Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 218 (2023) 45–56

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Machine Learning and Data Engineering

Sentiment Analysis using Bidirectional LSTM Network

U. B. Mahadevaswamy, Swathi P.*

Sri Jayachamarajendra College of Engineering, Mysore, India-570006

Abstract

Sentiment analysis is a cognitive tool to extract the emotional tone of a piece of text. It is an area of research that is actively pursued in Natural Language Processing. Social media, forums and blogs, among other places, have seen a massive surge in the number of personalized reviews since the emergence of Internet-based applications. Sentiment analysis is primarily concerned with the classification and prediction of users' thoughts and emotions from these reviews. In recent years, numerous deep learning techniques have emerged to achieve this task. This paper provides a technical summary of Sentiment Analysis using a Bidirectional LSTM network. This model is capable of handling long-term dependencies by introducing memory into the model for making better predictions. The Amazon Product Review dataset served as the source of data for this research. Concretely, the analysis is performed on 104,975 product reviews reflecting users' attitudes toward mobile electronics products. The proposed model attempts to classify the reviews into two categories: positive and negative. Finally, the paper outlines the results of the analysis and suggests potential avenues for future research.

*Keywords:*Opinion mining; Sentiment Analysis; RNN; LSTM; Bidirectional LSTM

1. Introduction

The areas of research in NLP (Natural Language Processing) have improved substantially since its inception. NLP is a branch of Artificial Intelligence (AI) and linguistics that focuses on how human language and computers interact [1]. The key application areas of NLP are speech recognition, NER (Named Entity Recognition), Opinion Mining, Machine translation, text generation and text summarization. Social media, forums and blogs, among other

* Corresponding author. Tel.: +91-9449199842
E-mail address: swathi.iemysore@gmail.com

10.1016/j.procs.2022.12.400

places, have seen a massive surge in the number of personalized reviews since the emergence of Internet-based applications [2]. Sentiment analysis is primarily concerned with the classification and prediction of users' thoughts and emotions from these reviews. In simple words, sentiment analysis is a tool to determine the polarity of a piece of text, i.e., either positive or negative [3]. Figure 1 illustrates the major trends in sentiment analysis, text analytics and text mining (from 2005 to 2013).

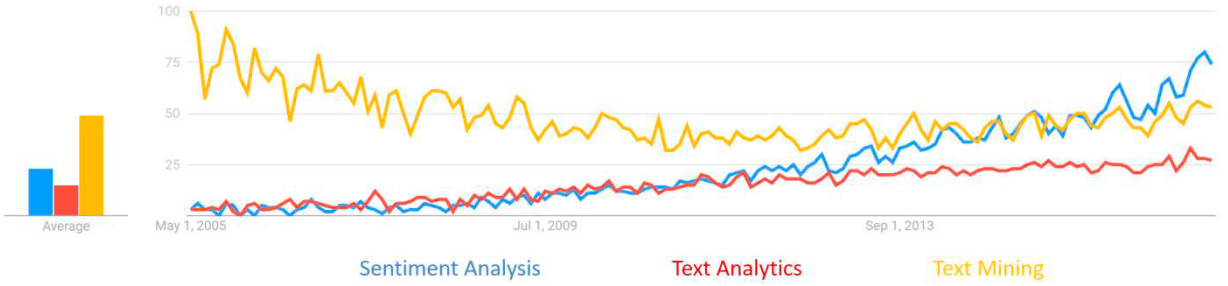

Fig 1: Recent trends in sentiment analysis, text analytics and text mining.

Almost every industry and corporation is going through a digital transformation today, resulting in massive amounts of structured and unstructured data. Sentiment analysis assists businesses in deriving meaningful insights from the unstructured data found on sources such as support tickets, emails, social media channels, blog posts, web chats, forums, and comments. These insights may help businesses while making important decisions[3]. Sentiment analysis also enables all kinds of market research and competitive analysis; whether a user is researching a new market, forecasting future trends, or looking for a competitive advantage, sentiment analysis is extremely beneficial.

Also, in recent times, organizations are no longer reliant on opinion polls, surveys, or focus groups because of the multitude of publicly available data. Given the likelihood to check each website, the task of opinion mining can be quite daunting to a human reader. Also, it can be very cumbersome for the human reader to identify the relevant sites and capture opinions from them. As a result, automated sentiment analysis is required. Sentiment analysis is also a valuable tool for monitoring social media sites since it offers a summary of public perception on any given issue. Monitoring social media sites, managing customer service, and analyzing consumer feedback are all popular sentiment analysis applications [3]. Automatic sentiment analysis may be employed to group Twitter and Facebook posts, survey responses, and chats, and scan emails and other documents.

