IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-1, Jan 2019 E-ISSN: 2347-2693

Sarcasm Recognition in Twitter

Sakshi Thakur[1*], Sarbjeet Singh[2], Makhan Singh[3]

[1,2,3]Computer Science and Engineering, University Institute of Engineering and Technology, Panjab University, Chandigarh, India

**Corresponding Author: sakshithakur18@gmail.com, Tel: +91-8283816789*

Available online at: www.ijcseonline.org

Accepted: 18/Jan/2019, Published: 31/Jan/2019

Abstract-- Sarcasm is a nuanced form of speech broadly utilized in different online platforms such as social networks, micro-blogs and sarcasm recognition refers to anticipate whether the content is sarcastic or not. Identifying sarcasm in content is among the significant issues confronting sentiment analysis. In sarcasm, individuals express their negative feelings by utilizing positive or strengthened positive words in the content. While talking, individuals regularly utilize intense tonal force and certain gestural pieces of information like rolling of the eyes, hand development, and so forth to reveal sarcasm. Due to these challenges, in the last few decades, researchers have been working rigorously on sarcasm recognition so as to amend the performance of automatic sentiment analysis of data. In this paper, a supervised learning approach, which learns from four different categories of features and their combinations, is presented. These feature sets are employed to classify instances as sarcastic and not-sarcastic using four different classifiers, namely – Naïve Bayes, SVMs, Random Forest and k-Nearest Neighbor classifiers. In particular, it has been tried to explore the impact of sarcastic patterns based on POS tags and the outcomes demonstrate that they are not useful as a feature set for recognizing sarcasm when compared to content words and function words. Using the finest feature set i.e. the combination of content words and function words, a precision and AUC of approximately 85% and 87%, respectively, were achieved. Additionally, the Naïve Bayes classifier gives better results over every single other classifier that has been utilized.

***Keywords*—**Sentiment analysis, Sarcasm, Supervised learning, Feature-sets

I. INTRODUCTION

Opinions are important to almost everyone and are key influencers in our decision-making. In today's era of digitalization, anyone with basic internet access can publish their thoughts and spread their ideas via various online platforms and social media. This results in a huge amount of opinionated content available online. This opinionated content has to be distilled out so as to extract opinions out of it, which could further be utilized by the researchers interested in Natural Language Processing (NLP) and data analysis. Sentiment analysis (SA) is an ongoing area of research which deals with natural language text, analyzes public opinions, sentiments, feelings and, emotions regarding a particular object, product, services, events or a person. Misinterpreting sarcasm in the field of SA formulates a big challenge as it reverses the polarity of an utterance, which eventually worsens the results of the task performed.

Sarcasm is a nuanced form of speech broadly utilized in different online platforms such as social networks and micro-blogs in order to act funny, to show anger, to criticize someone or, to avoid giving a clear answer. It does not have any specific definition. The English Oxford dictionary defines it as "*the use of irony to mock or convey contempt*" and the Merriam-Webster states sarcasm as "*a sharp and often satirical or ironic utterance designed to cut or give pain*". So, in simple words, sarcasm can be defined as a kind of sentiment where an individual expresses his/her negative feelings by utilizing positive words in content and vice-versa. For example, consider the following utterance:
"I just love feeling earthquakes!!!"

In a verbal communication, an individual can easily interpret the hidden sarcasm in this utterance due to the presence of intense tonal force and certain gestural pieces of information like rolling of eyes, hand movements, and so forth, whereas in the written form, due to the absence of these cues, it becomes difficult to anticipate whether the utterance is sarcastic or not. A naïve sentiment analysis system that is incapable of detecting sarcasm would falsely classify this utterance as positive and this misclassification degrades the performance of the system. Due to these challenges, identifying sarcasm in the text is among the significant issues confronting sentiment analysis.

In this work, a set of supervised learning experiments have been conducted so as to investigate the impact of sarcastic patterns based on POS tags to detect sarcasm in the text. The feature sets employed for classifying tweets includes POS tags, sarcastic patterns, content words, function words, and, their different combinations. The work draws on a manually annotated dataset of 2000 tweets. Following are a few sarcastic tweets present in our tweets dataset:

i. I love working midnights the same week I work evenings and days.
ii. I just made my mom cry I love today
iii. Stalking is the best way to know someone
iv. Man: why do I find so many stones in my pulaav? waitress: sir, if I am not wrong, you ordered 'Kashmiri pulaav' man: o shit...

