Array
Volume 13, March 2022, 100123

Sentiment analysis on Bangla text using extended lexicon dictionary and deep learning algorithms

Nitish Ranjan Bhowmik [a], Mohammad Arifuzzaman [b], M. Rubaiyat Hossain Mondal [a]

[a] Institute of Information and Communication Technology (IICT), Bangladesh University of Engineering and Technology(BUET), Dhaka, Bangladesh
[b] East West University, Dhaka, Bangladesh

Received 13 August 2021, Revised 3 November 2021, Accepted 11 December 2021, Available online 1 January 2022.

https://doi.org/10.1016/j.array.2021.100123

Abstract

Sentiment analysis (SA) is a subset of natural language processing (NLP) research. In the case of categorical weighted based dictionary with rule-based sentiment score generation, no work in SA has been done yet in Bangla language using deep learning (DL) approaches. This paper proposes DL models for SA on Bangla text using an extended lexicon data dictionary (LDD). We implement the rule-based method Bangla text sentiment score (BTSC) algorithm for extracting polarity from large texts. These polarities are then fed into the neural network along with 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 top highest probability word is applied on embedding layer as a weighted matrix to fit the DL models. This paper also presents a remarkably detailed analysis of selective DL models with some 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. 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.

Previous Next

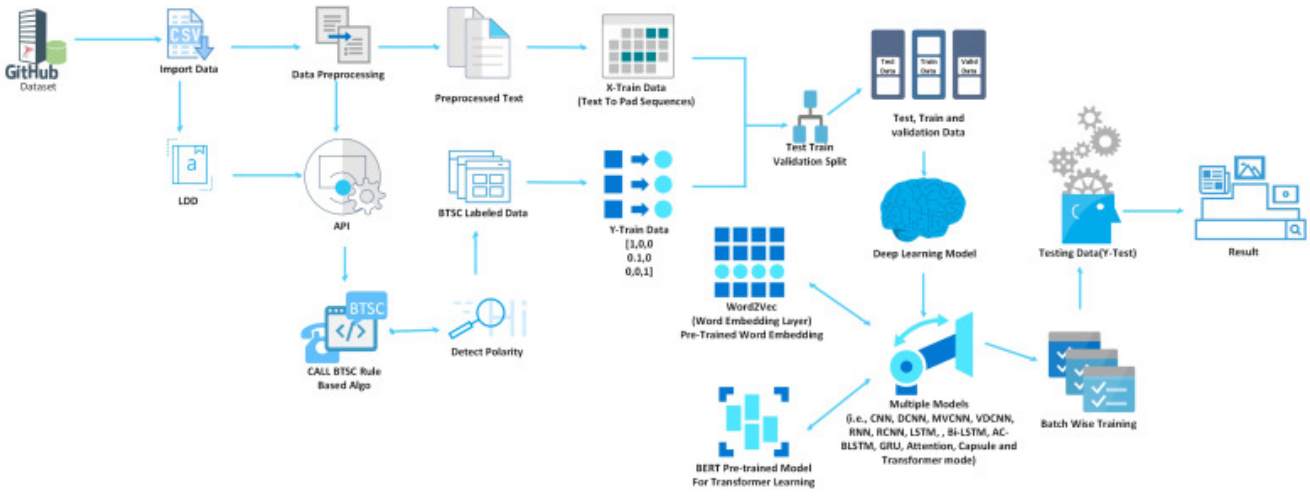
Keywords

Sentiment analysis; Bangla NLP; Deep learning; BTSC; Word2Vec; Fine-tuning; CNN; RNN; LSTM; GRU; Hybrid neural network; Attention mechanism; Capsule layers; BERT

1. Introduction

Social media has a wealth of information in the form of 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 huge label of opinionated corpus continuously emerges with the evocation of data classification and scalability. Natural language processing (NLP) can be used to purify out the noisy word and to discover pertinent insights from this flourishing rapidly data. In the recent year, many NLP researchers develop to find out the properties from the text, including emotion, polarity or subjectivity detection and document or context classification. Sentiment Analysis (SA) has fulfilled this demand for researchers to predict a context on positive, negative or neutral. SA or opinion extraction is narrated as collecting information from public content to generate people's attitude, expression 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 jam, road accidents, and best route policy such as feedback on every situation can be analyzed by people opinions from the social media sites. Again in the level of national military defense or law enforcement organization(i.e., police cyber-crime unit [2]) paid observation and attention to the public opinions (i.e., suicide activity [3]) what are their doing or saying activities in electronic media

net. The orientation and proactivity of a certain text are argued based upon the polarity and context which is 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 [4]. DL is a subset of multiple layers of neurons that perceives from nonlinear neural network with matrix representations and converts the output at a one level into an intense and abstract peak. A few machine learning techniques is driven out on the Bangla text for predicting opinions. In this paper, we combine rule-based with lexicon dictionary approach and several hybrid DL models to predict the text sentiment from Bangla text. The key contributions of our article include (i) the implementation of a rule based algorithm BTSC [5] for generating score from the text automatically with the help of categorical weighted lexicon data dictionary (LDD), (ii) Aggregation of our proposed BTSC polarity with our input text corpus and building a word embedding model (Word2Vec) in three dimensional space [128d, 200d, 300d] for learning representation in order to fit and train on DL models, and (iii) Developing several combinations of hybrid novel DL models of long short term memory network (LSTM) known as hierarchical attention based LSTM (HAN-LSTM), Dynamic routing based capsule neural network with Bi LSTM (D-CAPSNET-BiLSTM) and bidirectional encoder representations from Transformers (BERT) LSTM with customized proper training and setting hyperparameter tuning factors. Our paper is structured as follows: Section 2 provides a survey of SA using DL. Section 3 discusses the methodology of our research work which includes a text preprocessing mechanism to fit the data in the neural network model. In Section 4, a description is provided for the detailed experiments of the proposed models. Section 5 shows the performance results of the proposed model and compares the results with the existing research. Finally, Section 6 provides concluding remarks.



Download : Download high-res image (448KB) Download : Download full-size image

Fig. 1. Visualization of the Proposed System Architecture.

2. Related work

In this section, we sketch out the available methods with taxonomy which explore the influences on the several DL architectures and discuss how those methods are enhancing to operate in SA. In the interdisciplinary domain of NLP, sentiment classification was portrayed in [6], described a connection between subjectivity detection and polarity classification. In [7], authors showed a neural probabilistic model for learning simultaneously a consecutive representation of words and a probabilistic function to the word sequences. A simple one layer based convolutional neural network (CNN) approach is given in [7] to conduct sensitivity analysis on the text. Artificial neural network (ANN) does not work in large scale of inputs; however, CNN or Hybrid based CNN i.e., Dynamic CNN (DCNN) [8], Very Deep CNN (VDCNN) [9], variable-size convolutional filters i.e., (MVCNN) [10] model can do much better. DCNN uses a dynamic K-max polling and a global pooling operation over the text sequence, while VDCNN and MVCNN use different dimension of word embeddings on multiple filter sizes respectively in character and text level. 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 [11]. As RNN has problem of vanishing gradient descent, gradient explosion and lack of back propagation, those are mitigated in a modified version of RNN such as Long Short Term Memory Network (LSTM) [12], Bi-Directional LSTM [13], Asymmetric Convolutional Bidirectional LSTM (AC_Bi-LSTM) [14], Recurrent Convolutional Neural Network (RCNN) [15], Gated Recurrent Unit (GRU) [16]. Hierarchical Attention Network (HAN) based mechanism [17] on Bi-GRU [18] and LSTM [19] are also applied for document text classification because it works on between the hidden (encoder and decoder) layer to give weighted sum all features fed as an input. A recent NLP task, Transfer Neural Network BERT is published by the Google researchers [20], which learns contextual relation from words or text is also applied for SA [21].

A greater 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 [22] authors perform SA on 4000 positive and negative movie reviews which is manually translated into Bangla and obtained accuracy on LSTM at 82.42%. Another LSTM based approach is conducted on 9337 reviews for classifying polarity positive and negative sentiments and achieved accuracy at 78% [23]. In [24], the authors extract six types of emotion form different types of Bangla YouTube video comments by a CNN and LSTM based approach and showed 65.97% and 54.24% accuracy on three and five labels sentiment, respectively. Another CNN based single channel approach [25] is implemented on different domains from Bangla dataset however, they cannot maintain proper tuning in layers. A RNN type of network Bi-LSTM approach is applied on manually hand label dataset of 10000 comments from Facebook and they obtained accuracy at 85.67%; however, it has many notable drawbacks in data preprocessing [26]. In [27] authors obtained accuracy of 75.5% on word2vec model by tuning word co-occurrence score in word vector similarity. In [28] authors experimented on Bangla Romanized dataset and tested on a deep recurrent model LSTM and achieved accuracy at 55% for three categories. In [29] authors examined on aspect based sentiment analysis data that achieved 95% accuracy; however, their common and proper noun of Bangla words

rephrased by a global common words which is a hindrance to extract sentiment in lexicon based dictionary approach. In [30] authors collected 1000 online restaurant reviews from Foodpanda website for performing SA and deployed combined CNN with LSTM architecture with 300 dimensional Word2Vec pretrained model having validation accuracy of 75.01%. In [31] authors performed SA on 1002 public comments from newspaper with the help of BERT pretrained model and achieved accuracy on GRU at 71% on 2 class sentiment. A recent paper, authors [32] implement an attention based CNN model to analyze sentiment from Bangla text. However, in lexicon based approach a word may have different meanings on different domain; 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 [33], [34] authors build a sentiment detection mechanism from tweets by using sentiment lexicon and according with a linguistic rule-based approach [35]. 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.