2. Background

2.1. RNN (Recurrent Neural Network)

In general, feed forward neural networks are well-suited for independent data points [4]. But if the data points appear sequentially, the neural network must be adapted to include the dependencies between the data points.Recurrent neural networks, often known as RNNs, are an adaptation of the traditional feed-forward artificial neural networks, which are capable of processing sequential data [5]. RNNs are equipped with a "memory" element, which enables them to store the states of earlier inputs in order to generate the next output in the sequence [6]. Figure 2 illustrates the compressed and the unfolded forms of a recurrent neural network. The unfolded network closely resembles a feed-forward neural network.

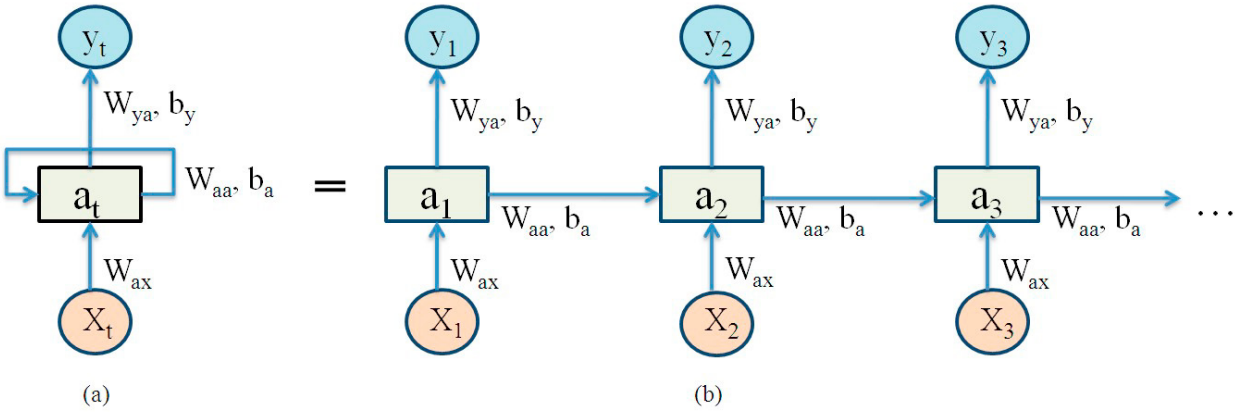

Fig 2: Recurrent Neural Network in (a) compressed form, (b) unfolded form

Let us consider the following notation for RNN. At time step t,

- X_t = the input to the network.
- y_t = the output from the network.
- a_t = the hidden state.
- W_a ,W_x and W_y = the weights associated with hidden units, input units and output units respectively.
- b_a, b_y = the biases associated with hidden layer and output layer respectively.

With an activation function f, the next hidden state is computed using equation (1).

$$a_{t+1} = f\left(W_X \cdot X_t + W_a \cdot a_t + b_a\right) \tag{1}$$

The output y at time t is computed using equation (2).

$$y_t = f\left(a_t, W_y \cdot\right) = f\left(W_y \cdot a_t + b_y\right) \tag{2}$$

The most widely used activation functions in the RNN are sigmoid, tanh and ReLU(Rectified Linear Unit) [7]. The weights in the RNN are updated using the backpropagation in time (BPTT) algorithm. This algorithm is a gradient based iterative technique used to train the RNN while minimizing the error [8]. However, when the BPTT algorithm is applied to a deep RNN, i.e., an RNN with a large no. of layers, some problems are encountered. As one backpropagates through each layer of the network, the exploding gradient problem appears when the gradients accumulate, whereas the vanishing gradient problem appears when the gradients progressively decline and reach zero [9].Generally, the vanishing gradient problem is seen more often and is very hard to mitigate because the weight updates become too insignificant, which means that no real learning is happening.

2.2. LSTM (Long Short Term Memory)

To overcome the vanishing gradient problem, a gating based RNN architecture called LSTM was developed. LSTM is a specific class of RNN that is well-suited to learn long term dependencies [10]. The structure of RNN appears as a sequence of neural network modules that iterate. This iterating module can take various forms, for example, a layer comprising of tanh function only. LSTMs also have a similar structure with recurring modules.

There are four layers instead of just one, and they interact in a very unique manner. The recurring module in LSTM is illustrated in figure 3.

