RANDOM FOREST-BASED CLASSIFIER FOR AUTOMATIC SARCASM CLASSIFICATION ON TWITTER DATA USING MULTIPLE FEATURES

CHRISTOPHER IFEANYI EKE [1], AZAH ANIR NORMAN[2], LIYANA SHUIB [2], FATOKUN FAITH B.[3], ZALIZAH AWANG LONG[3]

[1] Department of Computer Science, Federal University of Lafia, Nasarawa State, Nigeria.

[2]Department of Information Systems, Faculty of Computer Science & IT University of Malaya Kuala Lumpur, Malaysia

[3]Malaysian Institute of Information Technology University of Kuala Lumpur Kuala Lumpur, Malaysia

**Corresponding author: eke.christopher@science.fulafia.edu.ng*

ABSTRACT: Sarcasm is one of the nonliteral languages usually employed in social networks and microblogging websites to convey implicit information in an individual communication message. This could lead to the misclassification of tweets. This paper focuses on sarcasm detection on tweets, which has been experimented with the use of textual features. The textual features comprise the Neural language fusion and Natural language features, which include sentiment-related features, semantic and synthetic features, punctuation-related features, and GloVe embedding features. The features mentioned above were extracted separately from the target tweet and fused to form fused features for the target tweet. The proposed predictive model attained an accuracy of 86.9% with a random forest classifier, which outperformed other models employed in the experiment, such as DT (83.9), SVM (80.5), KNN (83.1), and LR (82.9).

KEYWORDS: *Natural language processing, Sarcasm detection, Classification algorithm, Random Forest, GloVe embedding*

1. INTRODUCTION

Sarcasm detection has become the main challenge in natural language processing applications. The development of information communication technology and the Internet has advanced social media usage, including Instagram, Twitter, YouTube, and Facebook. People connect to social media in order to exchange information and ideas as well as to discuss the importance of trends happening around the globe (Bharti et al., 2016). Hence, a huge amount of user-generated data are obtained in social networks daily, which needs to be analyzed.

Automatic identification of sarcasm has not been widely studied (González-Ibánez et al., 2011; Onan, 2017b). Twitter, one of the microblogging sites, enables individuals to show their views, ideas, and feelings in a short message form, usually referred to as tweets. Twitter is one of the biggest online microblogging platforms that publish over 143,199 posts per second (Chen et al., 2016). Users employ

Twitter for different reasons, including conversation, giving out information, election purpose, and reading breaking news (Davidson et al., 2020; Java et al., 2007). Lately, Twitter has acted as an essential means of information for scholars and experts, considering the huge volume of messages that users generate on a Twitter daily (Onan, 2017a).

Sarcasm detection is the task of employing a natural language processing approach for text classification of expressions that contain attributes and properties that are sarcastic (Yavanoglu et al., 2018). When sarcasm is employed in an expression, it is difficult to effectively recognize using the conventional data mining approach due to the variation in its explicit and implicit meaning in the expression (Yee Liau & Pei Tan, 2014). Due to sarcasm's ambiguous nature, finding the differences between sarcastic expressions and non-sarcastic expression is very difficult for an individual (Muresan et al., 2016). In addition, there is a lack of correctly labelled naturally occurring sentences as sarcastic that can be employed in the training of supervised machine learning algorithms. However, when a microblogging platform such as Twitter is used, hashtags are used to annotate messages, a sentiment indicator demonstrated in the tweets' utterances. These hashtags serve as a reliable indicator of emotion being explicitly conveyed in the author's tweets (e.g. #love, #unhappy, #amazing).

Sarcasm identification task has been investigated by various scholars by paying much attention to different feature usage and fusion approaches in their studies. For instance, Mukherjee & Bala (2017) utilized the content features in their study on sarcasm detection. However, the authors extracted emoticon features, word usage, and generally the expression structures in differentiating the sarcastic expressions from non-sarcastic counterparts. Hence, the study did not consider the neural language model, which could improve the predictive performance.

In this study, a random forest-based sarcasm detection using multiple features is proposed. The study, therefore, examined features that include sentiment-based, pragmatic (punctuation), syntactic, and GloVe embedding for sarcasm identification. In addition, these features have been represented using vector representation. In the classification phase, a set of standard supervised classifiers such as support vector machine, bagging, random forest, decision tree, k-nearest neighbor, and logistic regression has been experimented with.

The rest of the research is arranged thus: In section 2, a description of the literature survey is given. Section 3 discusses the proposed approach of the studies. In section 4, the experimental procedures, the empirical results, and discussions are presented. Section 5 finally brings the study to a conclusion.

2. LITERATURE SURVEY

Various kinds of research were conducted by several researchers on sarcasm classification in social media data using several features and classification models (Abulaish & Kamal, 2018; Sreelakshmi & Rafeeque, 2018; Suhaimin et al., 2018). For instance, Mukherjee and Bala (2017) proposed an approach that requires supplying knowledge to systems that describes author-style features for sarcasm detection on Twitter. In addition, various linguistic features were also considered in their study. They utilized the combinations of supervised (Naïve Bayes) and unsupervised (fuzzy clustering) learning algorithms in the modeling stage.

However, the prediction results show that the utilization of both learning algorithms and the use of features that do not rely on the text, enhances sarcasm detection results. Consequently, the method's drawback is the use of the author's style feature, which could be problematic when other types of features are utilized. To utilize the benefits of feature fusion, Castro et al. (2019) experimented with data fusion for sarcasm detection. The author integrated various types of features including the audio, video, and text features to test the effect of the combined features in sarcasm detection. In the text data, the author employed BERT for feature extraction (Devlin et al., 2018). To extract speech features, the Librosa library was used, which is a standard library for sound extraction that considers only the low-level audio data to utilize audio modality information. Furthermore, pool 5 layers of an ImageNet (Deng et al., 2009) was used to extract visual features in video pronouncement. Thus, the predictive results show an improvement of over 12.9% error rate reduction when features from audio, image, and text are combined than when only one feature is used. In a related study, the effectiveness of multiple feature fusion on the sarcasm detection framework was investigated by Eke et al. (2021b), which divided the classification into two different stages. The constructed classification model was tested in various experiments to compare the performance of each model. The comparison results indicated that the classifier's prediction based on the developed framework and feature fusion achieved the highest result of 0.947 precision with the RF algorithm, which outperformed the existing baseline methods. Razali et al. (2021) conducted a study on sarcasm classification using contextual features. The features were extracted from tweets dataset and modelled with diverse learning models, including SVM, DT, KNN, LR, DISCR. However, the experimental analysis indicated that LR outperformed all the tested models by attaining a detection accuracy of 94%. The contextual sarcasm detection in online discussion forums study was conducted by Hazarika et al., (2018). The author employed Reddit dataset to extract features, which were applied on various combination of learning models such as CNN, SVM, and CUE. However, the model performance attained an optimum performance of 86% with F-score measure. The performance of TF and TF-IDF features for sarcasm classification was tested by Nayel et al. (2021) using Twitter data. The convetional machine learning models such as SVM, NB, LR were experimented and the result showed that SVM outperformed other models by attaining an accuracy of 84.22%.