The first tweet expresses anger at the number of continuous hours the individual had been working by saying the opposite that he/she loves it. The second tweet apparently expresses the guilt for hurting someone. The third tweet conveys the disgust at the stalking thing because nobody likes it. The fourth tweet is used to act funny by mocking about the Kashmir issue. Hence, all the four tweets express sarcasm for different purposes.

The rest of this paper is organized as follows: Section II provides a summarized overview of the work carried out in the field of sarcasm detection. Section III describes the proposed approach for sarcasm detection. In Section IV, the results obtained from the experiments have been presented and discussed. Finally, Section V concludes this work.

II. RELATED WORK

In computational works, sarcasm has been stated as one of the significant challenges facing sentiment analysis. For a comprehensive look-up into the research and challenges in SA and opinion mining, refer Kumar and Vadlamani [1]. In this section, some state-of-the-art work associated with the study of sarcasm recognition has been reviewed. The work in this field has broadly been carried out using rule-based, machine-learning-based and, deep-learning-based approaches.

Riloff et al. [2] presented a bootstrapping algorithm to identify sarcasm arising from the contradiction amid a positive opinion and a negative condition. This bootstrapping process started with a seed word “love” plus some sarcastic tweets and grasped 26 positive opinion expressions along with 239 negative situation expressions. This work employed a dataset of 175,000 tweets: 35,000 labeled as sarcastic and 140,000 as non-sarcastic for training and a dataset of 2278 manually annotated tweets for testing and achieved a precision and F-score of 62% and 51%, respectively.

Maynard and Greenwood [3] considered the significance of sarcasm enclosed in hashtags and created a hashtag tokenizer (an algorithm) for extracting individual tokens from concatenated hashtags. For investigating these hashtags, so as to recognize the scope of sarcasm, some rules were developed; for example, if the opinion conveyed in the text is positive or neutral and there exists only one sarcasm hashtag then flip the polarity to negative and many more. The experiments were accomplished on a dataset of manually annotated general tweets and an F-score and precision of 91.03% was achieved.

Bharti et al. [4] proposed two rule-based classifiers. One of these approaches used PBLGA (Parse-Based Lexicon Generation Algorithm), which created parse trees of utterances and identified situation expressions bearing opinion. In this algorithm, whenever a negative expression appeared in a positive utterance, that utterance was predicted as sarcastic. And the other one aimed to capture hyperbolic sarcasm by analyzing the use of interjections such as ‘oops’, ‘hurray’ and ‘ouch’ and intensifiers such as ‘extremely’, ‘strongly’ and ‘completely’ that occurred together. On the dataset of tweets with the sarcastic hashtag, a precision of 89% and 85% was achieved using both the approaches, respectively.

Joshi et al. [5] projected a rule-based technique for sarcasm detection, taking an author’s historical tweets into consideration. The used approach consisted of three modules: Contrast-based predictor, that identifies sarcasm with the help of opinion contradiction (as in [2]); Historical tweet-based predictor, that used the targeted tweet along with author’s name to determine whether the opinion expressed in the tweet is different from the historical sentiment, and; Integrator, that combined the predictions from both the contrast-based and historical tweet-based predictors. For experiments, two lexicons were used: L1 that is a thesaurus containing positive and negative words from [6] plus L2 that is a thesaurus containing positive and negative word from [7] and the best precision and F-score was achieved in case of L2 with a value of 88.0% and 88.2%, respectively.

Justo et al. [8] proposed a set of set of supervised learning approaches to detect sarcasm plus nastiness in dialogic language over the net. In this work, various feature sets have been extracted using different criteria: Mechanical Turk Cues; Statistical Cues; Linguistic information; Semantic information; Length information; Concept and Polarity information. Two subsets of IAC (Internet Argument Corpus): Sarcasm dataset (comprising 3,230 sarcastic and 3,230 non-sarcastic instances) and Nastiness dataset (consisting of 1,382 nasty and 1,382 non-nasty instances) were employed by Justo et al. in their work and an accuracy of 68.7% and 78.6%, respectively, were achieved using the Naïve Bayes classifier.

