ISSN- 2394-5125 VOL 10, ISSUE 02, 2023

NLP-BASED EXTENDED LEXICON MODEL FOR SARCASM DETECTION WITH TWEETS AND EMOJIS

K. Ritika[1], N. Deekshitha Reddy[1], N. Akshaya Reddy[1], K. Jaya Rajan[2]

[1,2]Department of Information Technology

[1,2]Malla Reddy Engineering College for Women (A), Maisammaguda, Medchal, Telangana

ABSTRACT

Lexicon algorithm is used to determine the sentiment expressed by a textual content. This sentiment might be negative, neutral, or positive. It is possible to be sarcastic using only positive or neutral sentiment textual contents. Hence, lexicon algorithm can be useful but insufficient for sarcasm detection. It is necessary to extend the lexicon algorithm to come up with systems that would be proven efficient for sarcasm detection on neutral and positive sentiment textual contents. In this paper, two sarcasm analysis systems both obtained from the extension of the lexicon algorithm have been proposed for that sake. The first system consists of the combination of a lexicon algorithm and a pure sarcasm analysis algorithm. The second system consists of the combination of a lexicon algorithm and a sentiment prediction algorithm. Finally, naive bayes are used to predict sarcasm detection using pretrained features.

Keywords: Lexicon algorithm, sarcasm detection, sentiment prediction algorithm, a pure sarcasm analysis algorithm.

1.INTRODUCTION

Communication is the process of exchanging information. As time goes by, many ways and platforms of communication are being developed. Since the industrial revolution, the original way of communicating; face-to-face communication has been used as a model to develop the various ways of communicating known to date. Transposing the principles and codes of natural face-to-face communication to today's online communication is a major challenge for developers. Sarcasm is the communication practice that consists of meaning the opposite of what is said to mock or insult someone [1]. Sarcasm makes use of positive lingual contents to convey a negative message. Different types of approaches have been developed to implement sarcasm detection on online communication platforms. However, the levels of efficiency of these approaches have been the principal worries of developers. In this paper, propositions are made on how the lexicon algorithm can be extended to come out with systems that would be proven more efficient for sarcasm detection on textual contents. The magnitude of data generated through social media today is colossal. They are good for data analysis since they are very personal [1]. For years, companies have been analyzing this type of data to leverage their position in the market of their choice [2]. This field is called sentiment analysis [3]. On the other hand, sarcasm is defined as a positive utterance or sentence with underlying negative intention [4]. It is regarded as one of the most challenging issues in the Natural Language Processing (NLP) field [5]. Spotting and handling them correctly is crucial in an automated NLP system, mainly since sarcasm can flip the polarity of a sentence [5], [6]. Traditional studies as from Davidov et al. [7] and Riloff et al. [4] used rule-based techniques to tackle sarcasm detection. However, more recent studies [8], [9] have shifted towards deep learning to automatically detect the discriminatory features. In this work, the features extracted by a deep learning architecture is combined with the ones that are manually created through specific contextual understanding and processes. Tweets are used as the main source of input. Unlike in writing, different tones and gestures can be utilized to portray sarcasm in the real world [7]. As a countermeasure for this short-coming, writers of tweets tend to leave contextual clues for sarcasm in creative ways such as hashtags and hyperboles [4], [7]. This kind of

ISSN- 2394-5125 VOL 10, ISSUE 02, 2023

clue is what this work is trying to find and exploit. Several NLP studies have tried to come up with automatic detection models for sarcasm. Features are either discovered through deep learning or manual handcrafting (feature engineering) methods [10], never both. There is too much reliance on deep learning architecture for some researchers [8], [9], and vice-versa for manual handcrafting. This leaves some room for experiments. The method also significantly improves the F1-measure from the existing study using the same dataset. This work also demonstrates the generality of a deep learning architecture. For future work, a few datasets will be used to further generalize the comparisons. The process of extracting and gathering meaningful features could be expanded further.

