Comparative Analysis of Sentiment Analysis Techniques using Deep Learning

Lavanya Sharma
University of Waterloo
l9sharma@uwaterloo.ca

***Abstract*—Many business domains require knowing the opinions (positive or negative) of their customers regarding their products for improving their services. This can be overwhelming task for online opinions because of huge data. Deep learning techniques are rapidly growing to solve such problems. This study investigates and compares performance of deep learning networks like convolution network (CNN) and three types of recurrent neural network (RNN) like long short-term memory (LSTM), bidirectional LSTM (Bi-LSTM) and gated recurrent unit (GRU) which are very popular for sentiment analysis.Two novel concepts are studied in this work. First, is the effect of change in vocabulary size on prediction accuracy, and second is the dependence of number of epochs on accuracy while keeping the learning rate same. Effect of these variables on accuracy was tested by training and evaluating the four deep learning network models. Same data set was used in the study. Performance evaluation was carried by using confusion matrix and area under the receiver operating characteristics (AUROC) curve. The result shows that accuracy of models is invariant to change in vocabulary size from 5000 to 20000 words for the movie review data set. Also, it was observed increasing the number of epochs doesn't increase the accuracy of deep learning models.**

***Index Terms*—Sentiment Analysis, Deep learning, Convolution Neural Network (CNN), Recurrent Neural Network (RNN), Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), Area Under The Receiver Operating Characteristics (AUROC), Confusion Matrix**

I. Introduction

Automatic sentiment analysis of text is a fast-growing field especially with the development of range of online services. There is a huge requirement to categorize sentiments of opinion and attitude of customers using the online services. Enormous amount of data makes it impossible to analyze it manually without any human error or bias. Automated analysis provides actionable insights and helps companies in taking timely decisions. Over the years several techniques for sentiment analysis are employed.

Traditional approach to do sentiment analysis is based on manually defined rules. These rules include techniques such as Stemming, Tokenization and Lexicons. Disadvantage of this approach is that accuracy depends on number of rules defined and hence results in poor generalization. Additionally, rule-based systems require lot of manual effort for creation and testing of rules.

Automatic methods rely on machine learning techniques. The model is trained to associate text to a category, e.g. positive, negative, or neutral. Here, a feature extractor transfers text input into a feature vector. Feature vectors and category are fed into the machine learning algorithm to generate a model. This model is used for predicting unseen texts into sentiment category. Machine learning algorithms like Naïve Bayes, Logistic Regression and Support Vector Machines are generally used to generate model. However, in recent times the increase in amount of data makes the processing difficult with traditional methods for sentiment classification.

Deep learning-based algorithms have demonstrated great performance in several areas like image recognition, natural language processing and speech recognition in recent times. This is due to easy availability of high computing power. Deep learning-based algorithms were also used for sentiment analysis. Some of the deep-learning networks like convolutional neural network and recurrent neural networks have shown improved performance over other machine learning algorithms for sentiment classification.

The research in sentiment analysis is still ongoing. With the availability of enormous computing power (e.g. GPUs) and huge training data sets, research in sentiment analysis is evolving continuously.

Although deep-learning-based binary sentiment classification models have demonstrated great performances as compared to traditional models, but it is still difficult to select appropriate deep-learning structure for a data set. Therefore, the goal of the study is to compare various deep-learning networks and select optimal network for binary sentiment classification for a data set.

This study is organized as follows. Related literature is briefly overviewed in Section II. Overall methodology used for analyzing sentiments from given text is described in Section III. Thereafter, in Section IV and V publicly available data set and process of converting it into input data suitable for deep learning models is explained. Deep learning models used for comparative analysis are outlined in Section VI. The experiment carried is explained in VII. Evaluation criterion for comparative analysis is specified in Section VIII. Experimental results and discussion are presented in Section IX. The paper is concluded in Section X. Future work is presented in Section XI.

II. Related works

A. Sentiment analysis

Some of the research on methods other than deep learning for sentiment analysis are described here. Initial research on sentiment analysis included studying emerging fields of affective computing and sentiment analysis, which leverage human-computer interaction, information retrieval, and multi modal signal processing for distilling people's sentiments [3]. As research on sentiment analysis evolved, aspect based sentiment analysis [4], [10] method was explored for identifying sentiments and polarity detection. Also, challenges and opportunities of using multi-modal sentiment analysis [5] based on text, speech, images and video were analyzed. Methods for analyzing sentiments based on taxonomy [6] were proposed for extracting information about opinion. Different methods were compared and evaluated for extracting insight of sentiments from common data of TripAdvisor [7].

B. Deep learning for sentiment analysis

In this section, deep learning techniques that have been applied for sentiment analysis in recent years are described. Initial experiments with simple convolution neural network [1], [13] for sentiment analysis showed very promising results. It was shown that simple CNN can perform remarkably well for pre-trained word vectors for sentence-level classification tasks. Subsequently, 7-layer CNN model using word2vec [2] was applied for analysing sentences. Layer-wise relevance technique for feed-forward network was proposed [8] for obtaining better results in predicting sentiments. A deep neural network that incorporated user behavioural pattern [11] was proposed for Twitter sentiment analysis. Target-dependent Convolution Neural Network [12] that leverages the distance information between the target word and its neighbouring words was proposed. Variable Convolution and Pooling Convolution Neural Network [14] proposed for the TextCNN network structure for text sentiment analysis. Various types of RNNs to resolve vanishing or exploding gradient problem by adding gates i.e., LSTM or GRU were used [18], [19]. Document-level sentiment classification using long short-term memory gave several good results [20].

