Hyperbolic Feature-based Sarcasm Detection in Tweets: A Machine Learning Approach

Santosh Kumar Bharti
Computer Science & Engineering
National Institute of Technology
Rourkela, Odisha, India – 769008
Email: sbharti1984@gmail.com

Reddy Naidu
Computer Science & Engineering
ANITS Engineering College
Sangivalasa, Visakhapatnam - 531162
Email: naidureddy47@gmail.com

Korra Sathya Babu
Computer Science & Engineering
National Institute of Technology
Rourkela, Odisha, India – 769008
Email: prof.ksb@gmail.com

***Abstract*— In recent times, sarcasm analysis has been one of the toughest challenges in Natural Language Processing (NLP). The property of sarcasm that makes it difficult to analyze and detect is the gap between its literal and intended meaning. Detecting sarcastic sentiment in the domain of social media such as Facebook, Twitter, online blogs, reviews, *etc*. has become an essential task as they influence every business organization. In this article, a hyperbolic feature-based sarcasm detector for Twitter data is proposed. The hyperbolic features consist of intensifiers and interjections of the text. The performance of the proposed system is analyzed using several standard machine learning approaches namely, Naive Bayes (NB), Decision Tree (DT), Support Vector Machine (SVM), Random Forest (RF), and AdaBoost. The system attains an accuracy (%) of 75.12, 80.27, 80.67, 80.79, and 80.07 using NB, DT, SVM, RF, and AdaBoost respectively.**

Keywords*— *Hyperbolic Feature, Machine Learning, Natural Language Processing, Sarcasm, Sentiment, Tweets.

I. INTRODUCTION

Online companion has gained tremendous momentum in recent times for business, politics, entertainment, *etc*. Social media such as Twitter, Facebook, WhatsApp, *etc*. became the popular medium for online companion and attaining the response from worldwide. These responses include one's sentiment or opinion towards any specific target such as individuals, events, topics, products, organizations, services, *etc*. [1]. The sentiment is an opinion of an individual towards a specific target. It may be either positive or negative. Manual extraction of the sentiment from social media is a tedious job for individuals or organizations. Therefore, an automated system is required to analyze the sentiment without human interference.

The major challenges involve for sentiment analysis is the presence of sarcastic sentiment. Due to this, most of the existing systems for sentiment analysis fails in detecting the actual sentiment value. Sarcasm is a particular type of sentiment that usually flips the orientation of the opinion in a given piece of text. According to Macmillan Dictionary, sarcasm is known as "the activity of saying or writing the opposite of what you mean, or of speaking in a way intended to make someone else feel stupid or show them that you are angry" [2]. For example: "Working on holidays is my passion #sarcastic". In this example, the author says that he likes to work on holidays, but due to hashtag (#) sarcastic, the actual meaning conveys that he does not like to work on holidays. Usually, no one likes to work on holidays. Therefore, it is hard to identify such kind of sentiment using existing system for sentiment analysis.

In this article, we proposed an automated system to detect sarcastic sentiment from Twitter data. It uses hyperbolic words as the feature to detect sarcastic tweets. A hyperbolic word usually contains either an intensifier or interjection. The presence of an adjective or an adverb often act as an intensifier in the textual data such as extremely happy, so much, thoroughly enjoyed, *etc*. Keywords like wow, aha, nah, yay, wah, oh, *etc*. are the interjection words. An algorithm is proposed to extract the bigram and trigram hyperbolic phrases as the feature set for the proposed system. With the use of extracted hyperbolic features, a set of standard classifiers namely, NB, DT, SVM, RF and AdaBoost are trained to classify the sarcastic and non-sarcastic tweets.

The rest of the article is organized as follows: Section II describes related work. The proposed scheme is discussed in Section III. Results are shown in Section IV and conclusion of the article is given in Section V.

II. RELATED WORK

In recent times, sarcasm detection gained rapid attention as several companies adopted sentiment analysis for their products and services. According to the literature, Twitter sarcasm detection techniques can be classified into four categories, namely lexicon-based approach, pattern-based approach, machine learning-based approach and context-based approach as shown in Fig. 1.