2.LITERATURE SURVEY

Prabu palanisamy et. al presented our system that they used for the SemEval-2013 Task 2 for doing Sentiment Analysis for Twitter data. They goted an F-score of 0.8004 on the test data set. They presented a lexicon-based method for Sentiment Analysis with Twitter data. They provided practical approaches to identifying and extracting sentiments from emoticons and hashtags. They also provided a method to convert non-grammatical words to grammatical words and normalize non-root to root words to extract sentiments. A lexicon-based approach is a simple, viable and practical approach to Sentiment Analysis of Twitter data without a need for training. A Lexicon based approach is as good as the lexicon it uses. To achieve better results, word sense disambiguation should be combined with the existing lexicon approach.

Anna Jurek et. al we presented a new approach to lexicon-based sentiment analysis of Twitter messages. In the new approach, the sentiment is normalized, which allows us to obtain the intensity of sentiment rather than positive/negative decision. A new evidence-based combining function was developed to improve performance of the algorithm in the cases where a mixed sentiment occurs in a message. The evaluation was performed with the Stanford Twitter test set and IMDB data set. It was found from the results that the two new functions improve performance of the standard lexicon-based sentiment analysis algorithm. It could be noticed that the method is more appropriate for short messages such as tweets. When applied with long documents the method performed significantly better on the sentence than on the document level. Following this, this intention was to investigate the relationship between the amount and the level of negative sentiment related to a public demonstration and the level of violence and disorder during the event. In other words, they aimed to ascertain if sentiment analysis could be applied as a supportive tool while predicting a level of disruption prior to public events. As a first step in this study, we decided to examine Twitter as a source of data. Four different demonstrations were selected, and the negative sentiment related to these events was analyzed over 6 days prior to each event. Following the case study and several analyses they were able to reveal that there was a relationship to some extent between the negative sentiment and the level of disorder during the EDL events. Further research is however required in this area to provide more accurate findings and conclusions. At the current stage we can, however, conjecture that sentiment analysis of social media content can provide valuable, security-related information regarding some upcoming public events. In the next step they wish to collect more data related to public events and further investigate the relationship between negative sentiment and the level of violence and disorder during events. Following this, we aim to develop a predictive model that can be used by police services as a single tool to help indicate violence propensity.

Kiilu et. al evaluated the performance for sentiment classification in terms of accuracy, precision and recall. In this work, they compared various supervised machine learning algorithms of Naïve Bayes' for sentiment analysis and detection of the hate tweets in twitter. Apart from the system's ability to predict for a given tweet is hateful or not, the system also generates a list of users who frequently post such content. This provides us with an interesting insight into the usage pattern of hate-mongers in

ISSN- 2394-5125 VOL 10, ISSUE 02, 2023

terms of how they express bigotry, racism, and propaganda. The experimental results show that the classifiers yielded better results for the hate tweets review with the Naïve Bayes' approach giving above 80% accuracies and outperforming other algorithms. This work had several key weaknesses that can be addressed. One major consideration would be to include emotions and video images in detecting hate tweets among various users in twitter targeting various groups or individuals. Another problem that could be addressed is the limitation of twitter API for commercial research where authorization is limited. Currently Twitter allows users to collect approximately 1600 tweets per day and will only provide data that has been uploaded in the last six days. To gain real value from a sentiment analysis it would be required to have massive amounts of data on the product or service which is currently not available without premium accounts or using third parties. Given the legal and moral implications of hate speech it is important that they can accurately distinguish between the two. Thus, we can say Naïve Bayes' classifier can be used successfully to analyze movie reviews.

Rathan and Suchithra et. al Sarcasm detection on twitter tweets is more complicated has it provides very less detailed results and developing a dictionary for these kind of text documents takes more time and resources. Social media posts are hard to analyze on the phrase or sentence level because of their unique structureand grammar. Since twitter allows users to enter 140 characters processing time also increases. The sarcasm detection was ignored for different languages (except English), repeated tweets and empty or a single letter/word tweet. Finally, by using different types of features and their combinational logic we were able to detect sarcasm in twitter training data set. Preprocessing being the very important part of our project, it was successfully completed. And the results were clean preprocessed and tagged tweets. In this phase they were able to remove anomalies or noises such as hyperlinks, emailed, links and mention. We got 100% accuracy in detection and deletion of these noises. In POS tagging most of the important words were successfully detected whereas the undetected ones were due to misspelled words or the words which may be missing from dictionary, or it may have been prepositions (in, of, as, a, the) which we are not considering overall we got 75% and over detection which gave us a better POS tagging dataset for feature extraction. The algorithm also detected emoticons and renamed them. Finally, the result data set is moved to post processing Out of 300 tweets we considered 100 #sarcastic tweets, 100 non-sarcastic tweets and 100 sarcastic tweets with no # tags. The results were found to be extraordinary. The algorithm was able to classify accurately over this combined dataset. We found 68% sarcastic in one dataset and 71.35% in another dataset. The future work will be focused on backtracking of tweets (analyzed based on user's past replies and comments) and multilingual language support.