C. Comparative studies on sentiment analysis using deep learning

A systematic comparative analyses [15] using eight deep learning model architectures using word and character levels with 12 online review data sets was performed. It was concluded that word level input and increase in volume of data produces great results for sentiment analysis. A comprehensive survey of the deep learning architectures [9], [17] for applications in sentiment analysis was carried out. They categorized their research at three levels of granularity: document level, sentence level, and aspect level . Accuracy for sentiment classification as a function of learning rate, hidden size and batch size for CNN, LSTM and GRU was studied [16].

III. SENTIMENT ANALYSIS

The general methodology to apply deep learning techniques for sentiment analysis is described in Fig. 1. First, the type of text is chosen for classification. The text can be chosen from various domains. Next step is to prepare the data. The preparation includes cleaning the text and capturing essential text. The text is then used to create vectors. Each word is converted into word vectors for better processing. The words converted into vectors together consist of word embeddings. The data is then divided into two sets. One set is for training the deep learning model and the other set is to test the deep learning model. In this study, namely four deep learning models are used to train the data. These models are Convolutional Neural Network, Long Short-Term Memory, Gated Recurrent Unit and Bidirectional Long Short-Term Memory. The models will be used for prediction. This is a binary classification problem. Therefore, the models will predict either zero (negative) or one (positive). The model is then tested using the testing data set to produce the accuracy with which the model classifies whether the text is positive sentiment or a negative sentiment.

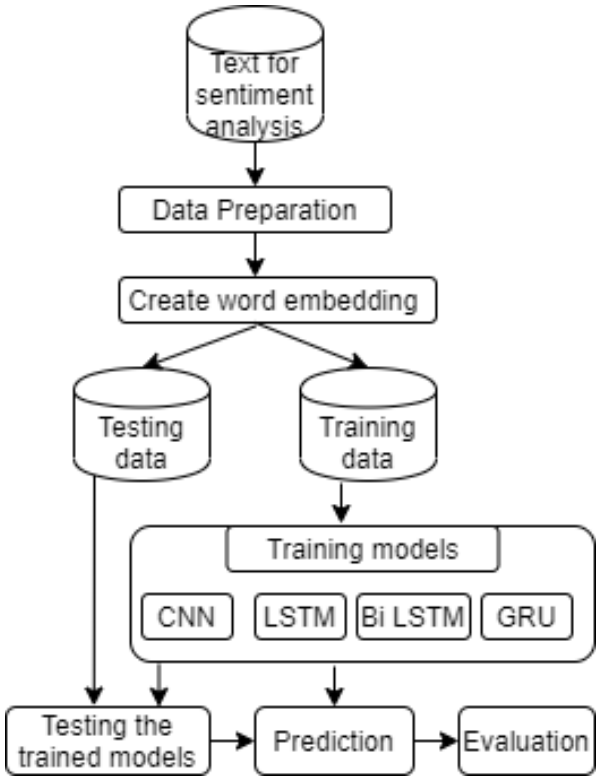


Fig. 1: Methodology of sentiment analysis.

IV. Dataset

Publicly available data set of 50,000 movies reviews from IMDB has been used for analysis [21].The data set consists of movie reviews and the sentiments related to review. If the review is positive then the movie review is classified as positive and if the review is negative then the movie review is classified as negative. The labels are one(positive) or

zero(negative). For analysis 25,000 movie reviews which are highly polar sentiments are used for training, and remaining 25,000 movie reviews are used for testing. The data set contains equal number of positive and negative reviews.

V. Data preparation

Deep learning framework, Keras has been used for data preparation. Data preparation includes cleaning the data. All the special characters need to be removed for better processing of the data. Then, the data needs to be normalized. Next, all the words need to be converted into numerical vectors. This is the most essential step as the input for deep learning models need to be scalar values. Each sentence is converted into a numerical representation. For sentiment classification of movie review data set, each word in the review is converted into word vectors. Several word vectors form a matrix. Next, word embedding is created. Word embedding is responsible for incorporating similarity between words with the context. Word embedding is a representation of text where words that have similar meaning have similar representation. It represents words in a coordinate system where related words are placed together. The embeddings are used to create the deep learning model. The embedding layer is used as the first layer in the neural network.

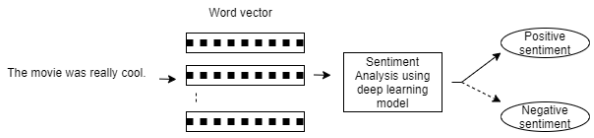


Fig. 2: Example of positive classification for sentiment analysis. The positive review is converted into word vector. The deep learning model classifies the review as positive.

General flow of how a review is classified as positive or negative using deep learning model is depicted in Fig. 2. The reviews are converted into word vectors for classification. These word vectors are passed through model and are classified as either positive or negative.

The movie review data is first encoded such that each word is represented by unique integer. The process of converting into numeric representation is called vectorization. Here, the reviews have been preprocessed, and each review is encoded as a sequence of word indexes. The words are indexed by overall frequency in the data set.This means that if a word is indexed by "5", then it is the fifth most frequent word in the data. This helps in selecting the most frequent words while training the neural networks.

Padding is added to make all the vectors of same length using "maxlen" parameter. The "maxlen" parameter represents maximum sequence length. For our study, "maxlen" is specified as 400. Any longer sequence than 400 is truncated by Keras. This truncated sequence is used in embedding layer. In Keras, the word embedding part is handled by embedding layer. Embedding layer stores a lookup table to map the words represented by numeric indexes to their dense vector representation. The Embedding layer requires the specification of the vocabulary size, the size of the real-valued vector space and the maximum length of input document. For the study, different embedding layers are created to provide better comparison between models. The vocabulary size chosen are 5,000 and 20,000. Also, the size of the real-valued vector is chosen as 128 for all models. After creation of the embedding layer, neural network model can be created.

