Sentiment Analysis of Short Informal Text by Tuning BERT - Bi-LSTM Model

Shreyas Agrawal
Dept. of Information Technology
Pune Institute of Computer Technology
Pune, India
shreyasagrawal2000@gmail.com

Sumanto Dutta
Dept. of Comp. Science and Eng.
Rourkela National Institute of Technology
Rourkela, India
sumanto.nitrkl@gmail.com

Bidyut Kr. Patra
Dept. of Comp. Science and Eng.
Rourkela National Institute of Technology
Rourkela, India
patrabk@nitrkl.ac.in

***Abstract*—Sentiment analysis is one of the significant tasks in processing natural language by a machine. However, it is difficult for a machine to understand the feelings of a person and opinion about a topic. Many approaches have been introduced for analyzing sentiment from long text in recent past. In contrast, these approaches fail to address the small length text problem like Twitter data efficiently. Recent advances in the pre-trained contextualized embeddings like Bidirectional Encoder Representations from Transformers (BERT) show far greater accuracy than traditional embeddings. In this paper, we develop a novel architecture to tune the BERT using a Bidirectional Long Short-Term Memory (Bi-LSTM) model. A task-specific layer is incorporated along with the BERT in the proposed model. Our model extracts sentiment from short texts, especially Twitter data. The extensive experiments show the superiority of our model over state-of-the-art models in sentiment analysis task across several gold standard datasets.**

Keywords*—Sentiment Analysis (SA), Contextual Embeddings, Bidirectional Encoder Representations from Transformers (BERT), Task-Specific Layer, Bidirectional Long Short-Term Memory (Bi-LSTM)

I. Introduction

Language is the most significant form of communication in the modern era. It consists of several words, syntaxes, contexts and expressions. However, it is a challenging task to analyze vast amounts of linguistic data. It becomes easier if a machine could do all the work; however, understanding human language is a difficult task for a machine. The natural language processing (NLP) plays a vital role to address the various challenges related to understanding the human language. NLP is the procedure of programming machines to process and analyzes vast amounts of natural language data. NLP has various tasks like text summarization, named entity recognition, relationship extraction, language translation, and sentiment analysis (SA). The SA is always a critical and challenging task for a machine to interpret human feelings and specific opinions about a topic. Primarily, it is started for analyzing sentiments in documents or feedback forms [1], [2]. However, in the current scenario, it is widely used to mine information from various social media websites, blogs, news, reviews, and comments like Facebook, IMDB, Instagram and Twitter [3]. Twitter is one such micro-blogging website where around 500 million (https://www.statista.com/) tweets are posted every day. It is one of the most widely used platforms to express human feelings. The most common length of a tweet is around 61 characters [4], and only 1% of all the tweets are above 140 characters. Nevertheless, it is challenging to get the sentiment from shorter length texts [5], [6]. Various methods are introduced for the SA task on social media data. These are broadly classified into two groups: aspect-based SA and text level SA. In the aspect-based SA, sentiments of a particular aspect of an entity are determined. In the text level, the overall sentiment of a sentence is determined.

In this paper, we propose a novel method of text-level SA using the deep learning approach, mainly targeting Twitter data. It handles three classes of sentiments- positive, negative, and neutral. We use context-aware, pre-trained embedding BERT (Devlin et al. [7]) to get context-aware word embedding for each tweet. We add a task-specific layer over BERT to achieve better sentiment from tweets. Our model outperforms the previous state-of-the-art methods in SA across several gold standard datasets. Our contributions in this paper are, we apply intelligent contextualized embedder BERT as an embedding method over traditional embedding methods. A novel textual feature extraction model using Bi-LSTM is proposed under task specific layer for text level Twitter SA. Effectiveness of the proposed method is established across different gold standard datasets over the existing state-of-the-art methods for Twitter SA.

The paper is organized as follows. In section II, we describe the related works in the SA. Section III gives a detailed description of the proposed model. In Section IV, we expand on the details of our experiment with parameter tuning and testing overfitting condition. In section V, we compare with the previous state-of-the-art models and present our results. Finally, in section VI, we conclude with future work.

