PROJECT REPORT on

Sarcasm Detection in Online Social Media Network Texts

THE REPORT SUBMITTED IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

M.Sc. COMPUTER SCIENCE

(2018-2020)

by

ATUL

Roll no- CUSB1802312004

Under Guidance of

Dr. PRABHAT RANJAN

ASSISTANT PROFESSOR & HEAD INCHARGE

DEPARTMENT OF COMPUTER SCIENCE

CENTRAL UNIVERSITY OF SOUTH BIHAR

DEPARTMENT OF COMPUTER SCIENCE

CENTRAL UNIVERSITY OF SOUTH BIHAR

SH-7 Gaya Panchanpur Road

Village- Karhara, Post-Fatehpur

GAYA, BIHAR – 824236

CENTRAL UNIVERSITY OF SOUTH BIHAR

PANCHANPUR, GAYA, BIHAR -824236

DEPARTMENT OF COMPUTER SCIENCE

Certificate

This is to certify that the project work entitled "**Sarcasm Detection in Online Social Media Network Texts"** is submitted by **ATUL** to the Department of Computer Science, Central University of South Bihar, Gaya in partial fulfillment for the award of degree of **Master of Science in Computer Science**.

I certify further that, to the best of my knowledge, the project work reported herein is not a part of any other project or software on the basis of which a degree or an award was given on an earlier occasion to any other candidate.

Guide and Supervisor

Dr PRABHAT RANJAN

B.E, M.Tech & PhD (Computer Science MNNIT, Allahabad)

Assistant Professor & Head Incharge

Department of Computer Science

Central University of South Bihar, Gaya

CENTRAL UNIVERSITY OF SOUTH BIHAR

PANCHANPUR, GAYA, BIHAR – 824236

DEPARTMENT OF COMPUTER SCIENCE

Certificate

This is to certify that the project work entitled "**Sarcasm Detection in Online Social Media Network Texts"** is submitted by **ATUL** to the Department of Computer Science, Central University of South Bihar, Gaya in partial fulfillment for the award of degree of **Master of Science in Computer Science**.

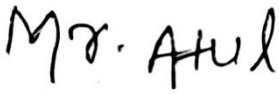
I certify further that, to the best of my knowledge, the project work reported herein is not a part of any other project or software on the basis of which a degree or an award was given on an earlier occasion to any other candidate.

Head of Department **Internal Examiners**

DECLARATION

I, **ATUL** a student of Department of Computer Science, Central University South Bihar, hereby declare that this project work entitled **"Sarcasm Detection in Online Social Media Network Texts"** being submitted to the department of Computer Science for the partial fulfillment for reward of degree of **Master of Science in Computer Science**.

This is a record of original and bonafide work done by me and it has not been submitted to any other Institute or University for the award of any other degree.



ATUL

MSc CS (2018-20)

CUSB1802312004

Acknowledgement

I would like to express my deep gratitude to my Guide and Head of Department and Assistant Professor Dr. Prabhat Ranjan for his patient guidance, enthusiastic encouragement and useful critiques of this research work. I would also like to thank all other professors of the department, for their advice and assistance in selecting a good and worthy topic to keep up with the current research works and keeping my progress on schedule.

I would also like to extend my thanks to the technicians of the laboratory of the Computer Science department for their help in offering me the resources in running the programs.

Finally, I wish to thank my parents, friends and family for their support and encouragement throughout my study.

ATUL

Table of Contents

List of Figures

Abstract

In the time of huge online data being put over the internet, specifically on online social media networks in the form of Post, Tweets, Comment, etc., almost everyone in the world right own at least a basic smartphone and usage is majorly for surfing on the Online Social media networks. As people post a lot of stuffs including about their personal lives, their daily routines, their wishes, their moods etc, along with these people are also interested in sharing a post which actually contradicts the literal meaning of the statement in the post. That statement be is called Sarcasm. Sarcasm is defined in many ways and the simple meaning of Sarcasm is that A Statement whose original meaning interpreted with human mind like intelligence contradicts with the literal meaning of that sentence, means opposite of what is being said is called sarcasm. A lot of number of researches in the field of Sarcasm Detection has been done and still being done which is improving the architecture to detect sarcasm day by day.
Even US Secret Services have shown their interest in getting some algorithms to detect sarcasm such that they can benefit by learning about the sentiment of people. Sarcasm is mostly used in the real life in such a way and manner that the person listening to it will understand the sarcasm behind by analyzing the tone and gesture of the speaker. But when it comes to detect sarcasm automatically on a machine is a tough job Because Machines lack the intelligence of human mind. That leads to training machine various patterns of sarcasm such that machine can extract patterns and match them with various new texts to declare them sarcasm or non sarcasm.