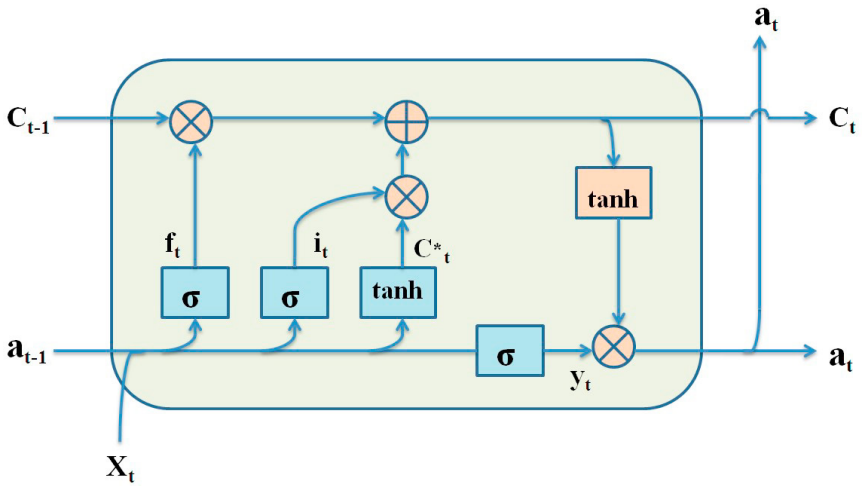

Fig 3: Recurring module in LSTM

The cell state is represented by the horizontal line at the top of the module. In some aspects, the cell state mimics an assembly line. It proceeds down the line with a few linear operations. LSTMs are capable of modifying the cell state by removing or adding information using structures called gates. Gates consist of sigmoid layer, tanh layer and point-wise multiplication operations. The LSTM module is made up of three gates: the input gate, the forget gate and the output gate [11].

Let us consider the following notation for LSTM. At time step t,

- X_t = the input to the network.
- y_t = the output from the network.
- a_{t-1} = the previous hidden state.
- C_t = the current cell state.
- C^*_t = the candidate function.
- σ= the sigmoid function.
- W_i ,W_f , W_c, W_y = the weights associated with the input gate, the forget gate, the candidate function and the output gate respectively.
- b_i , b_f , b_y = the biases associated with the input gate, the forget gate, and the output gate respectively.

Forget gate: This gate determines whether to keep the information or to discard it. The current input and the previous hidden state are processed by a sigmoid layer. This layer outputs a value between 0 and 1. If the output value is closer to 1, then keep the information. Else, forget the information.

The output of the forget gate is computed using equation(3).

$$f_t = \sigma\left(W_f \cdot [a_{t-1}, X_t] + b_f\right) \tag{3}$$

Input gate: The cell state is updated using the input gate. First, the current input and the previous hidden state are processed by a sigmoid layer and a tanh layer separately. The sigmoid layer converts a data value to a value between 0 and 1. The tanh layer converts a data value to a value between -1 and 1. A point-wise multiplication operation is performed on the outputs of the sigmoid layer and the tanh layer. The new cell state value is then computed.

The output of the input gate is computed using equation(4).

$$i_t = \sigma\left(W_i \cdot [a_{t-1}, X_t] + b_i\right) \tag{4}$$

The output of the tanh layer is computed using equation(5).

$$C^*_t = \tanh\left(W_c \cdot [a_{t-1}, X_t] + b_c\right) \quad (5)$$

The new cell state is computed using equation (6)

$$C_t = f_t \cdot C_{t-1} + i_t \cdot C^*_t \quad (6)$$

Output gate - The output gate determines the next hidden state. First, the current input and the previous hidden state are processed by a sigmoid layer. Then, the new cell state is passed to a tanh layer. A point-wise multiplication operation is performed on the outputs of the sigmoid layer and the tanh layer to determine the next hidden state. The next hidden state and the new cell state are then passed onto the subsequent time step.

The output gate computes the output using equation (7)

$$y_t = \sigma\left(W_y \cdot [a_{t-1}, X_t] + b_y\right) \quad (7)$$

The next hidden state is computed using equation (8)

$$a_t = y_t . \tanh(C_t)$$

2.3. Bidirectional LSTM network (Bi-LSTM)

In order to create bidirectional LSTM networks, the LSTM neurons are divided into two directions: one for forward states and the other for backward states [12]. Figure 4 illustrates the configurations of LSTM and Bi-LSTM networks. The two directions in the network allow input data from both past and future of the present time frame, unlike a regular LSTM network which has delays due to fetching the information from the future. Since the two directional neurons do not interact, Bi-LSTMs can be taught using LSTM-like techniques. However, additional procedures are required when back-propagation over time is used because updating the input and output layers cannot be done simultaneously [13].