VI. Deep learning models

In this section, brief description of the deep learning models used for binary classification is given. One CNN based model and three RNN-based model used for comparative analysis are described.

A. Convolution Neural Network (CNN)

Convolution is to apply a sliding window function to a matrix. The sliding window is called a kernel, filter or feature detector. CNN takes advantage of convolutional filters that automatically learn features suitable for the given task. For example, if we use the CNN for the sentiment classification, the convolutional filters capture inherent syntactic and semantic features of sentiment expressions [15].

CNNs are typically used on image data sets. CNNs extract the latent information of the image when the convolutional filter moves over the image. CNNs can be similarly used for text data as well. Here, the filter moves over the text data to extract the important features of the text. For text, a rectangular-shaped convolution filter is used whose width is same as the width of the input matrix.

The Fig. 3 shows the graphical representation of the network. Here, the network consists of an embedding layer, 2 convolutional layers, pooling layer and fully connected layer.

The embedding layer is a matrix where each row represents vector of each word from the sentence. The word vectors are placed in such a way that words with similar meanings are located close and words with opposite meanings are located far apart. Embedding layer can be obtained by several methods. For example, embedding layer can be obtained using Word2vec model.

The convolutional layer is responsible to store local information which is needed to classify the sentiment class. The first convolutional layer slides over the embedding layer with an arbitrary stride, calculates dot product and passes the dot product result to next layer. The second convolutional layer is responsible to extract features from the contextual information of the main word based on the local information stored in the first convolutional layer. The two convolutional layers can have n_1 and n_2 number of distinct filters to capture unique contextual information. To summarize, the first convolutional layer is used to extract simple contextual information while striding over the embedding layer, while the second convolutional layer is utilized to capture key features and then extract them that contain sentiments affecting classification [15].

Next is the pooling layer. The input to the pooling layer is matrix that passed through the two convolutional layers. The

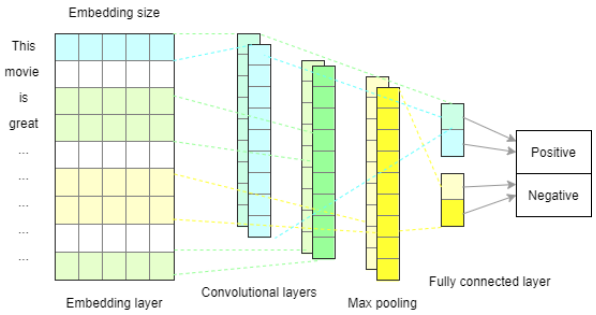


Fig. 3: Different layers of convolution neural network [15]. Embedding layer, 2 convolutional layers, max pooling layer and fully connected layer of CNN are shown.

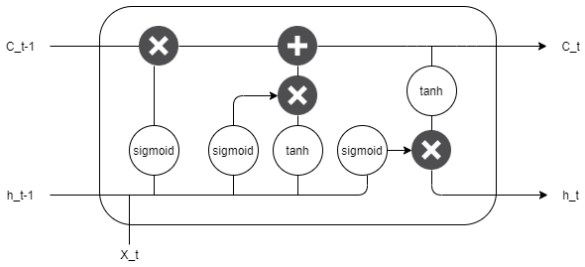


Fig. 4: LSTM Cell [15]: The various inputs to cell are previous hidden state(h_{t-1}), previous cell state(c_{t-1}) and input state(x_t).The various gates decide what information to keep and what to throw away. The outputs of the cell are new cell state(c_t) and hidden state(h_t).

pooling layer slides over the values of matrix to give output vectors. The output vectors of the pooling layer are of small size. The pooling layer selects most prominent features from contexts and extractions of convolutional layers.

After the pooling layer, there is a flattening process to convert the 2-D feature map from the output into a 1-D format. This is passed to Fully Connected layer. The Fully Connected layer connects all input and output neurons. Fully Connected layer gives output which is classified as positive or negative.

B. Recurrent Neural Network (RNN)

RNN is a deep-learning model that specializes in processing sequential data. Since text is sequential data, RNN is usually used in text analytics. However, RNN is not able to perform well when there is long term dependency in text. The reason for bad performance is vanishing or exploding gradient problem. These problems are resolved by using variants of traditional RNNs. Long short-term memory and gated recurrent units are variants of RNNs. They have additional gates to improve the performances. They demonstrate better performance than vanilla RNN which has no memory cell. LSTM consists of many gates like forget gate and input gate. GRU is a simplified LSTM which reduces the number of parameters for learning by combing the input and forget gates of LSTM [9]. Bidirectional LSTM is another variant of LSTM which consists of both forward states and backward states.

1) LSTM (Long short-term memory): LSTM's have internal mechanisms called gates. These gates learn to keep important sequences and throw away unimportant information. Therefore, LSTMs have ability to use relevant information for long sequences to make predictions. LSTM keeps only relevant information to make predictions. LSTM differ from RNN due to various operations within LSTM cells.

The most important part of LSTM is cell state and its various gates. The cell state is like the "memory" of the network. The cell state carries the relevant information throughout the processing of the sequence. The cell state keeps on updating itself by adding or removing information using gates. The gates decide which information to keep and which to forget.

There are three different gates that regulate information [9]. These are forget, input and output gate. The Fig. 4 shows a general LSTM cell with all the gates. The information form previous hidden state and current input is passed through the sigmoid function. For the input gate, we pass the previous hidden state and current input into sigmoid function and tanh function. The sigmoid function gives values between 0 and 1. The tanh function limits the values between -1 and 1 to regulate the network. The sigmoid output decides which information is important from the tanh output. The information from these two gates decides the cell state.