In this Project, We have firstly read various research papers and included summaries of some considerable works in this field. Also we tried to work on practical implementation of steps of Sarcasm detection which are used most commonly. We have used R software (platform and environment) along with various libraries to work out desired practical. While we were studying papers we came to an idea of introducing a new form of sarcasm. This form of sarcasm changes its behavior with time and it is almost hard for machines to understand sarcasm behind it. We have also proposed an architecture which can be followed to detect sarcasm in those types of texts. We have named this sarcasm as Factual Sarcasm or Dynamic Sarcasm.

List of Abbreviations

SVM: Support Vector Machine

NLP: Natural Language processing

SA: Sentiment Analysis

SD: Sarcasm Detection

SDA: Sarcasm Detection Algorithm

FB: Facebook

N_S: Level of Sarcasm

NS: Non Sarcasm

OSMN: Online Social Media Network

SM: Social Media

API: Application Programming Interface

FPM: Followers Per Month

1 Introduction

1.1 Motivation

Sarcasm is form of speech which is said in a way such that the statement looks like a simple statement at first but the contextual meaning of the speech tends to contradict what is being said.
There are various definitions for Sarcasm for example:

Google define Sarcasm as the use of irony to mock or convey contempt.

Wikipedia defines as **Sarcasm** is "a sharp, bitter, or cutting expression or remark; a bitter gibe or taunt"

Cambridge defines Sarcasm as the use of remarks that clearly mean the opposite of what they say, made in order to hurt someone's feelings or to criticize something in a humorous way.

Oxford dictionary defines Sarcasm as a way of using words that are the opposite of what you mean in order to be unpleasant to somebody or to make fun of them.

People use Sarcasm all the time in real life to express their view against some event, organization or person. Sarcasm is generally used to make a remark. Since the evolution of Social media, People have started putting their statements on Social media in the Sarcasm form. Sarcasm on the Social Media can be easily understood as people used to put similies or emoticons or specials characters which make similies or the distorted character like upside down question mark or hashtags like #sarcastic, #sarcasm or #irony, etc. A human can understand sarcasm easily by looking at the statement and grasping the context of the statement but a Machine faces various difficulties in identifying Sarcasm in the statements posted online. A machine has to find some common patterns in the statements to classify new statements to be sarcastic or non sarcastic.

Social Media Network has now a day become a most impactful place that people instead of visiting any organization physically, post sarcastically against the product and get their solution done within minutes because of the fear of organizational reputation. People use Social Media Networks to review about products, Movies, persons, events even political agendas. Two famous Online Social Media Networks these days are Twitter and Facebook. Posts on these two platforms are very impactful on the known and unknown audience of the post. People tend to use sarcasm when they want the victim to be target indirectly and hence recognizing a victim from the sarcastic comment is also a major concern while doing a work on the Sarcasm detection. People on SM doesn’t only make sarcastic comments but also they share about their own personal lives and tend to scroll to know more about other persons either in their friendslist or not.

An Organization can benefit from the Sarcasm Detection techniques as they can come to know about the negative aspects of their products and make further adjustment to mend the gap to satisfy public needs. Also in the same manner, Celebrities and public figures can do Sentiment Analysis of the public to know their views regarding them in a simpler manner by just collecting relevant post and comments about themselves. A country leadership can also benefit from Sentiment analysis by retrieving country related post and comments and analyzing if people have positive, neutral or negative view on the events or programs carried out by the leadership. A movie can easily estimate their earning after the movie launch and the reviews they get on the very first day of the movie. Cause People impact other on the SM easily and without any limit of the viewership.

There are two different terms when it come to analyze some statement, Sentiment Analysis and Sarcasm detection, Both of them seem to almost same but the major difference lies that Sentiment analysis outputs the result as the statement is Positive, Negative of Neutral in nature while Sarcasm detection detects if there any sarcasm exists in the given statement.

While Sarcasm is made, there is always some point which should be always keep in mind that, Sarcasm is always made against something(event) or someone(person) and Sarcasm is always made due to some event that has occurred and impacted the person making comment. When sarcasm is made against some person that sarcasm can be considered as a Shaming comment and that could intend to publicly shame the victim. There are high chance that the comment which seems sarcastic to the third person is not originally sarcastic between the person making comment and person who is being discussed in the comment, because of the relationship between them.