Researchers in this field now have the chance to perform research on the automatic detection of sarcasm thanks to the popularity of deep learning algorithms (Eke et al., 2021a; Nweke et al., 2018). This type of learning uses neural networks to automatically learn from big datasets as a subset of machine learning. For instance, Eke et al. (2021a) investigated the detection accuracy of context-based sarcasm classification using a deep learning and BERT models. The empirical analysis of the model obtained a promising result. The performance of the proposed technique was implemented using two publicly available datasets, which produced a highest precision of 98.5% on Twitter dataset and 81.2% on IAC-v2 dataset, which outperformed the baseline method for sarcasm detection. In a related study, Savini and Caragea (2022) performed an intermediate-task transfer learning with BERT for sarcasm detection. The author employed both Twitter and Reddit data to train the BERT model and achieved an F1-score of 97.43%. To compare the performance of the English and the Filipino datasets on sarcasm detection, Samonte et al. (2018) conducted an experiment with traditional machine learning

models that consist of SVM, NB, and ME. However, the predictry result showed that SVM outperformed other models in both datasets and a better performance was experienced in the Filipino data than in the English data.

In another study, Baruah et al. (2020) experimented with the idea of using historical knowledge to identify sarcasm using a deep learning Bidirectional Encoder Representations from Transformers (BERT) architecture. They made use of traditional conversational elements including response and last expression, last two expressions, and last three expressions. In a related study, Ilić et al. (2018) employed a deep learning model based on character-level word representations obtained from the Embedding from Language Models (ELMo) in another project. ELMo is a vector representation approach obtained from a bidirectional Long Short Term Memory (LSTM) (Peters et al., 2018). This approach employed a hashtag-created dataset (Ptáček et al., 2014). Mehndiratta et al. (2017) proposed an approach for sarcasm analysis by utilizing the deep convolutional neural model. The authors utilized the sentiment and word embedding (word2vec &skip-gram) features, which were fed as input to the DCNN model for classification. The study produced promising results. However, the word sense was not captured independently in the approach. In a related study, Liu et al. (2019) conducted a study on A2Text-Net, a novel DNN for sarcasm on identification. In the study, the author employed three different datasets, which include news headlines,Twitter, and Reddit. Several algorithms were tested, including DNN, LSTM, SVM, RF, LR, GRU, and A2Text-Net. The experimental results showed the highest performance of 99.7% AUC with LSTM (Bedi et al., 2021) on the Twitter dataset. Other researchers have employed different deep learning models such as LSTM , Bi-LSTM (Du et al., 2022; Kumar et al., 2019), ANN (Babanejad et al., 2020), and CNN (Manjusha & Raseek, 2018) for sarcasm detection study and obtained promising results. To have a look at more studies on sarcasm detection, refer to Christopher Ifeanyi Eke et al. (2019), who carried out a comprehensive systematic review on sarcasm prediction on textual data by analyzing the predictive performance of different studies by considering the datasets, feature extraction, feature representation, classification algorithm, and performance measures employed for sarcasm detection.

The summary of the literature survey is provided in Table 1 with respect to the data source, model employed, and performance attained.

Table 1: Summary of literature survey

S/N	Data source	Model employed	Performance attained	References
1	Twitter	NB and Fuzzy C-means clustering	65% accuracy	Mukherjee and Bala (2017)
2	Twitter	Deep convolutional neural network	89.9%. accuracy	Mehndiratta et al. (2017)
3	Twitter	NB, Bagging, DT	99% Precision with Bagging, 92% Recall with NB, and 94% F-score with DT classifier	(Abulaish & Kamal, 2018)
4	Twitter	SVM, NB, ME	English dataset (93.1% accuracy with SVM), and Filipino dataset (98.7% accuracy with SVM)	(Samonte et al., 2018)

5	Twitter	SVM, DT	79% accuracy with SVM, and 74.1% with DT classifier	(Sreelakshmi & Rafeeque, 2018)
6	Twitter	SVM, DT, KNN, CNN	79% F-score with CNN	(Manjusha & Raseek, 2018)
7	Twitter, Reddit, Online dialogue	ElMo, BiLSTM	87.6% on Twitter data, 76% on online dialogue data, and 78.5% on Reddit data.	(Ilić et al., 2018)
8	Reddit	CNN, CNN–SVM, CUE–CNN, CNN based designed model	F-score of 86%	(Hazarika et al., 2018)
9	Twitter	Bi-LSTM	97.87% Accuracy	(Kumar et al., 2019)
10	Social media	SVM	90.5% F-measure	(Suhaimin et al., 2019)
11	News headlines, Twitter, Reddit.	DNN, LSTM, SVM, RF, LR, GRU A2Text-Net	93.7% AUC with A2Text-Neton News headings data, 99.7% AUC with LSTM on Twitter data, 77.9% AUC with A2Text-Net on Reddit data	(Liu et al., 2019)
12	Twitter, Reddit.	BERT, Bi-LSTM and SVM	F-score of 74.3% and 65.8% for the Twitter and Reddit data	(Baruah et al., 2020)
13	Internet, Twitter	ANN, BERT	92.2% F-score	(Babanejad et al., 2020)
14	Twitter, Internet	Bi-LSTM, BERT	98.5% and 98.0% Precision with Twitter data. 81.2% Precision with IAC data	(Eke et al., 2021a)
15	Twitter	SVM, DT, KNN, LR, and RF	94.7% Precision with RF	(Eke et al., 2021b)
16	Twitter	SVM, NB, LR	84.22% Accuracy with SVM	(Nayel et al., 2021)
17	TV show	LSTM	86.2% Accuracy	(Bedi et al., 2021)
18	Twitter	SVM, KNN, LR, DT, DISCR	94% Accuracy with LR	(Razali et al., 2021)
19	Twitter, Reddit	BERT	97.43% F-1 score	(Savini & Caragea, 2022)
20	Twitter, Reddit	Bi-LSTM	71% Accuracy on both Twitter and Reddit data	(Du et al., 2022)

3. PROPOSED APPROACH

This segment provides the research design to construct a machine learning algorithm for sarcasm analysis using the Twitter dataset. In a given tweet, the goal is to perform a text classification on them in order to recognize which one is from a positive class or a negative class. The proposed method is based on the supervised learning for sarcasm detection. The framework for the overall approach is depicted in **Error! Reference source not found.**. The methodology design consists of five main steps: data collection, data preprocessing, feature extraction, sarcasm classification techniques, and performance evaluation metrics. A detailed description of each step is given in the sub-sections below.

