

Social Media Sentiment Analysis Using CNN-BiLSTM

Rhea Bharal¹, O. V. Vamsi Krishna²

^{1,2} B. Tech Student, Computer Science and Engineering, SRM Institute of Science and Technology, Modinagar, Uttar Pradesh, India

Abstract: Sentiment analysis is application of natural language processing for understanding the opinions or views of public on various topics. This is also popularly known as opinion mining, the system collects, analyses and examines the sentiments present in the form of tweets. Our proposed model extracts the sentiment of the tweets and classifies them using CNN-BiLSTM which is a technique of deep learning and uses Word2Vec as word embedding layer. The Sentiment140 dataset is generated from Twitter API which consists 1.6 million tweets. BiLSTM cell state based on memory is used for tweets classification Sentiments are published on Social media in the form of texts for expressing social support, happiness, anger, friendship etc. Using deep learning approach, we will be classifying the tweets as positive or negative. CNN-BiLSTM is an effective technique as compared to others like SVM, Naive Bayes Classifier and CNN.

Keywords: CNN-BiLSTM, Word2Vec, Sentiment Analysis, Machine Learning, Deep Learning, Twitter, Natural Language Processing

1. Introduction

With the increasing popularity of social media in the recent years, people are attracted towards the social networking sites like Twitter, Instagram, WhatsApp, Facebook, etc. Thus, performance of sentiment analysis tools has become increasingly critical to investigate the methods that recognize positive and negative sentiments in textual data. This is used in fields like, opinion trends, business intelligence, recommendation system, customer feedback, message filtering. Sentiment analysis is used to analyze user opinioned tweets because they give useful information such as response for specific product, opinion for candidates, user intent etc. Conventional methods in its early phase sentiment classification have been extensively performed with Naive Bayes, Support Vector Machines, Logistic Regression, bag-of-words model using TF – IDF which usually need well explained features but in case the of social media, the content is of short length and belongs to diversified topics, which makes it difficult to extract useful features for classification and thus, in this paper we will be performing Sentiment Analysis on Twitter data using deep learning techniques. The focus here is to determine whether the opinion of the tweet is “positive” or “negative” with Neural Networks classification, this could be beneficial for anyone who wants analysis of the overall sentiment of tweets at a glance which target a specific product, company, organization, or other entity. Additionally, our study also involves prediction of “words similar” to the mentioned keyword.

We propose two different Neural Network models to solve this problem, our approach focuses on, Convolutional Neural Networks (CNNs), a deep neural network (BiLSTM) that memorizes past and future experiences and learn features to extract high level features of text, while Bidirectional Long-Short Term Memory Neural Networks (BiLSTM) a subtype of Recurrent Neural Networks (RNN) which processes the layers recurrently in sequential data and capture long term dependency which helps to understand the context.

Thus, combination of these methods is used to develop new hybrid techniques which are used for future research. Therefore, in this paper we will be using a hybrid model CNN-BiLSTM to classify the tweets and compare the accuracy with the other conventional techniques. The paper is structured as: Section 2 Related work, Section3 Dataset, Section 4Methodology, Section 5 Implementation, Section 6 Conclusion.

2. Related Work

Sentiment analysis is the procedure of analysis of the text at different levels. In the last years, there has been various studies and researches on Sentiment Analysis. This is due to the growth of the internet, as now more people are expressing their opinions, thoughts and feelings. Firstly, the classification task was done to determine the class of an object based on its attributes (Turney, 2002; Pang and Lee, 2004) [1] and later it was analyzed for sentence level for classification (Hu and Liu, 2004) [2], (Kim and Hovy, 2004) [3].

The most common approaches for sentiment analysis problem include machine learning and natural language processing techniques. Naive Bayes and SVM were used to analyze the sentiment in movie reviews (Pang et al, 2002) [1]; they classified them as positive or negative and further compared it with others models.

