিনায়ক       | 2         | অধিনায়ক                   | 2.397895273                |
| বাজে           | 2         | বাজে                       | 1.791759469                |
| টীম            | 2         | টীম                        | 1.791759469                |
| ইনশাআল্লাহ     | 1         | ইনশাআল্লাহ                 | 2.397895273                |
| জয়            | 1         | জয়                        | 2.397895273                |
| বাংলা          | 1         | বাংলা                      | 2.397895273                |
| হারল           | 1         | হারল                       | 2.397895273                |
| বোলিং          | 1         | বোলিং                      | 2.397895273                |
| সাধারণ         | 1         | সাধারণ                     | 2.397895273                |
| রিয়াদ         | 1         | রিয়াদ                     | 2.397895273                |
| পরিপক্ব        | 1         | পরিপক্ব                    | 2.397895273                |
| দায়িত্বশীল    | 1         | দায়িত্বশীল                | 2.397895273                |
| রান            | 1         | রান                        | 2.397895273                |
| শ্রীলংকা       | 1         | শ্রীলংকা                   | 2.397895273                |
| পারা           | 1         | পারা                       | 2.397895273                |
| সম্ভাবনা       | 1         | সম্ভাবনা                   | 2.397895273                |
| দল             | 1         | দল                         | 2.397895273                |
| দাম            | 1         | দাম                        | 2.397895273                |
| তামিম          | 1         | তামিম                      | 2.397895273                |
| মুশিফক         | 1         | মুশিফক                     | 2.397895273                |
| খেলল           | 1         | খেলল                       | 2.397895273                |
| অবস্থা         | 1         | অবস্থা                     | 2.397895273                |
| থাকব           | 1         | থাকব                       | 2.397895273                |
| ক্রিকেট        | 1         | ক্রিকেট                    | 2.397895273                |
| নাটক           | 1         | নাটক                       | 2.397895273                |
| গ্রহণযোগ্য     | 1         | গ্রহণযোগ্য                 | 2.397895273                |
| ধরাশায়ী       | 1         | ধরাশায়ী                   | 2.397895273                |
| শ্রীলংকা       | 1         | শ্রীলংকা                   | 2.397895273                |
| মাশরাফি        | 1         | মাশরাফি                    | 2.397895273                |
| সেরা           | 1         | সেরা                       | 2.397895273                |
| প্রমাণ         | 1         | প্রমাণ                     | 2.397895273                |









# Bibliography

- [1] Liu B. *Sentiment analysis and opinion mining*. Morgan & Claypool Publishers, 2012.
- [2] Alexander Pak and Patrick Paroubek. “Twitter as a corpus for sentiment analysis and opinion mining”. In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC’10)*. 2010.
- [3] Maite Taboada et al. “Lexicon-based methods for sentiment analysis”. In: *Computational linguistics* 37.2 (2011), pp. 267–307.
- [4] Xiaoyong Liu and W Bruce Croft. “Statistical language modeling for information retrieval”. In: *Annual Review of Information Science and Technology* 39.1 (2005), pp. 1–31.
- [5] Wikipedia. *Bengali language*. [https://en.wikipedia.org/wiki/Bengali\\_language](https://en.wikipedia.org/wiki/Bengali_language). accessed on 17.04.2020.
- [6] Richard Rosenfeld and Robert Fornango. “The impact of economic conditions on robbery and property crime: The role of consumer sentiment”. In: *Criminology* 45.4 (2007), pp. 735–769.
- [7] Ronan Collobert et al. “Natural language processing (almost) from scratch”. In: *Journal of machine learning research* 12.ARTICLE (2011), pp. 2493–2537.
- [8] Mohammad A Karim. *Technical challenges and design issues in bangla language processing*. IGI Global, 2013.
- [9] Guixian Xu et al. “Chinese text sentiment analysis based on extended sentiment dictionary”. In: *IEEE Access* 7 (2019), pp. 43749–43762.
- [10] Nawaf A Abdulla et al. “Arabic sentiment analysis: Lexicon-based and corpus-based”. In: *2013 IEEE Jordan conference on applied electrical engineering and computing technologies (AEECT)*. IEEE. 2013, pp. 1–6.
- [11] Eissa M Alshari et al. “Effective method for sentiment lexical dictionary enrichment based on Word2Vec for sentiment analysis”. In: *2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP)*. IEEE. 2018, pp. 1–5.