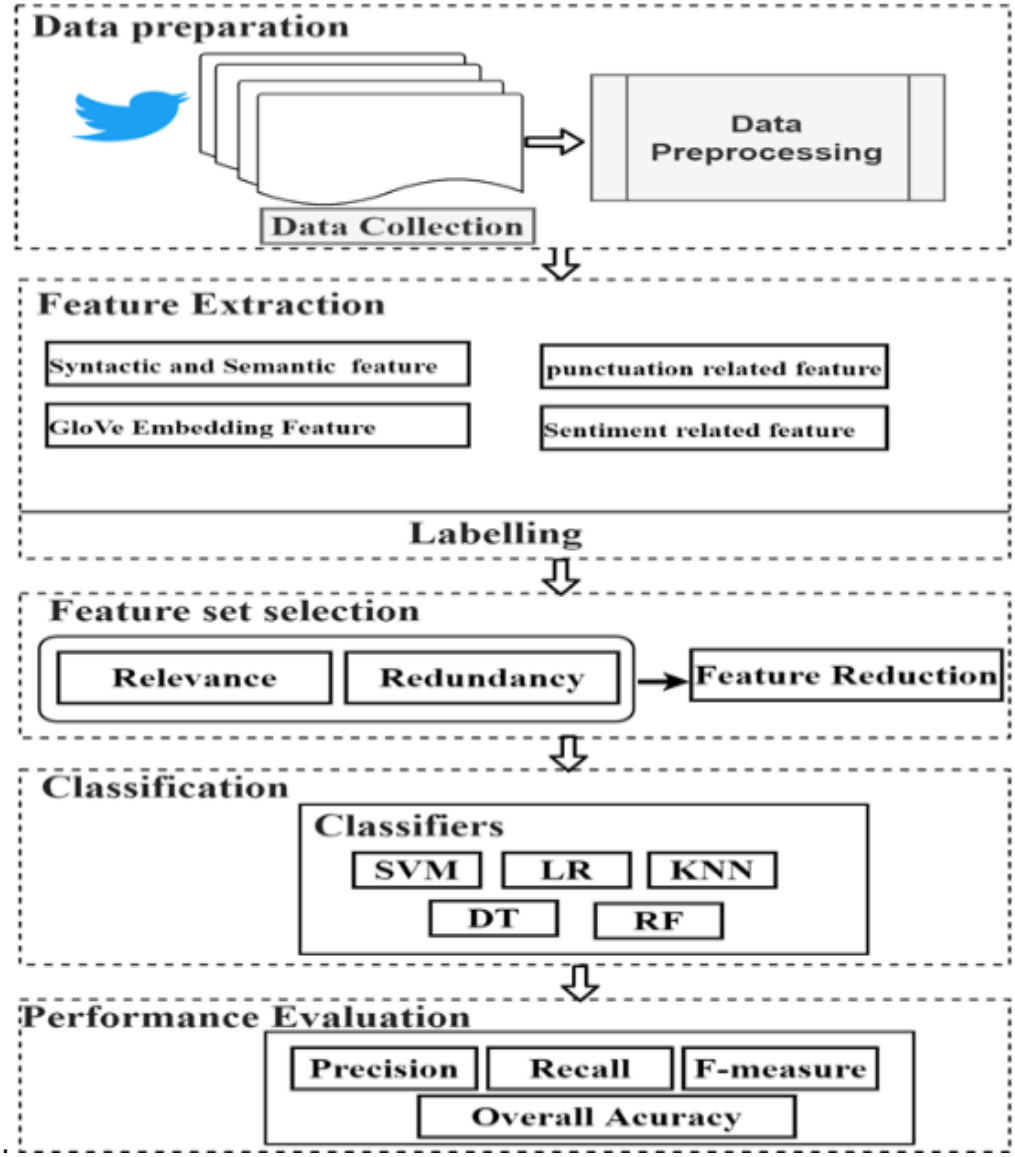

Fig.1. Proposed feature fusion for sarcasm identification in Twitter

3.1 Data Preparation

The experimental dataset was obtained from Twitter for both positive and negative classes. To perform the collection process, a Twitter application programming interface was used, which connects the users and the Twitter servers to easily retrieve tweets from the tweet archive. To create the sarcasm dataset, the tweets labelled by the tweets composer were used. However, the tweets annotated with a hashtag (#) are regarded as positive class tweets, whereas the tweets with no hashtag (#) annotation are assumed to be negative class tweets. Moreover, to remove quotes, non-English tweets, duplicates, and remove spam, tweets below three words in length were considered. The advantage of utilizing the Twitter API is that it enables us to obtain as many samples as possible. This is because people compose tweets every day and make use of sarcasm that can easily be collected and saved in a database. This study collected 30,000 volumes of tweets consisting of about 15,000 sarcastic and 15,000 non-sarcastic with the keywords #sarcastic or #sarcasm (Mukherjee & Bala, 2017; Schifanella et al., 2016) for positive while

the keywords without sarcasm or sarcastic hashtag (Sreelakshmi & Rafeeque, 2018) or with keyword #notsarcasm or #notsarcastic (Mukherjee & Bala, 2017) were used to represent non-sarcastic. The collection approach was based on the automatic retrieval of tweets using keywords (ARTK). The datasets consist of real-time tweets covering the aspects of politics, education, and technology. The dataset collection was made within a period of four months ranging from the month of June 2019 to September 2019.

3.2 Data Preprocessing

In the data preprocessing stage, various data preprocessing methods were implemented on the data by utilizing the NLP techniques, including stop-word removal, parts of speech tagging, lemmatization, and stemming. A short description of each of the methods is provided thus.

Stop words: Stop words consist of prepositional words and articles, which have negligible or no effect on the context of the sentence, and also lack contribution in text mining. They include the, into, in, on, at, under, etc. Therefore, the stop words that are found in the natural language processing toolkit were used to eliminate those words from the dataset.

Tokenization: Tokenization is a method of breaking sentences or words order into smaller segments, also referred to as tokens. They include symbols, words, and phrases that can stand on their own. This method also removes an empty white space, a character that usually occurs in a text. A token is an order of characters that exists in the text that combines to form a suitable semantic unit that is important during the data analysis stage. Therefore, the output of tokenization represents the input for further investigation. Thus, the whole tokenization job can be carried out by employing the natural language processing toolkit.

Stemming: This refers to the process of returning the root form of the word, also referred to as stem from its derivative state. This can be achieved by eliminating the suffixes and prefixes from the word. The process decreases the keyword's volume from the space of keywords, which improves the predictive accuracy whereby a particular keyword is derived from various keywords. For instance, the word 'scaling' can be stemmed from the word 'scale.'

Lemmatizing: When the suffixes and prefixes are eliminated from the derived word, in most cases, causes the word to be meaningless. Therefore, Lemmatizer fixes the lost character on the stemmed word to make the meaning out of it. For example, the stemming word 'improved' to the word 'improv' can result in the word improvement after lemmatization by inputting the character 'e' to the stemmed word 'improv.'

Parts-Of-Speech tagging: Parts-of-speech tagging is performed using parts of speech tagger, which scans through the test file and assigns parts of the speech, such as verb, interjection, pronoun, adverb, noun, etc. to every word token according to their definition. However, the fine-grained parts of speech tagging are required in most computer science applications. For example, a noun can be further tagged into singular, proper, plural, and progressive nouns by using NN, NNS, and NP notation. Thus, parts of speech tagger use a rule-based and stochastic algorithm for tagging.

