Comparative Analysis of Sentiment Classification using LSTM and BERT Model

MS Student
Department of Computing,
Islamabad, Pakistan

HoD Research, PhD
Department of Computing,
Islamabad, Pakistan

Abstract **– The aim of the paper is to compare the LSTM and BERT Model for the NLP task called Sentiment Classification. The RNNs model suffer from vanishing and exploding gradients and are unable to capture long-term dependencies. This can sometimes lead to false classification of text. To overcome this issue of RNNs, the transformer models were introduced which uses the attention mechanism to learn the context of the sentence over a larger window of dependency. The paper provides an overview of various deep learning techniques like Sentiment Analysis. BERT (Bidirectional Encoder Representations from Transformers) is a kind of transformer architecture. It can capture the context both from past and future word sequences. It uses a heuristic approach to train the model and learn the whole context of the dataset and generate results based on that learning. Our paper contains 2 variations of BERT model in contrast with the LSTM model. The BERT models outperformed the LSTM model with an accuracy increase of around 5-6%.**

Keywords – Sentiment Analysis, BERT, Transformer, LSTMs, RNNs, Long-term dependencies, Attention mechanism, IMDB dataset.

I. INTRODUCTION

Natural Language Processing is one the hot topics of data science domain. Sentiment classification is an important task of NLP. The method of evaluating whether any sequence of text is positive, negative, or neutral is Sentiment Classification. A text sentiment analysis system incorporates the machine learning techniques to classify the persons, subjects, themes and categories within a sentence or phrase to weighted sentiment scores. Sentiment analysis helps large business data analysts gauge the public sentiment, analyze the complex market, trace out the credibility of brands and goods, and analyze the consumer experiences. Sentiment Analysis can be used to extract useful information about any domain. Using various approaches, this classification task can be carried out to achieve a different degree of accuracy. A comparative study of different sentimental analysis techniques with their output evaluation will be discussed in this paper. Current techniques used for Sentiment Analysis include variations of RNNs like LSTMs and GRUs. These RNNs are difficult to train and suffer from exploding/vanishing gradients problem. These gradient values either become this small that learning stops, or the gradient values grow exponentially that the maximum limit is exceeded by the weights, in both scenarios learning becomes difficult. Introducing a gating mechanism in the RNN is an efficient solution to this issue. The GRU manages the flow of information, but without needing to use a memory unit, like the LSTM unit. Accuracy on sentiment analysis can be increased by using deep learning techniques which can overcome the long-term dependency issue of RNNs. So we need a mechanism which can capture long-term dependencies even at paragraph level. For this, we need to introduce an Attention mechanism in our model. One such model is Transformer model.

Transformer model is used to overcome the long-term dependency issue of RNNs. It is a pre-trained and a bidirectional model which means it can remember text sequences from future to past, past to future or both combined. This pre-trained model can be easily tuned to perform tasks like Text Summarization, Sentiment Analysis, etc. as it includes the heuristic approach to train the model and learn the whole context of the dataset and generate results based on that learning. Models like transformers, LSTMs, GRU's and neural networks can be used to give promise results in the task of sentiment analysis which may lead to the state-of-the-art results. A comparative evaluation of LSTM model and a Transformer model using IMDB's movie review dataset will be discussed in this paper.

II. LITERATURE REVIEW

In [1], I. Kaibi, E. H. Nfaoui, and H. Satori used the binary classification approach for the task of sentiment analysis. The researchers have focused on the three most popular word embeddings techniques for sentiment analysis, which includes the Fasttext, Glove, and Word2vec, on the Twitter dataset. The algorithms understudy are NuSVC, LinearSVC, Logistic Regression, Random Forest, GaussainNB, and SGD. The better results in terms of accuracy is of Fasttext model with 84.89% accuracy value, NuSVC classifier with 84.29% accuracy value, and the average of tweet word vectors with 83.85% accuracy value.