These days Sarcasm doesn't only expressed a single person but a single post may be so impactful that SM users actually share that same post with or without modifications that post is said to be viral.

A considerable amount of work has been already done to detect sarcasm in the statements by the use of NLP, Machine Learning Deep Learning etc. even though the huge amount of works which have been carried out are still insufficient to guarantee Sarcasm detection by 100% because People tend to write sarcasm in a varying ways depending on the trends.

In this paper we have tried to implement the exiting Sarcasm detection model and along with it, we even craved for some more deep knowledge and came to a know about a new type of Sarcastic post which tends to be a simple statement having some sentiment at some point of time but if the same post is made after the event has already happened then the statement turns sarcastic. This type of sarcasm can be called Dynamic Sarcasm or Factual sarcasm as it changes its behavior with time and contains contradicted facts.

We have read several research papers in the field of sarcasm detection and we are putting the summaries of those papers in this report. The papers we have gone through gave a deep knowledge about how the sarcasm detection technique is being used in the current times and what could be the future works on these to improve the technique. We have also come up with a paper which discussed not only the sarcasm detection but also from the point of view of the victim what actions the victim can take on the shamer's comment, i.e, Delete, mute shamer, block sharmer or none. We can suggest a more action which could be taken into consideration is the "report comment" action, such that twitter can review those comments and warn the shamers about suspension of the account if they don't stop shaming the victim.

1.2 Objective

As we have already learnt that detecting sarcasm on the OSMN can be of great help hence we had agreed for following objectives.

1. Our objective is to read about the currently existing techniques in the field of Sarcasm Detection of Online Social Media Networks
2. To find out the scope for further development or improvement of the technique.
3. Also if we can find some novel criterion for statements to be sarcasm then propose a model to detect that sarcasm.
4. Our objectives also include carrying out practical works on the currently existing methodologies for the Sarcasm detection on SM.

1.3 Scope of Work

1.3.1 Problem Statement

Sarcasm on the social media create much confusion in understanding the context behind the comment, which makes it harder for individual or organization to take proper care of their product and public appearances and attitudes. Hence a technique to detect sarcasm automatically on the Social media platform will be of great significance in understanding the reality behind the statement.

1.3.2 Goals

To gather information about current studies on the Sarcasm detection topic.

To implement currently existing techniques.

To improve currently available technique if possible

To derive new technique if existing techniques are not good in detecting some sarcasm.

1.3.3 Objective

Task: Read research papers on the topic of sarcasm detection.

Deliverable: Summaries of read papers. Novel method to detect sarcasm if possible

1.3.4 Administration

Have to report the status of project to the guide at several intervals

1.3.5 Timeline

January 2020 to April 2020- Reading existing research papers on the topic of Sarcasm Detection

May 2020 to June 2020- Practical workout of exiting methods

July 2020 to August 2020- Novel method description and working example for finding sarcasm in Dynamic Sarcasm/Factual Sarcasm

Our work require knowledge of Machine Learning, Deep Learning, Online Social Media Network, Sentiment Analysis, Neural Network and various libraries to be used for series of works in techniques.