All these categories deal with large text. On the other hand, Tweets are of shorter length text with maximum of 140 characters in it and they are difficult to analyze because of its unique language and different structure. Twitter has received much attention in the recent years for sentiment analysis, as it has become a big platform for people’s opinions with their feelings and emotions in tweets. Initially (Alec, Richa and Lei, 2009) [9] used twitter platform for the sentiment analysis with SVM, Maximum Entropy on automated training data with supervised learning. (Aproov, Boyi, Llia, Owen and Rebecca, 2011) [11] used lexicons and POS for classification. More recent studies investigated the role of emoticons on tweets (Yuki, Tadahiko and Akiyo,

2014) [4] these Lexicons of Emoticons are used to increase the performance of the analysis. (Prerna, Soujanya and Erik, 2015) [5] proposed a SVM model with rule-based classifier with increased accuracy. A study on meta-information of the words that composes the tweets and its noisy labels was done (Luciano and Junlan, 2010) [6], to retrieve more abstract representation of tweets, their study was robust on biased and noisy data. Apart from Naïve Bayes and SVM, other methods were also used in order to classify the tweets. (Nadia, Eduardo and Estevam, 2014) [7] used multinomial Naïve Bayes Classifier, Logistic Regression and Random Forest to perform classification of tweets in public tweets dataset. Multilingual analysis of twitter was also done (Anqi, Min, Yiqun and Shaoping, 2012) [10] emotions token and their polarities. K-Fold cross validation (Ramadhan, Astri and Casi, 2017) [8] used TF-IDF which increased performance. (Andrew and Raymond, 2011) [13] introduced learning word vectors for classification. Word embedding were also used in specific word learning by converting it into vectors (Duyu and Furu, 2014) [12].

In the recent years, with improved hardware functionality the complex sentiment analysis is done using neural networks. One of the first analysis using neural network was done using LSTM (Minlie, Yuije and Chao, 2016) [14]. CNN was also used for higher accuracy of analysis (Shiyang and Junbo, 2016) [15]. A Hybrid model LSTM-CNN was used for advanced analysis with multi-layer framework (Nan and Wang, 2018) [16]. Bi-LSTM was used to classify with TF-IDF (Xu and Meng, 2019) [17] which resulted in better F1 score than existing RNN, CNN and LSTM models but this was not done on twitter dataset. Although CNN-BiLSTM model was used for multi-channel lexicon sentiment analysis (Joosung and Kim, 2017) [18] but no research has been done in the Stanford Sentiment 140 dataset [19], So we will be training and testing our proposed CNN-BiLSTM model on Stanford Sentiment 140 dataset with 1.6 million tweets.

3. Dataset

Stanford Sentiment140 Dataset [19] is used for our experiment. A diverse and large dataset was used to evaluate the results using CNN-BiLSTM Classifier. The following set of data was obtained from Real-time Twitter API, collected various beneficial tweets about various subjects including sports, work, music, stocks, social media platforms and many more. A total of 1.6 million tweets were extracted which included 800,000 positive and negative tweets. This dataset was prepared and classified using emoticons. The dataset was partially pre-processed with no emoticon's symbols.

Table 1: Depicts the distribution of tweets according to their polarity in Sentiment140 Dataset

	Positive	Negative
TWEETS	800,000	800,000

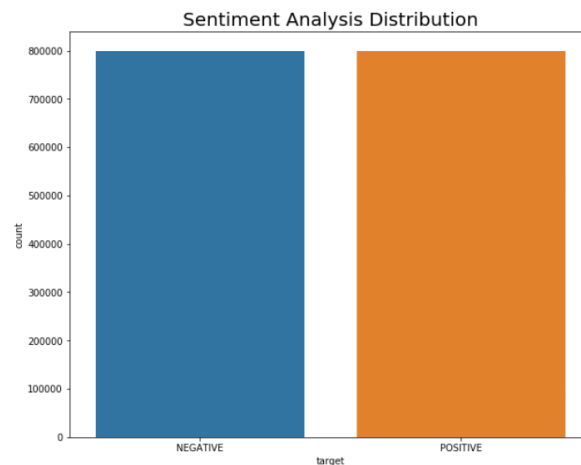


Figure 1: Negative vs Positive Tweets

4. Methodology

Deep learning is a branch of machine learning as well as a set of algorithms in which multiple neural network layers are used to solve complex functions to represent abstract data. It also updates the internal model weights with back-propagation algorithm that is used to learn the complex dataset characteristics and then further pass it on to the next layer of the neural network. In this section we will briefly elaborate deep learning model building of CNN-BiLSTM Bidirectional Long-Short Term Memory Recurrent Neural Networks.