The classical bag-of-words model have few deficiencies (like due to the large scale of vocabulary, bag of words contributes to a high dimensional function vector) and affects the accuracy of sentiment classifications. In this paper [2], the researchers improved the accuracy of the classification of sentiment by using the technique of Word Embeddings. The Skip-Gram Model is used whereas the underlying approach is the Word2Vec to create high-dimensional word vectors that learn words from contextual information. To evaluate the polarities of the tweets random forest classifier was used. An overall accuracy of 81% indicates that the classifier did well to predict the polarities of sentiment due to the consistency of word vectors generated by the model of the skip-gram.

In paper [3], Rincy J. and Varghese S.C. adopted a lexicon-based sentiment analysis approach that exploits sense meanings. This paper uses SentiWordNet and WordNet lexical tools alongside Word Sense Disambiguation. Sentiment classification is performed using WordNet and SentiWordNet on Twitter info. This technique gave an accuracy of 78.6%.

R. Jose and V. S. Chooralil, in their paper [4], the researchers have combined the outcomes of a series of classifiers in order to decrease the chance of selecting an incorrect classifier. The overall approach was to perform sentiment classification using Hidden-Markov Model, Naive Bayes, and SentiWord-Net, and lastly classification using an ensemble approach. Using the ensemble gave an accuracy of 71.48%.

In this paper [5], Das, Bijoyan & Chakraborty, Sarit proposed a method for classifying text sentiment using the Next Word Negation along with the Term Frequency-Inverse Document Frequency. The researchers have also compared the performance of the TF-IDF, and BOW model with the 'next word negation' text classification model. The model was trained for 3 most popular classification algorithms; Multinomial Naïve Bayes, Linear SVM, and Max Entropy Random Forest. The accuracy of all the algorithms is Linear SVM was highest among all.

In paper [6], the researchers have applied a Machine Learning Tool to evaluate the polarity of Twitter messages. This research used the Recursive Neural Tensor Network to (RNTN) to arrange all terms using a binary tree. The trained RNTN model was applied to classify the dataset. The researchers used the original RNTN model to create a baseline, which had not been trained by their manually annotated data.

In [7], the various techniques used for sentiment analysis are lexicon-based approach, machine learning based approaches, word embeddings, CNNs, RNNs, and LSTMs. The RNN model gave an overall efficiency of 87.53%. The recurrent model of the neural network generates output on the basis of using sequential knowledge in previous computations. This can an efficient approach for any sentiment classification problem, but there is still an issue of long-term dependencies.

In paper [8], B. N. Saha, A. Senapati and A. Mahajan used a modified version of RNN known as LSTM for sentiment analysis of election data. This approach is then compared to the supervised classifiers including naive bayes, and SVM. For classification task, the optimizer named ADAM was used. Results have shown that the deep RNN architecture based on LSTM outperforms all other classifiers discussed.

RNNs are difficult to train and suffer from vanishing and exploding gradients problem. The deep learning models like transformers, LSTMs, GRU's and neural networks can be used to give promise results in the task of sentiment analysis which may lead to the state-of-the-art results.

III. METHODOLOGY

Our paper contains a comparative evaluation of LSTM model and a BERT model with 2 variations having different number of attention heads, hidden layers, and hidden size.

A. The LSTM architecture

To avoid the long-term dependence problem, LSTMs were introduced. The main goal is to recall knowledge for long periods of time. The LSTMs were designed to capture the data of the sequential / time series.

LSTMs architecture have a chain like structure with a repeating module. There are 4 neural network layers, each interacting with one another in a meaningful fashion. The main component in LSTM architecture is the cell state, which is the horizontal belt running across the top of the cell (shown in figure 1). This cell state is a conveyor belt. It connects the entire module chain with just some small linear interactions with other components of the cell. The information can flow unchanged very easily along this belt.

The discussed LSTM architecture has the capacity to store or forget the information to the cell state, each controlled via structures known as gates. These gates allows the data to move through, forgetting the redundant information and preserving the important information. They are composed of a layer of sigmoid function and a multiplication operation pointwise.

In this variation of RNN, we multiply the weight attached with the previous state's input and the weight attached with the previous state's output. And then, to get the new state, we pass them to the Tanh function (also called the squashing function, used to evaluate candidate values to be added to the internal state of LSTM cell unit). In order to obtain the output vector, we multiply the new state with the output of the Tanh function.