2 Literature Review

We have read and reviewed 23 papers out of papers which were available for the topic of Automated Sarcasm Detection. In paper [1] authors Rajesh Basak et al. tried to automate the task of Public shaming detection in twitter from the perspective of victims and explore two aspects: Events and Shamer. They have defined 6 types of shaming tweets namely: Abusive, Comparison, Passing Judgement, Religious/Ethnic, Sarcasm/Joke, Whataboutery. They also observed in their research that People tend to shame the victim using sarcasm also follower count of shamers increases fast than normal. They have developed a web application BLOCKSHAME, which provides users a facility to detect shamers and take any automated actions like Muting or Block that account based on the user's choice.Automated classification of shaming tweets follows Preprocessing, Feature Extraction and Classification in sequential manner. Preprocessing is done as usual but first the Stanford Core NLP Library is used for recognizing Name Entity, Coreference resolution and dependency parsing. Preprocessing removes user mentions, retweets, markers, hashtags, and URLs. POS tagging is done using Stanford Core NLP library. Feature extraction step is the most crucial step. A feature vector is generated for each tweet. Feature vector consists of total 849 features (800 unigrams and 49 other features). Various methods are used for recognizing features. Then finally the classification is done on the basis of Support Vector Machine where 6 SVMs are arranged in hierarchical order, means the highest precision SVM is on top. The classification is considered for the first positive SVM for the tweet. If all six SVMs fail then the tweet is Non Shaming.
They proposed an automated software named BlockShame which can take actions based on user choice and category of shaming tweets. Paper also provides statistics for the shamer accounts like increase in their popularity in terms of followers per month calculated by current followers count and dividing it by the number of months the account had been created since. Authors have also discussed about some future works which could be carried out to improve the Sarcasm detection and mitigation mechanism. While the paper already analyzed a large number of user accounts for FollowersPerMonth analysis still there is a huge scope for checking the criteria for more vast number of features. Authors also talk about the dynamics of shaming Events, which means that how likely the specific shaming event will be at the peak and how much expected time is required to get off that shaming event. We can track accounts and keep records of their followers count for every shamer and non shamer account for a long time to be more confident if the shamer accounts actually gets higher FPM than that of Non shamers. Shaming is subjective which means that the same comment from two individual can be either shaming or non-shaming based on the relationship between the commenter and victim. A large number of annotated data can be used for more precise classification. After the initial outrage, the Volume of Apologetic and reconciliatory comment gradually increases. A considerable proportion of user makes multiple comments in view to the same event which can be categorized as Shaming or Non Shaming.

Bala Durga and Jayanag Bayana [2] start with defining Sarcasm as the "Sarcasm means the person speaks the contradictory of what the individual means, expressing gloomy feeling applying happy positive words". Sarcasm used in Social networking, micro blogging, marketing. Existing systems use Logistic Regression, Which cannot predict for continuous variable. Proposed system uses Naïve Bayes Classification, Sentiment analysis and Ada Boost Algorithm. Authors used Naïve Bayes Classification for Categorizing Tweets, AdaBoost Algorithm for converting Weak statements to Strong statements and Sentiment Analysis to mine the opinion of consumers. Sarcasm is more prevalent in the places where there are Capital letters, Emoticons and Exclamation marks etc. Proposed methodology consists of three modules: Data Preprocessing: Consists of Noise removal, Tokenization, Stemming and POS Tagging, Data Modeling: Consists Feature Selection and Classification: using Sentiment Analysis, Naïve Bayes Classifier and AdaBoost Algorithm. One additional Module namely Sentiment Analysis is used to recognize if the Sarcasm is Positive or Negative. Naïve Bayes classification considers probabilities. Various topics were chosen for classification of tweets as sarcastic or non sarcastic. Accuracy of model is calculated by gathering training accuracy and validation accuracy.

Authors further classify a sarcastic tweet as positive or negative such that the marketers can get a quick idea about their product. Marking sarcastic comment as positive or negative can also help in more real life situations like when someone is being trolled in the first instant but that could be validated as a positive sarcasm that means the trolling statement would be indirectly supporting the victim.

Ravinder Ahuja et al. [3] applied 12 different classification algorithms on 4 datasets and varied the split ratio of the datasets to check the accuracy of every algorithm in different situations. Hashtags has been used to construct a marked set of naturally transpiring negative, sarcastic and positive tweets which will be used as Corpus.
Sarcasms can be in 5 forms in expansions: Sarcasm as a constraint of sentiments, Sarcasm as a complex form of expression, Sarcasm as a means of conveying emotions, Sarcasm as a function of duality and Sarcasm as a written form of expressions. Paper considers i, ii and iv forms of sarcasms only. 12 different classifications algorithms used are: Gaussian Naïve Bayes, Extra Trees, Decision Tree, Random Forest, AdaBoost, L1 Regularized Logistic regression, L2 Regularized Logistic regression, Gradient Boosting, Bagging Model, svm(riof kernel), K-Neighbor hybrid and K-Neighbor First three feature sets are extracted from dataset (i)Using n grams (bigrams and trigrams) along with preprocessing (ii)Using wordlength distribution and syllables(iii)Using POS tagging. They have fetched tweets using hashtags for showcase: Direct Positive Sentiment (#Happy, #Cheer), Sarcasm (#sarcasm, #sarcastic), Direct Negative Sentiment , #Sorrow, #angry). Initial dataset contains Tweet ID (Unique) and their respective classification tag. Another dataset is used from a web tags “positive”, “negative” and “neutral”. Authors divide features into four feature sets: Sarcasm as a contrast of sentiment, Sarcasm as a complex form of expression, Sarcasm as a written form expression, All of the them combined. All 12 algorithms are applied to these feature sets. Performance measures used are Accuracy, percent error (for accurate measurement calculated as % error= (accepted-=experimental)*100%/accepted). Accuracy given by classifiers on four data sets are: Set1: SVM 54%, Gradient Descent Ensemble 79%--- Moderate Predictor, Set2: Gradient Boosting 65.95%--- Poor Predictor, Set3: Highest with 85%, Gaussian Naïve Bayes 75%--- Best predictor, Set4: Gradient Boosting 85.71%, Train split 75:25, K- Neighbor and SVM 55%. Authors did a comparative study and reached to a conclusion that even partial implementation of SCUBA Framework results in a better prediction of 12 all considered classification methods, Gradient Boost gives far better result than KNN or SVM, which means that ensemble methods throw better results than multilayered perceptron. Though this paper has told what they used but they have no mention of how the feature sets are created and what are the basic differences between sets with examples. Hence it is difficult to understand the internal mechanism of feature set.