Mukherjee et al. [9] used Naïve Bayes and fuzzy clustering techniques for detecting sarcasm in microblogs and proposed that for effectual sarcasm recognition, both content words and authorial style play a crucial role. In this work, numerous features have been employed: Content words; Function word; POS tags; and, their combinations. On employing a dataset of 2,000 tweet instances (annotated manually as sarcastic and non-sarcastic), an accuracy and F-score of 65% and 76%, respectively, were achieved using Naïve Bayes classifier. It was observed that the clustering algorithms were not that effective for sarcasm detection due to the small dataset used.

Bouazizi and Otsuki [10] proposed a supervised method which learns sarcastic patterns to detect sarcasm in tweets. Four categories of features were extracted: sentiment-related features (14), punctuation-related features (7), syntactic and semantic features (5), and pattern-related features. After feature extraction, the classification was performed using four different classifiers, of which, the random forest gave the best results with accuracy and f-score of 83.1% and 81.3%, respectively.

Tsur et al. [11] presented a SASI (Semi-Supervised Algorithm for Sarcasm Identification) to detect sarcasm in product reviews. They employed two basic types of features: first, on the basis of patterns, and other, based on punctuations (that included the length of a statement in words and number of exclamations '!', capitalized words, question marks '?', quotes in the sentence). About 66,000 Amazon product reviews were employed to carry out experiments. Using SASI, the projected approach, precision, and F-score of 76.6% and 78.8% were obtained, respectively. Davidov et al. [12] presented a work similar to Tsur et al. [11]. They made use of semi-supervised learning for detecting sarcasm using 66,000 Amazon merchandise reviews plus, about 5.9 million tweets gathered from Twitter and achieved accuracy and precision of 89.6% and 72.7%, respectively. Tepperman et al. [13] investigated sarcasm recognition for spoken systems. They produced a system that could discern sarcasm as effectively as a human, focusing on the use of the phrase `Yeah right'. Liebrecht et al. [14] used Netherland's e-science center's database containing Dutch tweets and extracted all the tweets with hashtag '#sarcasme', approximately 78,000 tweets. They employed 1-grams, 2-grams, and 3-grams as features and classified the tweets via a balanced Winnow classifier [15].

Gonzalez et al. [16] projected a technique to differentiate sarcasm from positive and negative opinions conveyed in tweets. They employed two lexical features: 1-grams; and dictionary-based and three pragmatic factors: positive emoticons; negative emoticons; and ToUser (that tells whether the tweet is a response to some other tweet). After extracting features, the tweets were classified into sarcastic, positive and negative using SVM, and accuracy of approximately 75% was achieved.

Rajadesingan et al. [17] proposed a behavioral modeling framework named SCUBA (Sarcasm Classification Using a Behavioural modeling Approach) for detecting sarcasm on twitter. Based on the following behavioral facets of sarcasm – dissimilarity in opinions; a complex form of expression; a form of written expression; a means of conveying emotion and; a possible function of familiarity, they constructed 335 features. The data was obtained using Twitter's Streaming API. Using this dataset, they trained the model (SCUBA). After training, the performance of SCUBA was evaluated and an accuracy of 83.05% was achieved.

From the previous state-of-the-art work associated with the study of sarcasm recognition, it can be drawn that the most commonly used feature sets for supervised learning approaches are: statistical features, syntactic features, pragmatic features and features based on sarcastic patterns. This work employs a supervised learning approach for investigating the impact of using a combination of features from different dimensions for sarcasm detection.

III. METHODOLOGY

The proposed detection model is built upon the thought that the presence of sarcasm in an utterance may highly alter the results of an SA system. The aim of the projected technique is to identify sarcasm in the written form. Figure 1 describes the framework of the proposed approach for detecting sarcasm. The design of the implemented sarcasm detection system is explained in the subsequent sub-sections. First, data is acquired from twitter and pre-processed for extracting relevant information. Then, certain features that have been chosen are detailed. After that, training and testing are done using inbuilt classifiers for the purpose of sarcasm detection. Finally, the system is evaluated for performance.