3. Methodology

Fig. 1 shows the pictorial representation of our full approach. The construction of LDD and BTSC algorithm is taken form a study [5]. Although we will not deep dive into the core details like mathematical description and construction of the neural network architecture, we will summarize our approach of of the neural network model used in our experiment.

3.1. Data augmentation

In classification of text or images data augmentation technique is used for improving the performance of DL models. In our experiment, we have used lexicon data dictionary (LDD) from our previous work [5]. We have done prerequisite preprocessing part in building LDD from dataset. It includes like noise reduction, substitute words in lexicon, words shuffling mainly used for short text which is mostly related to data sampling analysis [36]. 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 that increases the data in training samples. We have used categorical aspect-based dataset (cricket) [37] that means a comment has a positive on $\boldsymbol{x}$ category and negative on $\boldsymbol{y}$ category or neutral on z category with specific polarity. However, according to paper [5], in Table no. 12, BTSC algorithm detects polarity and only extracts sentiments according to our global extended lexicon dictionary not using categorical sentimental dictionary.

Table 1. Demonstration on neural network based data preprocessing.

Method	Data	Comment
Text	জয় বাংলা কাপ্তাও আবার স্বাধীনতার মাস মার্চে। যার মাথা থেকে এমন চমৎকার আইডিয়া এসেছে তাকে স্যালুট।	Raw Text

Method	Data	Comment
Tokenizer	['জয়','বাংলা', 'কাপ', 'স্বাধীনতার', 'মাস', 'মার্চে', 'মাথা', 'চমৎকার', 'আইডিয়া', 'স্যালুট']	Sentence are tokenized
Stemming	['জয়', 'বাংলা', 'কাপ', 'স্বাধীনতা', 'মাস', 'মার্চ', 'মাথা', 'চমৎকার', 'আইডিয়া', 'স্যালুট']	Stemming word
text_to_sequences	[16, 170, 504, 81, 105, 450, 188, 64, 206, 4161, 788]	Encoded each word as a numeric number representation
pad_sequences (maxlen = 40)	[16, 170, 504, 81, 105, 450, 188, 64, 206, 4161, 788, 0, 0, 0, 0, 0, 0, 0, 0, ...]	padding the sequence as 40 length followed by extra 0's

3.2. Data preprocessing

According to this paper [5] we preprocess our data by removing stop words, unnecessary characters, performing tokenization, stemming, parts of speech (pos) tagging operation. We have collected data from Github repository [38] and used cricket dataset for conducting our experiment. However, in order to represent the text into neural network, we used a tensor-based matrix representation of the corpus (review) with its polarity together. For comparing to other text representation mechanisms, this sparse dense matrix takes less memory for fitting in neural network. We employ the Tensorflow neural network libraries simultaneously with Keras [39] for preprocessing our data. We use Keras tools tokenizer, text_to_sequences [40] and pad_sequences [41].

3.2.1. Neural network based data preprocessing

We tokenize the words of our training data for keeping a maximum number of words, text_to_sequences method to map the tokenized words in our vocabulary in a numeric representation. Then we find a maximum length (maxlen) of the text over encoded sequences. Finally, the resulted encoded sequences need to be the same length (maxlen value) by following 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 $[i_{conpusLength}, j_{maxlen}]$, where indexes i and j denote as row and column, respectively. For example, Table 1 shows the full demonstration of data preprocessing for neural network training.

Table 2. 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 তালে স্যালুট।	Raw Text
Sentence Piece Tokenizer	[['জয় বাংলা কাপ!'],, ['তাও আবার স্বাধীনতার মাস মার্চে।'],,	3 Sentence are tokenized

Method	Data	Comments
	['যার মাথা থেকে এমন চমৎকার আইডিয়া এসেছে তালে স্যালুট।']]	
Tokenization + Stemming	[['জয়', 'বাংলা', 'কাপ'] ['স্বাধীনতা ', 'মাস', 'মার্চ'],, ['মাথা', 'চমৎকার', 'আইডিয়া', 'স্যালুট']]	Tokenize, Stemming word in every sentence
text_to_sequences	[[16, 170, 504], [81, 105, 450], [188, 64, 206, 788]]	Encoded each sentence as a numeric number representation
pad_sequences (maxSeqLen= 25 maxSentenceLen = 3)	[[16, 170, 504, 0, 0, 0, 0, 0, . . .], [81, 105, 450,0 ,0 ,0 ,0 , 0, . . .], [188, 64, 206, 788, 0, 0, 0, 0, 0, . . .]]	padding each sequence as 25 length followed by extra 0's

3.2.2. Data preprocessing on attention based mechanism

The difference in our attention-based neural network data preprocessing [39] is dividing each sentence as sequence followed by sentence piece tokenization. This sequence is encoded as a numerical vector representation. We find the maximum length (maxSentLen) of each raw text sentence piece tokenization [40] for specifying out our 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 [41]. Finally, the output of tensor data shape is three dimensional $[i_{conpusLength}, j_{maxSentLen}, k_{maxSeqLen}]$, where indexes i, j, k denote respectively as row, column, height. Here Table 2 shows the whole process for attention-based mechanism data preprocessing in training on neural network approach.

Table 3. Classification of token in transformer neural network.

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

Table 4. Demonstration on transformer based neural network data preprocessing.

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

3.2.3. Data preprocessing on transformer neural network(BERT) based mechanism

For preprocessing on Transformer neural network BERT, there are some special tokens in pre-trained language model. In our experiment, we used Bangla Bert base form huggingface library [42], [43], a PyTorch version to preprocess our text for learning in transformer encoder network. The special tokens are showed in Table 3. [CLS] tokens are at the beginning of the sentence, [SEP] tokens are at the ending of the sentence and [PAD] tokens are to pad and truncate the sentence in maximum length of sentence in the corpus. First, we tokenize the sentence text using the transformer package BertTokenizer [44]. We use the encode_plus [45] function to generate token_ids, then convert_ids _to_token and attention mask. The attention mask is used to identify which tokens are used (represent as 1) or not (represent as 0's). Finally, the input matrix is encoded as ['input_ids', 'attention_mask']. The whole demonstration is shown at Table 4.

3.3. Word embedding

In Neural network, word embedding is a measurement of language modeling and feature learning which maps the textual word into low dimensionality dense vectors. As a word embedding system, Word2Vec [46] research by Google, is computationally efficient and practicable in 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) [46] and Skip-Gram (SG) [47]. The procedure of CBoW model is to portend the current or target word from the neighboring co-occurrence word, whereas

SG model portends the entire context word from the target word. The basic difference on these two models is that, in each target–context pair, a newly annotation is considered in 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. We build three dimensional (d) vector space [128d, 200d, 300d] with window size 5 (window = 5) that 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.

4. Experiments

Our main aim of rule-based deep learning sentiment extraction experiments is to analyze the effectiveness of the well-known deep learning models. We used Tensorflow == 2.4.1, Keras == 2.4.3 and Transformer == 3.0 for developing our deep learning model. We split the training, testing, validation data set of 80.96% (2412 reviews), 10.03% (298 reviews) and 9.02% (269 reviews) of total reviews, respectively. Here, the training data set is used to train the model, whereas validation set checks to tune the hyper-parameters (i.e. learning rate, batch size, kernel regularization) of the model. Finally, the trained model will be evaluated with the test set. We trained a model on different hyper parameter settings such as embedding dimension, dropout rate, kernel and filter size, batch size, learning rate (lr), and epoch number. We iterate our model on this hyperparameter until we find its optimum value for training, which avoids overfitting on the dataset. In our experiment, we set the number of epoch to 50 and batch size to 256. Except for transformer learning training mechanism, we use a batch size of 16.

4.1. 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 (followed by Table 1). After a series of convolutional stacked with multiple filters and pooling operation, 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 input length of the text, embedding size and a weight of embedded matrix with 128d. Then we apply a learning sequence on to our vocabulary by using a convolutional 1D layer with 300 filters, kernel(k) size of 5(k = 5) value and ReLU activation unit. Convolution layer can shift the window over the sentence and the weight matrix and to let the CNN learn the weights for applying in the neighboring words in tensor input data. For effectively operating in the learning rate, we use a SpatialDropout1D parameter with 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 CNN model is sequential, we add batch normalization layer for learning efficiently from previous output layers. Finally, we add a dense layer with Sigmoid activation function because we are doing a trinary-based classification.