- [12] Shamsul Arafin Mahtab, Nazmul Islam, and Md Mahfuzur Rahaman. “Sentiment analysis on bangladesh cricket with support vector machine”. In: *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*. IEEE. 2018, pp. 1–4.
- [13] Clayton J Hutto and Eric Gilbert. “Vader: A parsimonious rule-based model for sentiment analysis of social media text”. In: *Eighth international AAAI conference on weblogs and social media*. 2014.
- [14] Shaika Chowdhury and Wasifa Chowdhury. “Performing sentiment analysis in Bangla microblog posts”. In: *2014 International Conference on Informatics, Electronics & Vision (ICIEV)*. IEEE. 2014, pp. 1–6.
- [15] KM Azharul Hasan, Mosiur Rahman, et al. “Sentiment detection from bangla text using contextual valency analysis”. In: *2014 17th International Conference on Computer and Information Technology (ICCIT)*. IEEE. 2014, pp. 292–295.
- [16] Md Saiful Islam et al. “Supervised approach of sentimentality extraction from bengali facebook status”. In: *2016 19th International Conference on Computer and Information Technology (ICCIT)*. IEEE. 2016, pp. 383–387.
- [17] Rashedul Amin Tuhin et al. “An Automated System of Sentiment Analysis from Bangla Text using Supervised Learning Techniques”. In: *2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*. IEEE. 2019, pp. 360–364.
- [18] Nusrath Tabassum and Muhammad Ibrahim Khan. “Design an Empirical Framework for Sentiment Analysis from Bangla Text using Machine Learning”. In: *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*. IEEE. 2019, pp. 1–5.
- [19] Shunxiang Zhang et al. “Sentiment analysis of Chinese micro-blog text based on extended sentiment dictionary”. In: *Future Generation Computer Systems* 81 (2018), pp. 395–403.
- [20] Sanjida Akter and Muhammad Tareq Aziz. “Sentiment analysis on facebook group using lexicon based approach”. In: *2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*. IEEE. 2016, pp. 1–4.
- [21] Bo Pang and Lillian Lee. “A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts”. In: *arXiv preprint cs/0409058* (2004).
- [22] Yoshua Bengio et al. “A neural probabilistic language model”. In: *The journal of machine learning research* 3 (2003), pp. 1137–1155.
- [23] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. “A convolutional neural network for modelling sentences”. In: *arXiv preprint arXiv:1404.2188* (2014).

- [24] Alexis Conneau et al. “Very deep convolutional networks for text classification”. In: *arXiv preprint arXiv:1606.01781* (2016).
- [25] Wenpeng Yin and Hinrich Schütze. “Multichannel variable-size convolution for sentence classification”. In: *arXiv preprint arXiv:1603.04513* (2016).
- [26] Tomáš Mikolov et al. “Recurrent neural network based language model”. In: *Eleventh annual conference of the international speech communication association*. 2010.
- [27] Lizhong Xiao, Guangzhong Wang, and Yang Zuo. “Research on patent text classification based on word2vec and LSTM”. In: *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*. Vol. 1. IEEE. 2018, pp. 71–74.
- [28] Peng Zhou et al. “Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling”. In: *arXiv preprint arXiv:1611.06639* (2016).
- [29] Depeng Liang and Yongdong Zhang. “AC-BLSTM: asymmetric convolutional bidirectional LSTM networks for text classification”. In: *arXiv preprint arXiv:1611.01884* (2016).
- [30] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. “Recurrent neural network for text classification with multi-task learning”. In: *arXiv preprint arXiv:1605.05101* (2016).
- [31] Rajib Rana. “Gated recurrent unit (GRU) for emotion classification from noisy speech”. In: *arXiv preprint arXiv:1612.07778* (2016).
- [32] Zichao Yang et al. “Hierarchical attention networks for document classification”. In: *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*. 2016, pp. 1480–1489.
- [33] Zhengxi Tian et al. “Attention aware bidirectional gated recurrent unit based framework for sentiment analysis”. In: *International Conference on Knowledge Science, Engineering and Management*. Springer. 2018, pp. 67–78.
- [34] Yuxiao Chen et al. “Twitter sentiment analysis via bi-sense emoji embedding and attention-based LSTM”. In: *Proceedings of the 26th ACM international conference on Multimedia*. 2018, pp. 117–125.
- [35] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).
- [36] Zhengjie Gao et al. “Target-dependent sentiment classification with BERT”. In: *IEEE Access* 7 (2019), pp. 154290–154299.