A. Data Acquisition and Pre-processing

About 14,000 tweets were downloaded from Twitter by using an application written in Java. In order to collect sarcastic utterances, tweets carrying hashtags #sarcasm plus #sarcastic and for non-sarcastic utterances, tweets with the opinion hashtags like #love, #sad, #hate, etc. were downloaded and manually annotated. It has been assumed that an individual usually labels his/her tweets correctly. This data was then filtered and a dataset of 2000 tweets, balanced between sarcastic and not-sarcastic, was obtained which was then pre-processed for training and testing purpose. For training and testing the classifiers, this dataset was randomly divided into two sets in a 3:1 ratio i.e. 75% of the dataset was employed for training and 25% for testing. In literature, the 3:1 ratio has been comprehensively applied [18].

To avoid the problem of overfitting, 10-fold cross-validation has been performed on the training set. In 10-fold cross-validation, data is divided into ten segments, where one of the segments is kept for testing and others for training. The entire algorithm is repeated 10 times such that each segment is kept for testing exactly once. The k-fold cross-validation is performed to avoid the problem of over-fitting and to maximize generalization accuracy [19].

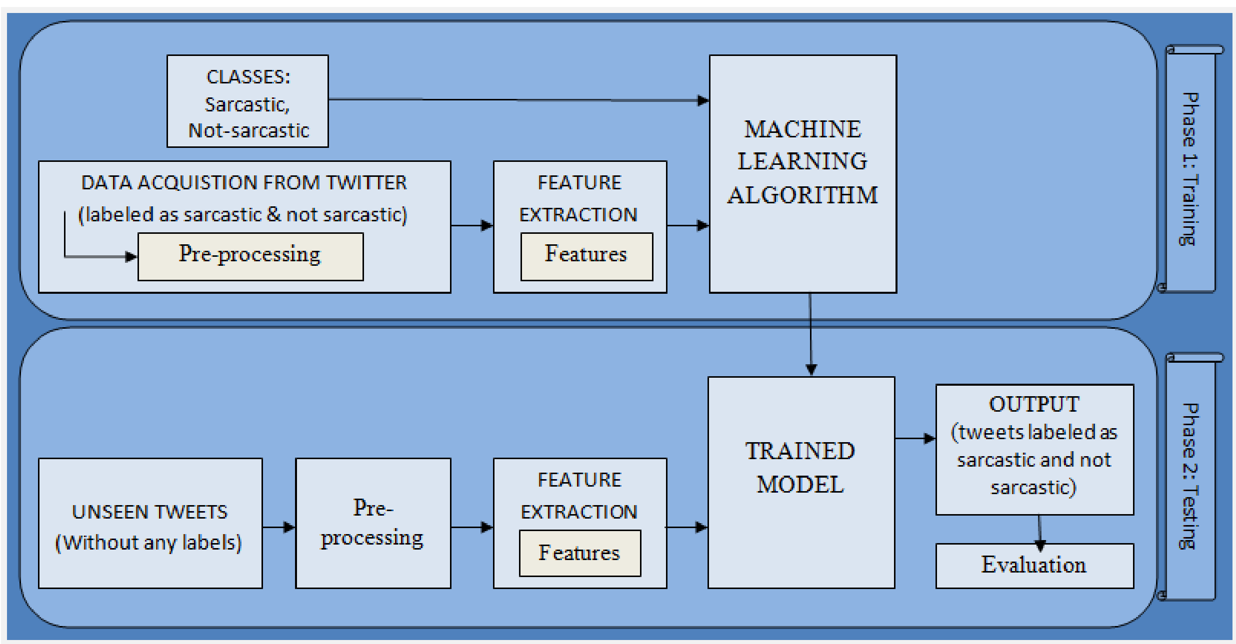

Figure 1. Framework of the proposed approach

Different features were then extracted from the training set and tested for performance on the test set using classification methods which have been discussed in subsequent sub-sections.