Let the weight and bias associated with the first sigmoid function be w_f and b_f. The weight and bias associated with the second sigmoid function be w_i and b_i. The weight and bias associated with the tanh function be w_c and b_c. And let the weight and bias associated last sigmoid function be w_o and b_o. Also, the previous hidden state and current input be h_{t-1} and x_t respectively. Then the equations for the forget gate(f_t), input gate(i_t) and output(o_t) can be given as follows [9]-

Forget gate:

$$f_t = \sigma(w_f \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} + b_f) \tag{1}$$

Input gate:

$$i_t = \sigma(w_i \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} + b_i) \tag{2}$$

$$\tilde{C}_t = tanh(w_c \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} + b_c) \tag{3}$$

Output gate:

$$o_t = \sigma(w_o \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} + b_o) \tag{4}$$

The equation for the new cell state c_t can be given as

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{C}_t \tag{5}$$

From (1),(2) and (3), we get (6)

$$c_t = \sigma(w_f X_t + b_f)c_{t-1} + \sigma(w_i X_t + b_i)tanh(w_c X_t + b_c) \quad (6)$$

The equation for the new hidden state (7), from (4) and (5) can be given as –

$$h_t = o_t \odot tanh(c_t) \quad (7)$$

2) Bidirectional LSTM: Bidirectional LSTMs are an extension of traditional LSTMs that can improve model performance on sentiment classification problems. In text classification problems it is sometimes necessary to understand both previous word and coming word. Therefore, bi directional LSTM train two instead of one network. The first is trained on the original input sequence and the second is trained on a time-reversed copy of the input sequence.

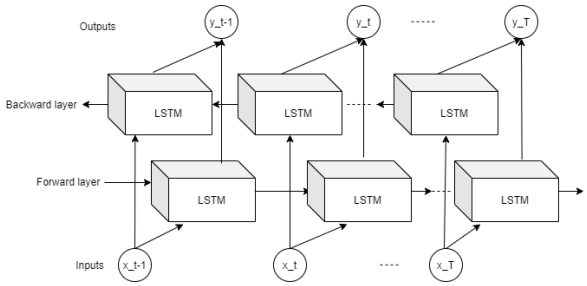


Fig. 5: Bidirectional LSTM [16]: The forward layer and the backward layer of the Bidirectional LSTM are shown. Both forward layer and backward layer have flows in opposite directions.

Bidirectional LSTMs are trained using similar algorithms to RNNs. The only difference is that neurons are split in such a way that some of them are responsible for forward states and some for the backward states [9]. Outputs from the forward states are not connected to inputs of backward states and vice versa.The Fig. 5 shows bidirectional LSTM with forward layer and backward layer.Both the layers have flow in the opposite direction. Without backward state, the structure simplifies to regular LSTM. With both time directions in the same network, input information in the past and future both are considered.

3) GRU: GRU is the newer generation of Recurrent Neural Networks and is very similar to LSTM. GRU is different as cell state is not used to transfer the information. Instead of cell state, GRU uses hidden state to transfer information. It consists of 2 gates, reset gate and update gate.

The update gate is very similar to forget and input gate of LSTM. Update gate is responsible for deciding what new information to add and what to throw away. The reset gate is used to decide how much past information to forget. GRU has fewer tensor operations. Therefore, they are much speedier to train than LSTMs. The Fig. 6 shows a GRU cell. It can be

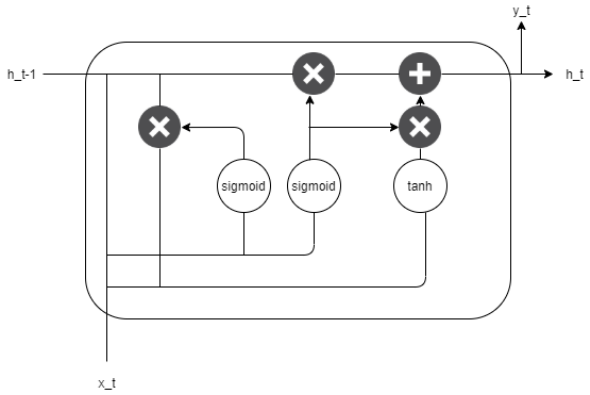


Fig. 6: GRU cell [15]:The inputs to cell are previous hidden state(h_{t-1}) and input state(x_t).The outputs of the cell are output state(y_t) and hidden state(h_t).

observed that it consists much less gates and operations as compared to LSTMs.

The input state and previous hidden state are defined as x_t and h_{t-1} respectively. The corresponding weights and biases are represented as w_z, w_i, w_c, b_z, b_i, b_c. Each gate and state of the GRU can be computed as follows [15]-

$$z_t = \sigma(w_z \left[h_{t-1}, x_t\right] + b_z) \quad (8)$$

$$r_t = \sigma(w_i \left[h_{t-1}, x_t\right] + b_i) \quad (9)$$

$$\tilde{h_t} = tanh(w_c \left[r_t * h_{t-1}, x_t\right] + b_c) \quad (10)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h_t} \quad (11)$$

The update state(z_t), reset state(r_t) and hidden state (h_t) are described by (8), (9) and (11) respectively.

VII. Experiment

The most essential step during deep learning is training of the neural network. The training of the neural network is achieved by continuously updating the weights in the network for reducing errors in the model. In the experiment, four different types (CNN, LSTM, Bi-LSTM and GRU) of deep neural networks were trained. Publicly available IMDB dataset of movie reviews was used for training.