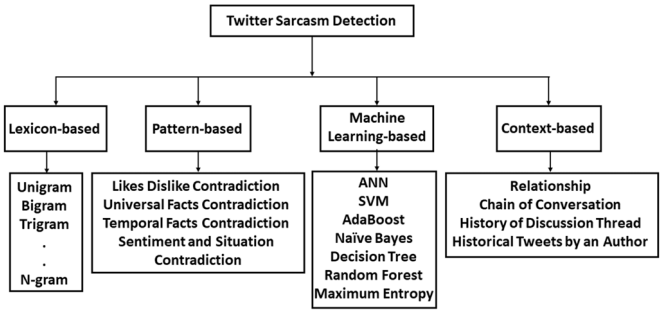


Fig. 1. Classification of Twitter sarcasm detection methodologies.

978-1-5386-4318-1/17/$31.00 ©2017 IEEE

1. Lexicon-based Approach:

This approach utilize a bags-of- lexicons which comprise of unigram, bigram, trigram, N-gram phrases to identify sarcasm in tweets. Riloff *et al.* [3] developed two bags-of-lexicons, namely positive sentiment and negative situation using bootstrapping technique. These lexicon file consists of unigram, bigram, and trigram phrases. Further, they utilized these phrases to identify sarcasm in tweets for the structure, positive sentiment in a negative situation. Similarly, Bharti *et al.* [4] developed four bags-of-lexicons, namely positive sentiment, negative situation, negative sentiment and positive situation using parsing technique. Further, they utilized these phrases to identify sarcasm in tweets for both structures, namely positive sentiment in a negative situation and negative sentiment in a positive situation.

2. Pattern-based Approach:

In this approach, one can observe the sarcastic dataset and identify the unique patterns for sarcastic text. Bouazizi and Ohtsuki [5] identified three patterns for sarcasm in tweets; namely, sarcasm is a wit, sarcasm is an evasion and sarcasm is a whimper. Based on these three patterns, they proposed an efficient sarcasm detection system that enhance the accuracy of sentiment analysis. Similarly, Bharti *et al.* [4] [6] proposed four patterns for sarcasm detection in tweets namely, a contradiction between sentiment and situation, user's likes and dislikes contradiction, and tweet contradicts universal and temporal facts. Based on these four patterns, they proposed a Hadoop-based framework for sarcasm detection in real-tweets. Riloff *et al.* [3] identified a unique pattern "sarcasm is a contract between positive sentiment and negative situation".

3. Machine Learning-based Approach:

In this approach, one needs to get labelled dataset to extract features. Further, these features are used to train the classifiers such as SVM, DT, NB, *etc*. Liebrecht *et al.* [7] designed a machine learning model to detect sarcasm in Dutch tweets. Peng *et al.* [8] used a supervised approach to classify movie reviews into two classes after performing subjective feature extractions. Tayal *et al.* [9] utilized machine learning classifier to detect the sarcastic political tweets. Tungthamthiti *et al.* [23], explored concept level knowledge using the hyperbolic words in sentences and gave an indirect contradiction between sentiment and situation such as raining, bad weather, that are conceptually the same. Therefore, if 'raining' is present in any sentence, then one can assume 'bad weather'. They built a system for sarcasm detection using SVM classifier and used these indirect contradiction features for training. Bharti *et al.* [10] developed a machine learning framework to detect sarcastic tweets. They deployed several classical classifiers such as SVM, DT, NB, Maximum Entropy (ME), *etc*. for sarcasm detection.

4. Context-based Approach:

The relationship between an author and audience followed by the immediate communicative context can be helpful to improve the sarcasm prediction accuracy [11]. Message-level sarcasm detection on Twitter using a context-based model were used for sarcasm detection [12]. Chains of tweets that work in a context were considered. They introduced a complex classification model that works over an entire tweet sequence and not on one tweet at a time. Integration between linguistic and contextual features extracted from the analysis of visuals embedded in multimodal posts was deployed for sarcasm detection [13]. A framework based on the linguistic theory of context incongruity and an introduction of inter-sentential incongruity by considering the history of the posts in the discussion thread was considered for sarcasm detection [14]. A quantitative evidence of historical tweets of an author can provide additional context for sarcasm detection [15]. The author's past sentiment on the entities in a tweet was exploited to detect the sarcastic intent.