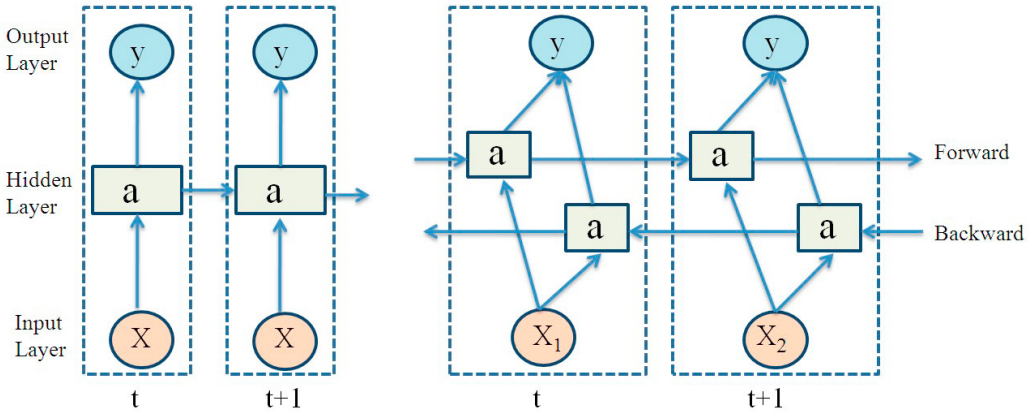

Fig 4: Configuration of (a) LSTM network (b) Bi-LSTM network

3. Implementation

3.1. Proposed Methodology

In the proposed work, sentiment analysis is performed on the Amazon Product Reviews dataset. The data in the dataset is mostly unstructured. It may also contain a lot of noise and all kinds of errors in the raw form. Hence, the datamust be pre-processed prior to performing any analysis. The data pre-processing operations include tokenization and case-folding of the review data. The processed data is then encoded using a text encoder which transforms the text data into word vectors. This data is divided into 2 sets: training and validation sets. These sets are further divided into batches of 128 reviews. Each review body is then padded with 'zeros' at the end in order to make all the reviews in the batch fit a standard length. A bidirectional LSTM network is constructed and trained to classify the reviews into 2 tiers- positive or negative. Finally, the performance evaluation metrics associated with sentiment classifier, such asaccuracy, precision, recall, F1 score, specificity and misclassification rate, are computed. Figure 5 illustrates the methodology for the proposed work.

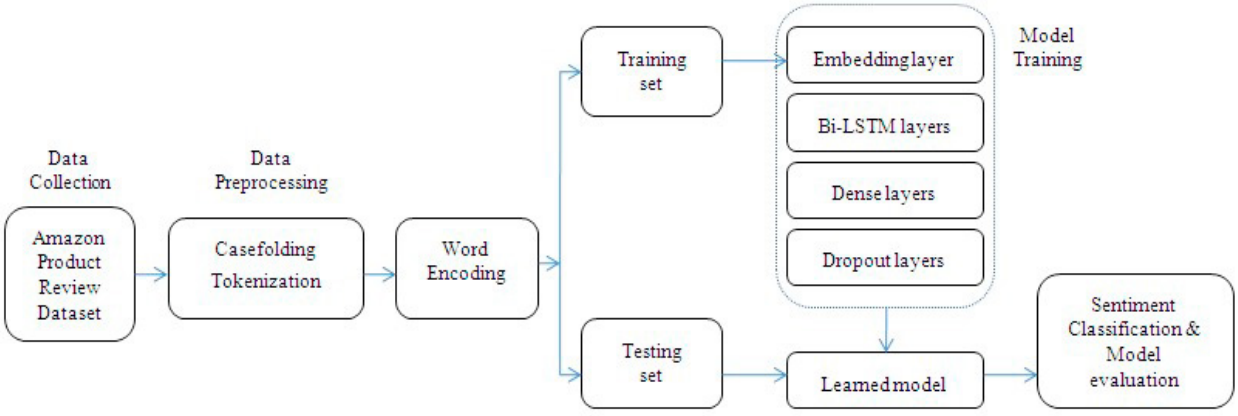

Fig 5: Proposed Methodology.

3.2. Procedure

The proposed method for performing sentiment analysis comprises of the following stages: data collection, pre-processing, word encoding, model design, andmodelevaluation

3.2.1. Data Collection

The dataset used to perform sentiment analysis is the Amazon product review dataset [14]. Under this dataset, the data corresponding to Mobile Electronics category is used. This dataset consists of product metadata and reviews. The product metadata consists of product descriptions, pricing, branding, and product images. The product reviews consist of star ratings, review body and helpfulness votes. It consists of 104975 reviews. The proposed method uses the review body and star ratings to perform sentiment analysis.