4.2. Dynamic Convolutional Neural Network (DCNN)

DCNN uses the convolutional layer with dynamic k-max pooling layers for extracting a feature map over a sentence. 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 dynamic k max pooling layer in the network. In our experiment, we have used five k ($k = 5$) max pooling layer 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.

4.3. Multichannel Variable-Size Convolution Neural Network (MVCNN)

MVCNN is a similar concept from CNN and DCNN except it has variable size filter mechanism with different size of word embedding layers. In out experiment we have used two embedding matrix [128D, 200D] dimension. The two embedding layer 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 layer is iterated again according to first layer mechanism. Finally, this three layer (layer_1 and layer_2, layer_3) is concatenated and flatten output followed as same before DCNN and CNN.

4.4. Very Deep Convolutional Neural Networks(VDCNN)

Difference form DCNN we have used a three dimensional word embedding layer [128D, 200D, 300D] having with ZeroPadding1D(filter_size -1, filter_size -1) add on three convolution1D layer with filter iteration as 3, 4, 5 sizes and GlobalMaxPool1D except k-max pooling layer. After three iterations, we get three layers (Layer_1, Layer_2, Layer_3) concatenated. This flatten the merge layer with l2(0.01) regularization, dropout of 0.5 and finally attach a three dense neurons of fully connected output with sigmoid activation.

4.5. Recurrent Neural Network(RNN)

RNN is a feed forward neural network for sequence modeling and data in which the output depends on the previous state. It maintains new state information during iteration over the sequence of elements and feed back to the previous layer to capture the correlation between current and previous time step. In our experiment, we use a simple RNN with 32 layers. At some timestamp, the previous input is fetched from the previous hidden layer and feed back to the current input of the current hidden layer. A SpatialDropout with 0.4 is plugged on the previous RNN layer. After the RNN layer, we add a BatchNormalization, Dropout with 0.4 and GlobalMaxPool1D sequentially. Finally a three connected layer with sigmoid activation function is added to the Dense layer.

4.6. Long Short Term Neural Network(LSTM)

LSTM is designed for reducing the problem of vanishing gradient descent problem and to remember the data as a long term period in a left to right context manner. Unlike RNN, LSTM has also a recurrent structure of interacting layer called Input gate, Forget gate and output

gate. At some timestamp, input gate with tanh layer generates a vector of all possible value which is triggered by sigmoid activation function (σ) and produces a new cell state. Input gate decides what much of information need to update or ignore. The forget gate decides what part of the information needs to be removed from the previous cell state of the previous hidden layer. 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. Our LSTM model consists of 32 unit hidden layers and Unlike RNN, at some timestamp, the previous input layer is fetched along with the previous cell state from the previous hidden layer and feed back to the current input at the current hidden layer and produce a new cell state. After LSTM layer we add a BatchNormalization layer and the rest of this architecture is followed by our RNN layer.

4.7. Bidirectional Long Short Term Neural Network(Bi-LSTM)

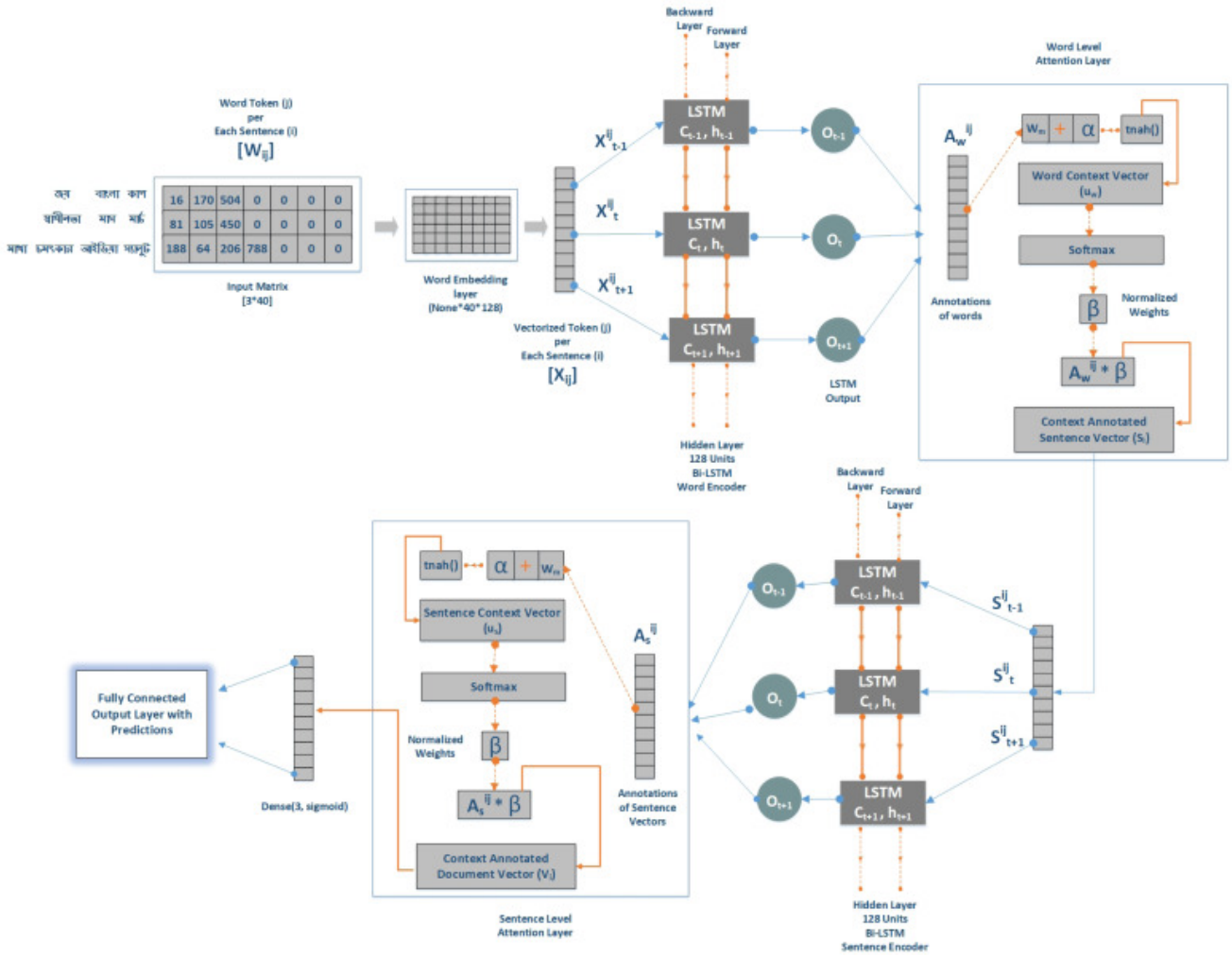
Bi-LSTM follows LSTM architecture, except it works on inputs in two ways one is left to right (capturing Forward Context), and the other is right to left (capturing backward context). It detects the feature from both past and the future context in sequential modeling. Same as our previous LSTM network, 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 in left to right (Forward) and right to left (Backward). At each timestamp, each hidden layer output is produced along with memory cell state and pass it to a convolutional1D layer of 64 filters of kernel_size of 4 and the rest of the network is followed by the previous LSTM network.

4.8. Asymmetric Convolutional Bidirectional LSTM(AC_Bi-LSTM)

AC_Bi-LSTM layer is a hybrid model combination of CNN and LSTM approach. In our sentiment classification, we applied that hybrid model. In our experiment, we use 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 of having different kernel sizes on the same filter and activation layer unit. Then these two different layers of conv1d are merged, and the input (x_t, x_{t+1}, x_{t+2} . . ., x_{t+n}) is passed to the LSTM layer of 32 units and the rest is followed by the previous LSTM network.

4.9. 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 applied four conv1D having with filter size of 100, kernel size of 2 and activated with tanh layer and each conv1D layers connected with each LSTM layer. This layer block is sequentially connected with another block layer. A flatten with 50 neurons, ReLU activation layer is forwarded from Bi-LSTM block. Then, it is finally attached with three dense neurons of fully connected output with a sigmoid activation function.



Download : Download high-res image (780KB) Download : Download full-size image

Fig. 2. Hierarchical Attention Based Neural Network (HAN) Architecture for Sentiment Classification.

4.10. Gated Recurrent Unit (GRU)

Similar architecture of LSTM where GRU consists of only two gates: update and reset gate and having a memory of only one state, which is a hidden state without considering a separate cell like LSTM. From a series of sequential inputs, update gate helps to learn long-term dependencies and to determine what amount of information from the previous hidden state needs to forward. Whereas reset gate is supervised to learn short-term dependencies and to generate how much information needs to forget. Same as the LSTM network, we use a GRU layer of 32 units. After GRU memory, we add a convolutional1D layer with 65 filter size, kernel_size of 5 and having a golort_uniform regularization of kernel_initializer. Then sequentially add a GlobalAveragePooling1D and GlobalMaxPooling1D layer. Finally, these are concatenated with three neurons of the dense sigmoid layer activation function.