II. Related Works

In this section, we perform a literature review of previous studies related to different SA approaches.

In the lexical based approaches, Esuli et al. [8] propose a method that uses a set of ternary classifiers to classify the polarity of each words in Synset database. Baccianella et al. [9] propose an improved method of Esuli et al. [8]. They perform a semi-supervised learning process and the random-walk process for the annotation of the words into positive, negative and neutral forms. Zhang et al. [10] use a dictionary of opinion lexicons to determine the sentiment of individual words. SVM classifier is used to get the final computation of the probabilities. Jurek et al. [11] propose a way of determining the sentiment by giving the text a sentiment range of -100 to +100. However, their work is limited to the long text.

Dehkharghani et al. [12] show the use of SentiTurkNet. It is based on Turkish lexicons that assign three polarity scores to each Synset in the Turkish WordNet. Their work lacks the use of dependency parse trees and covering negation in Turkish gloss. Zabha et al. [13], propose a cross-lingual sentiment analysis method combining two lexicons of English and Malay languages. However, their work lacks in analyzing dialect, detecting short and slang words. Overall, all these methods are highly word-dependent.

Dos et al. [14] propose a Deep Convolutional Neural Networks (DCNN) to perform sentiment analysis. However, their work lacks in performing character level representations with domain-specific unsupervised pre-training. Michael Cai [15] propose a very-deep convolutional neural networks (VDCNNs) to classify tweets in the Sentiment140 dataset. Cai compare VDCNN with simple BERT without any task specific layer. Baziotis et al. [16] present a 2-layer siamese Bi-LSTM model, with an attention mechanism over a base of GloVe embeddings. Their results shows poor accuracy for ambiguous word. Various sentiment classifiers using BERT as base embedding layer are proposed. These models are solely dependent upon the embedding layer for the results, thereby lacking a task-specific layer (Hoang et al. [17] Munikar et al. [18], Adhikari et al. [19], Lee et al. [20]).

III. Proposed Model

We propose a model to extract sentiments from Twitter data. In an amalgamation of BERT and Bi-LSTM in our model, BERT helps to extract contextualize intrinsic feature of the word, whereas Bi-LSTM helps to get the knowledge of local as well as global context of a sentence. Our model addresses three types of sentiments, such as positive, negative, and neutral. The Fig. 1 illustrates our proposed model. The proposed model has three main modules, namely preprocessing, BERT as a contextualized embedding layer, and task-specific layer which are explained below.

A. Preprocessing

Let $T = \{t_1, t_2, t_3, ..., t_n\}$ be the set of tweets under consideration, where each tweet $t_i = \{w_1, w_2, w_3, ..., w_m\}$ where w_i is either a word or emoji and in our model, $m = 55$. The tweets are preprocessed using standard techniques such as removal of stopwords, URLs, handles, hashtags, and numbers . Emot algorithm [21] is used to convert emojis and emoticons to their respective words by preserving sentiment information. We also use Ekphrasis spell corrector [22] for the spell correction and lowercase all the tweets. BertTokenizer is used to tokenize all the $t_i's$ into size of $[m \times H]$, where H is hidden size of BertTokenizer i.e. dimensionality of a hidden layer. In our case, $H = 768$ neurons in each hidden layer before passing it to BERT.

B. BERT as an embedding layer

BERT is a bidirectional transformer [23] based language model that uses encoders to contextualize the embeddings. The contextualized embedding helps to add syntax information and address polysemy in a model. Our implementation of contextualized embeddings through BERT is built on BERT$_{base}$ model using PyTorch's pre-trained BERT library. It contains 12 transformer blocks (L), 768 as hidden size (H) i.e. neurons in a hidden layer, and 12 self-attention heads (A). In our model, the first token is always a special classification token ([CLS]) and sentence separation is denoted by ([SEP]) token. The input tokens $[m \times H]$ are passed through three main embedding layers - token embedding, segment embedding, and position embedding. These layers allow our model to learn the embeddings based on the surrounding words rather than just giving a specific constant vector to a word. It gives the right amount of generalized data to get our contextualized vector sequences. A contextualize embedding, E is obtained by combining the 12 layers to make one vector of size $[mH], \forall t_i, i = 1, 2...n$. E is then passed to Bi-LSTM layer for task specific computation.

