IJCI V9-01 (2022) 01–14
International Journal of Computers and Information (IJCI)
Available online at *https://ijci.journals.ekb.eg/*

Performance Investigation of Features Extraction and Classification Approaches for Sentiment Analysis Systems

Raghda Elnadrey, Ashrif Elsisi, Walid Attwa

Computer Science Department, Faculty of Computers and Information,Menoufia University, Shebin Elkom 32511, Egypt
raghdah_shrif@yahoo.com, ashraf.elsisi@ci.menofia.edu.eg, walid_mufic@yahoo.com

Abstract

Data pre-processing and feature extraction of micro-blogging data in sentiment analysis systems becomes an effective field of analysis. Object identification, negation expressions, sarcasm, outlines, misspellings are the major issues faced during sentiment analysis. So, data pre-processing in a sentiment analysis system is a conclusive step to improve data quality, raise the extraction, and classification of meaningful data. This paper presents a sentiment analysis system for performance investigation. Several pre-processing and feature extraction techniques apply to optimise the sentiment analysis. Our system comprises three different components: data pre-processing, feature extraction, and sentiment analysis. The pre-processing and feature extraction approaches enhance the sentiment analysis system performance. We compare different sentiment analysis approaches using a dataset of US Airlines from Twitter. Results show achieving high performance when using the Word2Vec approach with XGBoost and random forest classification algorithms. Also, results show the classification technique, Naive Bayes is the lowest performance.

Keywords: Sentiment Analysis; Classification; Features Extraction; Microblogging; Machine learning.

1. Introduction

Recently, microblogging data such as social media platforms help users share their opinion on all kinds of topics and events. According to the Forbes website, each day, there are 2.5 million bytes of data created, and Twitter has received 456,000 tweets every minute of the day. Users frequent social media because it provides the amount of freedom to express their opinion while protecting anonymity. Because of these, they cause tons of hate speeches, derogatory or discriminatory content targeting certain groups, and racist comments, as discussed in [1]. For these reasons, it is essential to have an efficient way to predict user sentiments about social public events, services, and products [2]. The target is to classify the text on social media by using machine learning algorithms. In medical science, text classification is used to analyze and categorise reports such as hospital records, and brief text in tweets. The government used textual sentiment analysis in blogs to find potential self-murder victims and terrorists. Microblogging makes difficult to apply techniques such as simple pattern matching, parsing, spelling, and knowledge reasoning using the semantic web. One of the popular applications for sentiment analysis is to solve the problems of major airline problems to classify positive, negative, and neutral tweets. Twitter data was scraped from February 2015 and contributors were asked to first classify positive, negative, and neutral tweets, followed by categorizing negative reasons (such as “late flight” or “rude service”). In this paper, we aim to analyze how travelers mentioned their feelings on Twitter in February 2015. It would fascinate for airlines to use this free data to provide better service to their customers.

In this paper, we present the sentiment analysis system for performance investigation using microblogging data. We use several preprocessing and feature extraction techniques to optimise the sentiment analysis. The impact of the applied pre-processing and feature extraction approaches on enhancing the sentiment analysis system performance is compared and discussed. The applied feature extraction approaches are Bag of Words (BOW), Term Frequency-Inverse Document Frequency (TF-IDF), N-Gram, Word2Vector, and Doc2Vector. And the applied classification algorithms are XGBoost, Random Forest (RF), Logistic Regression (LR), support vector machine (SVM), and Naive Bayes (NB). Using the airline dataset selected from Twitter, the devolved sentiment analysis system has been evaluated.

The rest of the paper is organised as follows. Section II presents several kinds of literature related to this work. In Section III, the theory and concepts of feature extraction and classification approaches are explained. Section IV, discuss experimental results and evaluation for the sentiment analysis system. Finally, in Section V, the conclusions and future work directions are summarised.

2. **Related Work**

The development of the sentiment analysis system is conducted as in Fig. 1. The input dataset is captured and collected from Twitter. Input dataset is preprocessed by several techniques. The most informative feature extraction approaches apply to pre-processed data. Then, building the fundamental process of the system named sentiment analysis and evaluates the performance of prediction data results. Many standard criteria are used to assess the prediction of sentiment classification performance and show its efficacy. In recent years, many research works together and determines the sentiment of tweets, and a good number of text mining algorithms have analysed sentiments as in [10]. Researchers usually incorporate methodologies of feature extraction with machine learning techniques in building sentiment analysis systems to achieve effective classification performance [5].

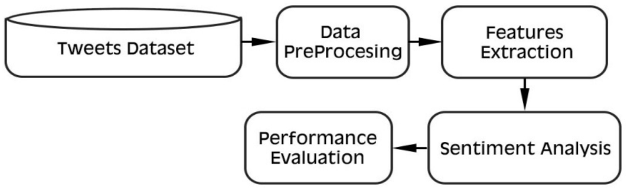


Fig. 1. General Processes of Sentiment Analysis System [5]

Performance comparison of different text representation techniques, such as TF-IDF, and Word2Vec was studied in [1]. Part-of-Speech tagging not effective in sentiment analysis by comparing different feature extraction techniques such as TF-IDF, and Word2Vec. Word2Vec achieves the highest accuracy in classifying compare to other feature techniques. The weakness is the insufficient dataset to train. So, only a portion of the dataset is randomly chosen to train the system.