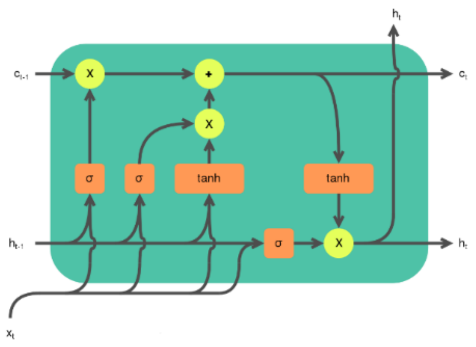

Figure 1. ***Single LSTM cell unit***

B. The BERT architecture

The BERT model was introduced to pre-train deep bidirectional representations from the unlabeled text by conditioning all layers together on both the left (past) and right (future) context. As a result, with an additional output layer (as per problem statement), the pre-trained BERT model can be fine-tuned for a wide variety of tasks including Sentiment Analysis.

The BERT model simultaneously considers every word of the input sentence and develops a contextual meaning of these words using an Attention mechanism (allows to remember text sequences and features of the input in our model hence capturing the long-term dependencies without any window size limitation). This attention mechanism was first introduced in the paper [10] 'Attention is all you need'. In deep learning, the attention mechanism is based on this notion of focusing the on specific words in the context, so that when processing the data, the model pays greater attention to certain word sequences.

The BERT model architecture is an encoder-decoder structure (as shown in figure 2) for multi-layer bidirectional transformers.

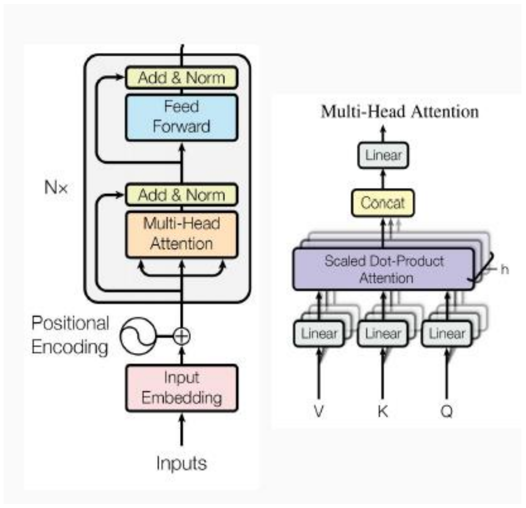

Figure 2. ***BERT model architecture***

The encoder part consists of a stack of identical layers. There are two sub layers in each layer. The first layer is a mechanism of multi-head self-attention, and the second is a fully connected feed-forward network that is position wise. There is a residual connection followed by layer normalization around each of the two sub layers.

The attention mechanism used by the transformer takes 3 input matrixes that are Query (Q), Key (K), Value (V). Attention scores are evaluated by dot product of the hidden states if encoder and decoder. The attention of the dot-product is scaled by a factor of square root of depth. The multiplication of the attention weights with V vector helps concentrate on words that are to be focused and the words that are meaningless are washed out. The transformer model follows as mechanism as sequence to sequence with attention model. For each word/token in the sequence, the input sentence is passed through N encoder layers that produce an output.

For sentiment analysis, only this encoder is used. Since the encoder portion of the transformer is used for a model that reads input sentence and produces some features that can be used for different NLP tasks.

IV. DATASET

The IMDB's movie review dataset compiled and prepared by Andrew L. Maas from the popular film rating service, IMDB. The IMDB reviews dataset uses a binary sentiment classification as per text polarity (whether it positive or negative). It includes 25,000 training movie reviews and 25,000 for testing. All of these 50,000 reviews are labelled data that can be used for deep learning techniques.

V. EXPERIMENTAL SETUP

The hyperparameters of the model, layers. and the workflow of both the LSTM and BERT will be discussed in this section.