Dr. Kalpesh Wandra and Mehul Barot [4] state Text analysis involves information retrieval, lexical analysis, pattern recognition, tagging/annotation, information extraction, data mining techniques, visualization and predictive analytics. Natural Language Processing is used to turn text into data. Paper defines sarcasm as “a form of ironic speech commonly used to convey implicit criticism with a particular victim as its target”. Mostly used technique is Machine Learning for sarcasm detection. Proposed architecture of this paper is as follow: Collecting relevant data using machine learning, Preprocessing, Feature selection, Classification (SVM, Logistic Regression), Once the classifiers have learnt, test data are fed into machine, Performance calculated. Authors explain SVM and Logistic Regression in details. Tweets are also manually marked as Natural or Sarcastic to train classifiers better. They have calculated accuracy and precision for its classifiers and compared the same with previously existing architecture/techniques. The resultant is that the accuracy of proposed architecture has shown improvement slight better than existing. This model uses N-gram features and vocabulary of 7000 words. Proposed: Accuracy 68.75% (SVM), 66.667% (Logistic Regression) and Precision 86.67 (SVM), 98.857% (Logistic Regression) vs Existing: Accuracy 65.4% and Precision 60.7%.

In [5] by Sana Parveen and Sachin Deshmukh, they say that “Sarcasm is the use of irony to mock or convey contempt”. Context of the sentence must be considered. The methodology proposed by them creates two data sets i.e. before adding sarcastic tweets and after adding sarcastic tweets to the training set. Each topic (total 6 topics) of training set has

1200 tweets. Classification methods used are Naïve Bayes, Maximum Entropy and Support Vector Machine. Preprocessing is done to remove non ASCII characters, blank lines, spaces, punctuations, special characters and remarks. Lemmetization is also performed in preprocessing. POS tagging is done after preprocessing using Penn Tree bank. Training of classifiers is done with 4 features extracted from training data: Sentiment related, Punctuation related, Syntactic related, Pattern related. When test was done on 7200 training data (6*1200) the sarcasm wasn't detected properly hence authors added 2800 (sarcastic) more data and now training and test resulted in better performance. Accuracy of classification: Naïve Bayes- 67.06%, SVM- 75.33%, Maximum entropy 82.13%.

Komalpreet Kaur and assistant professor Ankita Gupta [6] used different twitter tags such as different punctuations, words, patterns in the text. Authors insist that sentiment analysis task consists of two major steps: Looking for different expressions, Determine polarity (negative, positive and neutral. They distinguished tweets as sarcastic and non sarcastic to find polarity of a sentence. Hashtags can play a great role in identifying tweets as sarcastic or non sarcastic. But, tweets also contain noises in the form of non ASCII characters hence these noises are removed and moved further for feature engineering. N-grams, Sentiments and pattern extraction are used for feature extraction. N-grams can be unigrams or bigrams. Then it goes for Lemmetization, Stemming and uncapitalization. Sentiments are extracted using SentiWord Net dictionary which gives positive or negative sentiment values to every single word. Also Text Blob can be used in Python for doing the same. Pattern extraction: Definition provided by Davidov and Rappoport is used. A pattern is said to be directed sequence of high frequency words and same slot for content words. Classifications are done using SVM with SMO and Logistic Regression. Evaluation of the algorithm defined in this paper is done by cross Validation. Authors used F-Score for assessment. Paper finally observed that sarcastic tweets are more about expressing emotions and feeling either positive or negative than Non sarcastic tweets.