The process of training deep neural network is challenging as it requires high computational power. Therefore, online GPU cloud tool (https://www.paperspace.com) was used for high computational power. RNN based models took approximately two and half hour for training while CNN trained in few minutes.

The models were trained with high-level neural network API, Keras which runs on top of TensorFlow. The model in Keras is constructed by stacking layers. Once the final model is created, the model can be compiled by specifying loss, optimizer and metrics parameters. Subsequently, the model

can be fitted by specifying the number of epochs and batch size. Furthermore, required hyperparameters can be specified to tune the model based on specific requirements.

All models were initially trained using three epochs. Subsequently, for improving the accuracy of models, twenty epochs were used for training. The models were also trained using different vocabulary sizes. The models were trained using 5,000 and 20,000 vocabulary size. All the models were sequential. The first layer was embedding layer in all the models. The embedding layer converts the input data into dense vector suitable for neural network. The second layer was the respective model layer that was being trained. Therefore, if the model that was trained was LSTM, then second layer was LSTM. All the models had one dense layer with activation function as sigmoid. The dense layer takes the dot product of input tensor and weight kernel matrix. After addition of bias, the result was passed through activation function. This output was passed to other layers. The dropout used was 0.2 to avoid overfitting. For CNN, there was an additional layer of pooling. Global max-pooling was used to train data set with CNN. For loss, binary cross entropy was used for all the models and the optimizer used was 'adam' for all the models. Also, the batch size was 32 for all the models.

VIII. Evaluation criterion

Performance measurement is an essential task for comparing various methodologies for sentiment analysis using deep learning. Since this is a classification problem, AUC- ROC (Area Under The Curve - Receiver Operating Characteristics) curve is suitable evaluation metrics for comparing performance. It is a measure of capability of model in differentiating between sentiments. Purely random classifer will have AUC equal to 0.5. AUC is the percentage of the area that is under ROC curve, ranging between 0 and 1. Higher AUC indicate how good is model in predicting negative sentiments as negative sentiments and positive sentiments as positive sentiments.If the AUC value is near 1, then it indicates model is good. The ROC curves are made by plotting sensitivity against specificity. Confusion matrix is another evaluation measure used to evaluate the performance of a classification algorithm. A confusion matrix is a table that is used to describe the performance of a classification model on a set of test data for which the true values are known. The confusion matrix gives the values of true positives(TP), true negatives(TN), false positives(FP), false negatives(FN). Sensitivity and specificity can be defined as follows:

$$Sensitivity = TPR = \frac{TP}{(TP+FN)}$$

$$Specificity = 1 - FPR \text{ , where } FPR = \frac{FP}{(FP+TN)}$$

IX. Results and discussion

In this section, the results of training of CNN, LSTM, Bi-LSTM and GRU neural network models and prediction based on these models are presented.Table I shows accuracy of training and AUC score for small vocabulary size of 5000 words, while Table II shows results for vocabulary size of 20,000 words. Learning rate is kept same for all the trainings. In Fig. 7 and Fig. 8, the confusion matrix for the CNN and LSTM for different vocabulary size and epochs are shown. Confusion matrix for Bi-LSTM and GRU are shown in Fig. 9 and Fig. 10. The corresponding ROC curves are shown in Fig. 11, Fig. 12, Fig. 13 and Fig. 14 for CNN, LSTM, Bi-LSTM and GRU respectively. Also, the corresponding AUC values are depicted.

From Table I, we observe maximum AUC score of 0.9548 for GRU which is marginally higher than CNN (0.9521) for 3 epochs. However, for all other cases, AUC score was highest for CNN architecture (Table I and Table II). Further, AUC score of more than 90% in all cases suggest good sentiment classification. Also, no significant change in test accuracy and AUC score was observed by changing the vocabulary size (Table I and Table II). Therefore, it can be concluded that accuracy of training of neural networks is invariant to vocabulary size.

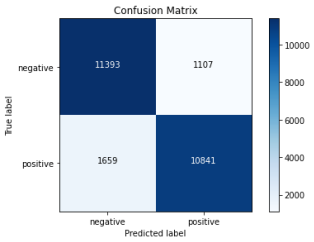
Also, the test accuracy for all the models seem pretty good (84% - 88%) suggesting that sentiment classification of the models is good. However,it was observed that accuracy is not increasing after increasing epochs, suggesting that plateau is reached early.Also, no significant change in test accuracy is observed after increasing the vocabulary size.

TABLE I: Training of deep neural network models for small vocabulary size

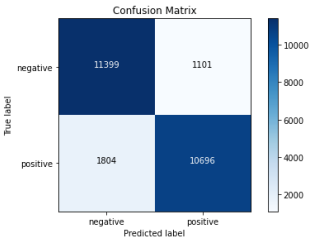
Vocabulary size = 5,000 words		
Model	***Test Accuracy***	***AUC score***
Epoch = 3		
CNN	0.8893	0.9521
LSTM	0.8535	0.9320
Bi-LSTM	0.8690	0.9393
GRU	0.8767	0.9548
Epoch = 20		
CNN	0.8828	0.9543
LSTM	0.8581	0.9321
Bi-LSTM	0.8586	0.9220
GRU	0.8514	0.9367

TABLE II: Training of deep neural network models for large vocabulary size

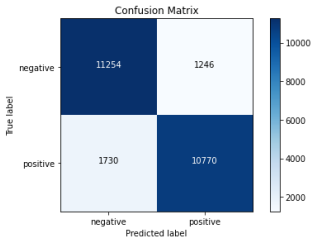
Vocabulary size = 20,000 words		
Model	***Test Accuracy***	***AUC score***
Epoch = 3		
CNN	0.8838	0.9567
LSTM	0.8549	0.9229
Bi-LSTM	0.8500	0.9280
GRU	0.8844	0.9498
Epoch = 20		
CNN	0.8833	0.9488
LSTM	0.8452	0.9141
Bi-LSTM	0.8545	0.9209
GRU	0.8624	0.9349