B. Feature Extraction

Feature extraction is a technique in which the raw input data is converted into a set of features that represent the data. It forms a key factor in determining the performance of sarcasm detection system. In this work, an extensive list of features has been used for classification purpose. Using more than one feature gives us the chance to analyze results obtained from different features and their combinations. The following categories of features have been used in the proposed system as described below:

1) *POS tags:* POS (Part-of-speech) tagging a procedure wherein a word in a corpus is marked with its corresponding part-of-speech based on its relationship with neighboring words in an utterance. In the proposed work, POS tags have been used as features for the training set. For example, *absolutely amazing start by heels* gets converted to absolutely/RB, amazing/JJ, start/NN, by/IN, Heels/NNS [20]. Here, RB means absolutely is an adverb, NN means start is a noun, JJ means amazing is an adjective, IN means by is a preposition and NNS means Heels is a noun (plural form).

2) *n-gram POS patterns:* In this feature set, part-of-speech tags corresponding to each word in a tweet are used. Broadly, these tags are associated with 8 lexical categories – nouns, pronouns, adjectives, verbs, adverbs, prepositions, conjunctions, and interjections. Bigram patterns, trigram patterns & higher-gram patterns were considered and it was found that with higher n-grams, performance doesn't improve. Thus, in this work, a set of POS bigram patterns has been used.

3) *Content words:* Content words are the lexical words, having an independent definition i.e. even if it is used outside any sentence, it holds meaning. Broadly, four classes of content words are there – nouns (words like boy, silver etc.), verbs (words like accept, read etc.), adjectives (words like tall, beautiful etc.) and adverbs (words like easily, slowly etc.) [21].

4) *Function words:* Function words are the grammatical words whose purpose is to contribute to the syntax of a sentence or phrase rather than its meaning. These are used to create a structural relation between content words and have little meaning on their own. For example, 'do'

in 'I do not like you'. Some of the main classes of function parts-of-speech are pronouns, prepositions, conjunctions, determiners and auxiliaries [21].

Also, various combinations of the above-mentioned features are employed for the purpose of classification.

Not all features are equally important to detect sarcasm. Hence, feature extraction techniques have been employed to find out the relevant ones. In this work, the chi-square and extra-trees method have been used for feature selection. Feature selection refers to a process in which the features that contribute the most to the required output are automatically selected. The advantage of using feature selection is that it lessens overfitting because less redundant data means fewer chances of making decisions on the basis of noise and also, it lessens the computational time required to train the system for classification without altering the performance.

C. Classification method

The above-extracted features and the available ground truth are then used to train the model for classifying the text into required categorization i.e. sarcastic and not-sarcastic. On the basis of this training, the classifier assigns a label to the text that does not have any label. The classifiers we have employed for evaluating the utility of the above-mentioned features in detecting whether an instance is sarcastic or not are:

1) *Naïve Bayes classifier:* Naïve Bayes [22] is a probabilistic model based on the Bayes theorem of conditional probability. Here, the conditional probability is the probability of occurrence of an event that will occur given that another event has already occurred and it is defined as given below:

$$P(Ev_1|Ev_2) = \frac{\{P(Ev_2|Ev_1) \times P(Ev_1)\}}{P(Ev_2)}$$

where $P(Ev_2|Ev_1)$ is the probability of the evidence given that the $Event_1$ is true, $P(Ev_1)$ is the probability of $Event_1$ being true, $P(Ev_2)$ is the probability of occurrence of $Event_2$.

2) *Support Vector Machine (SVM):* SVM [23] is a supervised ML algorithm whose main goal is designing a hyperplane that classifies all training vectors in two classes. After this, the hyperplane which separates two classes very well is used to perform classification. The proposed work makes use of a linear kernel as it is faster and performs well for linearly separable data. And also, while training an SVM with linear kernel, only C regularization parameter needs to be optimized, which reduces time.

3) *Random Forest classifier:* Random Forest [24] is based on the basic structural principle of Decision Trees. At the time of training, a multitude of Decision Trees is constructed and the class related to the mean or mode of the individual trees is outputted. To measure the quality of this split, Gini index or information gain is used. Thus in this way, it forms an ensemble learning method which works by generating a multitude of decision trees while training. In our work, n_estimators = 100 has been considered.

4) *k-Nearest Neighbor classifier:* k-NN [25] is a non-parametric lazy learning algorithm, by this, we mean that it does not presume anything on the basis of the training data. It classifies test instances based on computed distances measures to labeled training instances. These distances reveal a set of nearest neighbors (k) which are used to vote on the predicted class. The distance metric that is usually used is Minkowski with p=2 i.e. the standard Euclidean metric. In the proposed work, 10-fold cross-validation is being used to find the most favorable value for k.