3.3 Feature Extraction

Feature engineering is a vital aspect of constructing any intelligent system. The feature engineering phase deals with identifying, extracting, and representing the variables/features that make tweets sarcastic or non-sarcastic. Feature engineering is a process of transforming the input data into feature vectors. The process reduces the number of resources needed to describe the dataset. In this process, measurable attributes are obtained by breaking down each given sample. The feature vectors serve as an input to the classification algorithm, so that good quality feature enhances the performance of the classifier. In this study, a wide range of features for sarcasm identification purposes have been utilized. Exploiting numerous features offers us an opportunity to compare different performance measures obtained for feature fusion. Features such as sentiment, punctuation (pragmatic), semantic and syntactic, and GloVe embedding features were utilized in this research for sarcasm identification. A brief description of these features is given below.

Sentiment-based features: There are some instances, whereby the positive sentiment is used for describing negative situations. This situation is commonly referred to as a form of sarcasm called a whimper. In such an instance, the creator of the sarcasm expression defines the negative situations by employing the positive sentiments. A related study was experimented with by Riloff et al. (2013) on sarcasm detection. For example, 'I delight working on public holidays'. This research tested the existence of conflict between the polarity of the word and other tweet components for sarcasm identification. Thus, various sentiment features are obtained from the tweets and calculated. A lexicon, usually known as SentiStrength is used to extract the sentiment polarity of a word (Thelwall et al., 2012). It uses lexical rules and information to identify the polarity of English words, which could be either negative or positive, including emoticon, slang booster, slang, emotion, idiom, questions, and negation. The range of scores, between -5 and +5, with the stronger polarity representing the large numbers, is used for the polarity of sentence representation. Moreover, more than six features are obtained that show contradiction between the sentiment components, which include positive sentiment words, negative sentiment words, highly emotional positive content, highly emotional negative content, hashtag feature, co-existence of positive sentiment & negative sentiment word, positive sentiment & negative sentiment word with the hashtag, and positive sentiment & negative sentiment word with an emoticon. To extract sentiment related features from the content of a tweet, a dictionary that consists of positive words and negative words is created using the SentiStrength (Thelwall et al., 2012) database. SentiStrength is a sentiment lexicon that utilizes linguistic rule and information to detect an English text sentiment. The lexicon usually provides the polarity sentiment (positive and negative) of words like question, negation, emotion, booster, idioms, slangs, and emoticons. The sentiment score uses an integer ranging from -5 to +5, in which the larger absolute value represents the stronger sentiment. The first two features are extracted using the two lists by computing the number of sentiment words that tend to be positive or negative. The next two features (highly positive and negative positive words) are extracted by checking if any of the positive or negative sentiment words are associated with highly emotional parts of speech (adjective (JJ), verb (RB), and adverb (VB)) tags tweets. If it occurs, an integer 1 is recorded; otherwise, 0 is

recorded. The hashtag features could be a positive hashtag or a negative hashtag. In this study, three sets of hashtag features are defined: a positive hashtag, a negative hashtag, and the co-existence of the positive and negative hashtags. The hashtag features are extracted by creating a dictionary that consists of a list of negative hashtag words such as "#hate, #pity, #waste, #discrimination, etc", and a list of a positive hashtag words such as "#happy, #perfect, #great, #goodness, etc." By using this dictionary, the number of positive and negative hashtags present in the tweeted text are computed and added as a feature. Lastly, the last three features are extracted by checking the co-existence of the positive sentiment & negative sentiment words, positive sentiment & negative sentiment words with the hashtag, and positive sentiment & negative sentiment words with an emoticon in the same tweet, by recording an integer 1 if there is co-occurrence, otherwise 0. Therefore, the sentiment-based feature contains seven subsets of the feature. Thus, eight features were extracted as a sentiment feature vector.

Punctuation (pragmatic) features: In this study, we utilized punctuation marks as the pragmatic features. Punctuation has an important effect on text analysis, especially in sentiment analysis. Punctuation symbols are mostly used as an explicit mark that brings out the sarcastic expression in the text. In punctuation related features, six different sets of features were considered and were extracted from the tweets content. To extract punctuation marks from the tweets, a regular expression is employed to check the punctuation marks present in the sarcastic expressions. After that, the number of times each of them is used is computed. First, the numbers of question marks were calculated and extracted as a feature (?). The second feature was obtained by counting the number of exclamation marks in the text (!). The third feature calculated the number of ellipses (.) in the text. The fourth feature considered the presence of capitalization in the text and computed the number of occurrences, i.e it searches for the word that is "All-capitals" and extracted it as a feature in the text. The fifth feature calculated the quoted words, which are the words that are in a quote, and added it as a feature. Lastly, the sixth feature calculated the repeated vowels in the text and added it as a feature. Thus, these six features formed a feature set for the related pragmatic features.

Syntactic features: Syntactic features perform a significant function in providing information regarding the tweets text syntactic structure. In this study, three sets of features that include the POS feature, interjection word, and laughing expression are defined as syntactic features and were extracted from the processed tweet's content. To extract the syntactic feature, this study employed the NLTK tokenizer library to perform tokenization tasks on the processed tweets. First, we extracted the POS feature using the parts of speech dictionary as the basis, and the count of its presence in the sarcastic text is taken. We only focused on the parts of speech details with some emotional contents such as nouns, adverbs, and adjectives. Furthermore, the mapping of each of the POS tags and each corresponding POS group was established, and only the tokenized words that correspond with the chosen three parts of speech groups as aforementioned were preserved in the text. The study employed the same framework used in Berry & Castellanos (2004) and extracted ADV+ADJ+N (adverb, adjective, and noun). Second, to extract the second feature, we identified laughter words that are used to express pleasures or joy. Thus, we

added laughing features, which is the sum of internet laughs, represented with lol, hahaha, hehe, rofl, and imao, which we refer to as a new punctuation way. The feature was extracted by creating a dictionary list that contains the most common laughing words and was used to find the frequency of such words. Then, the frequency of such words present in the text was computed and added as a feature. The third feature is extracted by identifying interjection words (Bouazizi & Ohtsuki, 2016) such as woo, oh, wow, etc in the tweets and the frequency of interjection words was computed and added as a feature.

GloVe embedding (GE) features: GloVe, as the name appears, stands for 'Global Vectors'. GE is a strong neural model that is employed for word vector representation via dimension reduction on a co-existence matrix. The GE scheme is created by creating a large co-existence of matric information with their corresponding contents on how often each 'word' saved in a row occurs in the column. It is a neural model in which the same words that are clustered together repel against one another. In the GE scheme, a semantic relatedness of the word can be obtained using the co-existence matrix (Pennington et al., 2014). GE offers numerous advantages over other neural language models. One of the benefits is that it can capture both the local context, usually referred to as the local statistics as well as the word's relatedness, usually referred to as the global statistics in a corpus in order to acquire word vectors. The feature of parallel implementation found in GE makes it possible to model on a large corpus. Besides, in order to create new feature vectors, it integrates the discriminative features obtained from the two model relations, which are the global factorization and local content window approaches (Eke et al., 2020; Pennington et al., 2014).