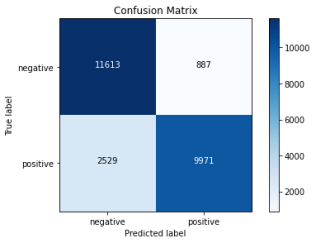
(a) Vocabulary size = 5,000, Epoch = 3



(b) Vocabulary size = 20,000, Epoch = 3

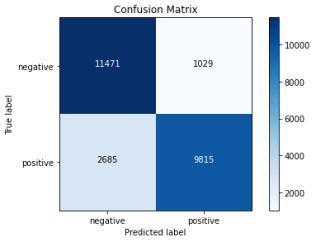


(c) Vocabulary size = 5,000, Epoch = 20

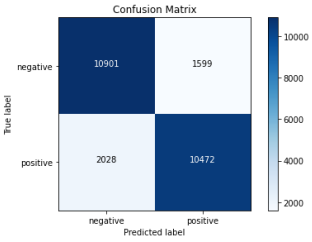


(d) Vocabulary size = 20,000, Epoch = 20

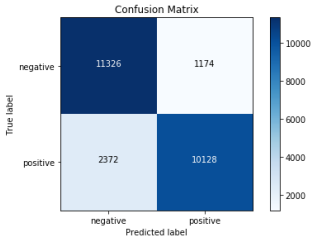
Fig. 7: Confusion matrix for CNN



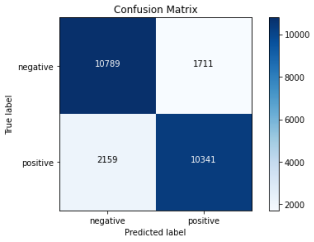
(a) Vocabulary size = 5,000, Epoch = 3



(b) Vocabulary size = 20,000, Epoch = 3

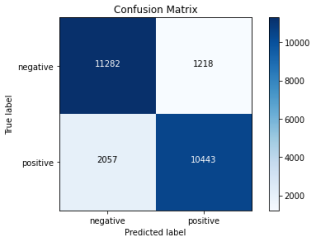


(c) Vocabulary size = 5,000, Epoch = 20

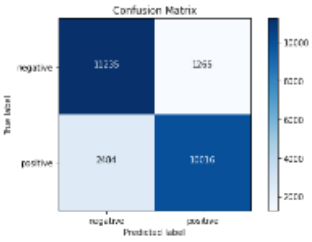


(d) Vocabulary size = 20,000, Epoch = 20

Fig. 8: Confusion matrix for LSTM

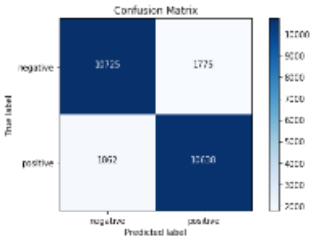


(a) Vocabulary size = 5,000, Epoch = 3



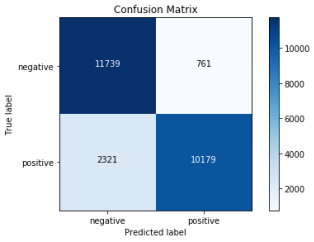
(b) Vocabulary size = 20,000, Epoch = 3

(c) Vocabulary size = 5,000, Epoch = 20

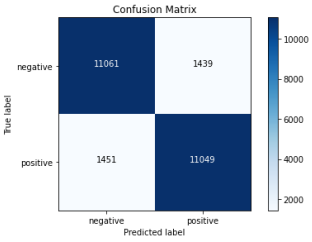


(d) Vocabulary size = 20,000, Epoch = 20

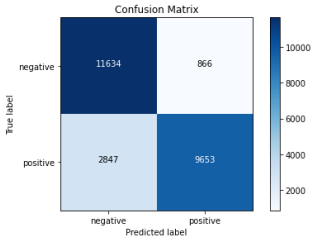
Fig. 9: Confusion matrix for Bi LSTM



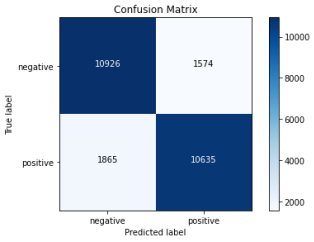
(a) Vocabulary size = 5,000, Epoch = 3



(b) Vocabulary size = 20,000, Epoch = 3

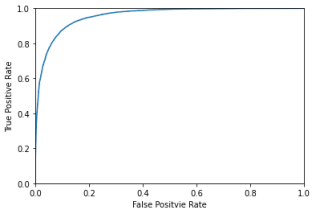


(c) Vocabulary size = 5,000, Epoch = 20

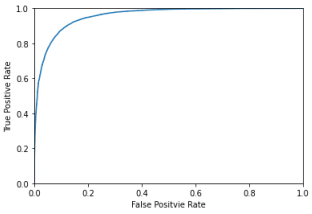


(d) Vocabulary size = 20,000, Epoch = 20

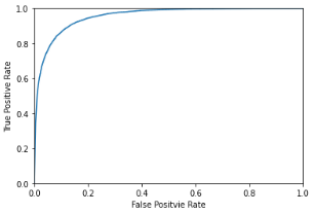
Fig. 10: Confusion matrix for GRU



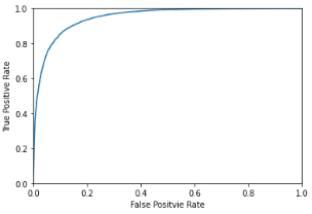
(a) Vocabulary size = 5,000, Epoch = 3 (AUC- 0.9521)