Dense Cohort of Terms (dCoT), an unsupervised algorithm to learn improved sparse Bag of Words (sBoW) document features, was proposed in [2]. With this approach, dCoT learns to reconstruct frequent words from co-occurring infrequent words and maps the high dimensional sparse sBoW vectors into a low-dimensional dense representation. Results show that, on several benchmark datasets, that dCoT features significantly improve the classification accuracy across several document classification tasks.

An improved Neural Bag-of-Words (NBOW) model was proposed in [6]. The word importance weights are learned by introducing a new weighted sum composition of the word vectors. Results show that, with experiments on standard topic and sentiment classification tasks, the proposed model learns meaningful word importance for a task and gives the best accuracies among the BOW approaches. Also, show that the learned word importance weights are comparable to TF-IDF based word weights when used as features in a BOW SVM classifier.

A novel Cross-media Bag-of-words Model (CBM) for Microblog sentiment analysis was proposed in [11]. In this model, the text and image of a Weibo tweet represent a unified Bag-of-words representation. Based on this model, Logistic Regression is used to classify the Microblog sentiment. It performs well in the sentiment classification task since it doesn't require the conditional dependence assumption. Also, SVM and Naïve Bayes were used to make a comparison.

An improved TF-IDF algorithm (TF-IDCRF) that considers the relationships between classes to complete the classification of texts was proposed in [12]. By modifying the calculation formulas of IDF to correct the problem of insufficient classification of feature categories, the Naive Bayes classification algorithm was used to complete the classification. Finally, the proposed algorithm was compared with two other improved TFIDF algorithms. The results of the three text classification evaluation indicators show that the proposed algorithm has certain advantages in text classification.

An approximate version of the TF–IDF measure suitable to work on continuous data stream was proposed in [13]. The algorithm for the calculation of this measure makes two assumptions: a fast response is required, and memory is both limited and infinitely smaller than the size of the data stream. Results show that the approximate version of the TF–IDF measure performs at a level that is comparable to the solution of the precise TF–IDF measure. Based on Term Frequency-Inverse Document Frequency (TF-IDF) and Support Vector Machine (SVM), a news classification method was proposed in [14]. This approach comprises three distinct steps: text preprocessing, feature extraction based on TF-IDF, and classification based on SVM. The approach was evaluated using two BBC datasets and five groups of 20 Newsgroup datasets. The classification precisions were got as 97.84% and 94.93% for BBC and 20 Newsgroup datasets, respectively.

A technique for text sentiment classification using term frequency-inverse document frequency (TF-IDF) along with Next Word Negation (NWN) was proposed in [15]. The performances of the binary bag of words, TF-IDF, and proposed technique for text classification compared. This technique is then applied to three different text mining algorithms. Results show that the Linear Support vector machine (LSVM) is the most appropriate to work with the proposed technique. Also, the achieved results show a significant increase in accuracy compared to earlier methods.

The sentiments of the users were studied by analysing the Twitter data in [16]. A classification model named SVM is used in the existing framework for classifying input data into seven classes using the SANTA Tool. The proposed study replaces the SVM classification model with the KNN classification model. This classification model classifies input data into seven classes. For feature extraction, the N-Gram method used. The tweets are classified into positive, negative, and neutral classes by employing the k-nearest neighbour classification model. The performance analysis of both approaches is performed based on accuracy.

3. **Features Extraction For Sentiment Analysis System**

In this section, five steps apply to develop our proposed system. Fig. (2) illustrates the framework of the sentiment analysis system of this paper.

3.1. Data Preprocessing

Data Pre-processing in a sentiment analysis system refers to preparing the input dataset to make it suitable for classifying and training machine learning systems to predict the testing data. Explore and clean the data are the first step as shown in Fig.2.

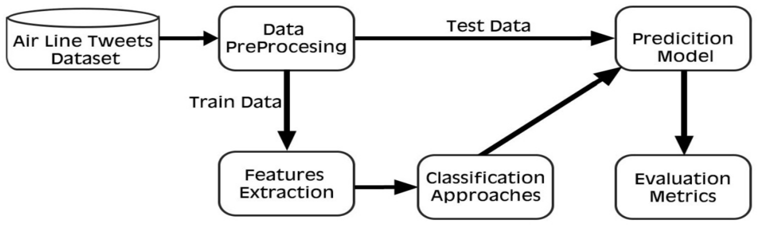


Fig. 2. Framework of sentiment analysis system

3.2. Features Extraction

Feature extraction in machine learning starts from an initial set of preprocessed data and builds features intended to be useful and non-redundant. The third step in this system is to apply feature extraction algorithms to take the input of training data and apply five algorithms, as shown in Fig. 2. This paper uses five feature extraction algorithms (bag of words, N-gram, TF-IDF, Doc2vec, and Word2vec) and comparing the results.