- [37] Rumman Rashid Chowdhury et al. “Analyzing sentiment of movie reviews in bangla by applying machine learning techniques”. In: *2019 International Conference on Bangla Speech and Language Processing (ICBSLP)*. IEEE. 2019, pp. 1–6.
- [38] Asif Hassan et al. “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.
- [39] Nafis Irtiza Tripto and Mohammed Eunos Ali. “Detecting multilabel sentiment and emotions from bangla youtube comments”. In: *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*. IEEE. 2018, pp. 1–6.
- [40] Md Habibul Alam, Md-Mizanur Rahoman, and Md Abul Kalam Azad. “Sentiment analysis for Bangla sentences using convolutional neural network”. In: *2017 20th International Conference of Computer and Information Technology (ICCIT)*. IEEE. 2017, pp. 1–6.
- [41] Abdullah Aziz Sharfuddin, Md Nafis Tihami, and Md Saiful Islam. “A deep recurrent neural network with bilstm model for sentiment classification”. In: *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*. IEEE. 2018, pp. 1–4.
- [42] Md Al-Amin, Md Saiful Islam, and Shapan Das Uzzal. “Sentiment analysis of Bengali comments with Word2Vec and sentiment information of words”. In: *2017 international conference on electrical, computer and communication engineering (ECCE)*. IEEE. 2017, pp. 186–190.
- [43] Asif Hassan et al. “Sentiment analysis on bangla and romanized bangla text (BRBT) using deep recurrent models”. In: *arXiv preprint arXiv:1610.00369* (2016).
- [44] Md Ferdous Wahid, Md Jahid Hasan, and Md Shahin Alom. “Cricket sentiment analysis from bangla text using recurrent neural network with long short term memory model”. In: *2019 International Conference on Bangla Speech and Language Processing (ICBSLP)*. IEEE. 2019, pp. 1–4.
- [45] Sadia Sharmin and Danial Chakma. “Attention-based convolutional neural network for Bangla sentiment analysis”. In: *AI & SOCIETY* 36.1 (2021), pp. 381–396.
- [46] Naimul Hossain et al. “Sentiment Analysis of Restaurant Reviews using Combined CNN-LSTM”. In: *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. IEEE. 2020, pp. 1–5.
- [47] Duyu Tang et al. “Building large-scale twitter-specific sentiment lexicon: A representation learning approach”. In: *Proceedings of coling 2014, the 25th international conference on computational linguistics: Technical papers*. 2014, pp. 172–182.