C. Task specific layer

Task specific layer plays an important role which performs a specific job such as sentence pair classification, text prediction, text summarization, sentiment analysis, name-entity recognition, question answering task utilizing pre-trained model or components in NLP. In our paper, we are performing SA of Twitter data. Our task specific layer contains four modules, namely textual feature extraction module, max-pooling layer, dense layer and sigmoid activation function. Textual feature extraction module is one of the vital parts of our proposed model and uses a deep neural network-based approach to extract textual feature vector from tweets. Previously various types of deep neural network based models such as Convolution neural network (CNN), RNN and LSTM are used for textual feature extraction. However, they suffer from various problems such as gradient disappearance or gradient explosion, forgets short or long term information and are inefficient to extract the contextual meaning of a sentence. We propose a novel textual feature extraction model using Bi-LSTM which addresses problems of previous models. It takes into account post word information. Let us consider two tweets having text ”Max went to bush for night camping” and ”Bush attacks Iraq today !!!”. Our Bi-LSTM model successfully distinguishes different context of the word ”bush”. However other methods overlook the context of word ”bush” [16], [24], [25]. The embedding E from BERT is given as input to task specific layer where firstly it is passed to Bi-LSTM for textual feature extraction. Forward and backward

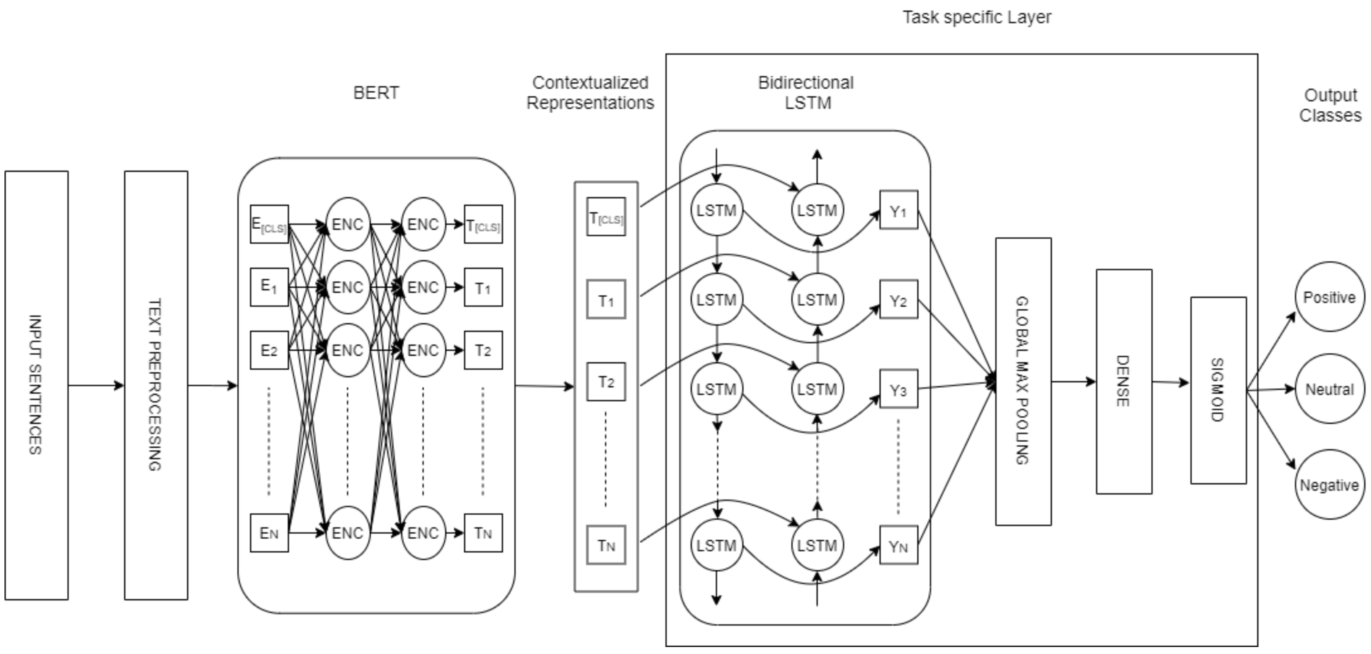

Fig. 1. Overview of proposed model

hidden states of Bi-LSTM, $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ are used to compute the information in the forward as well as in the backward direction, respectively. The results of $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ are stacked together to generate a textual feature of size $[m \times u]$ where u is the dimension of Bi-LSTM. We consider the dimension of LSTM as 128 which gives the output dimension of $u = 256$ (LSTM being bidirectional). The output is obtained by using $\hat{y_t} = g(W_t[\overrightarrow{h_t}, \overleftarrow{h_t}] + b_y)$, where W_t is weight parameter, b_y is bias and g is activation function. As LSTM is a building block in our Bi-LSTM therefore, calculation of our hidden states are performed in the same way as that of the LSTM hidden state using $c_t = f_t \cdot c_{t-1} + i_t \cdot tanh(W_c x_t + U_c h_{t-1} + b_c)$ and $h_t = o_t \cdot tanh(c_t)$. These textual feature vectors are passed to max pooling layer, dense layer and then sigmoid activation function to compute class probabilities.