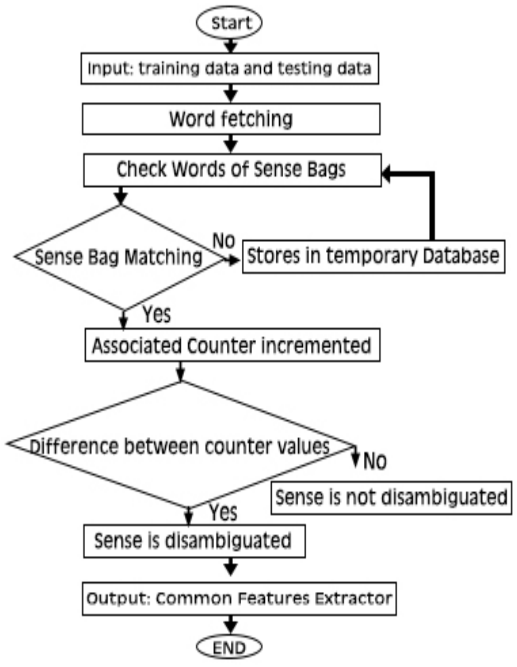
3.2.1. Bag Of Words (BOW)

The bag of words algorithm is a simplifying representation used in information retrieval and natural language processing. Fig. 3. shows the flowchart bag of words. Word is fetching from the training and testing data and check if these words bag. If, 'yes' the associated counter incremented one value. If, 'no' store it in the temporary database and check words bag again. After these steps, check the difference between counter values, and extract the features of the dataset.

3.2.2. TF-IDF

TF-IDF is a numerical statistic, intended to reflect how important a word is to a document in a corpus. That calculates the inverse likelihood of finding a word in a document. Term frequency (TF) represents a repeated number of times a term or word in a document, and inverse document frequency (IDF) represents the inverse frequency of a document [15]. The score of any word in any document can be represented as per the following equation:

$$TFIDF\ (word, \text{doc}) = TF\ (word, \text{doc}) * IDF\ (\text{word}) \tag{1}$$



Start
Input: training data and testing data
Word fetching
Check Words of Sense Bags
Sense Bag Matching
No
Stores in temporary Database
Yes
Associated Counter incremented
Difference between counter values
No
Sense is not disambiguated
Yes
Sense is disambiguated
Output: Common Features Extractor
END

Fig. 3. Flowchart of BOW algorithm

The methods for word frequency (TF) and inverse tweet frequency (IDF) are as follows [12]:

$$\text{tf}_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{2}$$

Where n_{ij} is the number of appearances of the word t_i in doc d_j, and $\Sigma_k\ n_{kj}$ is the sum of the appearances of all words in the doc d_j [12].

$$\text{idf}_i = \log \frac{|D|}{|\{j : t_i \in d_j\}|} \tag{3}$$

Where, |D| represents the number of the dataset in the framework, | *{j: ti ∈dj}* | is the tweet number containing the word tp. If the word is not in the framework, that will cause the dividend to be zero. So use, 1+| *{j: ti ∈ dj}* |. TF-IDF algorithm calculates the term frequency and orders the word by term frequency (TF) from the training and testing data. Then, calculate the n-words weight in each tweet and build vectors for tweets. Then extract the features of the dataset.

3.2.3. N-Gram.

N-gram is a contiguous sequence of n items from a sample of text. These items include letters, words, and phonetics upon application [1]. For microblogging N-grams, used to splitting a tweet into substring of fixed length. It is also called, 'unigram', 'bigram', and, 'trigram'. Then the size of *N* is 1, 2, and 3. This paper uses a bigram with a fixed length (1,2).

3.2.4. Word Embedding.

In natural language processing (NLP), word embedding is used for the representation of words for text analysis. Word embeddings can be got using a set of language modelling and feature learning techniques where words from the vocabulary are mapped to vectors of real numbers. It involves the mathematical embedding from space with many dimensions per word to a continuous vector space with a much lower dimension. Mapped each word one vector and learned the vector values in a resemble neural network with a similar meaning word to have a similar representation [17]. There are two algorithms in the word embedding feature extraction for text (Word2Vec) and (Doc2Vec).

3.2.4.1 Word 2 Vector (W2V)

Word2vec algorithm is based on a word appearing around the word in the tweet. It gives an individual weight to each word. Features of the word2vec algorithm represent a vector that means the distinct word with a particular list of numbers. This paper uses the average weight of the Word2vec algorithm to target learning word relationships of the tweet by using a neural network model from a large corpus. Word2vec can discover similar words for a partial sentence or suggest additional words. The vectors are chosen in mathematical function to show the similarity semantic level between the words described by those vectors.

3.2.4.2 Doc 2 Vector (D2V)

Doc2vec is an unsupervised learning approach to generate vectors for sentence, paragraphs, or documents. The input (tweets) is varied while the output is fixed-length vectors. First, pass the training data to build vocabulary and request the training phase to compute word vectors. Then, encode it by providing training testing data, and pass vectors to a classifier of sentiment analysis. In the proposed system, use PV-DM in the Doc2Vec algorithm. The Doc2vectors are achieved by training a neural network.