3.2.2. Data pre-processing

Generally, the data acquired from many sources, such as social media sites and blogs, is unstructured. These data may contain a lot of noise and all kinds of typographical and grammatical errors in their raw form. Hence, it is crucial to clean and pre-process the text prior to performing any analysis [15]. The goal of the pre-processing stage

is not just to improve analysis but also to reduce the dimensionality of input data, since many words are redundant and should be discarded because they have no effect on text polarity [16]. In the proposed work, the pre-processing techniques used are tokenization and case-folding. Tokenization decomposes the raw text into words or sentences, referred to as tokens. A set of delimiters is used to split the raw text into tokens[17]. Tokenization aids in interpreting the meaning of the text by analyzing the sequence of words. The tokenizer class tfds.deprecated.text.Tokenizer allows us to convert each word in the text corpus into word vectors [18]. This tokenizer class splits a string into tokens, and rejoins them. All the unique tokens from the text are now captured in a vocabulary set. Case-folding is a popular technique in which all the characters in the text are collapsed to lowercase.The size of the vocabulary set obtained after tokenization and case-folding is 73738.

3.2.3. Word Encoding

Word encoding is a technique which creates word vector representations of lower dimensions from volumes of text data [19]. The text encoder used in the proposed work is tfds.deprecated.text.TokenTextEncoder from TensorFlow [20]. This encoder encodes the tokens in the vocabulary into word vectors of dimension n*m, where n = size of the hidden layer and m = size of the vocabulary. In the proposed work, m = 73738 and n=128.

3.2.4. Training and Testing Procedure

The need to partition the dataset into training set and testing set is to prevent the model from overfitting and to evaluate the performance of the trained model. The training set consists of data items used to fit the model. The testing set consists of data items solely used for making predictions. In the proposed work, the training set consists of 99,975 reviews and the testing set consists of 5,000 reviews.

3.2.5. Padding

Padding is performeddue to the requirement of encoding sequence data into continuous batches, and to make all sequences in the batch fit the standard length. So, each review body in its encoded form is padded with ’zeros’ at the end to ensure all the review bodies fit the standard length of the batch

3.2.6. Model Design

Sentiment classification is performed using a Bidirectional LSTM model. It consists of the following layers:

1. The model consists of an embedding layer as the first layer. It takes the parameters-batch size and vocabulary size- as inputs. In the proposed model, vocabulary size =73738 and batch size =128.
2. Next, the model consists of bidirectional LSTM layers. There are 2 Bi-LSTM layers in the proposed model. These layers consist of 128 units and tanh activation function.
3. Next, the model consists of dense layers. There are 3 dense layers in the proposed model. These layers use ReLU activation function. Dense layer 1and 2 comprise of 64 units, and layer 3 consists of 16 units.
4. The model also has dropout layers. These layers are used to prevent the model from overfitting.
5. The last layer of the model is the output layer. It is a dense layer with a single neuron.

Since the analysis consists of binary classification problem, the loss function used is binary cross-entropy function. The model is compiled with ‘Adam’ optimizer and the evaluation metric is accuracy. The model is fit with epochs = 5.

3.2.7. Model Evaluation

The model performance is compared to that of baseline models on the basis of multiple factors. These factors are known as model evaluation metrics. A confusion matrix provides the count of correct and incorrect predictions

based on the actual values [21]. It consists of four values, namely, true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Figure 6 illustrates a confusion matrix.

- True Positives (TP): represents the number of correctly classified reviews that are negative in polarity.
- True Negatives (TN): represents the number of correctly classified reviews that are positive in polarity.
- False Positives (FP): represents the number of reviews that are misclassified into negative polarity, but are actually positive.
- False Negatives (FN): represents the number of reviews that are misclassified into positive polarity, but are actually negative

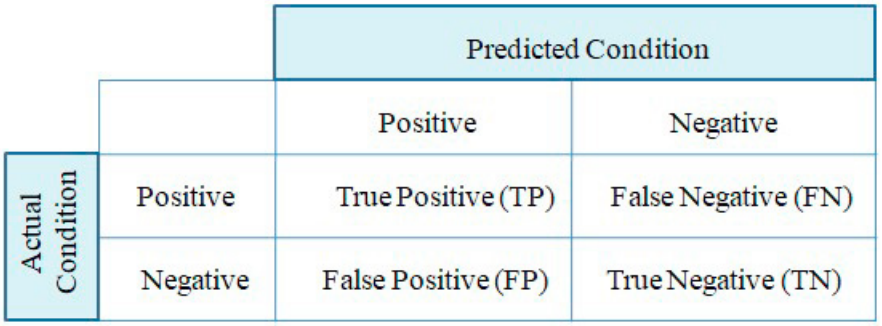

Fig 6: Confusion matrix

The performance evaluation metrics associated with the model are then computed using the values in confusion matrix. These metrics are accuracy, misclassification rate, precision, recall, f1 score, and specificity.