III. PROPOSED SCHEME

The most crucial task for sarcasm detection in text using machine learning is the database collection and feature extraction. An overall architecture for training and testing is shown in Fig. 2.

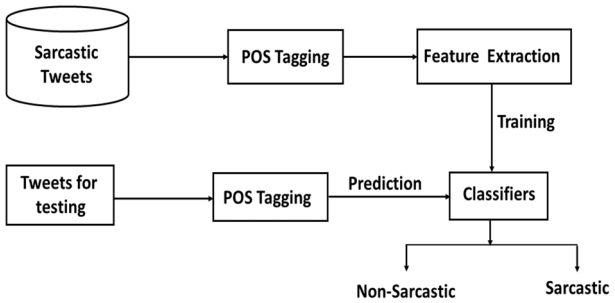


Fig. 2. System model for sarcasm detection in tweets.

A. Tweets Collection and Pre-processing

This section describes the procedure for collection of tweets from Twitter using RESTful Twitter developer API as shown in Fig 3. This API provides a connection between the Twitter user and Twitter server to access the archive tweets easily.

In this research, we have collected approximately 1, 00,000 sarcastic tweets using keywords such as "#sarcasm", "#sarcastic" *etc*. for training and testing using 5-fold cross validation. For preprocessing, URLs, hashtags, @username, Re-tweets are removed. Additionally, stop words like articles, prepositions, conjunctions *etc*. are also removed.

B. Part-of-speech (POS) Tagging

Part-of-speech tagging is a process to divide input texts into atomic words and assign them with appropriate POS tag information on the basis of context and relationship with surrounding words in the text. For example, the POS tags information of a tweet "Working on holidays is my passion." are Working-VBG | on-IN | holidays-NNS | is-VBZ | my-PRP | passion-NN. Here, VBG, IN, NNS, VBZ, PRP$, NN stands for Verb, Preposition, Noun, Verb, Possessive pronoun, and Noun respectively.

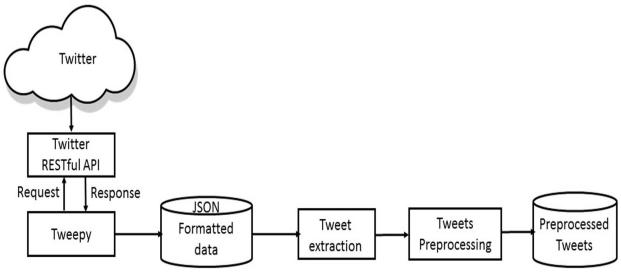


Fig. 3. Pipeline process for tweet collection & preprocessing.

In this research, Hidden Markov Model (HMM) based POS Tagger [6] is used to identify POS tag information in the text. For tag notation, Pen Treebank tagset [16] is used which comprise of 44 unique tags for English.

C. Feature Extraction

The feature selection and extraction are the most important tasks in machine learning approach. In this article, we have used interjection words and intensifiers to build the feature set. An intensifier is considered as either adjective, adverb or combination of both. Words like, wow, aha, nah, oh, yay, *etc*. are used as interjection words. To extract these interjections and intensifiers, POS tagging is used. The POS tag notations for an interjection, an adjective, and an adverb are "UH", "JJ", and "RB" respectively. The procedure for the feature extraction is given in Algorithm 1.