3. Proposed Method

Lexicon algorithm is used to determine the sentiment expressed by a textual content. This sentiment might be negative, neutral, or positive. It is possible to be sarcastic using only positive or neutral sentiment textual contents. Hence, lexicon algorithm can be useful but yet insufficient for sarcasm detection. It is necessary to extend the lexicon algorithm to come out with systems that would be proven efficient for sarcasm detection on neutral and positive sentiment textual contents. In this paper, two sarcasm analysis systems both obtained from the extension of the lexicon algorithm have been proposed for that sake. The first system consists of the combination of a lexicon algorithm and a pure sarcasm analysis algorithm. The second system consists of the combination of a lexicon algorithm and a sentiment prediction algorithm. In this work, the lexicon algorithm has been extended in two ways to generate two systems that could be more efficient for sarcasm analysis, especially on neutral and positive sentiment textual contents.

ISSN- 2394-5125 VOL 10, ISSUE 02, 2023

3.1. First system

The first system (Fig. 1) is the combination of a lexicon algorithm and a pure sarcasm analysis algorithm. This system takes textual contents as input. These contents could be from various social media platforms like Twitter or Facebook. The textual contents are parsed into the lexicon algorithm for polarity computation. Then the positive sentiment contents are parsed into the pure sarcasm analysis algorithm for sarcasm detection. The final output of this system is a list of sarcastic and non-sarcastic lingual contents.

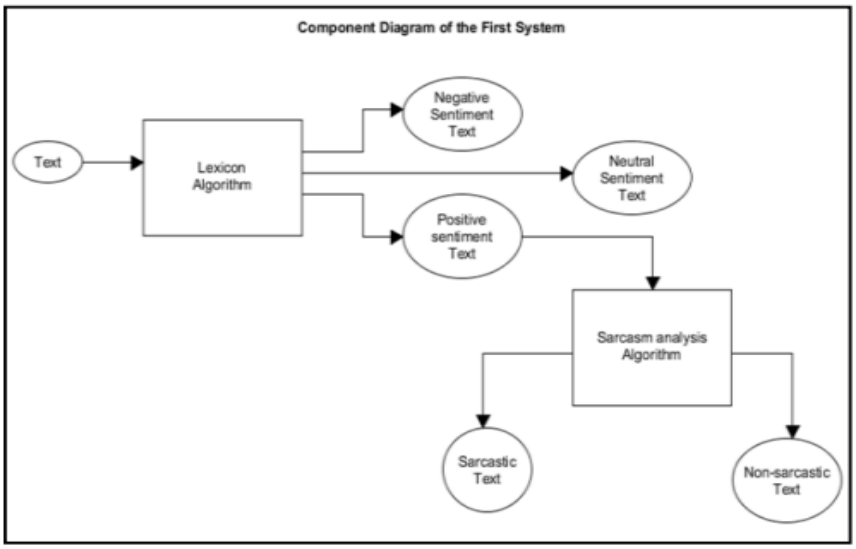

Fig. 1: Component diagram of the first system. An overview of the arrangement of components of the first system.