3.2.8. *System Specifications*

The proposed model makes use of a Graphical Processing Unit (GPU) that is available on Google Colab. Table 1 lists the GPU specifications for performing sentiment analysis using the proposed method.

Table1. GPUSpecifications

Criteria	Specification
Cloud workstation	Colab Notebook
GPU Name	Nvidia Tesla T4
Release Year	2018
CPU Cores	2
RAM	12GB (upgradable to 26.75GB)
Disk Space	358GB
GPU Memory	12GB
GPU Memory Clock	0.82GHz / 1.59GHz

4. Results

Figure 7 shows a summary of the proposed model. It lists the various layers within the Bidirectional LSTM network.

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, None, 128)         9438592

 bidirectional (Bidirectiona  (None, None, 256)        263168
 l)

 bidirectional_1 (Bidirectio  (None, 256)              394240
 nal)

 dense (Dense)               (None, 64)                16448

 dropout (Dropout)           (None, 64)                0

 dense_1 (Dense)             (None, 64)                4160

 dropout_1 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 16)                1040

 dropout_2 (Dropout)         (None, 16)                0

 dense_3 (Dense)             (None, 1)                 17

=================================================================
Total params: 10,117,665
Trainable params: 10,117,665
Non-trainable params: 0
_________________________________________________________________
```

Fig 7: Model summary.

Figure 8 shows the Bi-LSTM network being trained using training dataset. A total of 5 epochs are executed to train the Bidirectional LSTM network.