The Algorithm 1 takes a corpus of preprocessed sarcastic tweets as an input to extract the hyperbolic features. Initially, it finds POS tag information of all the tweets and stores it in the tagged file (TF). Next, it extracts first tag (FT), immediate next tag (INT) for every tweet in the tagged file. Additionally, it divides every tweet in the tagged file into bigram and trigram chunks. Then, it checks the presence of interjection tag (UH) in FT and either ADJ || ADV in INT. If both are present, then the corresponding tokens are added as bigram hyperbolic feature to the feature list (FL). Similarly, if the first tag is interjection and remaining tags in corresponding tweets contains a bigram of either (ADV + ADJ) || (ADJ + N) || (ADV + V) || (ADJ +ADV) then the corresponding tokens are added as trigram hyperbolic feature to the feature list (FL). Otherwise, if trigram (TTP) contains either (ADJ+V+V) || (ADV+V+V) then the corresponding tokens are added as trigram hyperbolic feature to the feature list (FL). Finally, if bigram tag pair (BTP) contains either ((ADJ +V) && (ADJ + N) || ((ADV+V) && (N + V) || ((ADV +ADJ) && (ADV +V) || ((V +N) && (N+V)) then, the corresponding tokens are added as quadgram hyperbolic feature to the feature list (FL). A sample of extracted features is shown in TABLE I.

D. Classifiers

Classifiers are used to classify tweets into sarcastic or non-sarcastic categories. In this article, we used five standard classifiers namely NB, DT, SVM, RF and AdaBoost for training and prediction of sarcastic tweets.

TABLE I. A sample output of HFE algorithm for feature extraction.

Sarcastic tweets	Hyperbolic phrases
Oh really!! I did not know that before. Thanks for making me informed.	'oh', 'really'
Aww! I love working on holidays.	'aww', 'love working'
Yay! Thoroughly enjoy the smell of pot permeating through my house while clean, thanks neighbors.	'yay', 'Thoroughly'
Being ignored makes me extremely happy. Yeah!!	'extremely', 'yeah'
I love waiting forever for my doctor.	'waiting forever'
Wow, so excited to see all kinds of couple pictures on Saturday and all the cute things they get for each other	'wow so'
Aha, great night	'Aha great'

Algorithm 1: *Hyperbolic Feature Extraction* (*HFE*)

Input: A corpus of preprocessed sarcastic tweets (**C**)
Output: List of hyperbolic features
Notation: *ADJ*: adjective, *V:* verb, *N*: noun, *ADV:* adverb, *FL*: feature list, *UH*: interjection, *T*: tweets, **C**: corpus, *TF*: tagged file, *t*: tag, *TWT*: tweet-wise tag, *FT*: first tag, *INT*: immediate next tag, *RTBP*: bigram pair of remaining tags.
Initialization: $TF = \{\Phi\}, FV = \{\Phi\}, ADV = \{RB \,||\, RBR \,||\, RBS\}$, $ADJ = \{JJ \,||\, JJR \,||\, JJS\}$

```
while (T in C) do
    k = find_postag(T)
    TF ← TF ∪ k
end
while (TWT in TF) do
    FT = find_first_tag(TWT)
    INT = find_immediate_next_tag(TWT)
    RTBP =
     find_bigram_tags_pair(TWT − (FT + INT))
    BTP = find_bigram_tags_pair(TWT)
    TTP = find_trigram_tags_pair(TWT)
    if ( FT = UH) && (INT = ADJ || ADV) then
        add corresponding text phrase to FL.
    end
    else if (FT = UH) && (RTBP =
     (ADV + ADJ) || (ADJ + N) || (ADV + V) ||
     (ADJ + ADV) then
        add corresponding text or text phrase to FL.
    end
    else if (TTP = ((ADJ + V + V) ||
     ((ADV + V + V)) then
        add corresponding text or text phrase to FL.
    end
    else if (BTP = ((ADJ + V)&&(ADJ + N) ||
     ((ADV + V)&&(N + V) ||
     ((ADV + ADJ)&&(ADV + V) ||
     ((V + N)&&(N + V)) then
        add corresponding text or text phrase to FL.
    end
end
```

1) Naïve Bayes

It is a statistical machine learning algorithm that uses Bayes theorem for classification [17]. To perform the classification, it does not require a large number of training data. It is simple to implement and produce good classification accuracy even with less number of training data. It also takes very less computational time to train the classifier. The NB classifier works extremely well for linearly separable problems. The Multinomial Bayes, Gaussian Naive Bayes, Bernoulli Bayes and Semi-supervised parameter estimation are the variants of Naive Bayes classifier. In this article, Gaussian Naive Bayes classifier is deployed.