In section IV, we present the details of our experimental procedures.

IV. Experiment

In this section we describe the datasets and experimental methodologies used to evaluate proposed tuned model.

A. Dataset distribution

We consider SemEval 2014, 2015, 2016, 2017 (Rosenthal et al. [26], Rosenthal et al. [27] Nakov et al. [28], Rosenthal et al. [29]) as the baseline datasets for Twitter sentiment evaluation . We compare our results with different state-of-the-art methods on these baseline datasets. The dataset statistics are described in Table I. It is clearly noticeable that the dataset is imbalanced and handling imbalanced dataset is important for better performance of the model.

TABLE I
Dataset statistics of SemEval 2014, 2015, 2016, 2017.

Year	Positive	Negative	Neutral	Total
SemEval 2014	7,743	3,347	9,279	20,369
SemEval 2015	8,783	3,712	10,266	22,761
SemEval 2016	13,715	6,471	12,575	32,761
SemEval 2017	19,902	7,840	22,591	50,333

1) Handling imbalanced dataset and data leakage problem: We find that our model underperforms without balancing the dataset. To make our dataset balanced and more robust, we use oversampling strategy. Other well know techniques such as data augmentation and undersampling can not be employed due to their limitation on textual data. In oversampling strategy, texts from the negative class are chosen randomly and duplicated to match with positive and neutral class. However, applying the oversampling strategy over the whole dataset may arise a data leakage problem where train and test sets may contain the same data point, even though they may correspond to different observations. We handle the issue by splitting the dataset into train and test sets before applying the oversampling strategy.

B. Training

We use *bert-base-uncased model* (L = 12, H = 768, A = 12) from PyTorch. The LSTM dimension is 128, which gives 2x128 = 256 as the output dimension (LSTM being bidirectional). As LSTM reads sentences word by word, the representations of these sentences in the matrix are independent. Hence we treat this matrix as an image. A one-dimensional global max-pooling layer is used, which extracts the maximum value vector from the matrix. The fixed vector is passed through a sigmoid layer to get the final classification. Dropout layers are used with a probability of 60% after each layer. We use Adam optimizer (Kigma et al. [30]) to minimize the loss with a learning rate of 0.001. The model is implemented using Tensorflow backend, and the experiments are performed on GeForce RTX 2080 Ti GPU at Intelligent Data Analysis Lab, Department of Computer Science and Engineering, National Institute of Technology, Rourkela.

