2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence) | 978-1-6654-1451-7/20/$31.00 ©2021 IEEE | DOI: 10.1109/Confluence51648.2021.9377070

A Comparative Evaluation of Traditional Machine Learning and Deep Learning Classification Techniques for Sentiment Analysis

Mr. Kaushik Dhola
Research Scholar, Information Technology Dept.
L. D. College of Engineering
Ahemdabad, India
kaushikdhola02@gmail.com

Mr. Mann Saradva
Research Scholar, Information Technology Dept.
L. D. College of Engineering
Ahemdabad, India
mannsaradva50@gmail.com

***Abstract*— With the technological advancement in the field of digital transformation, the use of the internet and social media has increased immensely. Many people use these platforms to share their views, opinions and experiences. Analyzing such information is significant for any organization as it apprises the organization to understand the need of their customers. Sentiment analysis is an intelligible way to interpret the emotions from the textual information and it helps to determine whether that emotion is positive or negative. This paper outlines the data cleaning and data preparation process for sentiment analysis and presents experimental findings that demonstrates the comparative performance analysis of various classification algorithms. In this context, we have analyzed various machine learning techniques (Support Vector Machine, and Multinomial Naive Bayes) and deep learning techniques (Bidirectional Encoder Representations from Transformers, and Long Short-Term Memory) for sentiment analysis.**

Keywords*— *Sentiment Analysis, Machine Learning, Deep Learning, Natural Language Processing, Classification

I. INTRODUCTION

Data is at the root of almost every organizations' decision-making process. Most companies are beginning to realize that analyzing and exploring this data will help benefit their business [1]. In this context, many Natural Language Processing (NLP) tasks are being implemented within the organization in order to analyze this massive amount of data. Sentiment analysis is one of the key functions of NLP where opiniated textual data is statistically analyzed and classified into positive, or negative sentiment. The most laborious phase of sentiment analysis is to identify the appropriate data to be used for the training purpose.

With the increasing demand for social media and the internet, people are free to share their experiences on online platforms. Although such opinions can affect future demands, they can also be used to improve the business by enhancing customer experience. In recent years there has been considerable amount of research has been done in the field of sentiment analysis particularly focused on identifying larger chunks of text, such as feedback and Tweets. Feedback represents the contextualized opinions of the writers, while tweets are more discreet and confined to 150 characters of a sentence. Tweets are not the same as feedback; however, it still provides additional direction to acquire the information.

Earlier studies in sentiment analysis have examined the success of various classifiers in reviewing the twitter data [2], [3]. These studies have used machine learning algorithms such as Naive Bayes, Maximum Entropy, and Support Vector Machine (SVM). Nasukawa et al. [4] proposed a sentiment analyzer that extracts the sentiment from the online textual documents for a given subject. Gamon [5] conducted research which was aimed to perform automatic sentiment analysis on the customer feedback data and concluded that deep linguistic analysis features contribute consistently to classification accuracy in this domain.

The sentiment classification is a binomial classification in which we categorize the statement in two criteria i.e. positive and negative. In this study, we compare several machine learning as well as deep learning techniques to perform sentiment classification on the dataset. The dataset used for this task is a corpus of English language Tweets. Our research work is split into two stages where the data is pre-processed in the first stage using various text-mining techniques. Subsequently in the second stage, we built multiple models with the use of machine learning and deep learning techniques to identify the sentiment of the proclamations on the Twitter data. The overarching goal of this research is to discover the most suitable approach for any individual or organization that may want to perform sentiment analysis on the textual data.

II. DATA PREPARATION

A. Tokenization

Data is systematically analyzed where sophistication increases with each growing stage of execution. A sentence consists of words-tokens, affixes, phrases, and compound words known as lexical items. Before knowing the meaning of the actual sentence, it is important to understand the meaning of each word. Every sentence has unique significance that depends upon the keywords used in the sentence and hence makes it difficult to describe. It can be well understood with the following example:

Example 1: "I want to take a leave."