3.2. Second system

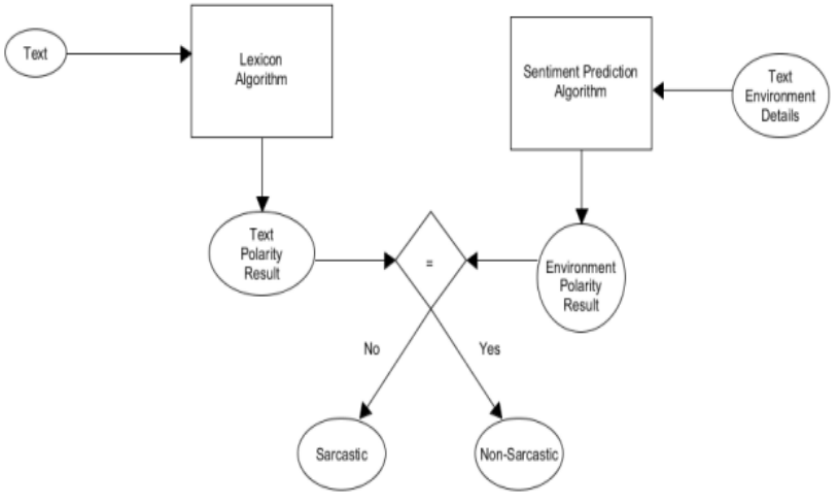

Fig. 2: Component diagram of the second system. An overview of the arrangement of the components of the second system.

The second system is the combination of a lexicon algorithm and a sentiment prediction algorithm. The lexicon algorithm is used here the same way as in the first system. The sentiment prediction algorithm consists of a mechanism that can predict the sentiment of a textual content that would be made under a specific environment. The sentiment prediction algorithm takes as input the details of the environment under which a lingual content would be made notably the state of the context, the

ISSN- 2394-5125 VOL 10, ISSUE 02, 2023

author's knowledge of the domain he/she would talk about, the author's level of education, the author's personality, the author's relationship with his/her interlocutor. The sentiment prediction algorithm processes these details and predicts the sentiment of the textual content that would be formed under that environment. The results from both the algorithms are compared. In Case the results are different for a textual content, this later is classified as sarcastic else it is classified as non-sarcastic (Fig. 2). These environment details are processed based on a training data set that was formed as follow:

- Each environment detail had a polarity. The polarity could be negative, neutral or positive (Table 1).
- An AND operation was performed in between the detail's polarities of each textual content environment to predict the sentiment of the textual content that would be made under each environment. The training data set of the sentiment prediction algorithm was formed of the environment details polarities and their predicted sentiment (Table 2)

All the algorithms used in this paper have been derived from the classic Naïve Bayes algorithm. This algorithm makes use of conditional probability to predict the likeness of future occurrence of events based on their historical information. Naïve Bayes is mainly used for classification purposes. It is an algorithm that discriminates different objects based on certain features. This algorithm is built after the Bayes theorem which assumes that all features within a class are independent from one another and that is why it is known as 'naive'.

Table. 1: Environment details and their polarity values.

Environment Details	**Polarity**		
	Negative (-)	**Neutral (0)**	**Positive (+)**
State of the context (c)	Tensed (c-)	Neutral (c0)	Calm (c+)
Author's knowledge of the domain (k)	Novice (k-)	Fair (k0)	Good (k+)
Author's level of education (le)	Primary (le-)	Secondary (le0)	University (le+)
Author's personality (ap)	Pessimist (ap-)	Realistic (ap0)	Optimist (ap+)
Author's relationship with his/her interlocutor (ri)	Public (ri-)	Just know (ri0)	Close (ri+)

Naïve Bayes is mainly used for classification purposes. It is an algorithm that discriminates different objects based on certain features [11]. This algorithm is built after the Bayes theorem which assumes that all features within a class are independent from one another and that is why it is known as 'naive' [12]. There are several types of Naïve Bayes models [12][13]: - Gaussian Naïve Bayes: where the predictors take up continuous value and are not discrete. - Bernouilli Naïve Bayes: where the parameters of the predictors are Boolean values; 'yes', 'no', '1' or '0'. - Multinomial Naïve Bayes: it is the generalization of Bernouilli where the features used by the classifier are the frequency of objects being processed. Multinomial Naïve Bayes is the model used in this paper. The steps of the Naïve Bayes algorithm can be resumed to the following [12]: Convert the data set into a frequency table, Create a likelihood table, and Use Naive Bayesian equation to calculate the posterior probability for each class.

ISSN- 2394-5125 VOL 10, ISSUE 02, 2023

Table. 2: Training data set.