4.1 Word2Vec

Word Embedding is a method used in neural networks for the automatic processing of natural language, it is based on the representation of words in a dataset by embedding a set of words of size y into a vector space of dimension K such that $K < y$, this is done for semantic analysis of each word in the set and to improve learning accuracy and performance. A Word Vector (Tomas and Ilya, 2013) [20] is a network of artificial neurons of two layers that learns how to represent each word in the form of real-number vector with its semantic features, which allows it to group similar words into a single vector. It is widely used for analysis in short texts like tweets in twitter, text from social networks.

```
In [7]: w2v_model.wv.most_similar("great")
Out[7]: [('fantastic', 0.7262570858001709),
('good', 0.7014542818069458),
('wonderful', 0.6890467405319214),
('awesome', 0.6784684062004089),
('amazing', 0.6550748348236084),
('fabulous', 0.6242499947547913),
('fab', 0.6024884581565857),
('excellent', 0.5791798233985901),
('nice', 0.5749059915542603),
('gr8', 0.5511847734451294)]
```



Figure 2: Most Similar Words of Great

Word2Vector is based on the fact that some words have similarities to one another which helps us to perform natural language processing.

The Word2vec can learn through two models:

-Continuous Bag of Words (CBOW) is based on word prediction from the context. It can be a single word or a set of words, thus this model is very strong as it does not need many resources. It is based on the calculation of negative logarithmic probability of word (w_t) with respect to a context (\hat{r}),

$$-\log P(w_t | \hat{r})$$

$$\text{where, } P(w_t | \hat{r}) = \frac{\exp(w_t^T \hat{r})}{\sum_{w \in V} \exp(w^T \hat{r})}$$

-Skip-Gram is similar to CBOW except that this model takes a single word as input and predicts all the other words as output.

4.2 CNN

Convolutional Neural Networks just like artificial Neural Networks, it enables learning of the model from abstract data and is made of large interconnected processing units like neurons that work together to find solution. The neurons contain learnable weights and biases of several convolution layers with nonlinear activation function that is applied at the end. A basic CNN work by feeding multidimensional data (images, word embedding, etc.) into a Convolutional layer that will be composed of multiple filters which are applied sequentially to different sections of the input to learn from it differently and combines their results, they use pooling layers, where the output is then pooled or subdivided to obtain smaller parts. Firstly, pooling of layers gives the required output of fixed size matrix for classification that allow the use of sentences and filters of varied size but gives a fixed dimensioned output every time to feed into next layer. Secondly, it keeps minimum fixed output dimension along with all the principal information by

using the max operation. important information is lost about where exactly it appeared