4.11. 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 layer without having to use a cell memory unit. In terms of simpler architecture form LSTM, Both works on resolving vanishing gradient descent problem; however, GRU capture and remember longer range of correlation and train faster more efficiently than LSTM [48]. 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 network has two states to resolve both contextual relations in left to right (Forward) and right to left (Backward) except maintaining no cell state mechanism. At time stamp, each hidden layer output is produced and pass it 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.

4.12. 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 or attend over 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.

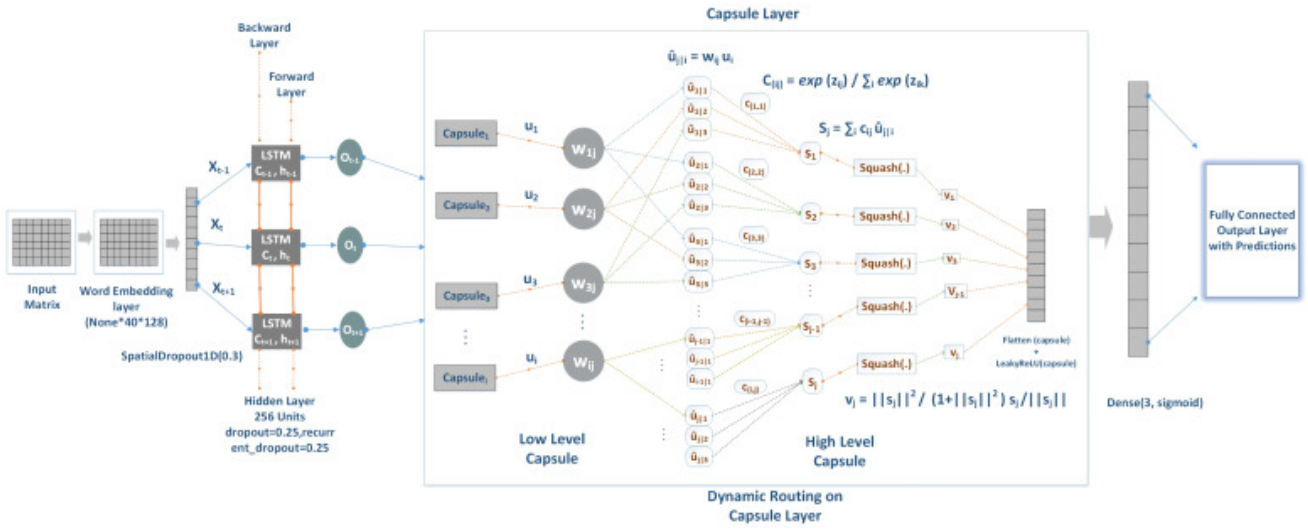
4.12.1. Hierarchical attention based neural network

The previous model in this methodology works on only sentence-level encoding; however, HAN work 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 in order to capture compositional features, hierarchical dependencies from a sequence of input as well as it contributes to the polarity of a text. A document is spitted by a number of sentences and each sentence word are tokenized to transform into a vector, and then these vectors are used as an input matrix in the neural network. Authors [12] proposed a hierarchical attention-based structure for word and sentence encoder. 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 by 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 layer, and sentence attention layer summarized the alignment of the sentence by extracting the relevant context of each sentence which classifies the document. The preprocessing of our text encoding sequence is followed by Table 2. The context can be achieved by a bidirectional RNN model. We use a Bi-LSTM in our HAN mechanism, shown in Fig. 2.

In Fig. 2, we demonstrated our HAN mechanism on bidirectional LSTM network. From input matrix, each word tokens (i) from each sentence (j) are placed on the word embedding layer noted as (W_{ij}) om 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 unit as a word

encoder layer. At time step t, it the input X_{t-1}^{ij} from previous hidden state (h_{t-1}) having 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 projected the output from Bi-LSTM word encoding layer on in its annotations word matrix (A_w^{ij}) as a continuous vector space which makes the base for attention mechanism. This one hidden layer operates as a Multilayer perception to do the model learn through a randomly initialized weights (W_m) and adding with biases (α) and put 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 (β). Then finally the normalized context vector with weights (β) is concatenated with the previously calculated context annotation matrix (A_w^{ij}) which produce the sentence vector (s_i).

After getting the sentence vector (s_i), a similar mechanism is followed for 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 forward it for calculating an improved context annotated document vector (v_i). Finally, these are concatenated with three neurons of dense sigmoid layer activation function.



Download : Download high-res image (568KB) Download : Download full-size image

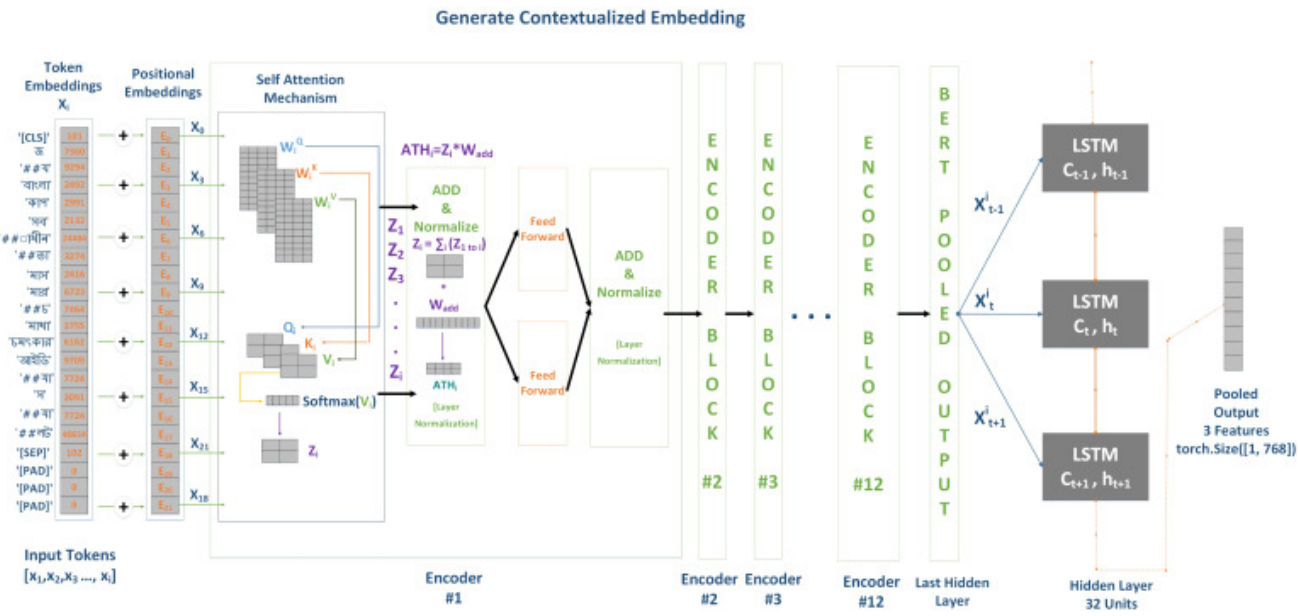
Fig. 3. Dynamic Routing Based Capsule Neural Network (D-CapsNet-Bi-LSTM) Architecture for Sentiment Classification.

4.13. Capsule neural network(CapsNet)

Capsule neural network is a group of neural network which solve the invariant of 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 matrix dimension, intercepts the various features, is captured by the pooling operation while losing a lot of 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 a number of 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 which also predict the lower and upper-level layer capsule. In our sentiment classification, we use 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 Fig. 1.

LSTM forwards its output into the each primary layer capsule (Capsule_i), 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 (receive form Bi-LSTM layer) into a vector output (u_1, u_2, u_3, …u_i) to be the input of the next capsule layer. This vector has two core elements: length and direction. The lower level capsule identifies the corresponding feature text probabilities by using this length, and 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, …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 words which is the higher level features. Here, $\hat{u}_{j\_i}$ indicates as a prediction vectors that what i^{th} features should predict the correct position for the j^{th} sentence. That means if all 16 capsule detects the same features is the lower-level capsule then 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 ($c_{(i,j)}$) and formed as next (higher) capsule level (s_j), which is determined the number of routing iteration process. The dot product differentiates the lower-level capsule i and higher level capsule j, although i capsule see its output in the j capsule. In our D-CapsNet-Bi-LSTM network, we set the number of routing iteration is 3. This coupling coefficient is calculated by a routing softmax function where the exponential coefficient $\exp(z_{ij})$ indicates some prior probabilities that which capsule layer i will be coupled to capsule layer j. 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 on Fig. 3



Download : Download high-res image (667KB) Download : Download full-size image

Fig. 4. Bidirectional Encoder Representation from Transformer with LSTM Neural Network (BERT-LSTM) Architecture for Sentiment Classification.