3.3. Sentiment Classification

Sentiment analysis allows us to gain a view of the broader public opinion of a specific topic, as discussed in [17]. The sentiment analysis is the fourth step in our system as shown in Fig.2. This paper uses five classifications approach to evaluate the proposed sentiment analysis system (XGBoost, Random forest (RF), Support Vector Machine (SVM), Naive Bayes (NB), and Logistic regression (LR)) and comparing the results.

3.3.1 XGBoost

XGBoost is the machine learning algorithm under the gradient boosting framework. Data scientists using the XGBoost to provide parallel tree boosting and achieve state-of-the-art results on many machine learning challenges. In this work, the XGBoost algorithm is used to classify the airline dataset [4,5] with using five feature extractions. Fig.4. shows the flowchart of the XGBoost classification algorithm. Read extracted features and trained label data, then define a set of hyper-parameters and using the XGBoost approach to cross-validation. The output of this step is the probability values of testing labels.

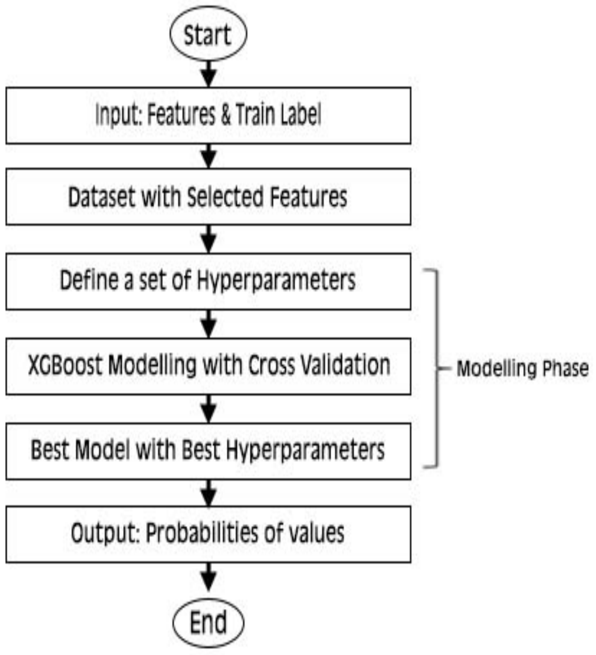

Start
Input: Features & Train Label
Dataset with Selected Features
Define a set of Hyperparameters
XGBoost Modelling with Cross Validation
Best Model with Best Hyperparameters
Modelling Phase
Output: Probabilities of values
End

Fig. 4. Flowchart of XGBoost algorithm

3.3.2 Random Forest (RF)

Random Forest is a supervised classification algorithm. It comprises many decision trees. This algorithm runs efficiently on large data sets, and uses features randomness, and bagging when building each tree. It tries to create a forest of trees to predict more accurately than any individual tree. For a set of tweets *t1, t2,.. tn*, and their sentiment labels *s1, s2,... sn*. Gets selects a random sample (*Tb, Sb*) with replacement. The output of this step will be the object's positions and the probabilities of values of test labels. All classification trees *f (b)* are trained using a different random sample *(Tb, Sb)* where *b* ranges from *1,... n.* Finally, a majority vote is taken of predictions of these B-trees, as discussed in papers [5,9].

3.3.3 Logistic Regression (LR)

Regression is a statistical process for evaluating the relationships among variables, the output often to predict any outcome (test labels). Logistic regression is also called maximum entropy. It is a supervised machine learning algorithm and regression, where the binary is the target variable. It can be used in several problems, including text classification. Fig.5. shows a flowchart of a logistic regression classification algorithm. Read input features, trained labels, and compute the regression coefficients of training data. It finds the relation between the training and testing data. The output will be the object's positions, and the probabilities values, and the prediction of testing labels.

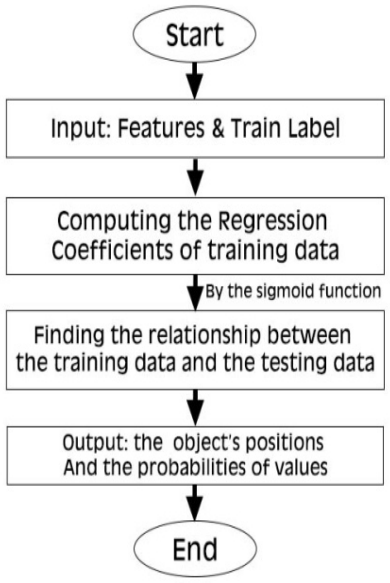

Start
Input: Features & Train Label
Computing the Regression Coefficients of training data
By the sigmoid function
Finding the relationship between the training data and the testing data
Output: the object's positions And the probabilities of values
End

Fig. 5. Flowchart of logistic regression algorithm

3.3.4 Support Vector Machine (SVM)

Support Vector Machine is a supervised classifier, that finds an optimal hyperplane that maximizes the margin between the two different classes. It classifies new samples or vectors by specifying where on the hyperplane based on the lowest risk principle structural. This algorithm creates a hyperplane separating the positive and negative samples during training data. The kernel transforms data not linearly separable in dimensional space to a higher dimension where it is linearly separable, as discussed in [19]. Fig.6 shows the flowchart of the SVM classification algorithm. First, read the features and trained labels. This paper uses linear SVM, and constructs the optimal classification surface, and solves for the optimal decision function. The output is the probability values of the test labels. Support vector machine should be used in a binary target variable, when the feature-to-row ratio is high, and in complex relationships, and many outliers.