The above-mentioned classifiers have extensively been used in the literature, therefore they have been chosen for this work. At the outset of the experiment, it is difficult to say which algorithm will outperform the others. Using multiple algorithms for performing classification is useful in order to select the specific algorithm that maximizes performance. In the next section, the performance of the system on the basis of various evaluation metrics is discussed.

IV. RESULTS AND DISCUSSION

The ability of classification algorithms for detecting sarcasm on the basis of various categories of features has been reported in this section. The performance of the sarcasm detection system was measured by using different evaluation metrics i.e. accuracy, precision, F-measure and AUC. Accuracy can be defined as the fraction of true classifications. According to the terminology used in a confusion matrix [26],

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

F-measure is a metric which is calculated as the weighted harmonic mean of precision and recall and, is evaluated using the below-mentioned formula:

$$\text{F} - \text{measure} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

In this work, precision can be defined as the number of successfully classified sarcastic instances out of total instances being classified as sarcastic and, recall refers to the number of instances truly classified as sarcastic out of the overall sarcastic instances. AUC stands for Area Under the Curve (where curve refers to the ROC curve). ROC curves are useful for visualizing the performance of classifiers.

More the area covered by the ROC curve, better the classifier is.

Based on the above-mentioned evaluation metrics, it has been found that when employed as a feature set, the combination of content words (CW) and function words (FW) performs better than the sarcastic patterns based on POS tags with a precision and AUC of 85% and 87% respectively (see Table 1). For sarcasm detection problem, precision is a more accurate metric for evaluation than others as precisely identifying the sarcastic instances is of more importance here. Hence, after evaluating all the metrics, it is drawn that sarcastic patterns are not as good a feature set as others.

Table 1. Comparison of CW+FW and sarcastic patterns on the basis of evaluation metrics.

Classifiers	**Feature sets**	**Metrics**			
		Accuracy	Precision	F1-score	AUC
Naïve Bayes	CW+FW	0.73	**0.85**	0.68	**0.87**
	Patterns	0.62	0.64	0.61	0.66
SVM	CW+FW	**0.78**	0.76	**0.79**	0.85
	Patterns	0.62	0.61	0.65	0.66
Random Forest	CW+FW	0.77	0.73	**0.79**	0.83
	Patterns	0.64	0.66	0.63	0.70
k-NN	CW+FW	0.71	0.66	0.75	0.79
	Patterns	0.61	0.58	0.68	0.65

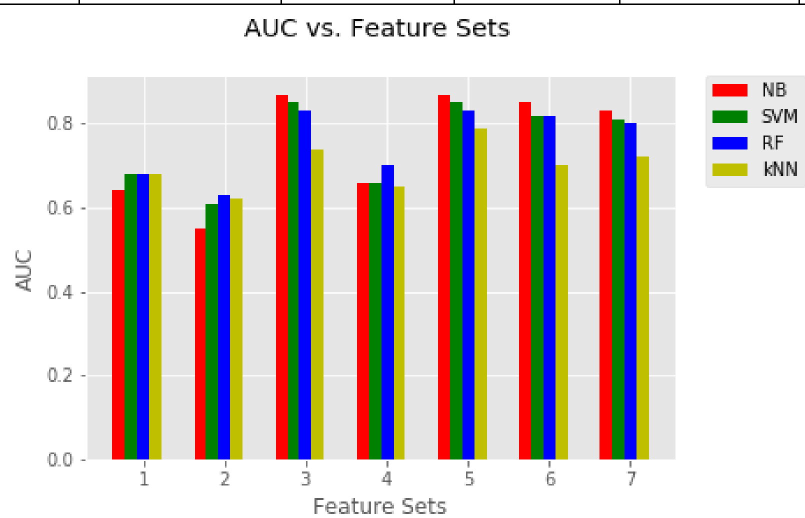

Figure 2. AUC vs. Feature sets across dataset

In Figure 2, on x-axis i.e. feature sets, 1, 2, 3, 4, 5, 6 and 7 stands for different features sets namely POS tags, Function Words, Content Words, n-gram POS patterns, Combination-1 i.e. Function Words and Content Words, Combination-2 i.e. Content Words, Function Words and n-gram POS patterns and Combination-3 containing all features, respectively. It can clearly be seen in Figure 2 that of all the features being used for sarcasm detection problem, CW and FW give the best results for almost all the classifiers.