- [48] Maite Taboada et al. “Lexicon-Based Methods for Sentiment Analysis”. In: *Computational Linguistics* 37 (June 2011), pp. 267–307. DOI: 10.1162/COLI\_a\_00049.
- [49] Hilke Reckman et al. “teragram: Rule-based detection of sentiment phrases using sas sentiment analysis”. In: *Second Joint Conference on Lexical and Computational Semantics (\*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. 2013, pp. 513–519.
- [50] Supaporn Buddeewong and Worapoj Kreesuradej. “A new association rule-based text classifier algorithm”. In: *17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)*. IEEE. 2005, 2–pp.
- [51] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. “Thumbs up?: sentiment classification using machine learning techniques”. In: *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics. 2002, pp. 79–86.
- [52] Kushal Dave, Steve Lawrence, and David M Pennock. “Mining the peanut gallery: Opinion extraction and semantic classification of product reviews”. In: *Proceedings of the 12th international conference on World Wide Web*. 2003, pp. 519–528.
- [53] AtikRahman. *Bangla ABSA Datasets for Sentiment Analysis*. [https://github.com/AtikRahman/Bangla\\_ABSA\\_Datasets](https://github.com/AtikRahman/Bangla_ABSA_Datasets). accessed on 05.19.2020.
- [54] Md Rahman, Emon Kumar Dey, et al. “Datasets for aspect-based sentiment analysis in bangla and its baseline evaluation”. In: *Data* 3.2 (2018), p. 15.
- [55] Grammar Hub. *Adjective*, নাম িবেশষণ. <http://www.grammarbd.com/en-grammar/adjective>. accessed on 17.04.2020.
- [56] Grammar Hub. *Adverb*, িকর্যা িবেশষণ. <http://www.grammarbd.com/en-grammar/adverb>. accessed on 17.04.2020.
- [57] Saimon Hossain. *BLTK, The Bengali Natural Language Processing Toolkit*. <https://pypi.org/project/bltk/>. accessed on 28.03.2020.
- [58] Ranks NL. *Bengali Stopwords - Ranks NL*. <https://www.ranks.nl/stopwords/bengali>. accessed on 08.01.2020.
- [59] TensorFlow API Developer. *Keras Preprocessing Library, Module:tf.keras.preprocessing*. [https://www.tensorflow.org/api\\_docs/python/tf/keras/preprocessing/](https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/). accessed on 07.31.2021.

- [60] TensorFlow API Developer. *Keras Preprocessing Library, Module: tf.keras.preprocessing.text.Tokenizer*. [https://www.tensorflow.org/api\\_docs/python/tf/keras/preprocessing/text/Tokenizer](https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer). accessed on 07.31.2021.
- [61] TensorFlow API Developer. *Keras Preprocessing Library, Module: tf.keras.preprocessing.sequence.pad\_sequences*. [https://www.tensorflow.org/api\\_docs/python/tf/keras/preprocessing/sequence/pad\\_sequences](https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/sequence/pad_sequences). accessed on 07.31.2021.
- [62] Sagor Sarker. *BanglaBERT: Bengali Mask Language Model for Bengali Language Understanding*. <https://github.com/sagorbrur/bangla-bert>. accessed on 07.31.2021.
- [63] Thomas Wolf et al. “Huggingface’s transformers: State-of-the-art natural language processing”. In: *arXiv preprint arXiv:1910.03771* (2019).
- [64] Huggingface Library Developer. *BertTokenizer*. [https://huggingface.co/transformers/model\\_doc/bert.html#berttokenizer](https://huggingface.co/transformers/model_doc/bert.html#berttokenizer). accessed on 07.31.2021.
- [65] Huggingface Library Developer. *Bert PreTrainedTokenizerBase encode\_plus function*. [https://huggingface.co/transformers/internal/tokenization\\_utils.html#transformers.tokenization\\_utils\\_base.PreTrainedTokenizerBase.encode\\_plus](https://huggingface.co/transformers/internal/tokenization_utils.html#transformers.tokenization_utils_base.PreTrainedTokenizerBase.encode_plus). accessed on 07.31.2021.
- [66] Tomas Mikolov et al. “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781* (2013).
- [67] Tomas Mikolov et al. “Distributed representations of words and phrases and their compositionality”. In: *Advances in neural information processing systems*. 2013, pp. 3111–3119.
- [68] Edward Ma. *NLP Augmentation*. <https://github.com/makcedward/nlpaug>. 2019.
- [69] Terry Traylor, Jeremy Straub, Nicholas Snell, et al. “Classifying fake news articles using natural language processing to identify in-article attribution as a supervised learning estimator”. In: *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*. IEEE. 2019, pp. 445–449.
- [70] A Gural Vural et al. “A framework for sentiment analysis in turkish: Application to polarity detection of movie reviews in turkish”. In: *Computer and Information Sciences III*. Springer, 2013, pp. 437–445.
- [71] Eric Nguyen. “Chapter 4 - Text Mining and Network Analysis of Digital Libraries in R”. In: *Data Mining Applications with R*. Ed. by Yanchang Zhao and Yonghua Cen. Boston: Academic Press, 2014, pp. 95–115. ISBN: 978-0-12-411511-8. DOI: <https://doi.org/10.1016/B978-0-12-411511-8.00004-9>. URL: <https://www.sciencedirect.com/science/article/pii/B9780124115118000049>.