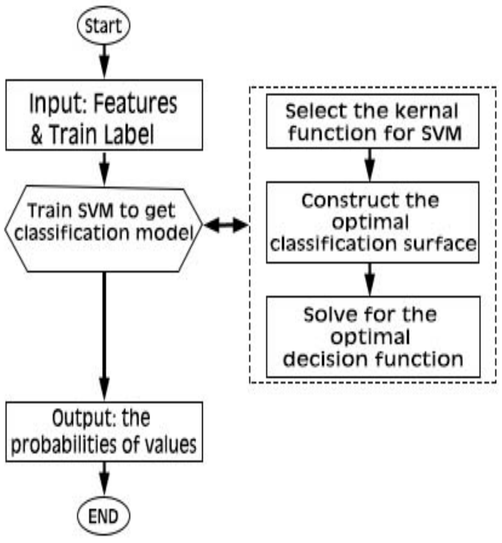

Start
Input: Features & Train Label
Train SVM to get classification model
Select the kernal function for SVM
Construct the optimal classification surface
Solve for the optimal decision function
Output: the probabilities of values
END

Fig. 6. Flowchart of SVM algorithm

3.3.5 Naïve Bayes (NB)

The theorem of Naïve Bayes uses the features in the dataset that are mutually independent. This algorithm can exceed the most powerful alternatives with small sample sizes. Because it is being relatively robust, easy to implement, fast, and accurate. The occurrence of one feature in Naïve Bayes does not affect the probability of occurrence of the other feature. This paper uses multinomial Naïve Bayes with a feature vector that depends on the term represents the number of times it appears. This algorithm deals with real-time inputs like tweets. The prediction is done for giving test classes. Fig.7 shows a flowchart of the Naive Bayes classification algorithm. In the first step features and trained labels are reading. For each attribute A, traverse attributes list for A of the examined node, then find probability using the value of A to be in a class, and then update class for A check if all values in A have? If 'No', then find probability using the value of A to be in a class, else check if there any attribute, if, yes, then read next attribute, if-else, the output is the probability values.

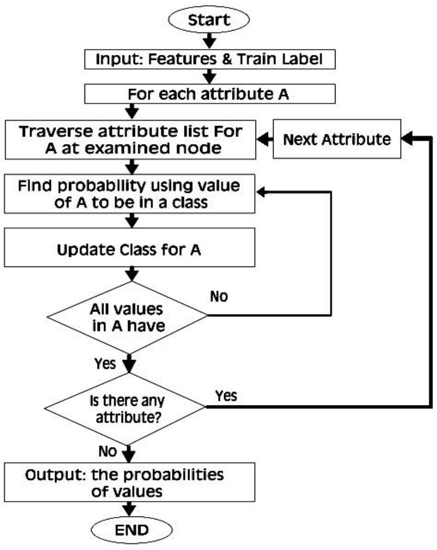

Start
Input: Features & Train Label
For each attribute A
Traverse attribute list For A at examined node
Next Attribute
Find probability using value of A to be in a class
Update Class for A
All values in A have
No
Yes
Is there any attribute?
Yes
No
Output: the probabilities of values
END

Fig. 7. Flowchart of Naïve Bayes algorithm

4. Experimental Results And Discussion

4.1. Experimental Setup and Data set

Experiments are implemented on a laptop with an Intel ® Core ™ i7 −2670M CPU 2.20GHz, 64-bit Windows 7 operating system, and 16 GB RAM. Also, some tools were used such as Python 3.7, Jupiter lab, Matplotlib, NumPy, pandas, anaconda3 (version 4.5.12), and Sci kit learn.

The Airline dataset contains 14640 reviews, with positive reviews (2363), negative reviews (9178), and neutral reviews (3099). For pre-processing dataset, it is split into two types of data (training and testing data), where 80% is training data and 20% use as testing data [21].

4.2. Performance Evaluation Metrics

To evaluate proposed system, standard metrics used to calculate the performance of system (accuracy, precision, recall, and f-score).

- *Accuracy*

Accuracy estimates how many tweets are predicted correctly as belonging to a category of the positive or negative word of tweets in the corpus [1,8].

$$accuracy = (Correctly\ Labeled\ Example)/\ (Total\ Example) \tag{4}$$

- *Precision*

Precision estimates how many tweets are predicted correctly as belonging to a category (positive, or negative). All the tweets that are predicted (correctly or incorrectly) as belonging to the category as presented in the airline dataset [4,8].

- *Recall*

Recall estimates how many tweets are correctly predicted as belonging to positive or negative sentiment. All the texts should have been predicted as belonging to the category. The more data feed classifiers, the better the recall will be [8].

- *F1-score*

F1-score is another metric that used for evaluating the accuracy of the proposed system with regarding to both the precision and the recall rate values [12].