A. The LSTM architecture

Firstly, the data is preprocessed (removing stop words, URLs, and special characters) and fed into the network. The first layer is the embedding layer. The embedding layer helps us to translate each word into a fixed size vector of a fixed length. It is dense vector containing real numbers. This embedding layer functions like a lookup table. In this table, the terms are the keys, while the dense word vectors are the values. The input embedding size used is 20000 (same as the number of maximum features to be used for training) and the output embedding size equal to 128. Next comes the LSTM layer, which generates a series output (means it capture the dependencies) rather than a single value output. For all input time steps one output per input time step, rather than one output time step. The output dimension of this layer is 128, with a dropout rate (to drop out linear transformation of the inputs) equal to 0.2 and a recurrent rate (prevents the over fitting of the data) equal to 0.2. The last layer of the model is the dense layer (containing the neurons), with an output dimensionality of 1 and a Sigmoid activation function. This dense layer performs a matrix-vector multiplication for output generation. These values are trainable parameters which are updated during backpropagation using the loss function to find the optimized weights for our network.

The optimizer used for classification is ‘Adam’ and the loss function used is ‘binary crossentropy’. The evaluation metric used is the Accuracy metric. Other specifications include:

Table 1. ***Hyperparameters of LSTM model***

Max_features	20000
Maximum length for the input	80
Batch size	32
Number of epochs for training dataset	07

A high-level design of the LSTM model used is given below:

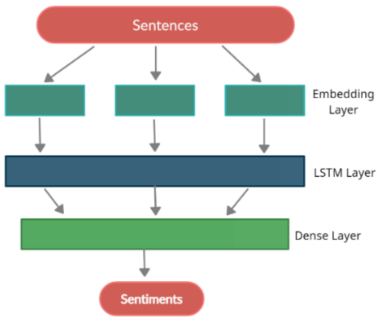

Figure 3. ***LSTM high-level workflow***

B. The BERT architecture

After importing the dataset from the Internet movie database, the preprocessing of the dataset is done. The imported data is already divided into training and test split. We then add 80:20 split on training data to obtain the validation set. The random seed value is passed to this split in order to ensure that there is no overlap in the training and validation split.

Before being inserted into BERT, text input must be converted into numeric token ids. BERT takes token embeddings, segment embeddings, positional embeddings, and mask tokens as input. In the input sentence, the token embeddings are numerical representations of words. There is also something called tokenization of sub-words that BERT uses to break down larger or complicated words into simple words first and then turn them into tokens. The segment embeddings are used in a single input to help BERT distinguish between the various sentences. For words from the same sentence, the components of this embedding vector are all the same and the meaning changes if the sentence is different. The mask tokens allow BERT to understand what is important to all input words and which ones are there only for padding. Finally, there are position embeddings in BERT that are generated internally and provide a sense of order for the input data. These values are fed into the feed forward network containing the dense layer and a dropout layer. A look ahead mask is used for ignoring the words that should not be considered for prediction.

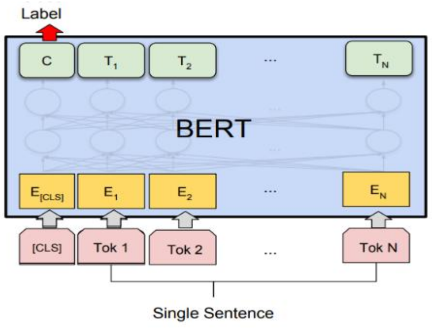

Figure 4. ***BERT high-level workflow***

There are multiple BERT models that can be used, they differ from each other in terms of number of attention heads, hidden size (output of each layer), and hidden layers. All these models are pre-trained on the BooksCorpus and Wikipedia. Our model contains the BERT encoder, a single dense layer, and a dropout layer.

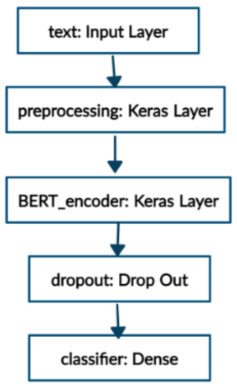

Figure 5. ***Layers used in BERT model***

The BERT models uses three important keys: pooled output (to represent each input sequence as a whole), sequence output (represents each input token in the context) and encoder outputs (are the intermediate activations of the transformer blocks). The dropout rate used at the dropout layer is 0.4 to avoid overfitting of data.