4.14. Bidirectional encoder representation from transformer(Bert)

Transformer belongs to an encoder–decoder architecture model having attention mechanisms [49] that are used for transfer learning in the field of machine translation as well as in NLP task and also leverage with long term dependencies finer than as a replacement of other conventional sequential model i.e., RNN, LSTM, GRU etc. In transfer learning, a model is trained on huge unlabeled content corpora utilizing self-supervised learning, and this pretrained model is negligibly balanced during fine tuning on a particular NLP task [50]. It is also more potential in training the model by removing the sequential dependencies of the past tokens. BERT recently developed by Google [20], an encoder based transformer architecture for language modeling that is used for dynamic embedding in NLP task which consider the both current and previous token on both left and right in a bidirectional way. As a contextual model, Bert generates a representation of each word which is based on every other sentence; however, in static embedding i.e., Word2Vec model generates a single word embedding representation for each word in vocabulary.

4.14.1. BERT-LSTM architecture for sentiment classification

In our sentiment classification, we use a BERT-BASE model with 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 token along with special tokens showed in Table 3. As, LSTM reads text input sequentially, whereas BERT takes the entire tokenization of words at once. In our experiment, we build as sentiment classifier by using huggingface library to

fine-tune with pretrained BERT model [42] on the upper layer of LSTM, showed in Fig. 4. From this library, we install the transformer version as 3.0 and load the BERT classifier and tokenizer for input processing.

The input sequences is a raw sentences which is split by the BERT tokenizer and that tokens converted into token_ids and with attention_mask showed in Table 4. BERT uses a self-attention mechanism over the input sequences, showed in Fig. 4 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, …, x_i). This sequence (x_i) is multiplied with three weight matrices (W_i^Q, W_i^K, W_i^V) in scalar dot matrix way to generate three vectors termed as respectively as query (Q_i), keys (K_i) and values (V_i). These weight matrices (W_i^Q, W_i^K, W_i^V) are produced during the training process on BERT. The main mechanism for self-attention is that it calculate each word score from input sequences and this score indicates that 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, …, 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, …, V_i) to produce another matrices (Z_i). In ADD and Normalize layer step [51] which is then multiply with the additional weight matrices (W_{add}) to produce attention head (ATH_i) which capture all the information from all attention heads, then it is forwarded in the feed-forward layer. Similar other encoder will follow this mechanism [52] 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.

4.15. Experimental model training and fitting

After construction of those models, we compile each model with our word embedding matrix [128D, 200D, 300D] as per requirement by setting the parameter loss function as categorical_crossentropy and optimizer as adam at a learning rate on different point shown in Table 5. Table 5 shows the summarization of our different parameters on each model. After compiling the model, we fit the model with our training and validation data having batch size of 256 with 50 epochs except using batch size of 16 with 50 epoch in BERT-LSTM model.

Fig. 5 shows 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 $\boldsymbol{X}$-axis indicates to the epoch number, while the $\boldsymbol{Y}$-axis indicates to the training, validation accuracy and loss. TA and VA determine whether the model was overfitting or not and TL and VL determines whether the model was overfitting or underfitting. The VA and VL for dataset how well the model will perform for the unseen future data.

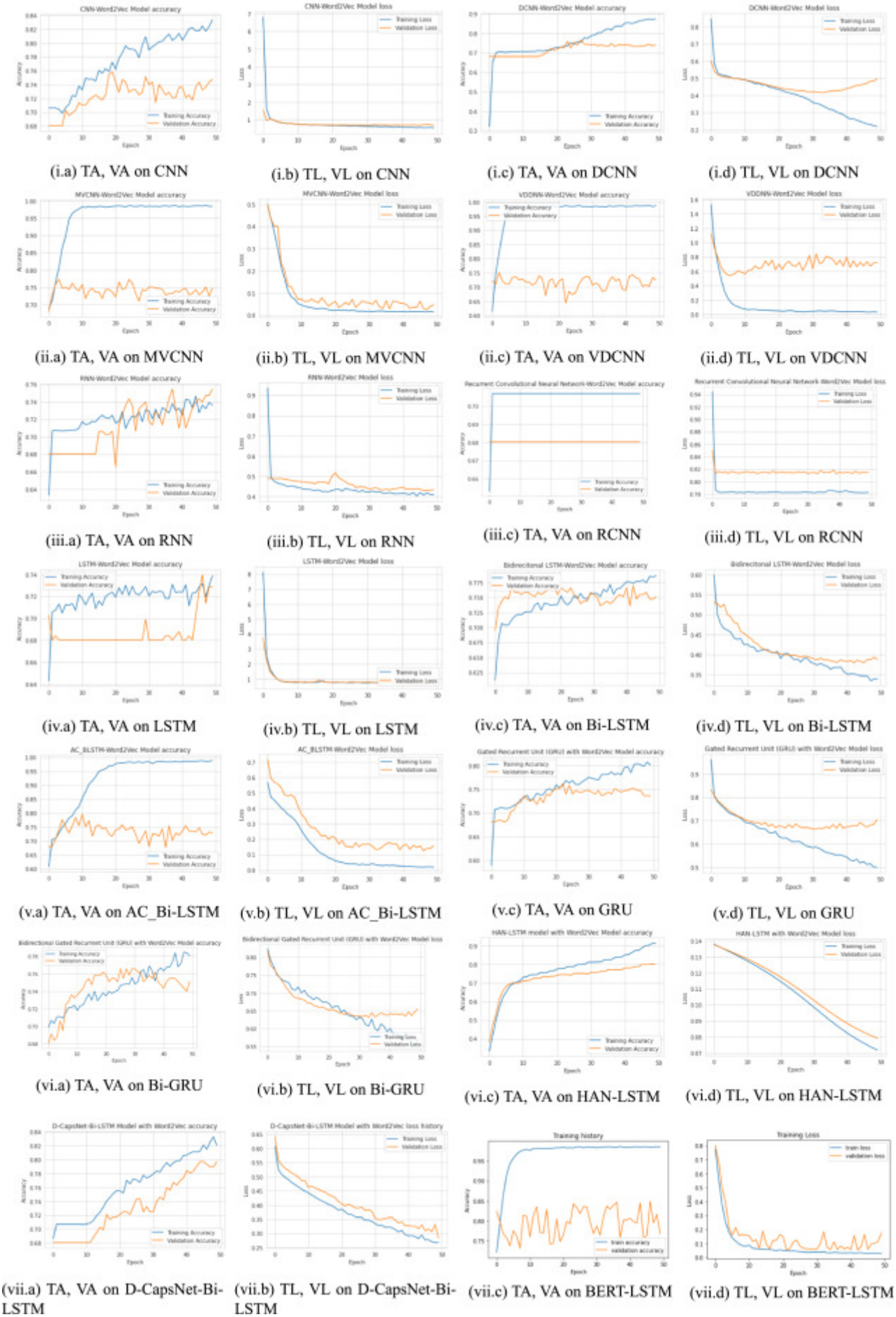
Table 5. Hyperparameter dependency and complexity on each model.

Model Name	Learning Rate (lr)	Word Embedding Dimension	Layer		Dropout
CNN	(lr = 0.01)	128D	Conv1D(filter = 300, Kernel_size = 5, ReLU), GlobalMaxpool1D		SpatialDropout = 0.5, Dropout 0.5
DCNN	(lr = 0.0001)	128D	ZeroPadding1D(49,49)	Conv1D(64, Kernel_size = 50),	SpatialDropout = 0.3, Dropout =
			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		SpatialDropout = 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)		SpatialDropout = 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)		SpatialDropout = 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),	

Model Name	Learning Rate (lr)	Word Embedding Dimension	Layer		Dropout
				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) GlobalAverageMaxpooling1D, GlobalMaxpool1D	SpatialDropout = 0.3, Dropout =
Bi-GRU	(lr = 1e-3)	128D	Conv1D(64, Kernel_size = 4) GlobalAverage Maxpooling1D, GlobalMaxpool1D	1 Block Bi-GRU (unit=32, dropout = 0.2, recurrent_dropout = 0.1)	SpatialDropout = 0.3, Dropout =
HAN-LSTM	(lr = 0.00001)	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)		SpatialDropout = 0.3, Dropout =
			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

5. Results & analysis

We conducted a total of fourteen experiments to offer a reasonable comparison of recent DL algorithms and traditional methods. Table 6 shows the experimental results of each individual 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 outperformed on different learning rate (lr) parameters to achieve outstanding results. From these results, researchers 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

(vii.a) TA, VA on D-CapsNet-Bi-LSTM (vii.b) TL, VL on D-CapsNet-Bi-LSTM (vii.c) TA, VA on BERT-LSTM (vii.d) TL, VL on BERT-LSTM

Download : Download high-res image (2MB) Download : Download full-size image

Fig. 5. Showing each Model Training Accuracy (TA), Validation Accuracy (VA) with respect to Epoch and Model Training Loss (TL), Validation Loss (VL) with respect to epoch.