4.3. Discussion and Results

Tables 1, 2, 3, 4, and 5 show the results and performance of a proposed system using different techniques of text representation and classification algorithms. Results in Table 1, show that, in the case Bag Of Words features, the system achieves the highest performance using the random forest (RF) technique for sentiment analysis with an accuracy of 93%. Comparing the results of the classification techniques, Naive Bayes is the lowest performance with an accuracy of 77%, as shown in Fig. (8).

Table 1. *Evaluation Metrics Using BOW Features*

	Bag Of Words			
	Accuracy	F-Score	Precision	Recall
XGBoost	0.87	0.87	0.87	0.87
RF	0.93	0.93	0.93	0.93
LR	0.80	0.80	0.79	0.80
SVM	0.80	0.80	0.80	0.80
NB	0.77	0.76	0.76	0.77

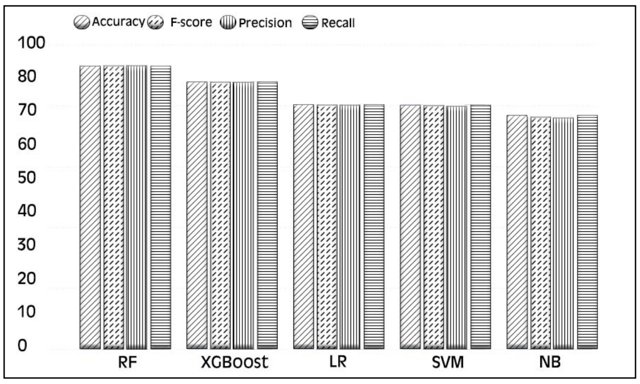


Fig. 8. Results of accuracy, f-Score, precision, and recall for (Bow) features

In Table 2, with TF-IDF features approach, the developed system achieves the highest performance using the random forest (RF) technique for sentiment analysis with an accuracy of 93%. Comparing the results of the classification techniques (LR, RF, SVM, and XGBoost), Naive Bayes is the lowest performance with an accuracy of 75%, as shown in Fig. 9.

Table 2. *Evaluation Metrics Using TF-IDF Features*

	TF-IDF			
	Accuracy	F-Score	Precision	*Recall*
XGBoost	0.91	0.91	0.91	0.91
RF	0.93	0.93	0.93	0.93
LR	0.80	0.79	0.79	0.80
SVM	0.80	0.79	0.79	0.80
NB	0.75	0.72	0.75	0.75

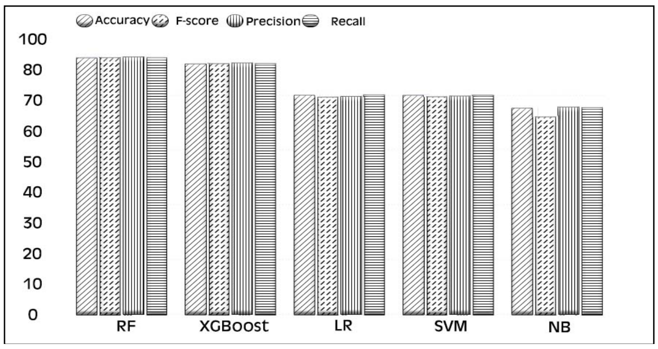


Fig. 9. Results of accuracy, f-score, precision, and recall for TF-IDF features

In Table 3, in the case of the N-gram features, the system achieves the highest performance using the XGBoost technique sentiment analysis with an accuracy of 94%. This paper compares the results of the classification techniques (LR, RF, SVM, and XGBoost). Naive Bayes is the lowest performance with having an accuracy of 67%, as shown in Fig.10.

Table 3. *Evaluation metrics using N-Gram features*

	N-gram			
	Accuracy	F-Score	Precision	Recall
XGBoost	0.94	0.94	0.94	0.94
RF	0.94	0.93	0.94	0.94
LR	0.75	0.74	0.74	0.75
SVM	0.75	0.74	0.74	0.75
NB	0.67	0.67	0.66	0.67

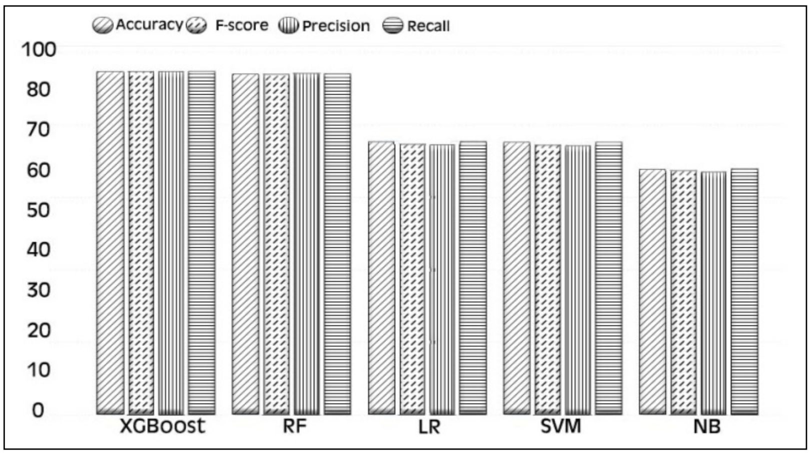


Fig. 10. Results of accuracy, f-Score, precision, and recall for n-gram features

In Table 4, with the Word2Vec feature, the system achieves the highest performance using XGBoost and RF techniques for sentiment analysis with an accuracy of 95%**.** Comparing the results of the classification techniques (LR, RF, SVM, and XGBoost), Naive Bayes is the lowest performance with an accuracy of 70%, as shown in Fig.11.