Moundher Bouazizi and Tomoaki Ohtsuki [7] said in their paper that detection of polarity of a statement is the major course in classification in Data Mining (opinion mining) or Sentiment analysis. But sarcastic statements make it somewhat difficult to identify them due to jumbled words. Authors have gone through various papers and deem to create a proposed way for sarcasm detection. For collection of data authors ranged its collection from December 2014 to March 2015 using Twitter streaming API. They used query of #sarcasm for sarcastic tweets and for non sarcastic tweets, they pulled data from various topics excluding sarcastic tweets. Authors prepared 3 data sets: Set1: [6000 tweets (3000 sarcastic, 3000 non sarcastic), Annotation is done manually. Level of sarcasm is marked from Level 1(highly non sarcastic) to Level 6(highly sarcastic). This is the training set. Sarcasm level is denoted by N_s], Set2: [1128 tweets with #sarcasm and 1128 non sarcastic tweets. No manual tagging is done. Noisy set, used for tuning (optimizing) training set.] and Set3: [500 sarcastic and 500 non sarcastic tweets. Manual annotation. This is the test set.]. They considered Sarcasm to have 3 categories: Sarcasm as wit: being funny, Sarcasm as a whimper: showing anger and Sarcasm as a avoidance: avoiding giving clear answer. Then four sets of features are extracted from tweets: Sentiment relation (Ratio)- Positive words and Negative words, Punctuation relation features (True/False): Punctuation and repetitive vowels, Lexical and semantic features: uncommon words, laughter, interjection and Pattern related features: POS tag for content word and grammatical. Authors extended its early training set from 6000 tweets to more 37918 tweets. They used Random forest for classification. The resulting accuracy, precision, recall and F-score are 83.1%, 91.1%, 73.4% and 81.3% respectively.

Lakshaya Kumar et al. divides past works on the basis of rule based, machine learning based and deep learning based approaches [8]. Rules based approaches attempts to identify sarcasm through specific evidences. These evidences are captured in terms of rules they rely on indicators of sarcasm. For Rule based approach: First paper reviewed is of Veale and Hao (2010) which identifies similes by using google search by 9 steps algorithm. Second paper is of Maynard and Greenwood(2014) which proposed that hashtag sentiment is a key indicator of sarcasm. Next paper is by Bharti et al (2015) which presented two rule based classifiers, first uses a parse based lexicon generation algorithm and second

algorithm aims to capture hyperboles. Then Last paper read is by Riloff et al.(2013) which looks for a positive verb and a negative situation phrase in a sentence (Bootstrapping is used). For Machine learning approach: Statistical approach also called Machine learning approach varies in terms of capturing different features and learning procedures. Tsur et al (2010) is discussed which designed pattern based features that indicates presence of discriminative patterns as extracted from a large sarcasm based corpus. Next paper is Liebrecht et al (2013) which tackled the problem by introducing bigrams and trigrams features. Buschmeier et al. (2014) incorporates ellipsis, hyperbole and imbalance in their set of features. Ptacek et al (2014) used words shape and pointedness features given in the form of 24 classes. Liu et al (2014) has introduced POS sequences and semantic imbalances as features. Also proposes a novel multi strategy ensemble learning approach MSELA to handle the imbalance feature. Authors discussed about a method where sarcasm detection can be done by using tweet's author's history. Also, wrote a detail information about which feature is used in feature sets in considered papers and which technology or dictionaries have been used. For Deep Learning Approach: In Deep learning approach section authors have written summery of following authors: Silvio Amir et al (2016) which presented a novel convolution network based that used embedding in addition to utterance based. Next and last paper discussed is of Zhan et al (2016) which investigates the use of neural network for the task and compared the effects of the continuous automatic features with discrete manual features. Hence authors have referenced various literatures of works done on the topic of sarcasm detection.