The first objective here is to extract the phrases used in the sentence and then understand its meaning. Verb plays an important role in the sentence. In Example 1, the meaning of the main verb want is to desire something. There are two verbs used in the sentence i.e. want and take, where the former is the main verb and the latter is used as an infinitive with to. The noun here is leave which could imply sick leave, casual leave, earned leave, maternity leave, or any other type of leave that is not explicitly specified in the sentence. The verbs want and leave are the primary source of information in the given example. It was a simple illustration but there can be more

complex sentences that might be difficult for the system to comprehend.

To overcome this problem, the sentence must be decomposed into small chunks and the process of dividing a sentence is called word tokenization. After having small chunks of the sentence, the stop-words (i.e. the, is, to, etc.) that are not useful for the processing of the sentence should be removed. Word tokenization divides the sentence into small chunks and removes the extra characters, whereas stop word removal is performed to filter out the words. Both processes are clarified with an example shown in Figure 1 and Figure 2.

Input: "Seema bought this dress and it is beautiful."
Output: ['Seema', 'bought', 'this', 'dress', 'and', 'it', 'is', 'beautiful', '.']

Fig. 1. Illustrates the input and output of the word tokenization

Input: ['Seema', 'bought', 'this', 'dress', 'and', 'it', 'is', 'beautiful', '.']
Output: ['Seema', 'bought , 'dress', 'beautiful']

Fig. 2. Illustrates the stop-word removal from the previously tokenized words

Once the sentence is tokenized and stop-words are removed, it is further treated using stemming process.

B. Stemming Process

During the stemming process, the extracted words are converted into their root form known as stem. To achieve this, the porter stemming algorithm is used which removes suffixes from the words in English [6]. This process helps in providing more details on how a word is related to the subject and eventually helps in getting better accuracy results. Table 1 demonstrates the stemming performed on various forms of word '*communication'*.

TABLE 1: SHOWS THAT ALL THE TERMS HAVE A SIMILAR MEANING, BUT THEIR FORM IS DIFFERENT (*Communication*, *Communicated*, *Communicates*, *Communicating*) AND THESE TERMS ARE THEN CONVERTED TO THEIR ORIGINAL FORM *Communicate*.

	Word	Stemmed Word
0	Communication	Communicate
1	Communicated	Communicate
2	Communicates	Communicate
3	Communicating	Communicate

After the stemming process, a sentence is further treated with the Lemmatization process where it is lowered down to its root, as it occurs in the vocabulary. The stems produced by lemmatization are valid vocabulary words and are grammatically complete, unlike the words generated by the stemmer.

Input Tweet: #HotelABC is really good. @Kaushik https://abc.co/
Output Tweet : HotelABC really good

Fig. 3. Illustrates the removal of tags and hyperlinks from the input tweet

Upon the completion of the lemmatization process, the removal of tags and hyperlinks is performed on the data. Our data collection is compiled from Twitter, a social media network where individuals can tag and mention other user's IDs and blogs from the internet. These tweets contain a lot of hyperlinks and Twitter references which needs to be removed for further analysis. Figure 3 demonstrates an example of the removal of such tags and hyperlinks from the tweet.

III. SENTIMENT CLASSIFICATION

A. Dataset

In order to carry out the comparative analysis, we have used the publicly available dataset which contains the Twitter posts [3]. The dataset contains over 1.6 million tweets where 7,98,988 are positive tweets and 8,01,011 are negative tweets. To validate the model on this dataset, it was split into train and test set with the ratio of 80% and 20% respectively (as shown in Table 2).

TABLE 2: COUNT OF THE TRAIN AND TEST DATASET WITH THE POSITIVE, NEGATIVE, AND TOTAL NUMBER OF TWEETS

	Positive	Negative	Total
Train-Set	639494	640505	1279999
Test-Set	159494	160506	320000
			1599999

B. Machine Learning Approach for Sentiment Classification

1. Support Vector Machine

SVM is a supervised machine learning algorithm that can be used for classification as well as regression problems. The SVM technique is a mathematical classification methodology focused on maximizing the margin between the instances and the separation hyper-plane [7]. SVM performs classification by locating a hyper-plane that separates the groups that we mapped in the n-dimensional region.