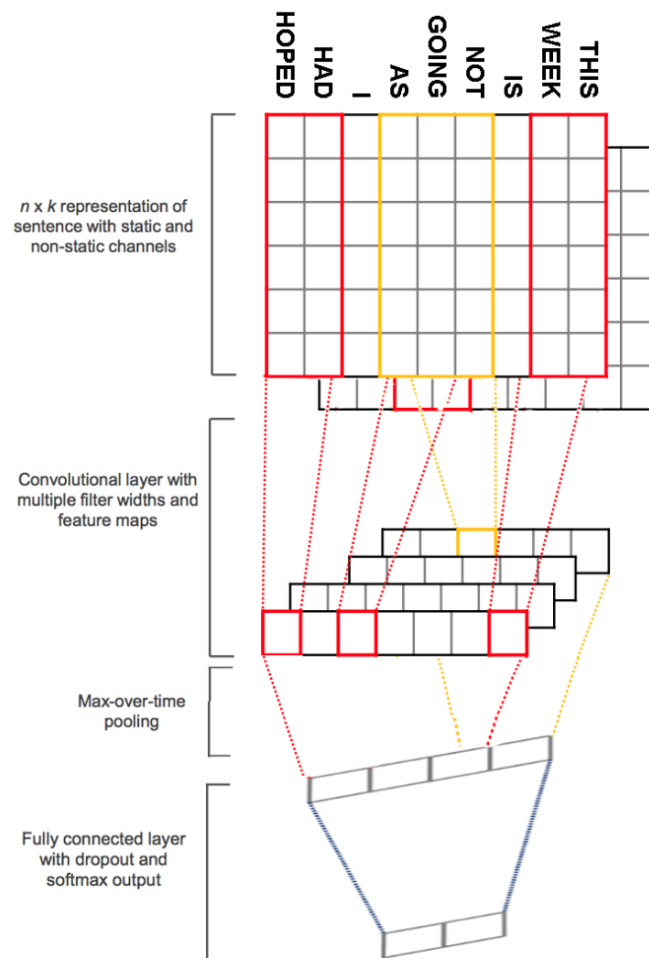


Figure 3: CNN Layer

4.3 Bi-LSTM

Unlike a CNN, RNN is suitable for temporal information, also called sequential data. However, it is difficult to train RNNs with learning long-term temporal dependencies as it causes time lag problems during training and disappearing gradient problems hence, Bi-LSTM, Using bidirectional network, the inputs will run in two directions, one from future to past and one from past to future and this is better than LSTM as it preserves information in memory from the future and uses the two hidden states together for retaining information from both past and future at any given time. Thus it has proven to give good results in natural language processing

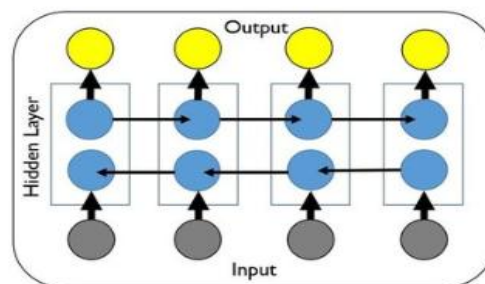


Figure 4: Bi-LSTM Cell

4.4 CNN-BiLSTM

The combination of CNN and Bi-LSTM models each have a specific architecture and its own Advantages:

- CNN is known for maximum feature extraction as much as possible from the input text using max pooling layer.
- Bi-LSTM preserves information in memory from the future and uses the two hidden states together to retain information from both past and future and it also has the ability to ignore unnecessary textual information using the forget gate.

The purpose of combining these two models is to generate a hybrid model that uses the strengths of CNN and Bi-LSTM, so that it captures the maximum features extracted using CNN, and uses them as a Bi-LSTM input. Here the CNN input layer takes vectors as input from word embedding by Word2Vec. After each filter, a layer of max pooling is applied to reduce and update the size of the data. Then, the results of all max pooling layers are concatenated to feed as Bi-LSTM input, this applies a layer to filter the information, using three gates. The output of this step is the input of the Bi-LSTM layer, which connects input information with output information. Finally, we apply the activation function to produce the desired output and classify tweets according to polarity.

Thus, we propose the model that is composed of three parts:

- Pre-processing: Here, data cleansing and pre-processing is done. Then Word2Vec embedding is used to prepare data for CNN. The resulting vector is given as an input to the next stage.
- CNN: Here, convolution and max pooling layers are applied for maximum feature extraction. The output of this stage is given as the input of the next stage.
- Bi-LSTM: Here, Bi-LSTM layers are applied for sentiment classification. The output of this stage is the classification of the tweets as positive, negative.