5.1. CNN based model

Through the comparison between our convolutional neural network types models (CNN, DCNN, MVCNN, VDCNN), it is noticed that VDCNN achieves the highest 77.85% accuracy, 80.16% precision, 79.56% recall, and 79.86% F1-score. VDCNN and MVCNN both are 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 CNN model.

It is shown in Fig. 5(i.a) that at about 20 epochs, CNN model achieves the highest testing accuracy; whereas as shown in Fig. 5(i.c), DCNN decreases testing accuracy from point 25 epochs. It is shown in Fig. 5(ii.a) and (ii.c) that MVCNN and VDCNN models have 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]. Here, MVCNN uses two dimensional [128D, 200D] and VDCNN uses three dimensional [128D, 200D, 300D] word embedding layers. We keep the dropout rate at 0.5 on each layer but we change kernel size when variable size of zero padding1D is added for performing spatial dimension to the output.

When we add a number of nodes in the layer in our model, the capacity increases that means accuracy, precision, and recall increase. As our training data is small, our model is pretty small; however, increasing model layers can drive to a more precise model. If more training data are given in the model, the larger the model should be. In our experiment, multilayer perception of CNN is applied in DCNN, VDCNN and MVCNN models; as there is no bound to use a specific number of layers so this stacked layer is susceptible to generalize our model better. By adding continuous layers of convolution and pooling operation in CNN, it might lose spatial information on classifying data.

Table 6. 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

MODEL NAME	Train accuracy	Test accuracy	Train precision	Test precision	Train recall	Test recall	Train F1-score	Test F1-score
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

5.2. RNN based model

Sequential models such as RNN and RCNN both achieved similar test accuracy of 73.83%; however, RNN achieved 78.88% precision which is greater than that of RCNN. Besides in preserving LSTM and Bi-LSTM, where Bi-LSTM model performs well on cricket dataset having optimal accuracy of 78.14%. However, LSTM model performs well with a testing precision of 78.53% that is greater than BiLSTM model.

At arbitrary time interval, the three gates of RNN remember the propagation of information into the cell. It is shown in Fig. 5(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 Fig. 5(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] but 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 Fig. 5(v.a) and has the highest testing recall of 95.58% and testing F1-score of 95.70%.

5.3. GRU based model

As GRU intends in the last hidden state to represent the sentence which means modeling the whole sentence causes neglecting main key parts of words, this might result in the incorrect prediction. From Table 6, while learning rate is so high in GRU and Bi-GRU models, training and testing accuracy, precision are not getting higher than LSTM and Bi-LSTM models. At most 75.84% accuracy is achieved on LSTM model with an average 71% precision, recall and

F1-score values. GRU limits the stream of data just like LSTM unit; however, except utilizing a memory unit in this case LSTM model performs well on this dataset. For this reason, we applied LSTM model as hybrid layer on attention, capsule and BERT based models.

In Fig. 5(v.c) and (vi.a), both GRU based models have similar cures with respect to testing accuracy of 75.84% and 75.17%. Bi-GRU model intersects on about at epoch point 5 and 35 that means it has both bi-directional dependency to co-relate words in text.

5.4. 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 is increasing with respect to epoch while training and testing loss is decreasing exponentially that shows an ideal state of the model. At most 78.52% and 80.12% testing accuracy and recall are produced in HAN-LSTM model and 80.82% and 81.00% testing accuracy, precision are produced in D-CAPSNET-Bi-LSTM.

The attention level in word and along with sentence increases the model accuracy with respect to other CNN, RNN type models. Calculation of attention vector which is co-related to word and sentence level that determines the less content for constructing the document representation. The main advantage of Capsule module is to resolve the max pooling level conversion for feature extractions that means the improvement of CNN and RNN type models. Because losing the information in polling layers might cause in lesgs accuracy. Again, augmenting the compositional capsule network with k-clustering mechanism will improve the classification accuracy of our HAN-LSTM model.

5.5. Transformer based model

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

Table 7. Comparison of major sentiment classifiers in both machine learning and deep learning with accuracy.

Paper	Context	Dataset	Methods	Accuracy	Our Approach Accuracy
[5]	Cricket ABSA Dataset	2979	Bi-Gram, SVM	82.21%	–

Paper	Context	Dataset	Methods	Accuracy	Our Approach Accuracy
[22]	Movie Reviews	4000	LSTM	82.42% for 2 category	74.16% for 3 category
[23]	Social media, news, product reviews	9337	RNN (LSTM)	78% for 2 category, 55% for 3 category	74.16% for 3 category
[26]	Facebook	10000	Bi-LSTM	85.67%	78.14% for 3 category
[32]	Cricket ABSA Dataset	2979	A-LSTM	66.06%	78.52% HAN-LSTM Model
			A-CNN	72.06%	
[30]	Food Panda Website	1000	CNN-LSTM	75.01%	75.50% AC_Bi-LSTM Model
[24]	Social Media	8910	LSTM	65.97%	73.49% CNN, 74.16% LSTM, 76.51% MVCNN, 77.85% VDCNN, 80.82% D-CAPSNET-Bi-LSTM
			CNN	60.89%	
			NB	60.79%	
			SVM	59.18%	
[53]	Microblogging sites	3000	Word2Vec with Hellinger PCA	70.00%	
[54]	Social Media	1000	LSTM	46.80%	
[31]	Newspaper Comments	1002	BERT-GRU	71.00% for 2 class	84.18% BERT-LSTM

5.6. Comparison with existing approaches

Next, we compare our work with other existing deep learning and machine learning methods. Table 7 compares our work with others in terms of accuracy. In [23], authors achieved 55% accuracy with three category sentiments on above nine thousand social comments and in [22], authors showed 82.42% accuracy on four thousand movie review dataset in two category sentiments in LSTM network, whereas our LSTM model achieves 74.16% accuracy. Similarly in [24], [54] LSTM network achieved 65.97% and 46.80% accuracy, respectively, in [32] attention-based LSTM (A-LSTM) achieved 65.97% accuracy; however, our hierarchical attention-based LSTM (HAN-LSTM) achieved 78.52% accuracy and in [30] combined CNN-LSTM achieved 75.01% accuracy where our AC_Bi-LSTM and D-CAPSNET-Bi-LSTM hybrid model achieves greater accuracy than those papers. There are drawbacks in preprocessing data on pronoun type word replacing; however, they still manage to achieve 85.67% accuracy in Bi-LSTM model on ten thousand comments where our Bi-LSTM gained 78.14% accuracy on

cricket ABSA dataset. Our CNN type model with multi-channel i.e., DCNN, VDCNN, MVCNN model achieves higher accuracy than [24], [32] type CNN model.

A recent supervised ML-based approach [5] on SA with rule-based achieved satisfactory accuracy on SVM classifier with 82.21% accuracy. However, the long term dependencies between words are not taken to consider on bi-gram approach. Traditional machine learning based bag of words (BOW) approach does not capture semantic relation between words where hidden layers in neural network boost the model with the help of contextual relationship between words which is retained by the word embedding. ML classifiers i.e., SVM, Naïve Base etc., do not perform well on unstructured noisy datasets, for example, target categories are overlapping with each other. Both approaches require a lot of large labeled corpus for training in order to deliberate best prediction. However, the semantic understanding between the sequences of data is preserved in DL approach. Although DL based model requires more initial tuning parameters and each model provides different results, one does not require to consider about feature engineering. The core advantage in DL is that it learns high level features from data in an exponential perspective through multi hidden layer with hyper tuned parameters and provides better results than others.

5.7. Complexity factors of DL model

Deep Learning model complexity implies to investigate the generalization capability and limitation of neural networks on the training process. Our hybrid DL models are hyper-parameterized but it has very little effect on model complexity [55]. Model complexity indicates how the neural network model expresses its behavior on distribution function or activation function [56], prevents overfitting on by adding L1, L2 regularization to the loss function [57] and the amplification coefficient (w_{ij}) which is defined by the multilayer perception of hidden neurons. Selecting an activation function i.e., ReLU, Sigmoid, tanh in network hidden layers which is an active module for learning and computing complex task by taking piecewise nonlinear transformation to the input. Pooling function such GlobalAverage Maxpooling1D, GlobalMaxpool1D with the use of variation of filter size on feature maps is needed to reduce fixed size vectors. However, the size of the DL model impacts on model complexity i.e., number of filters, kernel size, hidden neurons, dropout rate, depth and width efficiency of model, training and testing time also. This is partially noted in Table 5. In our work, we have limited our experiment in detecting accuracy of SA in Bangla text with the help of LDD and BTSC. In future, we will evaluate the complexity of our deep learning algorithms.