For the hypothesis, we will use the hyperplane to do the forecasting as defined below:

$$h(x) = \begin{cases} +1, & w * x + b \geq 0 \\ -1, & w * x + b < 0 \end{cases} \quad (1)$$

where, w is normal of the line and b is bias.

The above hyperplane will be classified as class +1, whereas the below hyperplane will be classified as class -1.

The distance D of x from the hyperplane is represented mathematically by the following equation:

$$D = \frac{|w*x+b|}{|w|} \quad (2)$$

2. Multinomial Naive Bayes Classifier

Multinomial Naive Bayes Classifier is a supervised learning algorithm which is based on probability and it is centered on the sentence classification contexts. This algorithm follows the concept of multinomial distribution in conditional probability [8]. The probability calculation of Multinomial Naive Bayes Classifier is shown as below [9]:

$$P(a|b) \propto p(a) \prod_{1 \leq k \leq n} p(tk|a) \quad (3)$$

where, $P\ (tk \mid a)$ is the conditional probability of the word tk that appears in class a.

Here, P (tk | a) is potential probability tk in class a. While P(a) is the previous probability appearing in class a and is calculated using the following equation:

$$P(a) = \frac{Na}{N} \tag{4}$$

where, *Na* is the sum of class *a* and *N* is the sum of all the Criteria.

The Potential probability calculation is shown below:

$$(tk|a) = \frac{Fta}{\sum_{t' \in V} F_{at'}} P \tag{5}$$

where, *Fta* is the number of frequencies of the word t in the sentence, and $\sum_{t' \in V} F_{at'}$ is the total number of frequencies of all words in class *a*.

C. Deep Learning Approach for Sentiment Classification

1. Bidirectional Encoder Representations from Transformers

Bidirectional Encoder Representations from Transformers (BERT) is the transformer architecture approach that can be applied to pre-train a general-purpose classification model and fine-tune it to a given task [10]. BERT uses undescribed text as an input where it masks 15 percent of words, and finally attempts to predict these words. BERT Model also keeps track of the context between sentences by training a basic task of predicting the input sentence. Figure 4 highlights the BERT model architecture.

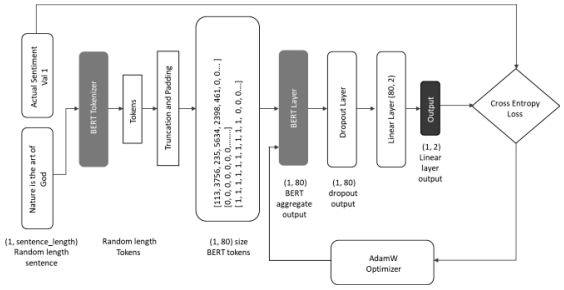

Fig. 4. BERT model architecture [14]

The sentences are permuted to the BERT Model input with the use of BERT Tokenizer where we have used 128 input tokens as the full length of the BERT Layer.

2. Long Short-Term Memory

Long Short-Term Memory (LSTM) networks are a special kind of Recurrent Neural Network (RNN) and capable of learning long-term dependencies [11]. The RNNs have a series of replicating neural network modules and these repeat modules have a rudimentary structure such as a single tanh layer. LSTM has a similar series like arrangement; however, the repeat module has a unique appearance. LSTM modules are being used broadly for encoding written texts. The LSTM model architecture comprises of four layers such as word embedding layer, LSTM layer, fully connected layer, and sigmoid activation layer (see Figure 5). It works uniquely rather than forming a single neural network layer [11].

The word embedding layer is a common representation of the document sentence structure. It is capable of capturing the part of speech in a sentence and finds out the lexical and pragmatic resemblance. The embedding layer also determines the connection with other words.