(b) Vocabulary size = 20,000, Epoch = 3 (AUC- 0.9567)

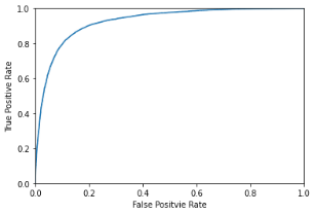


(c) Vocabulary size = 5,000, Epoch = 20 (AUC- 0.9543)

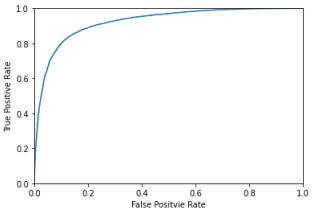


(d) Vocabulary size = 20,000, Epoch = 20 (AUC- 0.9488)

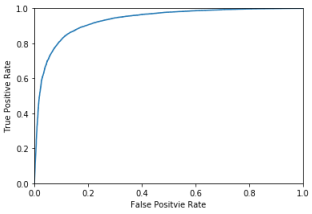
Fig. 11: RUC curves for CNN



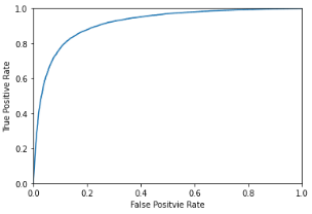
(a) Vocabulary size = 5,000, Epoch = 3 (AUC- 0.9320)



(b) Vocabulary size = 20,000, Epoch = 3 (AUC- 0.9229)

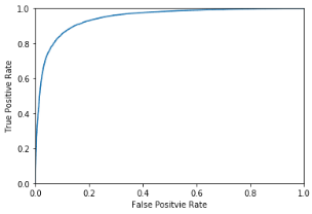


(c) Vocabulary size = 5,000, Epoch = 20 (AUC- 0.9321)

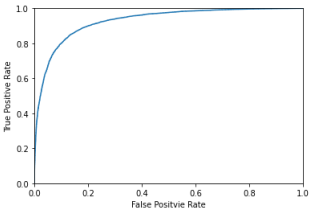


(d) Vocabulary size = 20,000, Epoch = 20 (AUC- 0.9141)

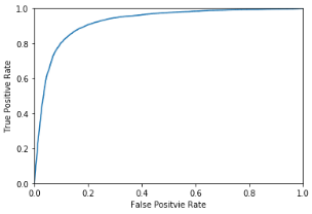
Fig. 12: RUC curves for LSTM



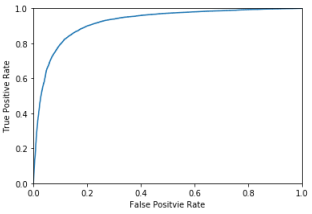
(a) Vocabulary size = 5,000, Epoch = 3 (AUC- 0.9393)



(b) Vocabulary size = 20,000, Epoch = 3 (AUC- 0.9280)

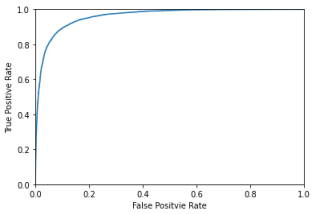


(c) Vocabulary size = 5,000, Epoch = 20 (AUC- 0.9220)

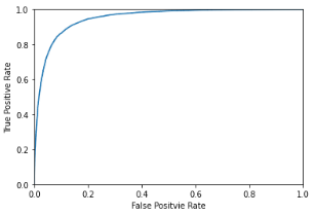


(d) Vocabulary size = 20,000, Epoch = 20 (AUC- 0.9209)

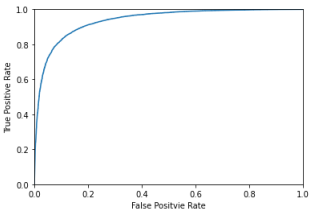
Fig. 13: RUC curves for Bi LSTM



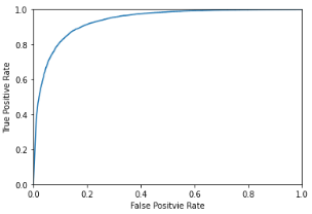
(a) Vocabulary size = 5,000, Epoch = 3 (AUC- 0.9548)



(b) Vocabulary size = 20,000, Epoch = 3 (AUC- 0.9498)



(c) Vocabulary size = 5,000, Epoch = 20 (AUC- 0.9367)



(d) Vocabulary size = 20,000, Epoch = 20 (AUC- 0.9349)

Fig. 14: RUC curves for GRU

X. Conclusion

In this study, deep learning based techniques for sentiment analysis were compared for performance analysis. One CNN and three RNN architectures were compared by predicting the accuracy. IMDB movie review dataset was used for modeling and evaluation of binary classification problem. Comparison was performed by using metrics like confusion matrix and the area under the receiver operating characteristics (AUROC) curve. The sensitivity and specificity of a binary classifier was characterized by AUROC curves.

The study has shown that CNN are GRU have performed well. For small vocabulary size (5000 words), irrespective of the deep learning architecture (CNN, LSTM, Bi-LSTM and GRU), accuracy generally remains same. Even for larger vocabulary size (20,000 words) the accuracy and performance in terms of AUC score remain more or less same.

The performance of deep learning models was investigated over number of epochs for learning. Performance of all models for 3 epochs and 20 epochs was studied. All models show that accuracy remains almost same.