Environment Details Polarities	Predicted Sentiment
c+ k- le+ ap+ ri+	N (Negative)
c+ k+ le+ ap+ ri+	P (Positive)
c- k- le- ap- ri-	N (Negative)
c- k- le+ ap+ ri-	N (Negative)
c+ k+ le- ap- ri+	P (Positive)
c+ k- le- ap- ri-	P (Positive)
c+ k- le0 ap+ ri+	Ne (Neutral)
c+ k0 le- ap- ri-	Ne (Neutral)
c0 k- le+ ap0 ri+	Ne (Neutral)

3.3 Naïve Bayes

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features. For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods. Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers.[6] Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests.[7] An advantage of naive Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification.

This article discusses the theory behind the Naive Bayes classifiers and their implementation. Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. To start with, let us consider a dataset. Consider a fictional dataset that describes the weather conditions for playing a game of golf. Given the weather conditions, each tuple classifies the conditions as fit("Yes") or unfit("No") for plaing golf. Here is a tabular representation of our dataset.

4. RESULTS

ISSN- 2394-5125 VOL 10, ISSUE 02, 2023

This section gives the detailed analysis of simulation results implemented using "python environment". Further, the performance of proposed method is compared with existing methods using same dataset.

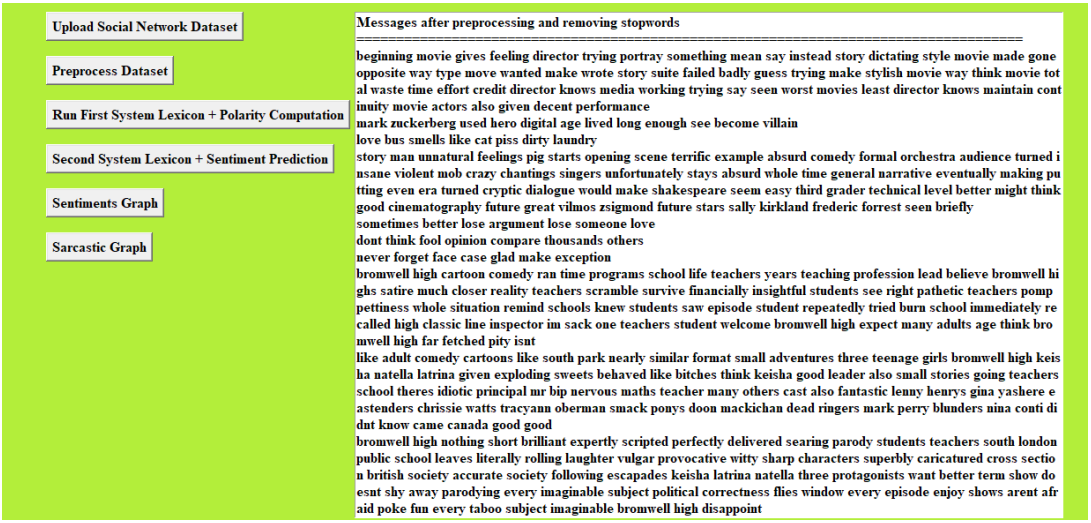

Fig. 3: Run first system lexicon + polarity computation.

In Fig. 3, we can see all messages after removing special symbols and stop words. Now click on 'Run First System Lexicon + Polarity Computation' button to calculate polarity of messages.