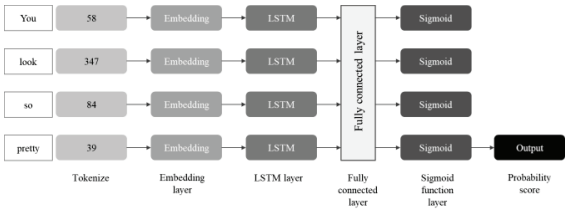

Fig. 5. LSTM model architecture [15]

LSTM layer and fully connected layers are used for additional tasks including word classifications which focuses on the encoded attribute. RNN uses the word embedding and its previous hidden state to quantify the next hidden state for each dimension in the sequence and each keyword in a sentence. The mathematical equation for LSTM operations is shown below:

$$h_t = \sigma(W_i X_t + U_i h_{t-1} + b_i) \tag{6}$$

where, *xt* is the current word embedding, $Wi \in Rm \times d$ and $Uh \in Rm \times m$ are weight matrices, $bi \in Rm$ is a bias term and *f(x)* is a non-linear function, usually chosen to be *tanh* [12].

This basic RNN has an exploded and perishing gradient problem during the back-propagation training stage. The gradient problem can be solved by implementing the mode complex internal structure of LSTMs that also allows remembering the information for long terms as well as short terms [13]. The mathematical equations are defined below:

$$f_t = \sigma(W_f X_t + U_f h_{t-1} + b_f)$$

$$i_t = \sigma(W_i X_t + U_i h_{t-1} + b_i)$$

$$o_t = \sigma(W_o X_t + U_o h_{t-1} + b_o) \tag{7}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tan h(W_c X_t + U_c h_{t-1} + b_c)$$

$$h_t = o_t \circ \tan h(c_t)$$

where,
i_t = the input gate,
f_t = the forget gate,
c_t = the cell state,
h_t = the regular hidden state,
σ = the sigmoid function,
$\circ$ = the Hadamard product

IV. EVALUATION

In order to evaluate the model performance, we have used Precision, Recall, and F1-score metrics. We demonstrate the results of all the models which gives a clear direction for selecting an appropriate algorithm for sentiment analysis on a large volume of data.

SVM Classifier reports an accuracy of 76.3% when tested on over 1.6 million Twitter Sentiments. Below we demonstrate the results obtained for the test dataset:

TABLE 3 CLASSIFICATION REPORT OF SVM CLASSIFIER

	precision	*recall*	*f1-score*	*support*
	0.73	0.82	0.77	160506
	0.79	0.70	0.74	159494
accuracy			0.76	320000
macro avg	0.76	0.76	0.76	320000
weighted avg	0.76	0.76	0.76	320000

The accuracy reported by Multinomial Naive Bayes Classifier turns out to be 76% and the classification report produced for the test dataset is shown below:

TABLE 4 CLASSIFICATION REPORT OF MULTINOMIAL NAIVE BAYES CLASSIFIER

	precision	*Recall*	*f1-score*	*support*
	0.77	0.74	0.76	160506
	0.75	0.78	0.77	159494
accuracy			0.76	320000
macro avg	0.76	0.76	0.76	320000
weighted avg	0.76	0.76	0.76	320000

However, the accuracy of Multinomial Naïve Bayes classifier can be improved by Hyper-parameter Tuning that optimizes the performance of the Classifier. We performed Hyper-parameter tuning that reported 76.9% accuracy on the same dataset. Below is the classification report after performing Hyper-parameter Tuning:

TABLE 5 CLASSIFICATION REPORT OF MULTINOMIAL NAIVE BAYES CLASSIFIER AFTER HYPER-PARAMETER TUNING

	precision	*recall*	*f1-score*	*support*
	0.78	0.75	0.77	160506
	0.76	0.79	0.78	159494
accuracy			0.77	320000
macro avg	0.77	0.77	0.77	320000
weighted avg	0.77	0.77	0.77	320000