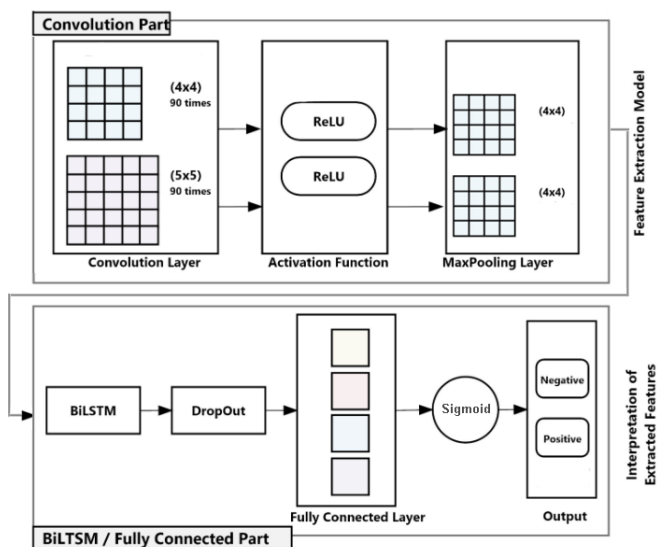


Figure 5: CNN-BiLSTM Layer

5. Implementation

Python is used to implement the algorithm on the tweets for classification with TensorFlow as backend. The first part is learning phase and followed by predicting phase. Before that we will splitting our dataset into train and test in 80: 20 ratio respectively. In learning phase, the train dataset is been trained and classifies it. The performance of this phase is recorded. Dataset of appropriate size is used to prevent overfitting and underfitting. The model learns to classify these input tweets based on its training and in testing phase the model is tested on test set.

```

Train on 1152000 samples, validate on 128000 samples
Epoch 1/15
1152000/1152000 [=====] - 201s 174us/step - loss: 0.4080 - acc: 0.7612 - val_loss: 0.4625 - val_acc: 0.7783
Epoch 2/15
1152000/1152000 [=====] - 194s 169us/step - loss: 0.4698 - acc: 0.7735 - val_loss: 0.4566 - val_acc: 0.7826
Epoch 3/15
1152000/1152000 [=====] - 194s 169us/step - loss: 0.4638 - acc: 0.7776 - val_loss: 0.4527 - val_acc: 0.7852
Epoch 4/15
1152000/1152000 [=====] - 196s 170us/step - loss: 0.4597 - acc: 0.7804 - val_loss: 0.4518 - val_acc: 0.7855
Epoch 5/15
1152000/1152000 [=====] - 197s 171us/step - loss: 0.4573 - acc: 0.7817 - val_loss: 0.4494 - val_acc: 0.7868
Epoch 6/15
1152000/1152000 [=====] - 197s 170us/step - loss: 0.4551 - acc: 0.7830 - val_loss: 0.4486 - val_acc: 0.7869
Epoch 7/15
1152000/1152000 [=====] - 196s 169us/step - loss: 0.4533 - acc: 0.7838 - val_loss: 0.4476 - val_acc: 0.7883
Epoch 8/15
1152000/1152000 [=====] - 194s 168us/step - loss: 0.4518 - acc: 0.7850 - val_loss: 0.4480 - val_acc: 0.7868
Epoch 9/15
1152000/1152000 [=====] - 193s 168us/step - loss: 0.4500 - acc: 0.7858 - val_loss: 0.4475 - val_acc: 0.7884
Epoch 10/15
1152000/1152000 [=====] - 193s 167us/step - loss: 0.4499 - acc: 0.7862 - val_loss: 0.4468 - val_acc: 0.7887
Epoch 11/15
1152000/1152000 [=====] - 194s 168us/step - loss: 0.4494 - acc: 0.7866 - val_loss: 0.4472 - val_acc: 0.7885
Epoch 12/15
1152000/1152000 [=====] - 194s 168us/step - loss: 0.4485 - acc: 0.7867 - val_loss: 0.4477 - val_acc: 0.7879
Epoch 13/15
1152000/1152000 [=====] - 193s 167us/step - loss: 0.4476 - acc: 0.7876 - val_loss: 0.4479 - val_acc: 0.7881
Epoch 14/15
1152000/1152000 [=====] - 194s 168us/step - loss: 0.4470 - acc: 0.7882 - val_loss: 0.4486 - val_acc: 0.7879
Epoch 15/15
1152000/1152000 [=====] - 193s 167us/step - loss: 0.4468 - acc: 0.7883 - val_loss: 0.4471 - val_acc: 0.7882
CPU times: user 1h 46s, sys: 4min 55s, total: 1h 50min 42s
Wall time: 48min 43s
    
```