A classifier is then applied to predict the sentiments of the text using the learned context of the input sentences. Other important details are as follows:

Table 2. ***Hyperparameters of BERT model***

Maximum length for the input	128
Batch size	32
Number of epochs for training dataset	07

The Adamw optimizer basically does regularization by weight decay and reduces prediction loss. The learning rate is initialized with a value of 3e-5. Since, sentiment analysis is a binary classification problem, the loss function used is the binary crossentropy. The output layer generates a single value which can be termed as a positive sentiment or a negative sentiment. The evaluation metric used is the Accuracy metric using the Binary Accuracy function.

For this paper, two BERT models were trained to compare the results of both model in terms of effect of number of attention heads, hidden size, and hidden layers on the accuracy value.

Table 3. ***Details of BERT models***

Model 1 (small_bert/bert_en_uncased_L-4_H-512_A-8)	
Attention heads_A	8
Hidden layers_L	4
Hidden size_H	512

Model 2 (small_bert/bert_en_uncased_L-4_H-768_A-12)	
Attention heads_A	12
Hidden layers_L	4
Hidden size_H	768

The results of the models in comparison with the LSTM model will be discussed in the discussion section of the paper.

VI. DISCUSSION AND RESULTS

The training process of all three model uses the Adamw optimizer and binary cross entropy to train the weights. This process reduces the loss value hence attaining training weights for the model. We set accuracy as the metric for measuring model's performance.

A. Discussion of LSTM results

The LSTM model was used to predict the labels for the test set and gave an accuracy of **81.42%.** Following is the table of accuracy values of this model.

Table 4. ***Results of LSTM model***

Validation accuracy	81.33%
Test accuracy	81.42%
Loss value	0.76

The graphs for model accuracy and loss value is given below:

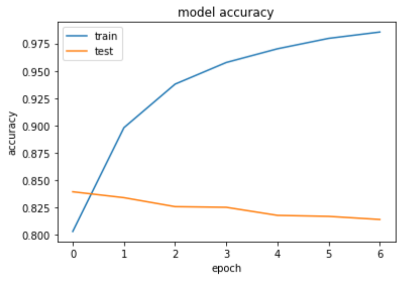

Figure 6. ***Accuracy curve of LSTM model***

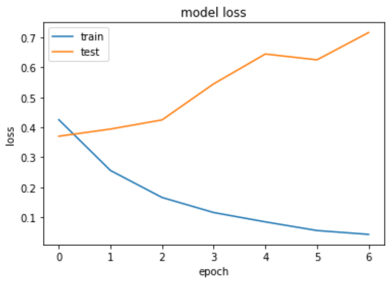

Figure 7. ***Loss value curve of LSTM model***

B. Discussion of BERT Model 1 results

The BERT model having 8 attention heads, 512 hidden size and 4 hidden layers gave a test accuracy of **85.23%.**

Table 5. ***Results of BERT model 1***

Validation accuracy	84.88%

Test accuracy	85.23%
Loss value	0.614

The graphs for model accuracy and loss value is given below.

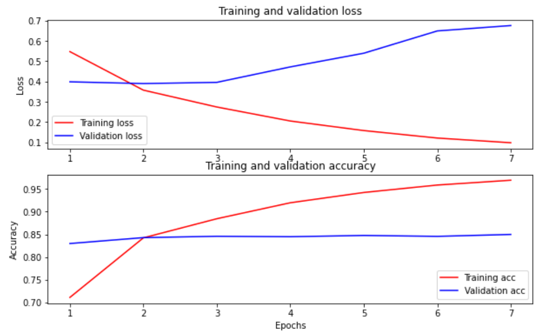

Figure 8. ***Accuracy and loss value curve of BERT model 1***

C. Discussion of BERT Model 2 results

The BERT model having 12 attention heads, 768 hidden size and 4 hidden layers gave a test accuracy of **86.35%.**

Table 6. ***Results of BERT model 2***

Validation accuracy	85.98
Test accuracy	86.35%
Loss value	0.73

The graphs for model accuracy and loss value is given below.