The experimental results after applying the BERT model reports an accuracy of 85.4%. We got the following evaluation metrics result:

TABLE 6 CLASSIFICATION REPORT OF BERT MODEL

	precision	*recall*	*f1-score*	*support*
	0.85	0.86	0.86	160506
	0.86	0.84	0.85	159494
Accuracy			0.85	320000
macro avg	0.85	0.85	0.85	320000
weighted avg	0.85	0.85	0.85	320000

The accuracy of the LSTM Model turns out to be 80% that is superior enough for the sentiment analysis on a large set of data. Although, accuracy can still be improved by performing more data pre-processing operations. The classification report is as shown below:

TABLE 7 CLASSIFICATION REPORT OF LSTM MODEL

	precision	*recall*	*f1-score*	*support*
	0.79	0.78	0.78	160506
	0.78	0.79	0.78	159494
accuracy			0.78	320000
macro avg	0.78	0.78	0.78	320000
weighted avg	0.78	0.78	0.78	320000

Table 8 presents the performance metrics of all the classifiers used in this research for the comparative analysis on both positive and negative sentiment dataset:

TABLE 8 COMPARISON OF EVALUATION METRICS FOR ALL THE CLASSIFIERS

Algorithm		**Precision**	**Recall**	**F1- Score**
SVM	Negative	0.73	0.82	0.77
	Positive	0.79	0.70	0.74
Multinomial Naïve Bayes	Negative	0.77	0.74	0.76
	Positive	0.75	0.78	0.77
LSTM	Negative	0.79	0.78	0.78
	Positive	0.78	0.79	0.78
BERT	Negative	0.85	0.86	0.86
	Positive	0.86	0.84	0.85

The same evaluation results for the classifiers are also visualized considering Precision, Recall and F1-score (see Figure 6).

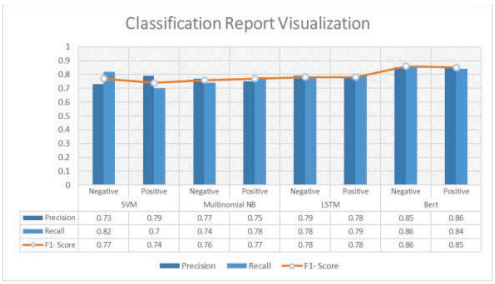

Fig. 6. Evaluation metrics for all the classifiers

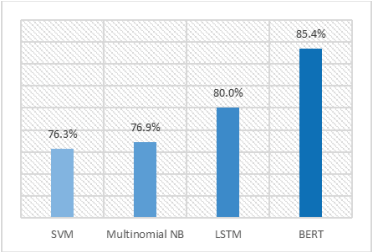

Fig. 7. Comparison of accuracy results reported by all the classifiers

As can be seen in Figure 7, deep learning techniques (LSTM and BERT) report higher accuracy than machine learning techniques (SVM and Multinomial Naïve Bayes). As the models are evaluated on a large amount of data (1.6 million tweets), it covers larger boundaries for sentiment analysis operations. Our findings confirm the advantage of using LSTM (80%) and BERT (85.4%) techniques over SVM (76.3%) and Multinomial Naïve Bayes classifier (76.9%) for sentiment analysis. This result has further strengthened our confidence in performing the sentiment classification using deep learning techniques.

V. CONCLUSION AND FUTURE WORK

In this research, we carried out a comparative study on various machine learning as well as deep learning techniques to perform the sentiment classification. These techniques were applied to the Twitter dataset with an aim to classify the sentiments with greater accuracy. The machine learning techniques applied on the data were SVM and Multinomial Naive Bayes Classifier, whereas, the deep learning techniques used were LSTM and BERT. We infer that both techniques work well for the sentiment classification. However, the

results of the individual classifiers on sentiment analysis showed that BERT and LSTM can achieve higher accuracy (9.95% higher) for classifying sentiment than the SVM and Multinomial Naive Bayes. Additionally, the accuracy for sentiment analysis can be further optimized by employing the model like Gated recurrent units (GRUs) which is similar to LSTM. This would be carried out in future and would improve the accuracy of classifier.