Figure 6: Training on dataset

TensorFlow is used as a backend for all the necessary functionalities. Using Keras we have built CNN-BiLSTM neural network with max pool layer, embedding layer and hidden state activation function sigmoid. We can see the performance increases with each epoch. The performance enhances with each epoch illustrating that model the learns with experience. We can here see classification report with f1-score in figure 7.

	precision	recall	f1-score	support
NEGATIVE	0.79	0.78	0.79	159494
POSITIVE	0.78	0.80	0.79	160506
accuracy			0.79	320000
macro avg	0.79	0.79	0.79	320000
weighted avg	0.79	0.79	0.79	320000

Figure 7: Classification Report

In prediction phase, the trained model is applied on the test dataset and classification reports shows the results and accuracy of the model. Here the accuracy of CNN-BiLSTM model is displayed and by how much precision and accuracy the model can classify the tweets can be seen. The proportions of correct and incorrect classifications are shown by confusion matrix. The classification results are represented using confusion matrix and which can be easily understood.

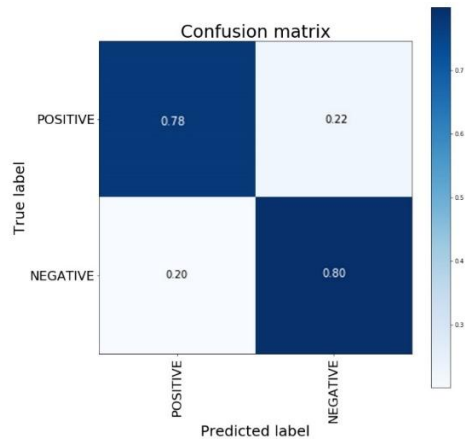


Figure 8: Confusion Matrix

In figure 8, We can see the incorrectly and correctly classified tweets. The proportion in the principle diagonal are the correctly classified tweets. The accuracy of the model is 78.86% by using CNN-BiLSTM model. The performance of model on this dataset is better than other machine learning techniques. Therefore, CNN-BiLSTM is very good for classifying the tweets and that too in shorter period of training time.

6. Conclusion

Our proposed model for sentiment classification approach based on CNN-BiLSTM for short texts Word2Vec model as a word embedding model. The model is very sensitive to overfitting, so the model was trained till 15th epoch only. In order to test our model against data and measure it, we needed to capture information on how many correct predictions for positive and negative occurred as well as incorrect predictions by using:

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total Predictions}}$$

Figure 9: Basic Formula for calculation of model Accuracy

Using the formula in Figure 9, we have made basic accuracy calculations and plots. Model testing against our sentiment140 dataset yielded validation accuracies up to **78.83%** after training for 80K iterations each and testing accuracy of **78.86%**. Further, the recall value **78%** for negative and **80%** for positive tweets indicates the correct predictions for one sentiment divided by the incorrect and correct predictions for the sentiment that helps us understand how good the model is for one sentiment; a higher recall shows the overall sentiment category is more complete or accurate. Precision value **79%** for the tweets is calculated by true positives divided by false positives and true positives that implies the results for that sentiment are quite reliable.

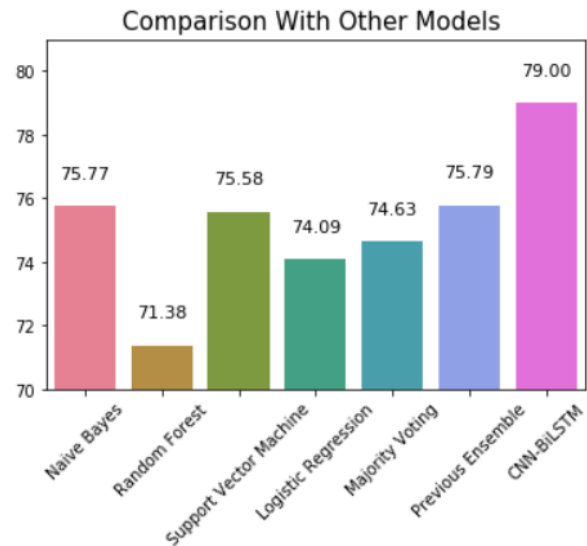


Figure 10: Accuracy comparison with different models

Deep learning methods such as CNN-BiLSTM show better performance of sentiment classification with more amounts of training data. As shown in figure 10, Our model's accuracy is better as compared to other models that have been trained on the same dataset Sentiment 140.