Figure 3 gives a visualization of the performance of all the four classifiers using the finest feature type i.e. a combination of content words and function words. As it is known that more the area under the ROC curve, better the classifier is in terms of performance, hence, after analyzing Figure 3, it is clear that Naïve Bayes classifier outperforms all the other classifiers with an AUC of 0.87 which shows that the classifier performs well for the sarcasm detection problem. Also, Figure 3 clearly states that k-Nearest Neighbour classifier is the worst choice for being used in detecting sarcasm in the text as even after optimizing the value of k through parameter tuning, it has the lowest AUC as compared to all other classifiers employed, with value 0.79.

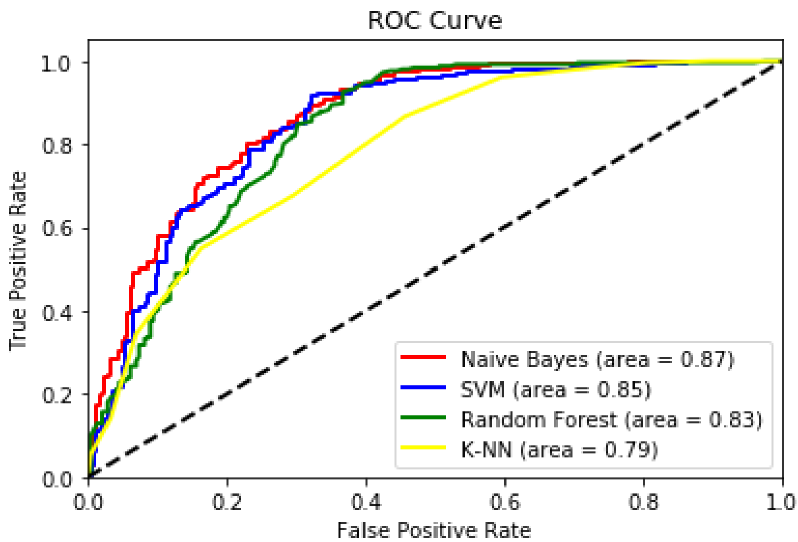

Figure 3. Performance of all classifiers on the basis of ROC curve

V. CONCLUSION AND FUTURE SCOPE

Presence of sarcasm is one of the major ongoing issues in the field of sentiment analysis. Its mere presence can entirely reverse the polarity of an utterance and therefore, successfully detecting sarcasm is of great importance in this field. In this work, a set of supervised learning experiments were conducted to investigate the impact of sarcastic patterns based on POS tags to detect sarcasm in the text. Through the experiments, it has been drawn that sarcastic patterns based on POS tags are not as useful for detecting sarcasm as content-words and functions words together. This combination gave a precision of 0.85, which is much higher than the one achieved from sarcastic patterns i.e. 0.64.

In the future, it is intended to perk up the performance of the sarcasm detection system and reduce the computational time. We argue that pragmatic factors when combined with other feature sets can add-on to the performance of the system, especially for social networking sites dataset where individuals frequently make use of pragmatic markers whenever they want to convey something sarcastically. Therefore, we hope to work on such feature sets.

REFERENCES

[1] R. Kumar, V. Ravi. "*A survey on opinion mining and sentiment analysis: tasks, approaches and applications*", Knowledge-Based Systems, Vol.89, pp.14-46, 2015.

[2] E. Riloff, A. Qadir, P. Surve, L. D. Silva, N. Gilbert, R. Huang, "*Sarcasm as contrast between a positive sentiment and negative situation*", In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp.704-714, 2013.

[3] D. G. Maynard, M. A. Greenwood, "*Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis*", In LREC 2014 Proceedings, ELRA, 2014.

[4] Bharti, S. Kumar, K. S. Babu, S. K. Jena, "*Parsing-based sarcasm sentiment recognition in twitter data*", In Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, ACM, pp. 1373-1380, 2015.