2) Decision Tree

According to Ross [18], DT is a logic-based algorithm, and it is a decision support tool which uses a tree-like model to draw the decisions. It is a non-parametric model and easily handles the feature interactions. These are trees that classify instances by sorting them based on feature values. In a DT, each node represents the feature of an instance to be classified, and value that the node can assume is represented by each branch. Instances are classified starting at the root node and sorted based on their feature values.

3) Support Vector Machine

Corinna and Vapnik [19] proposed a Support vector machine model which is a non-probabilistic binary linear classifier. It is a form of supervised learning model which separates the data into classes by constructing a set of hyperplanes in a high-dimensional space. It is a popular classifier for the text classification problems; it is a kind of memory intensive classifier and is hard to interpret. For a given linearly separable data, it chooses that hyperplane which maximizes the margin, where the margin is the distance to separate hyperplanes. So, by choosing the best hyperplane, the instances of the problem will classify correctly.

4) Random Forests

This classifier is also known as Random Decision Forests (RDFs). The RDFs is an ensemble learning process for both classifications as well as regression problems. It operates by constructing a large number of decision trees at the time of training. For the classification problem, it assigns a class to the given instance which is the mode of the classes, and for the regression problem, it assigns mean prediction of the individual trees [20]. The RDFs resolve the overfitting practice of the decision tree to their training data [21].

5) AdaBoost

According to Freund and Schapire [22], AdaBoost (with decision trees as the weak learners) is often referred to as the best out-of-the-box classifier. AdaBoost is an acronym for Adaptive Boosting. To improve the performance of classification approaches, the AdaBoost can be used as a conjunction with those approaches. It constructs a conglomerate hypothesis by voting many individual hypotheses. So, to produce the final prediction, the combined prediction is obtained through a weighted majority sum of all the predictions.

E. Training

To convert the extracted features into a feature vector, we considered a common feature vector of maximum length five. The feature vector consists of five parameters, namely, FT, INT, RTBP, BTP and TTP. It is represented as [FT, INT, RTBP, BTP, TTP]. Here, FT, INT, RTBP, BTP and TTP indicates first tag, immediate next tag, bigram pair in other than FT and INT, bigram tag pair and trigram tag pair of a tweet respectively. To train all the five classifiers, a common feature vector has been used as shown in TABLE II.

TABLE II. A common feature vector used for all the five classifiers

FT	INT	RTBP	BTP	TTP
UH	ADJ ADV	(ADV + ADJ) (ADJ + N) (ADV + V) (ADJ + ADV)	((ADJ + V) && (ADJ + N)) ((ADV + V) && (N + V)) ((ADV + ADJ) && (ADV + V)) ((V + N) && (N + V))	(ADJ + V + V) (ADV + V + V)

While training, it checks for the presence of FT, INT, RTBP, BTP and TTP in the feature instances. If a tag is present in the feature instance, then the instance of feature vector will be set as 1, otherwise 0. For example, if any feature corresponding feature vector will be [1, 1, 0, 0, 0]. If the value of a feature vector is satisfying with the value given in TABLE II then, the label of the vector is set as 1, otherwise 0. The label 1 indicates the sarcastic and 0 indicates non-sarcastic.