```
Epoch 1/5
781/782 [============================>.] - ETA: 0s - loss: 0.5092 - accuracy: 0.6580
Epoch 1: val_loss improved from inf to 0.33174, saving model to /tmp/sentiment_analysis.hdf5
782/782 [==============================] - 389s 445ms/step - loss: 0.5092 - accuracy: 0.6580
Epoch 2/5
781/782 [============================>.] - ETA: 0s - loss: 0.3364 - accuracy: 0.8567
Epoch 2: val_loss improved from 0.33174 to 0.30016, saving model to /tmp/sentiment_analysis.hdf5
782/782 [==============================] - 377s 445ms/step - loss: 0.3364 - accuracy: 0.8567
Epoch 3/5
781/782 [============================>.] - ETA: 0s - loss: 0.2952 - accuracy: 0.8798
Epoch 3: val_loss improved from 0.30016 to 0.29835, saving model to /tmp/sentiment_analysis.hdf5
782/782 [==============================] - 374s 442ms/step - loss: 0.2952 - accuracy: 0.8798
Epoch 4/5
781/782 [============================>.] - ETA: 0s - loss: 0.2683 - accuracy: 0.8928
Epoch 4: val_loss improved from 0.29835 to 0.29388, saving model to /tmp/sentiment_analysis.hdf5
782/782 [==============================] - 374s 441ms/step - loss: 0.2683 - accuracy: 0.8928
Epoch 5/5
781/782 [============================>.] - ETA: 0s - loss: 0.2476 - accuracy: 0.9013
Epoch 5: val_loss did not improve from 0.29388
782/782 [==============================] - 376s 444ms/step - loss: 0.2476 - accuracy: 0.9014
```

Fig 8: Training of Bi-LSTM network

In epoch 1, the model is able to make predictions with an accuracy of 65.8% and loss of 50.92%. It also achieves a validation loss of 33.17% and validation accuracy of 85.34%. In epoch 2, the model is able to make predictions with an accuracy of 85.67% and loss of 33.64%. It also achieves a validation loss of 30.02% and validation accuracy of 86.26%. In epoch 3, the model is able to make predictions with an accuracy of 87.98% and loss of 29.52%. It also achieves a validation loss of 29.83% and validation accuracy of 86.98%. In epoch 4, the model is able to make predictions with an accuracy of 89.28% and loss of 26.83%. It also achieves a validation loss of 29.39% and validation accuracy of 87.30%. In epoch 5, the model is able to make predictions with an accuracy of 90.14% and loss of 24.76%. It also achieves a validation loss of 32.20% and validation accuracy of 88.08%.The proposed model has trained with 10,117,665 parameters.

The graphical representation of accuracy and validation accuracy is illustrated in figure 9(a). Similarly, the graphical representation of loss and validation loss is illustrated in figure 9(b).

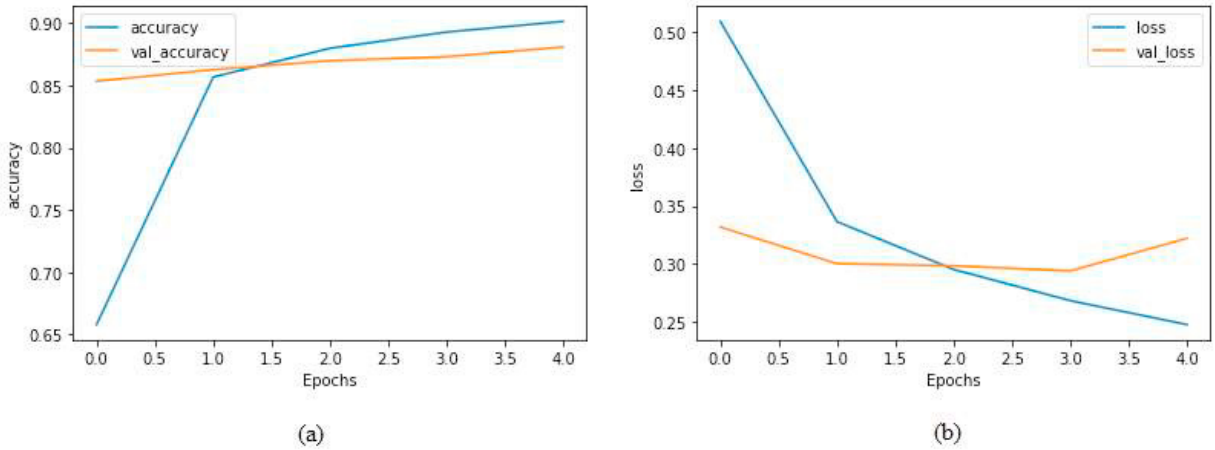

Fig 9: (a) Accuracy, (b) Loss during training phase.

Sentiment analysis is then performed on the test dataset using the trained model to gauge its performance. The evaluation metrics associated with the model performance are computed. These metrics are accuracy, misclassification rate, precision, recall, f1 score, and specificity. To compute these parameters, a confusion matrix is constructed. Figure 10 shows the confusion matrix of the proposed model.

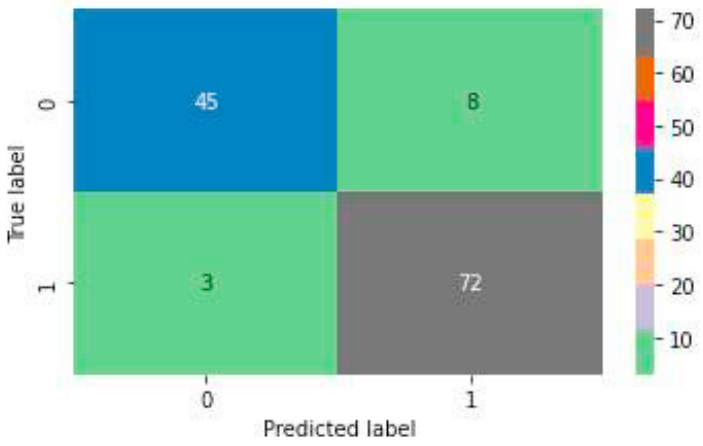

Fig 10: Confusion Matrix of the proposed model.

From the confusion matrix of the model, True Positives = 72, True Negatives = 45, False Positives = 8, and False Negatives = 3. Table 2 lists the performance evaluation metrics associated with the Bi-LSTM network.

Table2. Performance evaluation metrics of the model

Metric	Formula	Value
Accuracy	(TP + TN) / (TP + TN + FP + FN)	0.914
Precision	TP / (TP + FP)	0.9
Recall	TP / (TP + FN)	0.96
F1 Score	(2 *Precision*Recall) / [Precision + Recall]	0.929
Specificity	TN / (TN + FP	0.849
Misclassification rate	(FP + FN) / (TP + TN + FP + FN)	0.085

The originality of the proposed work is that the no. of epochs used to train the model is very less compared to other models, which typically use about 20 to 50 epochs to train. Also, the execution time to train the proposed model is 33.71 minutes.

6. Conclusion

Deep learning provides a means to employ a substantial amount of data and compute with minimal engineering. Advanced deep learning models are now the cutting-edge solutions to problems in NLP. One such problem addressed in NLP is opinion mining. Opinion mining, commonly referred to as sentiment analysis, emphasizes categorizing and predicting people's opinions concerning a certain issue. This paper intends to provide a technical summary of Sentiment Analysis using a Bidirectional-LSTM network. Review body and star ratings are the essential features used to make predictions on users' sentiments. Table 3 lists the experimental results of various techniques used to perform sentiment analysis. Clearly, the proposed method comprising the Bidirectional LSTM model achieves a higher accuracy of 91.4%. Future research may include a more fine-grained sentiment analysis, i.e., classification of sentiments into 3 to 5 tiers. The use of ensemble methods might potentially increase the classifier's performance. Also, the emotional tone of the text, i.e., happy, excited, sad, angry, etc., could be identified.