C. *Hyperparameter Tuning*

Hyperparameters are the values which affect the training and accuracies of a model. Tuning of these hyperparameters helps in better training of the model resulting in better accuracies. We tune the following hyperparameters to improve our model's performance. The experiments are carried out on SemEval 2017 task 5 Gold Standard dataset.

1) Dropouts: Dropout is an important factor to consider when deciding a model structure. Dropout reduces the dependencies in the neural network. This helps in reducing overfitting of the model. Four different dropouts, 0.5, 0.55, 0.6, 0.65 are used for comparison. We get the best results with 0.6 as the dropout rate. Hence this dropout is fixed for further comparisons. The comparison table is shown in Table II.

2) Activation Functions: The activation functions are used to add some non linear property to neural network. We compare our model with several activation functions namely, softmax, relu, sigmoid and tanh. In this comparison, the sigmoid function gives the best results. Hence, sigmoid is fixed and used for further comparisons. The comparison table is shown in Table II.

3) Learning Rates: Learning rate helps in minimizing and optimizing the loss function. It ranges between 0 and 1. We compare our results with different learning rates like 0.002, 0.001, 0.0001, and 0.00001. The results are more accurate with a learning rate of 0.001 hence we fix this learning rate for our model. The comparison table is shown in Table II.

TABLE II
HYPERPARAMETER TUNING RESULT.

Hyperparameter	Parameter Values			
Dropout	0.50	0.55	**0.60**	0.65
F1/ Recall scores	59/56	62/60	**64/61**	63/61
Activation Funtion	softmax	relu	**sigmoid**	tanh
F1/ Recall scores	60/74	63/68	**63/72**	22/17
Learning Rate	0.002	**0.001**	0.0001	0.00001
F1/ Recall scores	64/62	**65/62**	57/54	23/34

D. *Checking overfitting issue*

We train our model with each dataset for 1000 epochs to check for overfitting. Fig. 2 shows the training and validation accuracies of SemEval 2014 dataset over 1000 epochs. Similar results are obtained for each dataset. Hence from the graphs, it is noticeable that the proposed model does not overfit, even for a large epoch size with the selected hyperparameters. This shows the superiority of BERT and robustness of our model to address the overfitting.

Section V, describes the comparisons with the current models accompanied with the results.

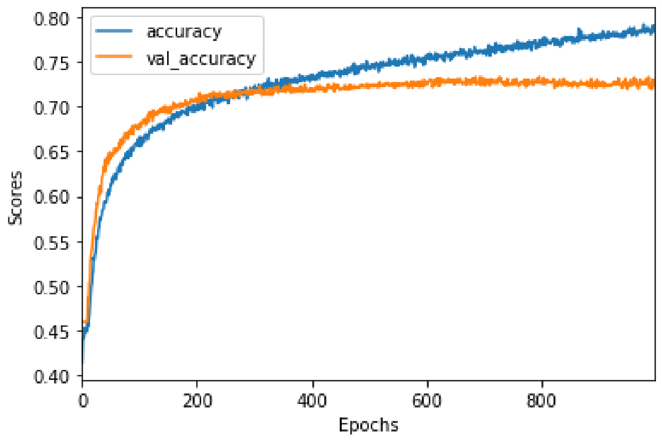

Fig. 2. Training and validation accuracies on SemEval 2014 dataset over 1000 epochs.

V. Comparison and Results

This section describes the comparisons with previous state-of-the-art models re-implemented in Twitter SA. The results are presented in Table III, where R = Recall (macro average), F1 = F1 score (macro average), Acc = Accuracy.

TABLE III
COMPARISON WITH THE DIFFERENT STATE-OF-ART METHODS.