IV. RESULTS AND DISCUSSION

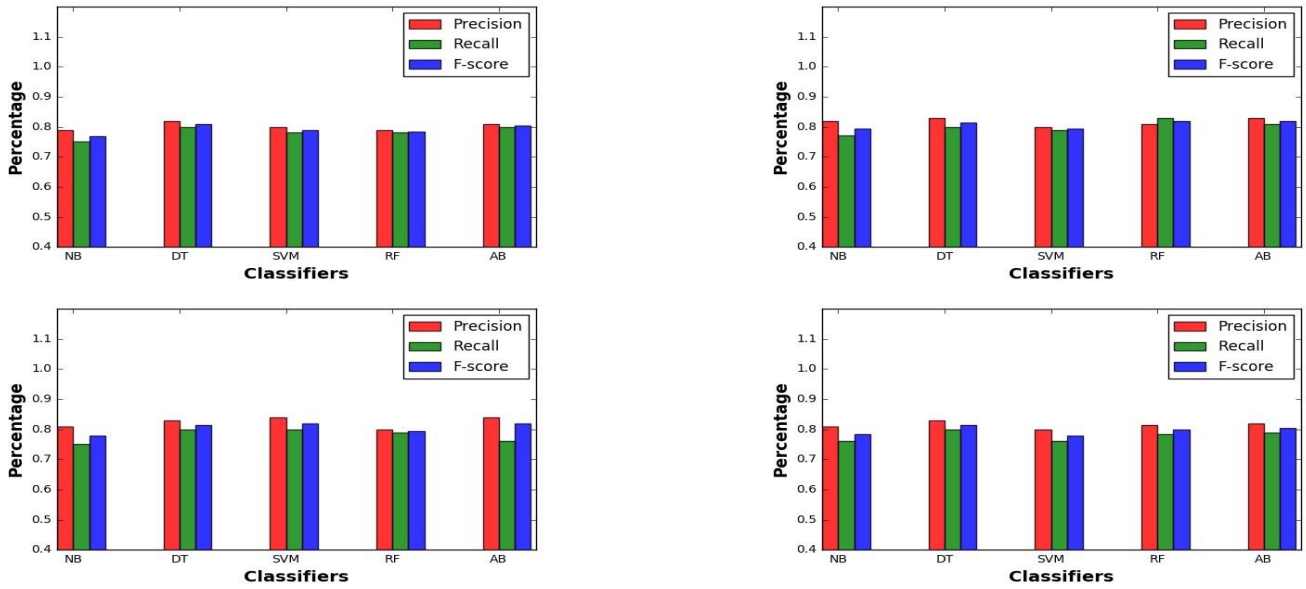
This section describes the experimental results of the proposed approach for sarcasm detection in tweets using several machine learning algorithms such as NB, DT, SVM, RF and AdaBoost. The performance of the proposed approach is measured using four statistical parameters namely, accuracy, precision, recall and F1-measure. The accuracy determines the overall performance of a system. A formula to calculate accuracy is given in Equation 1.

$$Accuracy = \frac{T_p + T_n}{T_p + T_n + F_n + F_p} \quad (1)$$

Precision and recall are the other parameters which determine the performance of a system. Here, precision determines how much relevant information is identified. Whereas, recall determines how much identified information is relevant. The formula to ascertain precision and recall appeared in equations 2 and 3.

$$Precision = \frac{T_p}{T_p + F_p} \quad (2)$$

$$Recall = \frac{T_p}{T_p + F_n} \quad (3)$$



The accuracy results of the all the trials are tabulated. The

Fig. 4: Precision, Recall and F1-score of all the four trials (Trail1-left top, Trail2-right top, Trail3-bottom left, Trail4-bottom right)

The value of precision and recall may vary with application to application. For example, an application attains high precision but low recall. Similarly, another application may attain low precision but high recall. To deal with this situation, one can rely on F1-measure. It is a harmonic mean of precision and recall. To obtain the F1-measure, a formula is given in Equation 4.

$$F1\text{-}measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad (4)$$

where,

Tp = true positive, Fp = false positive, Fn = false negative.

This article uses 5-fold cross validation [23] to evaluate the performance of machine learning approaches mentioned above, over a corpus of 1, 00,000 tweets with sarcastic hashtag (#). In the 5-fold cross validation, the dataset is divided into five equal chunks where one part (20,000 tweets) is used as a testing and remaining four parts (80,000 tweets) are used as training set. In this experiment, we perform five iterations, and for each iteration, the testing set is different (one of the five chunks). The accuracy results of all the five iterations are averaged to produce a single estimation. Similarly, the whole process is repeated four times as four trials, and in each trial, the tweets in the corpus are shuffled to validate the result effectively.