REFERENCES

[1] Prof.MD Sameeruddin Khan, Prof.GVNKV Subbarao, and GNV VIBHAV REDDY, 'Hace Theorem based Data Mining using Big Data', International Journal of Engineering And Science, vol. 6, no. 5, p. PP-83-87, May 2016.

[2] V. A. and S. S. Sonawane, 'Sentiment Analysis of Twitter Data: A Survey of Techniques', International Journal of Computer Applications, vol. 139, no. 11, pp. 5–15, Apr. 2016, doi: 10.5120/ijca2016908625.

[3] A. Go, R. Bhayani, and L. Huang, 'Twitter sentiment classification using distant supervision', Processing, vol. 150, 2009.

[4] T. Nasukawa and J. Yi, 'Sentiment analysis: capturing favorability using natural language processing', in Proceedings of the international conference on Knowledge capture - K-CAP '03, Sanibel Island, FL, USA, 2003, p. 70, doi: 10.1145/945645.945658.

[5] M. Gamon, 'Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis', in Proceedings of the 20th international conference on Computational Linguistics - COLING '04, Geneva, Switzerland, 2004, pp. 841-es, doi: 10.3115/1220355.1220476.

[6] M. F. Porter, 'An algorithm for suffix stripping', Program, vol. 40, no. 3, pp. 211–218, Jul. 2006, doi: 10.1108/00330330610681286.

[7] Y. Al Amrani, M. Lazaar, and K. E. El Kadiri, 'Random Forest and Support Vector Machine based Hybrid Approach to Sentiment Analysis', Procedia Computer Science, vol. 127, pp. 511–520, 2018, doi: 10.1016/j.procs.2018.01.150.

[8] I. C. Mogotsi, 'Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze: Introduction to information retrieval: Cambridge University Press, Cambridge, England, 2008, 482 pp, ISBN: 978-0-521-86571-5', Information Retrieval, vol. 13, no. 2, pp. 192–195, Apr. 2010, doi: 10.1007/s10791-009-9115-y.

[9] A. A. Farisi, Y. Sibaroni, and S. A. Faraby, 'Sentiment analysis on hotel reviews using Multinomial Naïve Bayes classifier', Journal of Physics: Conference Series, vol. 1192, p. 012024, Mar. 2019, doi: 10.1088/1742-6596/1192/1/012024.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, 'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding', arXiv:1810.04805 [cs], May 2019, Accessed: Nov. 28, 2020. [Online]. Available: http://arxiv.org/abs/1810.04805.

[11] 'Understanding LSTM Networks', Aug. 2015, [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[12] M. Cliche, 'BB_twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs', arXiv:1704.06125 [cs, stat], Apr. 2017, Accessed: Nov. 28, 2020. [Online]. Available: http://arxiv.org/abs/1704.06125.

[13] S. Hochreiter and J. Schmidhuber, 'Long Short-Term Memory', Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[14] Mahendras8894. (2020, October 05). BERT Sentiment Analysis. Retrieved December 23, 2020, from https://www.kaggle.com/mahendras8894/bert-sentiment-analysis/

[15] Agrawal, S. (2020, January 10). Sentiment Analysis using LSTM Step-by-Step. Retrieved December 23, 2020, from https://towardsdatascience.com/sentiment-analysis-using-lstm-step-by-step-50d074f09948

[16] Yadav, A., Vishwakarma, D.K. Sentiment analysis using deep learning architectures: a review. Artif Intell Rev 53, 4335–4385 (2020). https://doi.org/10.1007/s10462-019-09794-5

[17] N. C. Dang, M. N. Moreno-García, and F. De la Prieta, "Sentiment Analysis Based on Deep Learning: A Comparative Study," Electronics, vol. 9, no. 3, p. 483, Mar. 2020.