	Dataset					
	SemVal 2014			SemVal 2015		
	R	F1	Acc	R	F1	Acc
Severyn et al. [24]	0.549	0.566	0.628	0.546	0.535	0.569
Baziotis et al. [16]	0.676	0.684	0.709	0.650	0.669	0.695
Jianqiang et a. [25]	0.573	0.572	0.606	0.568	0.577	0.614
Proposed Method	**0.688**	**0.700**	**0.726**	**0.665**	**0.680**	**0.710**
	SemVal 2016			SemVal 2017		
	R	F1	Acc	R	F1	Acc
Severyn et al. [24]	0.538	0.552	0.614	0.557	0.565	0.611
Baziotis et al. [16]	0.670	0.674	0.682	0.629	0.632	0.654
Jianqiang et a. [25]	0.560	0.561	0.588	0.546	0.553	0.584
Proposed Method	**0.706**	**0.708**	**0.703**	**0.763**	**0.766**	**0.766**

From the Table III, we observe that our model outperforms all the previous state-of-the-art models, in analysing short texts. It is established from results that the contextual representations obtained from BERT and adding task specific Bi-LSTM layer over BERT, play a substantial role in deeply understanding the sentences. The experiments are performed over 100 epochs to obtain the final results.

A. *Impact of tuning BERT*

We tune BERT by using a task-specific layer, consisting of Bi-LSTM. BERT in our model, helps in getting the contextual embeddings. So it becomes necessary to validate the addition of a task-specifc layer. Table IV shows the comparision between simple BERT and BERT + Bi-LSTM model for sentiment analysis. We present scores of F1 metric only due to space constraints. We can clearly observe the underperformance of BERT without having a task-specific layer.

TABLE IV
IMPACT OF FINETUNING ON BERT

Model	Dataset (F1 score)			
	2014	2015	2016	2017
BERT	0.370	0.430	0.650	0.480
BERT+Task Specific Layer	**0.700**	**0.680**	**0.708**	**0.766**

VI. Conclusion and Future Work

In this paper, we presented a model that utilizes contextual embeddings from BERT to train a bidirectional LSTM based sentiment classifier. Adding a simple task-specific layer outperforms all the previous state-of-the-art models. We also experimented using 1000 epochs to check for overfitting on large epoch size. In the future, we intend to apply this method for aspect-based sentiment analysis task.

Acknowledgment

This work is partially funded by SERB (Science and Engineering Research Board , Government Of India) for the project file no. EEQ/2018/001105 dated 12/02/2019.

References

[1] D. J. Blood and P. C. Phillips, "Recession headline news, consumer sentiment, the state of the economy and presidential popularity: A time series analysis 1989–1993," *International Journal of Public Opinion Research*, vol. 7, pp. 2–22, 1995.

[2] M. Bertrand and S. Mullainathan, "Do people mean what they say? implications for subjective survey data," *American Economic Review*, vol. 91, pp. 67–72, 2001.

[3] K. Chakraborty, S. Bhattacharyya, and R. Bag, "A survey of sentiment analysis from social media data," *IEEE Transactions on Computational Social Systems*, vol. 7, pp. 450–464, 2020.

[4] M. Mayo, "A clustering analysis of tweet length and its relation to sentiment," *CoRR*, vol. abs/1406.3287, pp. 1–6, 2014.

[5] S. Vanaja and M. Belwal, "Aspect-level sentiment analysis on e-commerce data," in *Proceedings of International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2018, pp. 1275–1279.

[6] M. V. Mäntylä, D. Graziotin, and M. Kuutila, "The evolution of sentiment analysis—a review of research topics, venues, and top cited papers," *Computer Science Review*, vol. 27, pp. 16–32, 2018.

[7] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, pp. 1–16, 2018.

[8] A. Esuli and F. Sebastiani, "Sentiwordnet: A publicly available lexical resource for opinion mining," in *Proceeding of International Conference on Language Resources and Evaluation*, 2006, pp. 417–422.

[9] S. Baccianella, A. Esuli, and F. Sebastiani, "Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining," in *Proceeding of International Conference on Language Resources and Evaluation*, 2010, pp. 2200–2204.