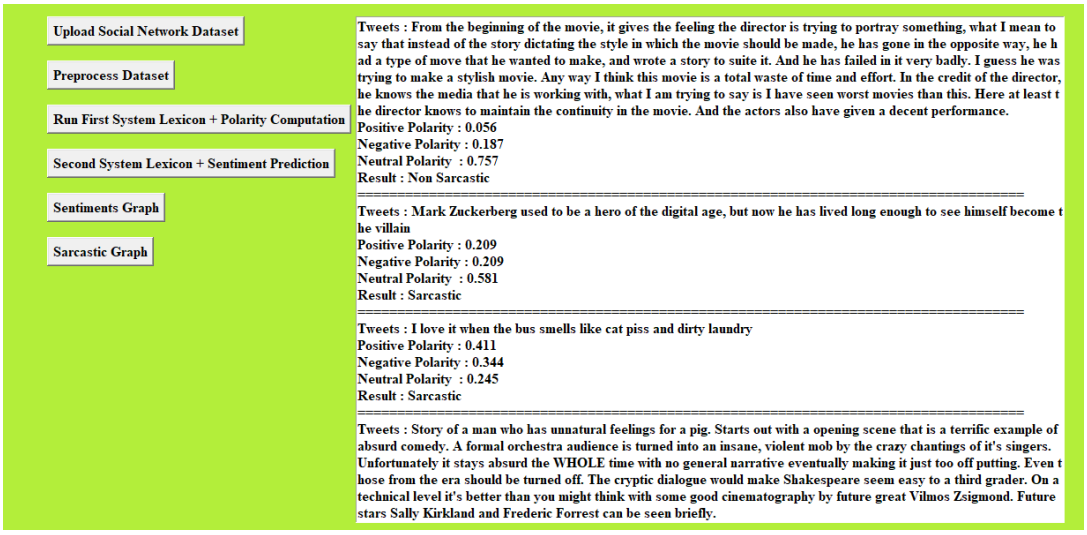

Fig. 4: First system predicted outcomes.

In Fig. 4 for each message, we can see tweet data and positive, negative, and neutral polarity score and the message in tweets is sarcastic or non-sarcastic. The tweets will be classified to positive, negative, or neutral based on its high score for example in first tweet neutral got high score as 0.757 so tweet will consider as neutral. If that neutral tweet contains some negative words, then consider it as sarcastic. You can scroll down to the above screen text area to see all messages details.

ISSN- 2394-5125 VOL 10, ISSUE 02, 2023

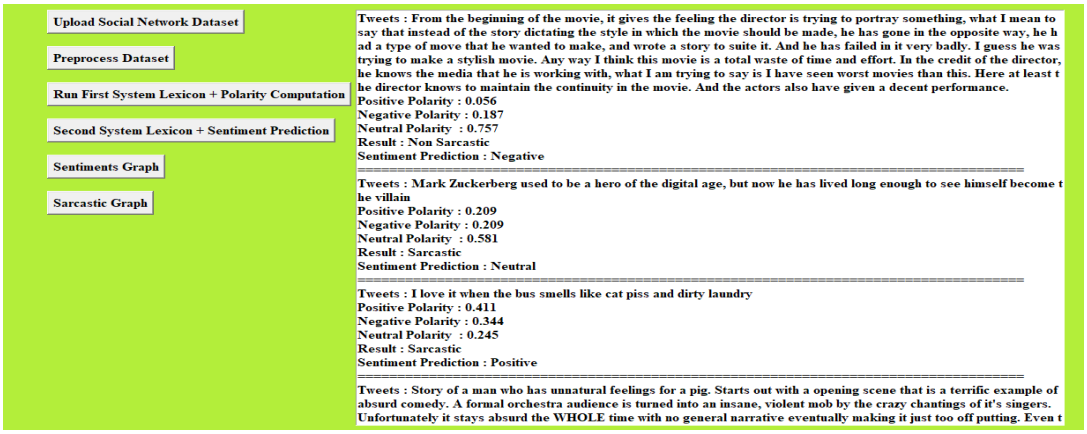

Fig. 5: Proposed predicted outcomes.

In Fig. 5, we can see the same results with extra details such as whether tweet/message is positive or negative or neutral. You can scroll down to the above text area to see all messages. The above Figure shows the results from the sentiment prediction algorithm. For each comment, the result of the sentiment prediction algorithm has been compared with result of the lexicon algorithm. In case these results are similar, the content is considered as non-sarcastic or else it is considered as sarcastic. The environmental details used in this experiment are assumptions that have been made since the required data were not available and the proposed system was an innovative one at the time of experiment.

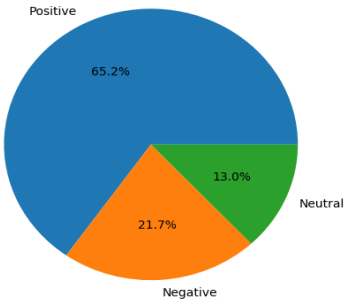

Fig. 6: Sentiment graph.