TABLE III: Results of the 5 iterations of Trial 1

Classifier	Itr1	Itr2	Itr3	Itr4	Itr5	Average (%)
NB	73.78	74.72	73.69	75.51	75.63	74.66
DT	79.99	78.96	79.20	79.16	80.23	79.50
SVM	79.69	81.06	80.23	80.91	81.45	80.67
RF	77.98	80.38	80.77	79.99	79.30	79.68
AdaBoost	80.91	80.13	78.28	80.77	80.28	80.07

Table III represents the accuracy result of trial 1, and an average is calculated and shown in the last column of the table for every classifier. Similarly, Table IV, Table V, Table VI represents the accuracy results of trials 2, 3, 4 respectively and an average of each trial is calculated and shown in the last column of every classifier.

TABLE IV: Results of the 5 iterations of Trial 2

Classifier	Itr1	Itr2	Itr3	Itr4	Itr5	Average (%)
NB	73.75	76.72	75.51	74.72	73.00	74.74
DT	78.23	80.47	79.40	79.01	80.08	79.44
SVM	80.91	80.38	79.01	79.50	78.77	79.71
RF	80.72	80.52	80.38	79.45	79.84	80.20
AdaBoost	80.18	79.64	80.77	79.89	79.16	79.93

TABLE V: Results of the 5 iterations of Trial 3

Classifier	Itr1	Itr2	Itr3	Itr4	Itr5	Average (%)
NB	75.72	73.47	77.67	74.59	74.15	75.12
DT	79.31	79.30	80.23	80.33	79.20	79.67
SVM	79.16	80.18	79.64	79.89	79.16	79.60
RF	80.62	80.47	80.18	82.18	80.52	80.79
AdaBoost	80.57	79.59	80.62	79.06	79.25	79.82

TABLE VI: Results of the 5 iterations of Trial 4

Classifier	Itr1	Itr2	Itr3	Itr4	Itr5	Average (%)
NB	74.93	75.08	72.15	74.93	73.23	74.06
DT	80.77	80.18	79.99	80.62	79.79	80.27
SVM	80.03	80.23	79.69	80.13	80.42	80.10
RF	80.38	78.77	79.59	78.33	79.45	79.30
AdaBoost	79.40	80.23	79.59	79.45	79.74	79.68

The attained values of precision, recall and F1-measure are shown in Fig. 4. The x-axis represents different machine learning classifiers and y-axis represents the attained precision, recall and F1-measure. Fig. 4 consists of four subfigures, in which each sub-figure represents one of the four trials. A comparison of the proposed scheme with existing state-of-art approaches is shown in TABLE VII. The comparison was done with various statistical parameters such as precision, recall, and F1-score.

V. CONCLUSION

The sarcastic text does not have any fixed structure. Due to this, the analysis of sarcasm in textual data using standard machine learning classifiers is a difficult task as compared to the other approaches like pattern-based, context-based, *etc*. In this article, we deployed five classical machine learning algorithms namely NB, DT, SVM, RF and AdaBoost to classify sarcastic and non-sarcastic tweets using hyperbolic feature set. The performance of classifiers as mentioned above is significantly better than the existing system as it attains the accuracy (%) of 75.12, 80.27, 80.67, 80.79, 80.07 using NB, DT, SVM, RF, and AdaBoost respectively.

TABLE VII: Comparison of proposed approach with some of the state-of-the-art techniques.

Approach	Precision	Recall	F1-score
Tungthamthiti *et al.*[23]			
with Contradiction in sentiment score	0.56	0.55	0.56
with proposed features in SVM	0.63	0.64	0.63
with Uni-gram features in SVM	0.73	0.72	0.73
Riloff *et al.* [3]			
system with positive verb	0.28	0.45	0.35
with negative situation	0.29	0.38	0.33
with SVM	0.42	0.63	0.50
Contrast (+VPs, -situation) unordered	0.11	0.56	0.18
Contrast (+VPs, -situation) ordered	0.09	0.70	0.15
Contrast (+preds, -situation)	0.13	0.63	0.22
Bharti *et al.* [10]			
with Naive Bayes	0.43	0.41	0.42
with SVM	0.47	0.68	0.55
with Maximum Entropy	0.47	0.46	0.46
with Decision Tree	0.51	0.42	0.46
Proposed approach			
with Naive Bayes	0.81	0.77	0.79
with Decision Tree	0.83	0.81	0.82
with SVM	0.83	0.80	0.81
with Random Forest	0.82	0.82	0.82
with AdaBoost	0.84	0.80	0.82