Table 4. *Evaluation metrics using Word2Vec features*

	Word2Vec			
	Accuracy	F-Score	Precision	Recall
XGBoost	0.95	0.95	0.95	0.95
RF	0.95	0.94	0.95	0.95
LR	0.77	0.76	0.76	0.77
SVM	0.78	0.77	0.77	0.78
NB	0.70	0.69	0.68	0.70

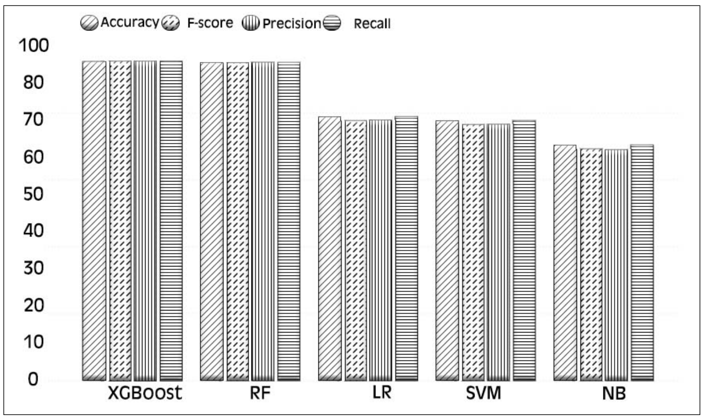


Fig. 11. Results of accuracy, f-Score, precision, and recall for Word2Vec features

In Table 5, with Doc2Vec features, the system achieves the highest performance using the XGBoost technique for sentiment analysis with an accuracy of 81%. Comparing the results of the classification techniques (LR, RF, SVM, and XGBoost) Naive Bayes is the lowest performance with an accuracy of 61%, as shown in Fig.12.

Table 5 Evaluation metrics using Doc2Vec features

	Doc2Vec			
	Accuracy	F-Score	Precision	Recall
XGBoost	0.81	0.79	0.80	0.81
RF	0.71	0.66	0.72	0.71
LR	0.69	0.67	0.67	0.69
SVM	0.68	0.61	0.66	0.68
NB	0.60	0.61	0.64	0.60

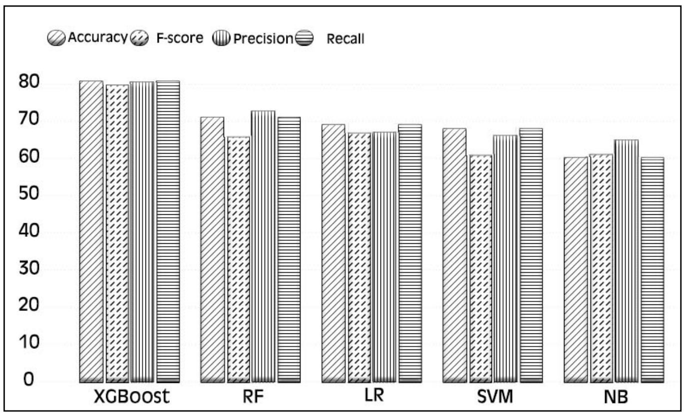


Fig. 12. Results of Accuracy, F-Score, Precision, and Recall for Doc2Vec Features

4.4. Comparison with related system

This paper compares the results between the related system in [4] and this work. For pre-processing in [4], firstly tokenize and remove stop words, URLs, digits, punctuation, and emoticons. In this paper, making distinct steps, first, system read data and save the dataset in a CSV file and make process called part of speech (POS) tagging, removing stop words dictionary, and numbers. Various forms of the same word were lemmatized by converting them to the keyword using the WordNet Lemmatize module of the Natural Language Toolkit Python library. Then, making the stemming data, visualize data, and normalized the data output. Also, this work uses BOW, TF-IDF, N-gram, Word2Vec, and Doc2Vec, as features extraction and RF, LR, SVM, and NB as classification data. Table.6, shows a comparison between the related system in [4] and the devolved system in this paper.

Results in Table 6 show that with Random forest, the best result of accuracy was 93% with BOW and 93% with TF-IDF. With using LR, the best accuracy was 80% with BOW and 80% with TF-IDF. But with using SVM, the best accuracy was 80% with BOW, and 80% with TF-IDF. Also in the case using NB, the best accuracy result was 77% with BOW, and 75% with TF-IDF. In this work, N-gram, Word2Vec, and Doc2Vec feature extractions are used for more performance investigation. In case of RF, the proposed system achieves the best accuracy result (94% with n-gram, 95% with Word2Vec), and the lowest result was 71% with Doc2Vec.