Table 2: Summary of the proposed features for sarcasm classification

NO	Groups	Features
1	Syntactic features	Laughing expression, POS (Noun, verb, adverb, and adjectives), and Interjection words
2	Punctuation features	Exclamation mark, Question mark, Ellipsis, Quoted word, All capitals, Repeated vowels.
3	Sentiment related features	Positive sentiment words, Negative sentiment words, Highly emotional positive content, highly emotional negative content, hashtag feature, co-existence of positive sentiment & negative sentiment word, positive sentiment & negative sentiment word with the hashtag, and positive sentiment & negative sentiment word with an emoticon
4	GLoVe	GLoVe Embedding features

3.4 Classification Algorithm

The classification step, also known as model training, is the next step after feature extraction in any text classification technique. The computers learn from algorithms, unlike the people who learn from experience because of their ability to

reason. The model can be trained by using various approaches such as the supervised approach (input mapped to desired output) and the unsupervised approach (auto-detection of data disregarding pattern to class assignment) (Eke et al., 2019). This study utilized the supervised learning approach. In the supervised learning, data is separated into testing sets and training sets. The supervised learning uses either a classification approach or a regression approach for modelling. A text classification deals with assigning a label to text documents such that the output variable is categorical. The classification performance can be measured by employing cross-validation techniques (that is, by using some portion of labelled data for training and another portion for testing the model). Evaluation indicators such as F1-score, precision, recall, and accuracy were utilized to evaluate the classifiers' performance. Regression, on the other hand, uses training sets to make a prediction as well but produces continuous variables as the output variable. In this study, the classification models were utilized for classifying tweets as sarcastic and non-sarcastic. Classifiers such as decision tree, random forest, k-nearest neighbor, support vector machine, and ensemble classifiers have been experimented in diverse analyses in order to choose the highest performing predictive model for the sarcasm classification. Below is a summary explanation of some of the classification algorithms utilized in this study.

Decision Tree: The decision tree classifier uses the structure of the tree to make a decision (Quinlan, 1987). A decision tree can simply manage the feature interaction even in the absence of a parameter. The model reliance is on the value of features like the sorting algorithm classification. The tree is made up of each instance that requires classification, usually referred to as nodes, and the value that nodes can assume is usually referred to as branches. The instance classification usually begins from the root node.

Support Vector Machine: Support vector machine, proposed by Cortes and Vapnik (1995) is a binary classification algorithm, which is supervised and linear in nature. It is a machine learning model that constructs a set of hyperplane using high dimensional space for splitting data into various classes. It is a text mining classifier that involves the suitable classification of instances of problems by selecting the best hyperplane.

Logistic Regression: Logistic regression, as the name implies, is a linear classification algorithm employed for the classification of the event occurrence probability as the linear function of a predictive class variable (Kantardzic, 2011). In a linear regression classifier, the linear function features are always constructed using the decision boundaries. This classifier aims to enhance the probability function for easy recognition of document class labels and to achieve conditional probability using feature selection techniques (Aggarwal & Zhai, 2012). Logistic regression is a standard classifier that attains optimum performance. However, it mostly generates class variables outside 0 and 1, which is unacceptable for the probability range.

K-Nearest Neighbor: K-nearest neighbor is a classification algorithm that is based on the instances for solving the regression as well as a classification task. This classification algorithm relies on the k-nearest neighbor of a particular instance to identify the class label for individual instances. As a result, the k-nearest neighbor uses the instance of majority voting technique to determine the class label of each instance. Thus, the majority vote of an individual neighbor is allocated to the

instance of its class in this classification scheme, which is the highest common k-nearest neighbor class instance (Han et al., 2011).

Random Forest: Random forest has currently gained recognition as a result of its robustness and resistance to noise when evaluated with single classifiers. A random forest is an ensemble of decision trees that are formed by integrating several decision trees. The reason for employing the combination of multiple decision trees is that working with a single tree classification model may produce noisy data or outliers that may likely influence the overall predictive performance. Moreover, a random forest classifier is very vigorous to outliers and noise due to the randomness that it offers. The random forest offers two forms of randomness, which include randomness with respect to data and randomness to features. Also, the idea of bootstrapping and bagging are employed in the random forest algorithm, which can be performed by increasing the difference tree that causes growth on diverse subsets of training data formed through bootstrap aggregating (Breiman, 1996).

3.5 Evaluation Measures

Evaluation measures, also known as performance metrics are indicators used to evaluate the experimental results. Different performance matrices are used to measure the classifiers' performance. Standard performance measures, which include, precision, recall, F1-score, and accuracy were employed to evaluate the predictive performance of the models. Every classifier shows its detection ability using the aforementioned metrics. A brief explanation of each of the metrics is provided below.

Classification accuracy (ACC) denotes the entire correctness of the classification result. It measures the fraction of true positives and true negatives attained by the classified instances over the total number of instances, which is represented in equation 1,

$$Acc = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TN} + \mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \tag{1}$$

where TP, TN, FN, and FP represent the true positive number, true negative number, false-negative number, and false-positive number, respectively.

Recall (REC) measures the fraction of true positives over the summation of a true positive and false negative. In other words, it computes the sum of the successfully classified sarcastic tweets over the total number of sarcastic tweets. It is depicted in equation 2.

$$REC = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2}$$

Precision (PRE) measures the fraction of true positives over the true positives and false positives. That is, it determines the number of tweets that have been effectively classified as sarcastic over the whole tweets that are classified as sarcastic. It is indicated in equation 3.

$$PRE = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \quad (3)$$

F- Measure (F-M) is a performance evaluation that unites precision and recall by computing their harmonic mean. It has been previously employed in the classification study to measure classifiers' performance as it considers precision and recall (Justo et al., 2014). F-M assumes the values of 0 and 1. It is represented in equation 4.

$$\mathrm{F-M} = 2 * \frac{\mathrm{PRE} * \mathrm{REC}}{\mathrm{PRE} + \mathrm{REC}} \quad (4)$$

4. EXPERIMENTAL SETTINGS

The classification experiment was carried out to analyze the sarcasm task (sarcastic and non-sarcastic) in a given tweet. The preprocessing and feature extraction tasks were carried out using a Python programming language. Subsets of features explained in section 3.3 have been employed in the sarcasm analysis experiment as the input to various classification algorithms. We experimented with six different classifiers: logistic regression, k-nearest neighbor, support vector machine, decision tree, bagging, and random forest by using a default parameter setting. The purpose of employing different models is to get the best performance result. The proposed technique was trained and tested using a 90% split ratio on the tweet dataset. In this pattern, the initial dataset is arbitrarily separated into two exclusive portions, where the first training portion is used for training the algorithm, and the other portion is for testing. The machine toolkit WEKA 3.9 (Waikato Environment for Knowledge Analysis), open-source software that consists of various machine learning algorithms executed in Java, was used for analysis. The motivation behind using Weka is due to its versatility as it combines both binary and continuous features. In addition, Weka has tools employed for preprocessing data, regression tasks, classification tasks, clustering, finding, association rules, and data visualization. The default settings of Weka have been used during the experiment. Various subsets of lexical features, sentiment features, pragmatic features, punctuation-related features, and GloVe embedding features have been experimented with. Four standard evaluation metrics such as accuracy, F-measure, precision, and recall were employed and weighted over both classes (sarcastic and non-sarcastic) during the experiment. The weights were obtained based on class ratios.

5. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this segment, the experimental results are described. The study employed 90% and 10% split, a similar approach found in González-Ibánez et al. (2011) for training and testing analysis on different classifiers, which includes support vector machine, logistic regression, k-nearest neighbor, random forest, and decision tree to estimate the existence of sarcastic sentiments in the given tweets. The proposed random forest-based sarcasm classification experiment was carried out to test the predictive performance of fused feature sets. Table 3 shows the values obtained on the simulated result by comparing different classifiers on sarcasm analysis, whereas

Fig.2 shows the visualization of the results. It is obvious from the results of the experiment that the best predictive performance is obtained on the ensemble classifier (random forest) with an accuracy of 0.869% and an F-score of 0.869% compared to other classifiers. This shows that the predictive performance was improved. It was also revealed that the least result is obtained on the SVM classifier with an accuracy of 80.5% and a precision of 80.4%. Thus, the results of the experiments show that the combination of features such as sentiment, punctuation related, semantic and syntactic, and GloVe embedding features produces better performance than when each feature set is used on its own. Thus, we conclude that the model's accuracy improves when feature fusion is used for sarcasm classification.

Table 3: Performance result of the proposed feature fusion classification using different classifiers

Classification Algorithm	Accuracy	Precision	Recall	F-measure
SVM	0.805	0.810	0.805	0.804
LR	0.829	0.830	0.829	0.829
KNN	0.831	0.832	0.832	0.832
DT	0.839	0.8401	0.839	0.839
RF	**0.869**	**0.869**	**0.869**	**0.869**

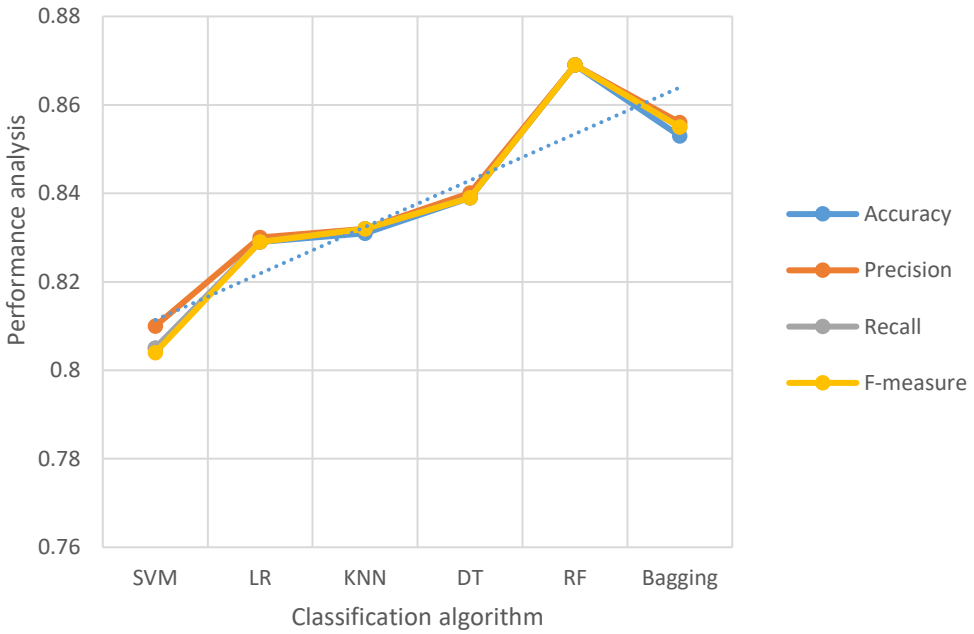

Fig.2. Result analysis of 90% split ratio with accuracy, precision, recall, and f-measure

5.1 Comparison with the baseline methods

Finally, we compare our proposed method with the three baseline methods for the identification of sarcasm on Twitter data. The three baseline methods were established to compare with our proposed approach. In baseline 1, we considered the method proposed by Riloff et al. (2013). Baseline 2 followed the Bouazizi and Ohtsuki's (2016) method, while baseline 3 followed the proposed method by Kumar et al. (2019). Due to the lack of comprehensive public datasets for evaluating the significance of the proposed approach, the three baseline approaches were experimented on the dataset utilized in this study. In this experimental setting, methods used in the baseline mentioned above were implemented on a processed sarcasm dataset and represented accordingly

The results of the comparison of the baseline approaches with our method are shown in Table 4. The performance measure of accuracy, recall, precision, and F-measure of different methods were presented. As observed from the table, baseline 1 attained an accuracy of 59.4%, baseline 2 attained an accuracy of 83.1%, and baseline 3 attained an accuracy of 84.4%. The last row of the table shows the performance of our proposed approach with the ensemble (bagging) classifier, an accuracy of 86.9%, and 86.9% F-measure with the ensemble (bagging) classifier. Thus, our proposed approach outperformed baseline 1 by 34.7% and 44% F-measure, baseline 2 by 11% accuracy and 12.8% F-measure, and baseline 3 by 8.7% accuracy and 8.6 % F-measure during the cross-validation. In addition, our method also shows a relatively higher precision when compared to the baselines.

Table 4: Performance of the proposed approach compared with the baseline methods

Methods	ACCURACY	PRECISION	RECALL	F-MEASURE
Baseline 1 Riloff et al. (2013)	59.40	65.00	40.80	50.1
Baseline 2 Bouazizi and Ohtsuki (2016)	83.10	91.10	73.40	81.30
Baseline 3 Kumar et al. (2019)	85.40	85.20	91.10	85.50
Our Proposed Approach	**86.93**	**86.90**	**86.90**	**86.90**