[5] A. Khattri, A. Joshi, P. Bhattacharyya, M. Carman, "*Your sentiment precedes you: Using an author's historical tweets to predict sarcasm*", In Proceedings of the 6th workshop on computational approaches to subjectivity, sentiment and social media analysis, pp. 25-30, 2015.

[6] B. Pang, L. Lee, "*A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts*", In Proceedings of the 42nd annual meeting on Association for Computational Linguistics, Association for Computational Linguistics, p.271, 2004.

[7] S. M. Mohammad, P. D. Turney, "*Crowdsourcing a word–emotion association lexicon*", Computational Intelligence, Vol.29, Issue.3, pp. 436-465, 2013.

[8] R. Justo, T. Corcoran, S. M. Lukin, M. Walker, M. I. Torres, "*Extracting relevant knowledge for the detection of sarcasm and nastiness in the social web*", Knowledge-Based Systems, Vol.69, pp.124-133, 2014.

[9] S. Mukherjee, P. K. Bala, "*Sarcasm detection in microblogs using Naïve Bayes and fuzzy clustering*", Technology in Society, Vol.48, pp.19-27, 2017.

[10] M. Bouazizi, T. O. Ohtsuki, "*A pattern-based approach for sarcasm detection on Twitter*", IEEE Access 4, pp.5477-5488, 2016.

[11] O. Tsur, D. Davidov, A. Rappoport, "*ICWSM-A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Online Product Reviews*", In ICWSM, pp.162-169, 2010.

[12] D. Davidov, O. Tsur, A. Rappoport, "*Semi-supervised recognition of sarcastic sentences in twitter and amazon*", In Proceedings of the fourteenth conference on computational natural language learning, Association for Computational Linguistics, pp.107-116, 2010.

[13] J. Tepperman, D. Traum, S. Narayanan, "*" Yeah Right": Sarcasm Recognition for Spoken Dialogue Systems*", In Ninth International Conference on Spoken Language Processing, 2006.

[14] C. C. Liebrecht, F. A. Kunneman, A. P. J. van Den Bosch. "*The perfect solution for detecting sarcasm in tweets# not*", 2013.

[15] N. Littlestone. "*Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm*", Machine learning, Vol.2, Issue.4, pp.285-318, 1988.

[16] R. González-Ibánez, S. Muresan, N. Wacholder, "*Identifying sarcasm in Twitter: a closer look*", In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers, Association for Computational Linguistics, Vol.2, pp.581-586, 2011.
[17] A. Rajadesingan, R. Zafarani, H. Liu, "*Sarcasm detection on twitter: A behavioral modeling approach*", In Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, ACM, pp.97-106, 2015.
[18] S. C. Schürer, S. M. Muskal, "*Kinome-wide activity modeling from diverse public high-quality data sets*", Journal of chemical information and modeling, Vol.53, Issue.1, pp.27-38, 2013.
[19] R. Kohavi, "*A study of cross-validation and bootstrap for accuracy estimation and model selection*" In Ijcai, Vol.14, Issue.2, pp.1137-1145, 1995.
[20] K. W. Church, "*A stochastic parts program and noun phrase parser for unrestricted text*", In Proceedings of the second conference on Applied natural language processing, Association for Computational Linguistics, pp.136-143, 1988.
[21] D. Biber, S. Johansson, G. Leech, S. Conrad, E. Finegan, "*Longman grammar of spoken and written English*", pp.89-110, 1999.
[22] K. P. Murphy, "*Naive bayes classifiers*", University of British Columbia, 2006.
[23] C. Hsu, C. Chang, C. Lin, "*A practical guide to support vector classification*", pp.1-16, 2003.
[24] L. Breiman, "*Random forests*", Machine learning, Vol.45, Issue.1, pp.5-32, 2001.
[25] M. J. Islam, QM Jonathan, M. Ahmadi, M. A. Sid-Ahmed, "*Investigating the performance of naive-bayes classifiers and k-nearest neighbor classifiers*", In Convergence Information Technology, 2007. International Conference on, IEEE, pp.1541-1546, 2007.
[26] H. Schütze, C. D. Manning, P. Raghavan, "*Introduction to information retrieval*", Cambridge University Press, Vol.1, 2008.