David Bamman and Noah. A Smith [9] start with by saying that the work in field of sarcasm detection on twitter has been done till date as a text categorization problem. Hence authors provide carious claims to prove that sarcasm is majorly a problem of context developed between the author and the audience. The notion of audience becomes complex when it is on social media, where the audience is often unknown, under specified or collapsed. They have done series of experiments to discern the effect of extra linguistic information on the detection of sarcasm, reasoning about features derived not only from the local context of message by also using information about author, their relationship to audience and immediate communicative context between them. They collected data (tweets) from August 2013 to July 2014 and considered most recent 3200 each tweets of authors who explicitly mention #sarcastic or #sarcasm. Collected 9767 positive training set data and equal negative data without mention hashtags(#sarcastic or #sarcasm). Binary logistic regression with L2 regularization using ten fold cross validation is used. Four classes of features is defined by them: Tweet feature: Scoped only over immediate tweet, Author feature: That reason over the author, Audience feature: That reason over addressee of tweet and Response feature: That consider the interaction between the tweet being predicted. Tweet features are word unigrams, brown cluster unigrams, unlabeled dependency bigrams, lexilalized, Part of Speech features, Pronunciation features, Capitalization, Tweet whole sentiment, tweet word sentiment, Intensifiers. Author features are author's historical salient terms, author's historical topics, profile information, author's historical sentiment, profile unigrams. Audience features are defined for author features (in percentage %), author/addressee interaction topics, historical communication between author and addressee.Environment features are pair wise brown features between the original message and response, unigram features of the original message.
Authors considered five features combinations to evaluate its models. Training the model on these five features alone yields an accuracy of 84.3% while all features together yields 85.1%.

The Greek referendum of 2015 and subsequent legislative elections are considered by Despoina Antonakaki et al. [10] for this report. Authors compiled novel dictionaries for sentiment and entity detection for the Greek language tailored to these events. Also authors performed volume analysis sentiment analysis and sarcasm correction. They faced difficulties like Online Greek language, Greekish, demographic subset of users. This papers aims to reveal hidden or hard to find content and relations for political domain exploric datasets from specific events. The dataset considered are with #dimopsifisma (Greek for referendum) and #greferendum from 25th June 2015 to 5th July 2015, which contains 301000 tweets out of which 84481 were neither retweets nor replies. The second dataset includes all tweets that contain the

#ekloges(election) and #ekloges_round2 of September 2015. This dataset contains 182000 tweets out of which 45750 are neither retweets nor replies. Authors did entity identification on the elections and referendum twitter corpus. Then sentiment analysis is done using Senti Strength. Senti Strength estimates the sentimental strength of short test. Authors used methods defined on http://www.thesarcasmdetector.com/ for sarcasm detection of the content. The classification result for the method adapted is average precision, recall and F1-scores are 70%, 71% and 70% respectively. They did an analysis on the tweets volume and find out that the final Yes or No ratio right before the referendum was 18% and close to preference of demographics of Greek users. Then entity co-occurrence is studied, co-occurrence means there exits at least one tweet that contain both entities and found that Europe related entities are close to Yes where as entities regarding domestic affairs including debt are closer to the No point. Next sarcasm, sentiment and hashtags are analyzed. Overall 61.8% of total referendum and 58.7% of total election tweets were sarcastic. The computation of sarcasm and sentiment levels allows plotting temporal variation of sentiment. Each tweet sentiment was corrected towards the neutral side proportionately to the percentage of sarcasm if contained. During the pre referendum period, positive sentiment towards Europe decreased and the negative sentiment towards Greek Prime Minister Alexis Tsipras increases and becomes almost stable after the enforcement of capital controls on June 29th. This trend was reversed since the leading SYRIZA, undergoes a decrease in positive sentiment and increase in negative. The results of this paper shed light on the often unnoticed societal and political trends that guide citizen's choices and actions.

V. Haripriya and Dr. Poornima G Patil [11] talks about available approaches for sarcasm detection on twitter. Sentiment classification can be done in two ways, Machine Learning and Lexicon based approach. Machine learning can be supervised or unsupervised. Supervised require training and test sets. Various classifiers like decision tree, SVM, Neural Networks are used in supervised learning. The lexicon approach can be divided in dictionary based or corpus based approach. Dictionary contains words and their positive/negative. Sentiment tags can be created using otology. Corpus based relies on probabilities of occurrence of a sentiment word. Authors gave a brief description of all the supervised approaches, unsupervised approaches. They also discussed some of the machine learning approaches like Hybrid approach, Deep Learning. Finally authors reached to conclusion that SVM lacks in transparency, Decision tree is Easy and simple but with drawbacks, Deep learning gives better results for large data set with pre trained data, Random forest works with less data and more attributes, Naïve bayes works with small amount of training data and Lexicon approach gives better results if it works with small dictionary not on large dataset.