Table 6. Comparison between related system and proposed system

	Related system Accuracy In [4]	Proposed system Accuracy				
		BOW	TF-IDF	N-gram	Word2Vec	Doc2Vec
RF	76%	93%	93%	94%	95**%**	71%
LR	78%	80%	80%	75%	77%	69%
SVM	78**%**	80**%**	80%	75%	78%	68%
NB	52%	77%	75%	67%	70%	60%

5. Conclusion

Sentiment analysis is the process of detecting positive or negative sentiment in text. We present the sentiment analysis of microblog data systems such as Twitter APIs. In this paper, several pre-processing and feature extraction techniques apply to optimize the sentiment analysis system. We compare different sentiment analysis approaches using a dataset of US Airlines from Twitter. Evaluation results stated that in case, Word2Vec feature extraction with (XGBoost and random forest) classifiers achieve the highest performance. And in case, Doc2Vec and the Naive Bayes classifier achieve the lowest performance. Future work is directed toward implementing deep learning techniques for building the sentiment analysis system.

References

[1] Lim, Yu Qing, Lim Chun Ming, Gan Keng Hoon, Samsudin Nur Hana, "Text Sentiment Analysis on Twitter to Identify Positive or Negative Context in Addressing Inept Regulations on Social Media Platform." IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE), 2020.

[2] Zhixiang Xu, Minmin Chen, Kilian Q. Weinberger, Fei Sha, "An alternative text representation to TF-IDF and bag of words." arXiv, 2013.

[3] Berry, Michael W. Kogan, Jacob. "Text mining: applications and theory." John Wiley & Sons, 2010.

[4] Ullah Mohammad Aman, Marium Syeda Maliha, Begum Shamim Ara, Dipa Nibadita Saha," An algorithm and method for sentiment analysis using the text and emoticon." ICT Express, 2020.

[5] Wan and Q. Gao. "An ensemble sentiment classification system of Twitter data for airline service analysis." International conference on data mining workshop (ICDMW), 2015.

[6] Sheikh Imran, Illina Irina, Fohr Dominique, Linares Georges, "Learning word importance with the neural bag of words model." Workshop on Representation Learning for NLP,2016.

[7] Chen, Tianqi, guestrin, Carlos. "Xgboost: A scalable tree boosting system." ACM signed an international conference on knowledge discovery and data mining. 2016.

[8] Ramadhan, w. P. novianty, stmt astri, setianingsih, Stmt casi. "Sentiment analysis using multinomial logistic regression." IEEE International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC), 2017.

[9] Sangeeta rani, Nasibsingh gill. "Hybrid model for Twitter data sentiment analysis based on the ensemble of dictionary-based classifier and stacked machine learning classifiers-SVM, KNN and C5." Journal of Theoretical and Applied Information Technology, 2020.

[10] El-din, Doaa Mohey. "Enhancement bag of-word model for solving the challenges of sentiment analysis." International Journal of Advanced Computer Science and Applications, 2016.

[11] Wang Min, Cao Donglin, Li Lingxiao, Li Shaozi, Ji Rongrong, "Microblog sentiment analysis based on cross-media bag of-word model." Proceedings of the international conference on internet multimedia computing and service, 2014.

[12] Liu Cai-zhi, Sheng Yan-xiu, Wei Zhi-qiang, Yang Yong-Quan, "Research of text classification based on improved TF-IDF algorithm." IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE), 2018.

[13] Erra Ugo, Senatore Sabrina, Minnella Fernando, Caggianese Giuseppe, "Approximate TF-IDF based on topic extraction from massive message stream using the GPU." Information Sciences, 2015.

[14] Dadgar, Seyyed Mohammad Hossein; araghi, Mohammad Shirzad; Farahani, Morteza Mastery. "A novel text mining approach based on TF-IDF and Support Vector Machine for news classification." International Conference on Engineering and Technology (ICE TECH), 2016.

[15] Das, Bijoyan, Chakraborty, Sarit. "An improved text sentiment classification model using TF-IDF and the next word negation." arXiv, 2018.

[16] Parikh, Satyen M, shah, Mitali K. "Classification of Sentiment Analysis Using Machine Learning." International Conference on Innovative Data Communication Technologies and Application. Springer, Cham, 2019.

[17] Mikolov Tomas, Chen Kai, Corrado Greg, Dean Jeffrey, "Efficient Estimation of Word Representations in Vector Space." arxiy, 2013.

[18] Ahuja Ravinder, Chug Aakarsha, Kohli Shruti, Gupta Shaurya, Ahuja Pratyush, "The impact of feature extraction on the sentiment analysis." Procedia Computer Science, 2019.

[19] Martineau, Justin, Finin, Tim. "Delta tfidf: An improved feature space for sentiment analysis." UMBC Student Collection, 2009.

[20] Samantha, Rai B. Shetty, Sweekriti M. AI, Prakhyath. "Sentiment Analysis Using Machine Learning Classifiers: Evaluation of Performance.", IEEE International Conference on Computer and Communication Systems (ICCCS),2019.

[21] https://www.kaggle.com/crowdflower/twitter-airline-sentiment acsses in 2 Dec. 2020.