Table3. Experimental results of various other models.

Authors/ year	Method	Classification	Accuracy
[22] Z. Jianqiang, G. Xiaolin, and Z. Xuejun, 2017	CNN	Positive/Negative	87.62%
[23] M.-H. Su, C.-H. Wu, K.-Y. Huang and Q.-B. Hong, 2018	LSTM	Positive/Negative	70.66%
	CNN	Positive/Negative	65.33%
[24] K. Liu and L. Chen., 2019	CNN	Positive/Negative	86.28%
	LSTM	Positive/Negative	85.74%
	Bi-LSTM	Positive/Negative	86.56%
Proposed model	Bi-LSTM	Positive/Negative	91.4%

Acknowledgements

The authors express their deepest gratitude toward the Principal, the Head of Department, Electronics and Communication Engineering and the staff of Sri Jayachamarajendra College of Engineering, JSS STU, Mysore, Karnataka, India, for their continuous support and timely cooperation throughout the project period.

References

[1] T. Patten and P. Jacobs, “Natural-language processing,” IEEE Expert, vol. 9, no. 1,p. 35, 1994.

[2] J. Eisenstein, Introduction to natural language processing. MIT press, 2019.

[3] V. Raina and S. Krishnamurthy, “Natural language processing,” in Building an Effective Data Science Practice, pp. 63–73, Springer, 2022.

[4] Goldberg, Yoav. "Neural network methods for natural language processing." Synthesis lectures on human language technologies 10.1 (2017): 1-309.

[5] Kaur, Manjot, and AakashMohta. "A review of deep learning with recurrent neural network."2019 International Conference on Smart Systems and Inventive Technology (ICSSIT).IEEE, 2019.

[6] Haykin, S. (2009). Neural networks and learning machines, volume 3. Pearson Education

[7] Parhi, Rahul, and Robert D. Nowak. "The role of neural network activation functions." IEEE Signal Processing Letters 27 (2020): 1779-1783.

[8] Thanaki, Jalaj. Machine learning solutions: expert techniques to tackle complex machine learning problems using python. Packt Publishing Ltd, 2018.

[9] Guo, Jiang. "Backpropagation through time."Unpubl.ms., Harbin Institute of Technology 40 (2013): 1-6.

[10] Olah, Christopher. "Understanding lstm networks." 2015, https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[11] Yu, Yong, et al. "A review of recurrent neural networks: LSTM cells and network architectures." Neural computation 31.7 (2019): 1235-1270.

[12] Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks." IEEE transactions on Signal Processing 45.11 (1997): 2673-2681.

[13] Berglund, Mathias, et al. "Bidirectional recurrent neural networks as generative models." Advances in neural information processing systems 28 (2015).

[14]https://s3.amazonaws.com/amazon-reviews-pds/readme.html

[15] Haddi, Emma, Xiaohui Liu, and Yong Shi. "The role of text pre-processing in sentiment analysis." Procedia computer science 17 (2013): 26-32.

[16] Noaman, Hatem M., Shahenda S. Sarhan, and Mohsen Rashwan. "Enhancing recurrent neural network-based language models by word tokenization." Human-centric Computing and Information Sciences 8.1 (2018): 1-13.

[17].Pradha, Saurav, Malka N. Halgamuge, and Nguyen Tran Quoc Vinh. "Effective text data preprocessing technique for sentiment analysis in social media data."2019 11th international conference on knowledge and systems engineering (KSE).IEEE, 2019.

[18] https://www.tensorflow.org/datasets/api_docs/python/tfds/deprecated/text/Tokenizer

[19] Millstein, Frank. Natural language processing with python: natural language processing using NLTK. Frank Millstein, 2020.

[20] https://www.tensorflow.org/datasets/api_docs/python/tfds/deprecated/text/TokenTextEncoder

[21] Düntsch, Ivo, and GüntherGediga. "Confusion matrices and rough set data analysis" Journal of Physics: Conference Series. Vol. 1229.No. 1.IOP Publishing, 2019.

[22] Z. Jianqiang, G. Xiaolin, and Z. Xuejun (2018), “Deep convolution neural networks for twitter sentiment analysis,” IEEE Access, vol. 6, pp. 23253–23260.

[23] M.-H. Su, C.-H.Wu, K.-Y.Huang, and Q.-B. Hong (2018), “Lstm-based text emotion recognition using semantic and emotional word vectors,” First Asian Conference on Affective Computing and Intelligent Interaction (ACII Asia), pp. 1–6, IEEE.

[24] K. Liu and L. Chen (2019), "Medical Social Media Text Classification Integrating Consumer Health Terminology," IEEE Access, vol. 7, pp. 78185-7819378185-78193