REFERENCES

[1] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[2] Macmillan, "Macmillan dictionary," 2007.

[3] E. Riloff, A. Qadir, P. Surve, L. De Silva, N. Gilbert, and R. Huang, "Sarcasm as contrast between a positive sentiment and negative situation.," in *EMNLP*, pp. 704–714, 2013.

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

[5] M. Bouazizi and T. O. Ohtsuki, "A pattern-based approach for sarcasm detection on twitter," *IEEE Access*, vol. 4, pp. 5477–5488, 2016.

[6] S. Bharti, B. Vachha, R. Pradhan, K. Babu, and S. Jena, "Sarcastic sentiment detection in tweets streamed in real time: A big data approach," *Digital Communications and Networks*, vol. 2, no. 3, pp. 108–121, 2016.

[7] C. Liebrecht, F. Kunneman, and A. van den Bosch, "The perfect solution for detecting sarcasm in tweets# not," in *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pp. 29–37, New Brunswick, NJ: ACL, 2013.

[8] P. Liu, W. Chen, G. Ou, T. Wang, D. Yang, and K. Lei, "Sarcasm detection in social media based on imbalanced classification," in *Web-Age Information Management*, pp. 459–471, 2014.

[9] D. Tayal, S. Yadav, K. Gupta, B. Rajput, and K. Kumari, "Polarity detection of sarcastic political tweets," in *proceedings of International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 625–628, IEEE, 2014.

[10] S. K. Bharti, R. Pradhan, K. S. Babu, and S. K. Jena, "Sarcasm analysis on twitter data using machine learning approaches," in *Trends in Social Network Analysis*, pp. 51–76, Springer, 2017.

[11] D. Bamman and N. A. Smith, "Contextualized sarcasm detection on twitter.," in *ICWSM*, pp. 574–577, 2015.

[12] Z. Wang, Z. Wu, R. Wang, and Y. Ren, "Twitter sarcasm detection exploiting a context-based model," in *International Conference on Web Information Systems Engineering*, pp. 77–91, Springer, 2015.

[13] R. Schifanella, P. de Juan, J. Tetreault, and L. Cao, "Detecting sarcasm in multimodal social platforms," in *Proceedings of the 2016 ACM on Multimedia Conference*, pp. 1136–1145, ACM, 2016.

[14] A. Joshi, V. Sharma, and P. Bhattacharyya, "Harnessing context incongruity for sarcasm detection.," in *ACL (2)*, pp. 757–762, 2015.

[15] A. Khattri, A. Joshi, P. Bhattacharyya, and M. J. Carman, "Your sentiment precedes you: Using an author's historical tweets to predict sarcasm.," in *WASSA@ EMNLP*, pp. 25–30, 2015.

[16] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of English: The Penn treebank," *Computational linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[17] M. Andrew and K. Nigam, "A comparison of event models for naive Bayes text classification," *workshop on learning for text categorization*, vol. 752, 1998.

[18] Q. J. Ross, "Simplifying decision trees," *International journal of man machine studies*, vol. 27, no. 3, pp. 221–234, 1987.

[19] C. Corinna and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[20] H. T. Kam, "Random decision forests," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[21] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics Springer, Berlin, 2nd ed., 2008.

[22] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *European conference on computational learning theory*, Springer Berlin Heidelberg, 1995.

[23] S.S. Mukku, N. Choudhary, and R. Mamidi, "Enhanced Sentiment Classification of Telugu Text using ML Techniques". In SAAIP@ IJCAI, pp. 29-34, 2016.