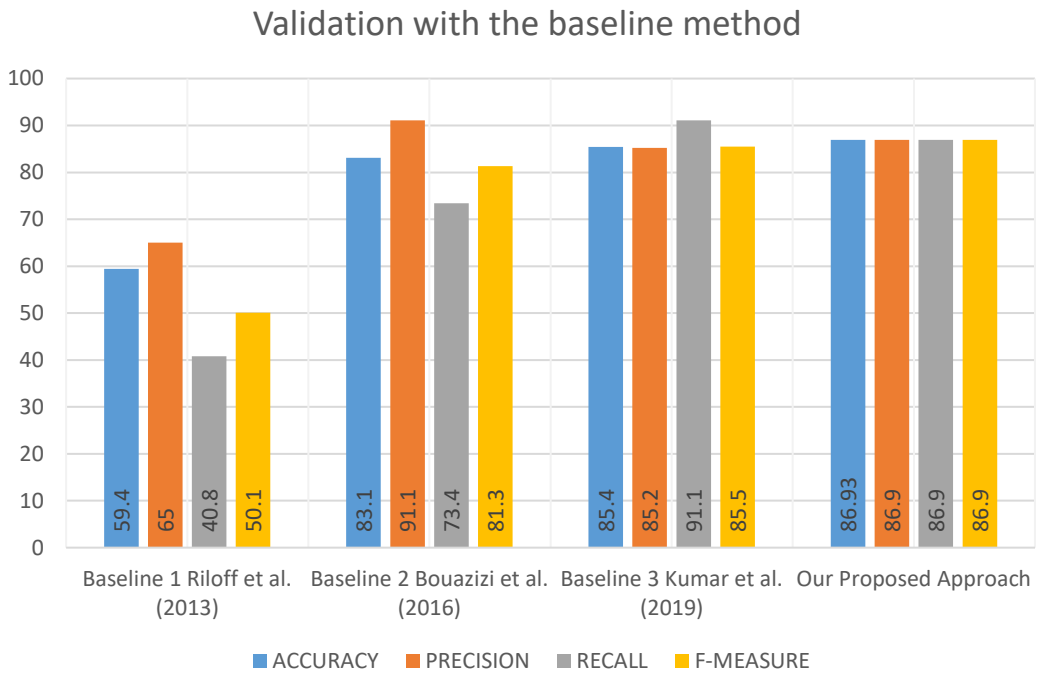

Fig. 3. Performance evaluation of the proposed method compared to the baseline

6. CONCLUSION

The advancement in information and communication technology has brought a remarkable evolution in microblogging and social media platforms. Microblogging platform helps in identifying the subjective message of the people such as opinion, sentiment, and behavior. Sarcasm identification has been a crucial challenge in natural language processing. In this study, we have considered an effective method for the identification of sarcasm in Twitter data using the collection of multiple features. The proposed method extracted various feature sets such as sentiment-based, punctuation-related, syntactic and semantic, and GloVe embedding features by taking into consideration the different forms of sarcasm and different components of tweets. The machine-learning algorithm was employed for classification by experimenting on a different subset of features to find the predictive performance of the models. However, the performance of seven predictive models such as Naïve Bayes, support vector machine, random forest, decision tree, logistic regression, bagging, k-nearest neighbour, and Naïve Bayes has been examined in the classification phase. The experimental result obtained a predictive performance of 86.9% accuracy by the fusion of five subset features using an RF ensemble classifier. Thus, the improved result accuracy shows the importance of multi-feature fusion and ensemble learning in sarcasm analysis. The proposed method can be employed to enhance sentiment analysis and opinion mining as a result of its ability to recognize sarcastic utterances in textual data. In our future study, we will conduct a comparative analysis of different word embedding schemes for sarcasm analysis, since the importance of GloVe embedding features has been observed in the current work to improve performance. A transfer learning technique based on BERT (Bidirectional Encoder Representation from Transformers) is another open research direction for sarcasm identification as it has recorded promising results in many NLP tasks. BERT is the first deep bidirectional and unsupervised language

model, which uses only plain text data to pre-train the model. Unlike the existing models constrained on unidirectional by employing a mask language model that randomly masks some tokens from the input, BERT removes such barriers and allows training on deep bidirectional transformers. In addition, it pre-trains text pair representation by employing the next sentence prediction (NSP) task. The configuration of BERT consists of two innovative prediction tasks such as Next Sentence Prediction and Masked LM. Studies have revealed that the pre-trained BERT model produces a better performance when compared with ELMO and OpenAI GPT in the sequence of the downstream task in NLP (Devlin et al., 2018). Thus, transfer learning that captures more discriminative features that can enhance the sarcasm classification performance is highly required.

REFERENCES

Abulaish, M., & Kamal, A. (2018). *Self-Deprecating Sarcasm Detection: An Amalgamation of Rule-Based and Machine Learning Approach*. Paper presented at the 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI).

Aggarwal, C. C., & Zhai, C. (2012). A survey of text classification algorithms. In *Mining text data* (pp. 163-222): Springer.

Babanejad, N., Davoudi, H., An, A., & Papagelis, M. (2020). *Affective and Contextual Embedding for Sarcasm Detection.* Paper presented at the Proceedings of the 28th International Conference on Computational Linguistics.

Baruah, A., Das, K., Barbhuiya, F., & Dey, K. (2020). *Context-aware sarcasm detection using bert.* Paper presented at the Proceedings of the Second Workshop on Figurative Language Processing.

Bedi, M., Kumar, S., Akhtar, M. S., & Chakraborty, T. J. I. T. o. A. C. (2021). Multi-modal sarcasm detection and humor classification in code-mixed conversations.

Berry, M. W., & Castellanos, M. (2004). Survey of text mining. *Computing Reviews, 45*(9), 548.

Bharti, S., Vachha, B., Pradhan, R., Babu, K. S., & Jena, S. (2016). Sarcastic sentiment detection in tweets streamed in real time: a big data approach. *Digital Communications and Networks, 2*(3), 108-121.

Bouazizi, M., & Ohtsuki, T. O. (2016). A pattern-based approach for sarcasm detection on twitter. *IEEE Access, 4*, 5477-5488.

Breiman, L. (1996). Bagging predictors. *Machine learning, 24*(2), 123-140.

Castro, S., Hazarika, D., Pérez-Rosas, V., Zimmermann, R., Mihalcea, R., & Poria, S. (2019). Towards Multimodal Sarcasm Detection (An _Obviously_ Perfect Paper). *arXiv preprint arXiv:1906.01815*.

Chen, J., Liu, Y., & Zou, M. (2016). Home location profiling for users in social media. *Information & Management, 53*(1), 135-143.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning, 20*(3), 273-297.

Davidson, I., Gourru, A., Velcin, J., & Wu, Y. (2020). Behavioral differences: insights, explanations and comparisons of French and US Twitter usage during elections. *Social Network Analysis and Mining, 10*(1), 6.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). *Imagenet: A large-scale hierarchical image database.* Paper presented at the 2009 IEEE conference on computer vision and pattern recognition.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Du, Y., Li, T., Pathan, M. S., Teklehaimanot, H. K., & Yang, Z. J. C. C. (2022). An effective sarcasm detection approach based on sentimental context and individual expression habits. *14*(1), 78-90.

Eke, C. I., Norman, A., Shuib, L., Fatokun, F. B., & Omame, I. (2020). *The Significance of Global Vectors Representation in Sarcasm Analysis.* Paper presented at the 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS).

Eke, C. I., Norman, A. A., Shuib, L., & Nweke, H. F. (2019). Sarcasm identification in textual data: systematic review, research challenges and open directions. *Artificial Intelligence Review*, 1-44.

Eke, C. I., Norman, A. A., Shuib, L., & Nweke, H. F. (2019). A Survey of User Profiling: State-of-the-Art, Challenges, and Solutions. *IEEE Access, 7*, 144907-144924. doi:10.1109/ACCESS.2019.2944243

Eke, C. I., Norman, A. A., & Shuib, L. J. I. A. (2021a). Context-Based Feature Technique for Sarcasm Identification in Benchmark Datasets Using Deep Learning and BERT Model. *9*, 48501-48518.

Eke, C. I., Norman, A. A., & Shuib, L. J. P. o. (2021b). Multi-feature fusion framework for sarcasm identification on twitter data: A machine learning based approach. *16*(6), e0252918.

González-Ibánez, R., Muresan, S., & Wacholder, N. (2011). *Identifying sarcasm in Twitter: a closer look.* Paper presented at the Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers-Volume 2.

Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*: Elsevier.

Hazarika, D., Poria, S., Gorantla, S., Cambria, E., Zimmermann, R., & Mihalcea, R. (2018). Cascade: Contextual sarcasm detection in online discussion forums. *arXiv preprint arXiv:1805.06413*.