6. Conclusion

In this paper, we investigated the most noteworthy work on sentiment analysis on cricket reviews as a low resourced Bangla dataset using various deep learning architectures. This empirical study is an initial dive on lexicon dictionary-based approach on neural network mechanism. We measured 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 basic learning 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 a customized and fine-tuned with hyperparameters on individual models. We found that LSTM had given better results over CNN and RNN type models. Then we used attention, capsule, transformer-based mechanism by adding LSTM layer, and the results showed significant improvement, which is indeed effective in the sentiment classification effect. By using attention, capsule mechanism, we proposed some hybrid DL models, termed as HAN-LSTM and D-CAPSNET-Bi-LSTM along with semantic learning representations (word2vec) having excellent performance in terms of accuracy, precision, recall and F1-score. Finally, a hybrid model termed as BERT-LSTM was used for the classification task, and it surpasses others in terms of accuracy and precision.

This LDD dataset will be useful for future research. Bangla cricket review dataset is relatively small with respect to benchmark dataset and there is lack of enough large training corpus in Bangla domain, this is the main drawbacks in sentiment analysis tasks in Bangla. We have identified trinary polarity in cricket reviews since this dataset is not properly balanced data; so, using balanced data for each polarity in training process might enhance accuracy of prediction results. Because increasing the amount of training data can assist to promote 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 and showing individual model with confusion matrix. Despite the fact that we gained satisfactory result, it still has a scope for further improvement. In the future, we will conduct our research using memory augmented network. This will be done by combining external memory of neural network and other transformer models i.e., ALBERT, ELECTRA, RoBERT, DistilBERT along with multilayer, and also hybrid capsule and multi headed attention layer-based models with other word embedding learning representation mechanism such as Glove, fastText, etc. We will design a graph neural network to capture internal and external graph structures of NLP. Finally, this research work will contribute in the improvement of emotional sentiment analysis on Bangla domain corpus.

CRediT authorship contribution statement

Nitish Ranjan Bhowmik: Conception and design of study, Acquisition of data, Analysis and/or interpretation of data, Writing – original draft, Writing – review & editing. **Mohammad Arifuzzaman:** Conception and design of study, Writing – review & editing. **M. Rubaiyat Hossain Mondal:** Conception and design of study, Writing – review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

All authors approved the version of the manuscript to be published.

References

[1] Liu Bing
Sentiment analysis and opinion mining
Synth Lect Hum Lang Technol, 5 (1) (2012), pp. 1-167
CrossRef View Record in Scopus Google Scholar

[2] Rosenfeld Richard, Fornango Robert
The impact of economic conditions on robbery and property crime: The role of consumer sentiment
Criminology, 45 (4) (2007), pp. 735-769
View Record in Scopus Google Scholar

[3] Bernert Rebecca A, Hilberg Amanda M, Melia Ruth, Kim Jane Paik, Shah Nigam H, Abnousi Freddy
Artificial intelligence and suicide prevention: a systematic review of machine learning investigations
Int J Environ Res Public Health, 17 (16) (2020), p. 5929
CrossRef Google Scholar

[4] Collobert Ronan, Weston Jason, Bottou Léon, Karlen Michael, Kavukcuoglu Koray, Kuksa Pavel
Natural language processing (almost) from scratch
J Mach Learn Res, 12 (ARTICLE) (2011), pp. 2493-2537
View Record in Scopus Google Scholar

[5] Bhowmik Nitish Ranjan, Arifuzzaman Mohammad, Mondal M Rubaiyat Hossain, Islam MS
Bangla text sentiment analysis using supervised machine learning with extended lexicon dictionary
Nat Lang Process Res, 1 (3–4) (2021), pp. 34-45
CrossRef View Record in Scopus Google Scholar

[6] Pang Bo, Lee Lillian
A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts
(2004)
arXiv preprint Cs/0409058
Google Scholar

[7] Bengio Yoshua, Ducharme Réjean, Vincent Pascal, Janvin Christian

A neural probabilistic language model
J Mach Learn Res, 3 (2003), pp. 1137-1155
View Record in Scopus Google Scholar

[8] Kalchbrenner Nal, Grefenstette Edward, Blunsom Phil
A convolutional neural network for modelling sentences
(2014)
arXiv preprint arXiv:1404.2188
Google Scholar

[9] Conneau Alexis, Schwenk Holger, Barrault Loïc, Lecun Yann
Very deep convolutional networks for text classification
(2016)
arXiv preprint arXiv:1606.01781
Google Scholar

[10] Yin Wenpeng, Schütze Hinrich
Multichannel variable-size convolution for sentence classification
(2016)
arXiv preprint arXiv:1603.04513
Google Scholar

[11] Mikolov Tomáš, Karafiát Martin, Burget Lukáš, Černockỳ Jan, Khudanpur Sanjeev. Recurrent neural network based language model. In: Eleventh annual conference of the international speech communication association. 2010.
Google Scholar

[12] Xiao Lizhong, Wang Guangzhong, Zuo Yang
Research on patent text classification based on word2vec and LSTM
2018 11th International symposium on computational intelligence and design, Vol. 1, IEEE (2018), pp. 71-74
CrossRef View Record in Scopus Google Scholar

[13] Zhou Peng, Qi Zhenyu, Zheng Suncong, Xu Jiaming, Bao Hongyun, Xu Bo
Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling
(2016)
arXiv preprint arXiv:1611.06639
Google Scholar

[14] Liang Depeng, Zhang Yongdong
AC-BLSTM: asymmetric convolutional bidirectional LSTM networks for text classification
(2016)
arXiv preprint arXiv:1611.01884

Google Scholar

[15] Liu Pengfei, Qiu Xipeng, Huang Xuanjing
Recurrent neural network for text classification with multi-task learning
(2016)
arXiv preprint arXiv:1605.05101
Google Scholar

[16] Rana Rajib
Gated recurrent unit (GRU) for emotion classification from noisy speech
(2016)
arXiv preprint arXiv:1612.07778
Google Scholar

[17] Yang Zichao, Yang Diyi, Dyer Chris, He Xiaodong, Smola Alex, Hovy Eduard. 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. p. 1480–9.
Google Scholar

[18] Tian Zhengxi, Rong Wenge, Shi Libin, Liu Jingshuang, Xiong Zhang
Attention aware bidirectional gated recurrent unit based framework for sentiment analysis
International conference on knowledge science, engineering and management, Springer (2018), pp. 67-78
CrossRef View Record in Scopus Google Scholar

[19] Chen Yuxiao, Yuan Jianbo, You Quanzeng, Luo Jiebo. Twitter sentiment analysis via bi-sense emoji embedding and attention-based LSTM. In: Proceedings of the 26th ACM international conference on multimedia. 2018. p. 117–25.
Google Scholar

[20] Devlin Jacob, Chang Ming-Wei, Lee Kenton, Toutanova Kristina
Bert: Pre-training of deep bidirectional transformers for language understanding
(2018)
arXiv preprint arXiv:1810.04805
Google Scholar

[21] Gao Zhengjie, Feng Ao, Song Xinyu, Wu Xi
Target-dependent sentiment classification with BERT
IEEE Access, 7 (2019), pp. 154290-154299
CrossRef View Record in Scopus Google Scholar

[22] Chowdhury Rumman Rashid, Hossain Mohammad Shahadat, Hossain Sazzad, Andersson Karl

Analyzing sentiment of movie reviews in bangla by applying machine learning techniques
2019 International conference on bangla speech and language processing, IEEE (2019), pp. 1-6
CrossRef Google Scholar

[23] Hassan Asif, Amin Mohammad Rashedul, Al Azad Abul Kalam, Mohammed Nabeel
Sentiment analysis on bangla and romanized bangla text using deep recurrent models
2016 International workshop on computational intelligence, IEEE (2016), pp. 51-56
CrossRef View Record in Scopus Google Scholar

[24] Tripto Nafis Irtiza, Ali Mohammed Eunus
Detecting multilabel sentiment and emotions from bangla youtube comments
2018 International conference on bangla speech and language processing, IEEE (2018), pp. 1-6
View Record in Scopus Google Scholar

[25] Alam Md Habibul, Rahoman Md-Mizanur, Azad Md Abul Kalam
Sentiment analysis for bangla sentences using convolutional neural network
2017 20th International conference of computer and information technology, IEEE (2017), pp. 1-6
CrossRef Google Scholar

[26] Sharfuddin Abdullah Aziz, Tihami Md Nafis, Islam Md Saiful
A deep recurrent neural network with bilstm model for sentiment classification
2018 International conference on bangla speech and language processing, IEEE (2018), pp. 1-4
View Record in Scopus Google Scholar

[27] Al-Amin Md, Islam Md Saiful, Uzzal Shapan Das
Sentiment analysis of bengali comments with Word2Vec and sentiment information of words
2017 International conference on electrical, computer and communication engineering, IEEE (2017), pp. 186-190
CrossRef Google Scholar

[28] Hassan Asif, Amin Mohammad Rashedul, Mohammed N, Azad AKA
Sentiment analysis on bangla and romanized bangla text (BRBT) using deep recurrent models
(2016)
arXiv preprint arXiv:1610.00369
Google Scholar