References

- [1] Bo Pang and Lillian Lee, "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts", Proceedings of the ACL, 2004
- [2] Minqing Hu and Bing Liu, "Mining and Summarizing Customer Reviews", KDD'04, August 22–25, 2004, Seattle, Washington, USA
- [3] Soo-Min Kim and Eduard Hovy, "Determining the Sentiment of Opinions", California
- [4] Yuki Yamamoto, Tadahiko Kumamoto and Akiyo Nadamoto, "Role of Emoticons for Multidimensional Sentiment Analysis of Twitter", Proceedings of the 16th International Conference on Information Integration and Web-based Applications & Services, 2014
- [5] Prerna Chikersal, Soujanya Poria and Erik Cambria, "SeNTU: Sentiment Analysis of Tweets by Combining a Rule-based Classifier with Supervised Learning", Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), 2015
- [6] Luciano Barbosa and Junlan Feng, "Robust Sentiment Detection on Twitter from Biased and Noisy Data", 2010 Available: <https://www.aclweb.org/anthology/C10-2005.pdf>
- [7] Nádia F. F. da Silva, Eduardo R. Hruschka and Estevam R. Hruschka Jr, "Tweet sentiment analysis with classifier ensembles", 2014 Available: <https://doi.org/10.1016/j.dss.2014.07.003>
- [8] W. P. Ramadhan, S. T. M. T. Astri Novianty and S. T. M. T. Casi Setianingsih, "Sentiment analysis using Multinomial logistic regression", International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC), 2017
- [9] Alec Go, Richa Bhayani and Lei Huang, "Twitter Sentiment Classification using Distant Supervision"

- [10] Anqi Cui, Min Zhang, Yiqun Liu and Shaoping Ma, “*Emotion Tokens: Bridging the Gap among Multilingual Twitter Sentiment Analysis*”, 2011.
- [11] Apoorv Agarwal, BoyiXie, Ilia Vovsha, Owen Rambow and Rebecca Passonneau, “*Sentiment Analysis of Twitter Data*”, Proceedings of the Workshop on Language in Social Media (LSM 2011), Portland, Oregon, 23 June 2011.
- [12] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou & Ting Liu†, Bing Qin, “*Learning Sentiment – Specific Word Embedding for Twitter Sentiment Classification*”, Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, Maryland, USA, June 23-25 2014.
- [13] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Ng & Christopher Potts, “*Learning word vectors for sentiment analysis*”, HLT '11: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, June 2011
- [14] Minlie Huang, Yujie Cao, Chao Dong, “*Modeling Rich Contexts for Sentiment Classification with LSTM*”, 2016 Available: <https://arxiv.org/abs/1605.01478>
- [15] Shiyang Liao, Junbo Wang, Ruiyun Yu, Koichi Sato & Zixue Cheng, “*CNN for situations understanding based on sentiment analysis of twitter data*”. Available: <https://doi.org/10.1016/j.procs.2017.06.037>
- [16] Nan Chen & Peikang Wang, “*Advanced Combined LSTM-CNN Model for Twitter Sentiment Analysis*”, 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), Nov.2018
- [17] Guixian Xu, Yueting Meng, Xiaoyu Qiu, Ziheng Yu & Xu Wu, “*Sentiment Analysis of Comment Texts Based on BiLSTM*”, IEEE Access (Volume: 7), April 2019
- [18] Joosung Yoon & Hyeoncheol Kim, “*Multi-Channel Lexicon Integrated CNN-BiLSTM Models for Sentiment Analysis*”, the 2017 Conference on Computational Linguistics and Speech Processing ROCLING 2017, 2017 Dataset: *Sentiment140*, Available: <http://help.sentiment140.com/for-students>
- [19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado and Jeffrey Dean “*Distributed Representations of Words and Phrases and their Compositionality*”, 2013