Ilić, S., Marrese-Taylor, E., Balazs, J. A., & Matsuo, Y. J. a. p. a. (2018). Deep contextualized word representations for detecting sarcasm and irony.

Java, A., Song, X., Finin, T., & Tseng, B. (2007). *Why we twitter: understanding microblogging usage and communities.* Paper presented at the Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis.

Justo, R., Corcoran, T., Lukin, S. M., Walker, M., & Torres, M. I. (2014). Extracting relevant knowledge for the detection of sarcasm and nastiness in the social web. *Knowledge-Based Systems, 69*, 124-133.

Kantardzic, M. (2011). *Data mining: concepts, models, methods, and algorithms*: John Wiley & Sons.

Kumar, A., Sangwan, S. R., Arora, A., Nayyar, A., & Abdel-Basset, M. (2019). Sarcasm Detection Using Soft Attention-Based Bidirectional Long Short-Term Memory Model With Convolution Network. *IEEE Access, 7*, 23319-23328.

Liu, L., Priestley, J. L., Zhou, Y., Ray, H. E., & Han, M. (2019). *A2text-net: A novel deep neural network for sarcasm detection.* Paper presented at the 2019 IEEE First International Conference on Cognitive Machine Intelligence (CogMI).

Manjusha, P., & Raseek, C. (2018). *Convolutional Neural Network Based Simile Classification System.* Paper presented at the 2018 International Conference on Emerging Trends and Innovations In Engineering And Technological Research (ICETIETR).

Mehndiratta, P., Sachdeva, S., & Soni, D. (2017). Detection of Sarcasm in Text Data using Deep Convolutional Neural Networks. *Scalable Computing: Practice and Experience, 18*(3), 219-228.

Mukherjee, S., & Bala, P. K. (2017). Sarcasm detection in microblogs using Naïve Bayes and fuzzy clustering. *Technology in Society, 48*, 19-27. doi:10.1016/j.techsoc.2016.10.003

Muresan, S., Gonzalez-Ibanez, R., Ghosh, D., & Wacholder, N. (2016). Identification of nonliteral language in social media: A case study on sarcasm. *Journal of the Association for Information Science and Technology, 67*(11), 2725-2737.

Nayel, H., Amer, E., Allam, A., & Abdallah, H. (2021). *Machine learning-based model for sentiment and sarcasm detection.* Paper presented at the Proceedings of the Sixth Arabic Natural Language Processing Workshop.

Nweke, H. F., Teh, Y. W., Al-Garadi, M. A., & Alo, U. R. (2018). Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges. *Expert Systems with Applications, 105*, 233-261.

Onan, A. (2017a). *A machine learning based approach to identify geo-location of Twitter users.* Paper presented at the Proceedings of the Second International Conference on Internet of Things, Data and Cloud Computing.

Onan, A. (2017b). *Sarcasm identification on twitter: a machine learning approach.* Paper presented at the Computer Science On-line Conference.

Pennington, J., Socher, R., & Manning, C. (2014). *Glove: Global vectors for word representation.* Paper presented at the Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP).

Peters, M. E., Neumann, M., Zettlemoyer, L., & Yih, W.-t. J. a. p. a. (2018). Dissecting contextual word embeddings: Architecture and representation.

Ptáček, T., Habernal, I., & Hong, J. (2014). *Sarcasm detection on czech and english twitter.* Paper presented at the Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers.

Quinlan, J. R. (1987). Simplifying decision trees. *International journal of man-machine studies, 27*(3), 221-234.

Razali, M. S., Halin, A. A., Ye, L., Doraisamy, S., & Norowi, N. M. J. I. A. (2021). Sarcasm detection using deep learning with contextual features. *9*, 68609-68618.

Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N., & Huang, R. (2013). *Sarcasm as contrast between a positive sentiment and negative situation.* Paper presented at the Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing.

Samonte, M. J. C., Dollete, C. J. T., Capanas, P. M. M., Flores, M. L. C., & Soriano, C. B. (2018). *Sentence-Level Sarcasm Detection in English and Filipino Tweets*. Paper presented at the Proceedings of the 4th International Conference on Industrial and Business Engineering - ICIBE' 18. http://delivery.acm.org/10.1145/3290000/3288172/p181-Samonte.pdf?ip=103.18.0.19&id=3288172&acc=ACTIVE%20SERVICE&key=69AF3716A20387ED%2EE7759EC8BE158239%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&__acm__=1562041412_216ad611ed7438dea30eb1738af6b7df

Savini, E., & Caragea, C. J. M. (2022). Intermediate-task transfer learning with BERT for sarcasm detection. *10*(5), 844.

Schifanella, R., de Juan, P., Tetreault, J., & Cao, L. (2016). *Detecting sarcasm in multimodal social platforms.* Paper presented at the Proceedings of the 2016 ACM on Multimedia Conference.

Sreelakshmi, K., & Rafeeque, P. (2018). *An Effective Approach for Detection of Sarcasm in Tweets.* Paper presented at the 2018 International CET Conference on Control, Communication, and Computing (IC4).

Suhaimin, M. S. M., Hijazi, M. H. A., Alfred, R., & Coenen, F. (2018). Mechanism for Sarcasm Detection and Classification in Malay Social Media. *Advanced Science Letters, 24*(2), 1388-1392.

Suhaimin, M. S. M., Hijazi, M. H. A., Alfred, R., & Coenen, F. (2019). Modified framework for sarcasm detection and classification in sentiment analysis. *Indonesian Journal of Electrical Engineering and Computer Science, 13*(3), 1175-1183.

Thelwall, M., Buckley, K., & Paltoglou, G. (2012). Sentiment strength detection for the social web. *Journal of the American Society for information Science and Technology, 63*(1), 163-173.

Yavanoglu, U., Ibisoglu, T. Y., & Wıcana, S. G. (2018). Technical Review: Sarcasm Detection Algorithms. *International Journal of Semantic Computing, 12*(03), 457-478.

Yee Liau, B., & Pei Tan, P. (2014). Gaining customer knowledge in low cost airlines through text mining. *Industrial Management & Data Systems, 114*(9), 1344-1359.

Appendix 1: Sample tweet and preparation for feature extraction and modeling

For example, in a given sample tweet below.

"@realObama Mr Obama love being cheated on! #sarcasm https://t.co/G5N24J5nMX"

The tweet sample is initially divided into a stream of tokens, as shown below. Thus, for the sake of uniformity, the tokens are changed to lower case.

['@realObama', 'mr', 'obama', 'love', 'being', 'cheated', 'on!', '#sarcasm', 'https://t.co/G5N24J5nMX']

Then, using the placeholders as shown below, URLs, user mentions, and numbers in a given tweet sample are replaced.

['AT_USER', 'obama', 'love', 'being', 'cheated', 'on!', 'URL']

Finally yet importantly, words are changed into their root form, active tense, singular, and present tense using text normalization techniques like stemming and lemmatization. This method makes it simple to parse the data and effectively extract its features.

['AT_USER', 'obama', 'love', 'being', 'cheat', 'on!', 'URL']

The tokenized word is employed to the machine learning techniques such as weka for feature extraction and modeling.