[29] Wahid Md Ferdous, Hasan Md Jahid, Alom Md Shahin

Cricket sentiment analysis from bangla text using recurrent neural network with long short term memory model
2019 International conference on bangla speech and language processing, IEEE (2019), pp. 1-4
View Record in Scopus Google Scholar

[30] Hossain Naimul, Bhuiyan Md Rafiuzzaman, Tumpa Zerin Nasrin, Hossain Syed Akhter
Sentiment analysis of restaurant reviews using combined CNN-LSTM
2020 11th International conference on computing, communication and networking technologies, IEEE (2020), pp. 1-5
CrossRef Google Scholar

[31] Islam Khondoker Ittehadul, Islam Md Saiful, Amin Md Ruhul
Sentiment analysis in bengali via transfer learning using multi-lingual BERT
2020 23rd International conference on computer and information technology, IEEE (2020), pp. 1-5
CrossRef Google Scholar

[32] Sharmin Sadia, Chakma Danial
Attention-based convolutional neural network for bangla sentiment analysis
AI Soc, 36 (1) (2021), pp. 381-396
CrossRef View Record in Scopus Google Scholar

[33] Tang Duyu, Wei Furu, Qin Bing, Zhou Ming, Liu Ting. 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. p. 172–82.
Google Scholar

[34] Taboada Maite, Brooke Julian, Tofiloski Milan, Voll Kimberly, Stede Manfred
Lexicon-based methods for sentiment analysis
Comput Linguist, 37 (2011), pp. 267-307, 10.1162/COLI_a_00049
View Record in Scopus Google Scholar

[35] Reckman Hilke, Baird Cheyanne, Crawford Jean, Crowell Richard, Micciulla Linnea, Sethi Saratendu 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. p. 513–9.
Google Scholar

[36] Wei Jason, Zou Kai
Eda: Easy data augmentation techniques for boosting performance on text classification tasks
(2019)

arXiv preprint arXiv:1901.11196
Google Scholar

[37] Rahman Md, Kumar Dey Emon, *et al.*
Datasets for aspect-based sentiment analysis in bangla and its baseline evaluation
Data, 3 (2) (2018), p. 15
CrossRef View Record in Scopus Google Scholar

[38] AtikRahman
Bangla ABSA datasets for sentiment analysis
(2021)
https://github.com/AtikRahman/Bangla_ABSA_Datasets. (Accessed 31 July 2021)
Google Scholar

[39] TensorFlow API Developer
Keras preprocessing library, module:tf.keras.preprocessing
(2021)
https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/. (Accessed 31 July 2021)
Google Scholar

[40] TensorFlow API Developer
Keras preprocessing library, module:tf.keras.preprocessing.text.tokenizer
(2021)
https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer. (Accessed 31 July 2021)
Google Scholar

[41] TensorFlow API Developer
Keras preprocessing library, module:tf.keras.preprocessing.sequence.pad_sequences
(2021)
https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/sequence/pad_sequences. (Accessed 31 July 2021)
Google Scholar

[42] Sarker Sagor
Banglabert: Bengali mask language model for bengali language understanding
(2021)
https://github.com/sagorbrur/bangla-bert. (Accessed 31 July 2021)
Google Scholar

[43] Wolf Thomas, Debut Lysandre, Sanh Victor, Chaumond Julien, Delangue Clement, Moi Anthony, *et al.*
Huggingface's transformers: State-of-the-art natural language processing
(2019)

arXiv preprint arXiv:1910.03771
Google Scholar

[44] Developer Huggingface Library
Berttokenizer
(2021)
https://huggingface.co/transformers/model_doc/bert.html#berttokenizer. (Accessed 31 July 2021)
Google Scholar

[45] Developer Huggingface Library
Bert PreTrained tokenizer base encode_plus function
(2021)
https://huggingface.co/transformers/internal/tokenization_utils.html#transformers.tokenization_utils_base.PreTrainedTokenizerBase.encode_plus. (Accessed 31 July 2021)
Google Scholar

[46] Mikolov Tomas, Chen Kai, Corrado Greg, Dean Jeffrey
Efficient estimation of word representations in vector space
(2013)
arXiv preprint arXiv:1301.3781
Google Scholar

[47] Mikolov Tomas, Sutskever Ilya, Chen Kai, Corrado Greg S, Dean Jeff
Distributed representations of words and phrases and their compositionality
Advances in neural information processing systems (2013), pp. 3111-3119
View Record in Scopus Google Scholar

[48] Chung Junyoung, Gulcehre Caglar, Cho KyungHyun, Bengio Yoshua
Empirical evaluation of gated recurrent neural networks on sequence modeling
(2014)
arXiv preprint arXiv:1412.3555
Google Scholar

[49] Vaswani Ashish, Shazeer Noam, Parmar Niki, Uszkoreit Jakob, Jones Llion, Gomez Aidan N, *et al.*
Attention is all you need
Advances in neural information processing systems (2017), pp. 5998-6008
View Record in Scopus Google Scholar

[50] Raffel Colin, Shazeer Noam, Roberts Adam, Lee Katherine, Narang Sharan, Matena Michael, *et al.*
Exploring the limits of transfer learning with a unified text-to-text transformer
(2019)
arXiv preprint arXiv:1910.10683

Google Scholar

[51] Ba Jimmy Lei, Kiros Jamie Ryan, Hinton Geoffrey E.
Layer normalization
(2016)
arXiv preprint arXiv:1607.06450
Google Scholar

[52] Uszkoreit Jakob. Transformer: A novel neural network architecture for language understanding. Google AI blog. Vol. 31.
Google Scholar

[53] Islam Md Saiful, Al-Amin Md, Uzzal Shapan Das
Word embedding with hellinger pca to detect the sentiment of bengali text
2016 19th International conference on computer and information technology, IEEE (2016), pp. 363-366
View Record in Scopus Google Scholar

[54] Sarkar Kamal. Sentiment polarity detection in bengali tweets using LSTM recurrent neural networks. In: 2019 Second international conference on advanced computational and communication paradigms. 2019. p. 1–6.
Google Scholar

[55] Kakade Sham M., Sridharan Karthik, Tewari Ambuj
On the complexity of linear prediction: Risk bounds, margin bounds, and regularization
(2008)
Google Scholar

[56] Bengio Yoshua, Delalleau Olivier
On the expressive power of deep architectures
International conference on algorithmic learning theory, Springer (2011), pp. 18-36
CrossRef View Record in Scopus Google Scholar

[57] Goodfellow Ian, Bengio Yoshua, Courville Aaron
Deep learning
MIT Press (2016)
Google Scholar

Nitish Ranjan Bhowmik was born in Bangladesh. He received the B.Sc.(Honors) degree in Computer Science & Engineering (CSE) from Dhaka University (DU), at Dhaka, Bangladesh. He is currently pursuing the master's degree in Information and Communication Technology(ICT) with the Bangladesh University of Engineering and Technology (BUET). His research interests include data mining, text mining, natural language processing, machine learning and deep learning.

Dr. Mohammad Arifuzzaman received the B.Sc. degree in Computer Science & Engineering from BUET (Bangladesh University of Engineering and Technology). He received Masters in Information and Telecommunication Studies, from Waseda University, Tokyo, Japan in 2012 and Ph.D. in Information and Telecommunication from the Waseda University, Tokyo, Japan in 2015. He worked as a postdoctoral fellow at Memorial University of Newfoundland, Canada. His research interest is in AI and Data Science, Future Internet Architecture, Green Wireless and Sensor Communication. Currently, he is an Associate Professor at Department of ECE, East West University, Dhaka, Bangladesh. Before that, he worked as a faculty member of IICT, BUET. Dr. Arif published 60+ research papers in International Journals (including *IEEE Sensors Journal, IEEE Access Journal, IEEE Transactions on Instrumentation & Measurement, Orphanet journal of rare diseases etc*) and conferences (including IEEE Globecom, ITU Kaleidoscope etc.). He achieved best paper award at ITU Kaleidoscope Conference, Cape Town, South Africa, 12–14 December 2011. Dr. Arif worked as a reviewer of different peer reviewed journals including IEEE Sensors Journal, IEEE Communications Letters, IEEE Communications Surveys & Tutorials, IEEE Internet of Things (IoT) Journal etc. For more details, visit https://fse.ewubd.edu/electronics-communications-engineering/faculty-view/mazaman.

M. Rubaiyat Hossain Mondal received the B.Sc. and M.Sc. degrees in electrical and electronic engineering from Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh. He obtained the Ph.D. degree in 2014 from the Department of electrical and computer systems engineering, Monash University, Melbourne, Australia. From 2005 to 2010, and from 2014 to date he has been working as a faculty member at the Institute of Information and Communication Technology (IICT) in BUET. His research interests include wireless communications, optical wireless communications, image processing, bioinformatics and machine learning. https://iict.buet.ac.bd/?page_id=106.