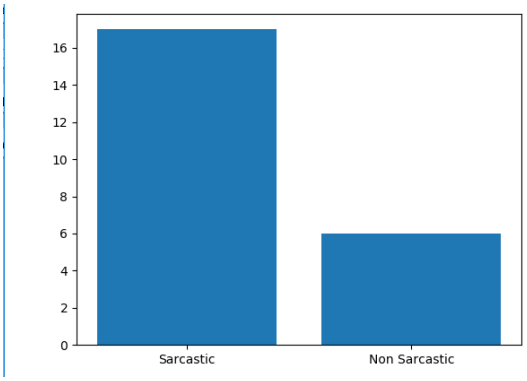

Fig. 7: Count of sarcastic or non-sarcastic tweets.

In Fig. 6 using pie chart we can see percentage of positive, negative, or neutral tweets. In Fig. 7, x-axis represents type of tweets and y-axis represents count of sarcastic or non-sarcastic tweets.

5. CONCLUSION

The aim of this study was to propose ways to extend the lexicon algorithm to build systems that would be more efficient for sarcasm detection. This aim has been successfully met as two systems

ISSN- 2394-5125 VOL 10, ISSUE 02, 2023

have been developed to address this situation. However, in the first system, it had been noticed that the training set of the sarcasm analysis algorithm must be relevant to the actual data that need to be analyzed to obtain meaningful results and to improve the accuracy of the system. The second system constitutes a vast area of study. Some work needs to be done to develop a system that would allow the collection of environmental details under which the textual contents would be made on social media platforms. A consolidated way of computing the sentiment polarity of the environments based on their details should also be developed.

REFERENCES

[1] Cambridge University Press, 2018. sarcasm. [Online] Available at: https://dictionary.cambridge.org/dictionary/english/sarc asm [Accessed 20 January 2018].

[2] Palanisamy, P., Yadav, V., & Elchuri, H. (2013). Serendio: Simple and Practical lexicon-based approach to Sentiment. 543-548.

[3] Jurek, A., Mulvenna, M. D., & Bi, Y. (2015). Improved lexicon-based sentiment analysis for social media analytics. SpringerOpen, 4-9.

[4] Kiilu, K. K., Okeyo, G., Rimiru, R., & Ogada, K. (2018). Using Naïve Bayes Algorithm in detection of Hate Tweets. International Journal of Scientific and Research Publications, 99-107.

[5] Rathan, K., & Suchithra, R. (2017). Sarcasm detection using combinational. Imperial Journal of Interdisciplinary Research, 546-551.

[6] Sathya, R., & Abraham, A. (2013). Comparison of Supervised and Unsupervised. International Journal of Advanced Research in Artificial Intelligence, 34-38.

[7] Dataquest, 2018. Top 10 Machine Learning Algorithms for Beginners. [Online] Available at: https://www.dataquest.io/blog/top-10- machine-learning-algorithms-for-beginners/ [Accessed 15 September 2018].

[8] Haripriya, V., & Patil, D. P. (2017). A Survey of Sarcasm Detection in Social Media. International Journal for Research in Applied Science & Engineering Technology, 1748-1753.

[9] Musto, C., Semeraro, G., & Polignano, M. (n.d.). A comparison of Lexicon-based approaches for Sentiment Analysis of microblog posts.

[10] Saxena, R., 2017. How the naive bayes classifier works in machine learning. [Online] Available at: http://dataaspirant.com/2017/02/06/naivebayes-classifier-machine-learning/ [Accessed 10 February 2019].

[11] Gandhi, R., 2018. Naive Bayes Classifier. [Online] Available at: https://towardsdatascience.com/naivebayes-classifier-81d512f50a7c [Accessed 10 February 2019].

[12] RAY, S., 2017. 6 Easy Steps to Learn Naive Bayes Algorithm (with codes in Python and R). [Online] Available at: https://www.analyticsvidhya.com/blog/2017/09/naivebayes-explained/ [Accessed 10 February 2019].

[13] Aggarwal, S., & Kaur, D. (2013). Naïve Bayes Classifier with Various Smoothing. International Journal of Computer Trends and Technology, 873-876.