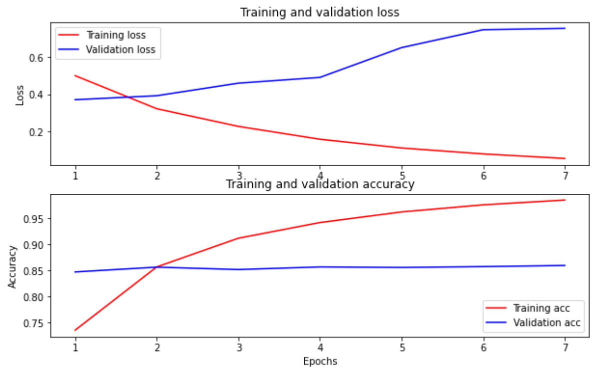

Figure 9. ***Accuracy and loss value curve of BERT model 2***

LSTM models are capable of learning order dependence in sequence prediction problems but still its window size for learning these dependencies is limited. It cannot learn the context of sentences at a paragraph level. The BERT (transformer model) overcomes this issue of long-term dependencies using the attention mechanism. The greater the number of attention heads used, the better would be the performance of the model. As clearly seen from our experiment results mentioned in the previous section, the model with attention_heads = 8 gave an accuracy of 85.23%, whereas the model with attention_heads = 12 gave an accuracy of 86.35%. Increased number of heads leads to parallelization, this means multiple attention heads allows for attending to parts of the sequence differently. As each word (each position in the input sequence) is processed by the model, attention enables it to search for clues at other positions in the input sequence that can help lead to better encoding for this word.

Transformers are better than LSTMs due to the presence of self-attention, multi-head attention mechanisms and positional embeddings in their architecture. They prevent recursion completely by processing sentences as a whole and by learning relationships between each input sentence.

VII. CONCLUSION

From the results of our experiment, it is evident that the BERT, a transformer model, outperformed the LSTM model. As transformer models are attention-based models, unlike LSTMs where the sentence is sequentially processed - one word per time stage, transformers see the entire sentence as a whole. LSTMs need to propagate the error back in time by one word at a time during training. On the contrary, the transformer sees all words/sentences simultaneously - so there is no backpropagation across time. The main benefit of transformer model is that they are not sequential, which means that they can be more easily parallelized, unlike LSTMs, and that by parallelizing the training, larger and larger models can be trained. Hence, transformer models can be used for various other text classification tasks for better accuracies.

REFERENCES

[1] Ibrahim K. , El Habib N., and Hassan S., "A Comparative Evaluation of Word Embeddings Techniques for Twitter Sentiment Analysis", (2019), International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS.).

[2] Oscar B.Deho , William A. A, Jeferry A.A , Felix L.A ,"Sentiment Analysis with Word Embedding", (2019).

[3] Rincy Jose, Varghese S.C. ,"Predication of Election Result by Enhanced Sentiment Analysis on Twitter Data using Word Sense Disambiguation", (2015), International Conference on Control Communication & Computing India.

[4] Rincy Jose, Varghese S.C. ,"Predication of Election Result by Enhanced Sentiment Analysis on Twitter Data using Classifier Ensemble Approach", (2016), International Conference on Data Mining and Advanced Computing.

[5] Bijoyan D., Sarit Chakraborty, "An Improved Text Sentiment Classification Model Using TF-IDF and Next Word Negation", IEEE, (2018).

[6] Meng-Hsiu T. , Yingfeng W. , Myungjae K. , Neil Rigole., "A Machine Learning Based Strategy for Election Result Predication" , Middle Georgia State University, (2019), International Conference on Computational Science and Computational Intelligence (CSCI).

[7] Patel, Alpna and Tiwari, Arvind Kumar, Sentiment Analysis by using Recurrent Neural Network (February 8, 2019). Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE) 2019.

[8] Baidya N.S, Apurbalal S. , Amnol M. ,"LSTM based Deep RNN Architecture for Election Sentiment Analysis From Bengali Newspaper", (2020), International Conference on Computational Performance Evaluation (ComPE).

[9] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I. “Attention is All you Need”, (2017)