By looking at the accuracy and AUC score for all the models,it can be concluded that CNN has performed slightly better than other deep learning architectures for binary classification. Further, it is concluded that accuracy of models is invariant to change in vocabulary size from 5000 to 20000 words for the movie review data set.

XI. Future work

The study performed have limitations because of embedding as they do not represent combination of two or more words in the study. One solution to this problem to identify phrases based on word co-occurrence n-gram embeddings from unlabeled data.

Another limitation comes from size of window of surrounding words. Some time these embedding clusters may have words with opposing polarities. This leads to model used for polarity analysis with poor performance. Hence, sentiment specific word embedding can be used.

Another way for improving performance of sentiment analysis is to implement attention mechanism. Here, large weights are assigned for those words that are influencing classification objective.

Further, different combination of words can indicate different sentiments. For example, the word like *good* may indicate a positive sentiment; while *not so good* indicate neutral sentiment and so *so good* positive sentiment. There are practically infinitely many combinations of commonly used words and phrases whose sentiment can go different ways. Attention mechanism in deep learning can learn words and their combination for achieving the sentiment classification.

Therefore, it is suggested to apply *Attention mechanism* on top of deep learning layers for improving the accuracy of sentiment analysis.

References

[1] Y. Kim, “Convolution neural networks for sentence classification”, arXiv:1408.5882, Aug 2014.

[2] X. Ouyang, P. Zhou, C. H. Li, L. Liu, ”Sentiment analysis using convolution neural network”, Proc. IEEE Int. Conf. Comput. Inf. Technol. Ubiquitous Comput. Commun. Dependable Autonomic Secure Comput. Pervasive Intell. Comput., pp. 2359-2364, Oct. 2015.

[3] E. Cambria,” Affective computing and sentiment analysis”, IEEE Intelligent Systems, vol. 31, issue 2, pp 102-107,Mar-April 2016.

[4] N. U. Pannala, C. P. Nawarathna, J. Jayakody, L. Rupasinghe, K. Krishnadeva, ”Supervised learning based approach to aspect based sentiment analysis”, Proc. IEEE Int. Conf. Comput. Inf. Technol. (CIT), pp. 662-666, Dec. 2016.

[5] M.Soleymani, D. Garcia, B. Jou, B. Shuller, S. Chang, M. Pantic,” A survey of multimodal sentiment analysis”, Image and Vision Computing, vol. 65, pp. 3-14, Sep 2017.

[6] A. Yadollahi, A. G. Shahraki, O. R. Zaiane,” Current state of text sentiment analysis from opinion to emotion mining”, ACM Computing Surveys, vol. 50, Issue 2, 2017

[7] [7] A. Valdivia, M. V. Luzon, F. Herrera,”Sentiment analysis in TripAdvisor”, IEEE Intelligent Systems, vol. 32, Issue 4, pp 72-77, 2017.

[8] L. Arras, G. Montavon, K. R. Muller, W. Samek,” Explaining recurrent neural network predictions in sentiment analysis”, Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis, arXiv 1706 07206v2, 2017.

[9] L. Zhang, S. Wang, B. Liu, ”Deep learning for sentiment analysis: A survey”, Wiley Interdiscipl. Rev. Data Mining Knowl. Discovery, vol. 8, no. 4, 2018.

[10] H. H. Do, P. Prasad, A. Maag, A. Alsadoon, ”Deep Learning for aspect based sentiment analysis: A Comparative Review”, Expert Syst. Appl., vol. 118, pp. 272-299, Mar. 2019.

[11] A. S. M. Alharbi, E. De Doncker, ”Twitter sentiment analysis with a deep neural network: An enhanced approach using user behavioural information”, Cognit. Syst. Res., vol. 54, pp. 50-61, May 2019.

[12] D. Hyun, C. Park, M.C. Yang, I. Song, J. T. Lee, H. Yu, “Target aware convolution neural network for target level sentimental analysis”, Inf. Sci., vol. 491, pp 166-178, July 2019.

[13] R. Wang, Z. Li, J. Cao, T. Chen, L. Wang, ”Convolution recurrent neural networks for text classification”, Proc. Int. Joint Conf. Neural Netw. (IJCNN), pp. 1-6, Jul. 2019.

[14] M. Dong, Y Li, X. Tang, J. Xu, S. Bi, Y. Cai, “Variable convolution and pooling convolutional neural network for text sentiment classification”, IEEE Access, vol. 8, DOI: 10.1109/ACCESS.2020.2966726, Jan 2020.

[15] S. Seo, C. Kim, H. Kim, K Mo, P. Kang, “Comparative study of deep learning-based sentiment classification”, IEEE Access, vol. 8, DOI: 10.1109/ ACCESS.2019.2963426, Jan 2020.

[16] W. Yin, K. Kann, M. Yu, and H. Schütze, “Comparative study of cnn and rnn for natural language processing,” arXiv preprint arXiv:1702.01923, 2017.

[17] T. Katic and N. Milicevic, “Comparing sentiment analysis and document representation methods of amazon reviews,” in 2018 IEEE 16th International Symposium on Intelligent Systems and Informatics (SISY), pp. 283-286, Sep. 2018.

[18] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” arXiv preprint arXiv:1406.1078, 2014

[20] J. Xu, D. Chen, X. Qiu, and X. Huang, “Cached long short-term memory neural networks for document-level sentiment classification,” arXiv preprint arXiv:1610.04989, 2016.

[21] M. Andrew, D. Raymond, P. Peter,H. Dan, N. Andrew, P.Christopher, ”Learning Word Vectors for Sentiment Analysis,” in Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 142-150, June. 2011. [Dataset]. Available:http://ai.stanford.edu/ amaas/data/sentiment/ [Accessed: March 10, 2020]