[10] L. Zhang, R. Ghosh, M. Dekhil, M. Hsu, and B. Liu, "Combining lexicon-based and learning-based methods for twitter sentiment analysis," *HP Laboratories, Technical Report HPL-2011*, vol. 89, pp. 1–8, 2011.

[11] A. Jurek, M. D. Mulvenna, and Y. Bi, "Improved lexicon-based sentiment analysis for social media analytics," *Security Informatics*, vol. 4, pp. 1–13, 2015.

[12] R. Dehkharghani, Y. Saygin, B. Yanikoglu, and K. Oflazer, "Sentiturknet: a turkish polarity lexicon for sentiment analysis," *Language Resources and Evaluation*, vol. 50, pp. 667–685, 2016.

[13] N. I. Zabha, Z. Ayop, S. Anawar, E. Hamid, and Z. Z. Abidin, "Developing cross-lingual sentiment analysis of malay twitter data using lexicon-based approach," *International Journal Of Advanced Computer Science And Applications*, vol. 10, pp. 346–351, 2019.

[14] C. Dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proceeding of 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 69–78.

[15] M. Cai, "Sentiment analysis of tweets using deep neural architectures," in *Proceedings of Conference on Neural Information Processing Systems (NIPS)*, 2018, pp. 1–8.

[16] C. Baziotis, N. Pelekis, and C. Doulkeridis, "Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis," in *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, 2017, pp. 747–754.

[17] M. Hoang, O. A. Bihorac, and J. Rouces, "Aspect-based sentiment analysis using bert," in *Proceedings of Nordic Conference on Computional Linguistics (NoDaLiDa)*, no. 167, 2019, pp. 187–196.

[18] M. Munikar, S. Shakya, and A. Shrestha, "Fine-grained sentiment classification using bert," in *Proceedings of Artificial Intelligence for Transforming Business and Society (AITB)*, 2019, pp. 1–5.

[19] A. Adhikari, A. Ram, R. Tang, and J. Lin, "Docbert: Bert for document classification," *arXiv preprint arXiv:1904.08398*, vol. abs/1904.08398, pp. 1–7, 2019.

[20] J.-S. Lee and J. Hsiang, "Patent classification by fine-tuning bert language model," *World Patent Information*, vol. 61, p. 101965, 2020.

[21] N. Shah and S. Rohilla, *emot library*, 2020. [Online]. Available: https://github.com/NeelShah18/emot

[22] C. Baziotis, *ekphrasis*, 2019. [Online]. Available: https://github.com/cbaziotis/ekphrasis

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of Advances in neural information processing systems*, 2017, pp. 5998–6008.

[24] A. Severyn and A. Moschitti, "Twitter sentiment analysis with deep convolutional neural networks," in *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2015, pp. 959–962.

[25] Z. Jianqiang, G. Xiaolin, and Z. Xuejun, "Deep convolution neural networks for twitter sentiment analysis," *IEEE Access*, vol. 6, pp. 23 253–23 260, 2018.

[26] S. Rosenthal, A. Ritter, P. Nakov, and V. Stoyanov, "Semeval-2014 task 9: Sentiment analysis in twitter," in *Proceedings of International Workshop on Semantic Evaluation (SemEval-2014)*, 2014, pp. 73–80.

[27] S. Rosenthal, P. Nakov, S. Kiritchenko, S. Mohammad, A. Ritter, and V. Stoyanov, "SemEval-2015 task 10: Sentiment analysis in twitter," in *Proceedings of International Workshop on Semantic Evaluation (SemEval 2015)*, 2015, pp. 451–463.

[28] P. Nakov, A. Ritter, S. Rosenthal, F. Sebastiani, and V. Stoyanov, "SemEval-2016 task 4: Sentiment analysis in twitter," in *Proceedings of International Workshop on Semantic Evaluation (SemEval-2016)*, 2016, pp. 1–18.

[29] S. Rosenthal, N. Farra, and P. Nakov, "SemEval-2017 task 4: Sentiment analysis in twitter," in *Proceedings of International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 502–518.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of International Conference on Learning Representations, ICLR*, 2015, pp. 502–518.