- [72] Brian D Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.
- [73] Michel Jose Anzanello and Flavio Sanson Fogliatto. “Learning curve models and applications: Literature review and research directions”. In: *International Journal of Industrial Ergonomics* 41.5 (2011), pp. 573–583.
- [74] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. “Deep learning (adaptive computation and machine learning series)”. In: *Cambridge Massachusetts* (2017), pp. 321–359.
- [75] Maryem Rhanoui et al. “A CNN-BiLSTM model for document-level sentiment analysis”. In: *Machine Learning and Knowledge Extraction* 1.3 (2019), pp. 832–847.
- [76] Ochilbek Rakhmanov. “On validity of sentiment analysis scores and development of classification model for student-lecturer comments using weight-based approach and deep learning”. In: *Proceedings of the 21st Annual Conference on Information Technology Education*. 2020, pp. 174–179.
- [77] Eftekhar Hossain et al. “Sentilstm: a deep learning approach for sentiment analysis of restaurant reviews”. In: *International Conference on Hybrid Intelligent Systems*. Springer. 2020, pp. 193–203.
- [78] Ahmed Alsayat. “Improving Sentiment Analysis for Social Media Applications Using an Ensemble Deep Learning Language Model”. In: *Arabian Journal for Science and Engineering* 47.2 (2022), pp. 2499–2511.
- [79] Jie Wang, Bingxin Xu, and Yujie Zu. “Deep learning for aspect-based sentiment analysis”. In: *2021 International Conference on Machine Learning and Intelligent Systems Engineering (MLISE)*. IEEE. 2021, pp. 267–271.
- [80] Junyoung Chung et al. “Empirical evaluation of gated recurrent neural networks on sequence modeling”. In: *arXiv preprint arXiv:1412.3555* (2014).
- [81] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.
- [82] Colin Raffel et al. “Exploring the limits of transfer learning with a unified text-to-text transformer”. In: *arXiv preprint arXiv:1910.10683* (2019).
- [83] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: *arXiv preprint arXiv:1607.06450* (2016).
- [84] Jakob Uszkoreit. “Transformer: A novel neural network architecture for language understanding”. In: *Google AI Blog* 31 (2017).



- [85] Omar Sharif, Mohammed Moshiul Hoque, and Eftekhari Hossain. “Sentiment Analysis of Bengali Texts on Online Restaurant Reviews Using Multinomial Naive Bayes”. In: *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*. IEEE. 2019, pp. 1–6.
- [86] Kamal Sarkar. “Sentiment Polarity Detection in Bengali Tweets Using LSTM Recurrent Neural Networks”. In: *2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP) (2019)*, pp. 1–6.
- [87] Md Saiful Islam, Md Al-Amin, and Shapan Das Uzzal. “Word embedding with hellinger pca to detect the sentiment of bengali text”. In: *2016 19th International Conference on Computer and Information Technology (ICCIT)*. IEEE. 2016, pp. 363–366.
- [88] Sham M Kakade, Karthik Sridharan, and Ambuj Tewari. “On the complexity of linear prediction: Risk bounds, margin bounds, and regularization”. In: (2008).
- [89] Yoshua Bengio and Olivier Delalleau. “On the expressive power of deep architectures”. In: *International conference on algorithmic learning theory*. Springer. 2011, pp. 18–36.
- [90] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.