As the topic of Sarcasm detection has been explored and yet being explored, almost of them are done for English language. Least or negligible has been done for Hindi language. Hence, Santosh Kumar Bharti et al. [12] tried to propose a context based approach to detect sarcasm in Hindi tweets. They used Sarcasm as a contradiction form. As the sentiment analysis is a part of NLP but sentiment analysis doesn't consider any sarcastic sentiment hence making difficult to detect sarcasm. In real scenarios natural Hindi tweets are different in nature unlike English tweets or Hindi translated from English. Authors collected single line news from twitter Hindi news sources such as ABP, aajtak, etc. similarly a corpus of Hindi tweets are collected on the news keywords and some timestamp. Then the sentiment of the tweet and context of related news is matched. If differs then sarcastic otherwise non sarcastic. Authors used POS tagging and Hindi SentiWordNet. They collected 2500 Hindi news tweets manually. Keywords are extracted from news corpus. POS tagging is applied on the corpus. Next 4000 Hindi tweets are extracted using keywords. Manual annotation is done on these 4000 tweets to identify them as sarcastic or non sarcastic by 3 professionals then inter annotator agreement is calculated by using Fleiss' Kappa coefficient. Among 4000 tweets 3000 are used as training and 1000 as test. Next sarcasm detection engine is explained. SDE takes tweet and find POS tags from news and tweet. Then sentiment identification algorithm is applied on tagged tweet file and context identification algorithm is applied on tweet as we as news to identify the sentiment and context. The proposed tweet contradicts sentiment and context says that if the context polarity of a tweet is contradicted with context polarity of the news then it is sarcastic otherwise non

sarcastic. Performance measures considered are accuracy, precision, recall and F1- score measure. Finally the proposed approach resulted in precision 84.8%, recall at 83.3%, F1 score at 84.2% and accuracy at 87.0%

The hyperbolic features consist of intensifiers and interjections of the text. The proposed approach by Santosh Kumar Bharti et al.[13] results in accuracies 75.12%, 80.27%, 80.67%, 80.79% and 80.07% for Naïve Bayes, Decision Tree, SVM, Random forest and AdaBoost respectively. 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, etc are the interjections. Authors collected 10000 tweets for training and test using 5 folds cross validation. They preprocess or clean the data. POS tagging is done after preprocessing. Then Feature extraction is carried out. In feature extraction the intensifiers and interjections are extracted from POS tagging. An algorithm is proposed for feature extraction. After feature selection, classifiers like Naïve Bayes, Decision tee, Support Vector Machine, Random Forest, Ada Boost are applied. A five length feature vector is generated from extracted features tag as FT, INT, RTBP, BTP and TTP. Values for tags are 1 for presence and 0 for no presence. The label of vector is put 1 if sarcastic otherwise 0.
All the trials for performance measures have been done 5 times. Ada boost has better a performance with accuracy 84% recall 80% and F1 score 82%.

The magnitude of data generated online is colossal. Md Saifullah Razali et al. [14] explains how sentiment analysis has been benefitted from sarcasm detection where the accuracy of certain Sentiment Analysis systems improved. Even though sentiment analysis is not the same as emotion recognition, these two fields are closely related. Emotions can be broken into many parts whilst sentiment analysis is usually only broken into positive, negative and neutral. Authors referenced various papers for sentiment analysis and sarcasm detection. Authors say that researches also stress on the importance of using multimodalities to solve the problem of lack of useful features in text for sarcasm detection. This lack of textual features can be supplemented by analyzing other types of online media. Using Tumblr as a data source, images and text features are combined to perform sarcasm detection. This is because text is sometimes embedded directly into images, making it part of image meaning. Authors also discuss about various approaches of different authors for sarcasm detection. They found that a recent paper that used multimodality was using 10000 positive photos from tumblr. They were manually annotated then crowflower is used to find ground truth of gold standard. The result tumblr set is that the fusion of text and images yields the best performance over two platforms. Hence author reacted to the conclusion that existing research indicates an increasing use of multimodality especially from social media.

Sentiment Analysis is a study of people's attitude, opinion and emotions to classify whether it is positive, negative or neutral. Use of emoticons on social media has increased rapidly in recent years. Various issues like sarcasm detection, multilingualism, handling acronyms and slang language lexical variation and dynamic dictionary handling are discussed. Ms. Payal Yadav and Prof. Dhatri Pandya [15] discussed about Text and emoticons as emoticons are powerful units to change the polarity of a statement. Such as smiling emoticon is positive. Also, authors described about related works done in this field along with the types of approaches being in existing systems. They forces on the idea that much researches have been done on analyzing sentiments based on attributes such as gender, age, location etc which means context based sentiment analysis. As people express their views based on their gender, location, age, time, etc.
They state that most of the existing systems remove special characters at preprocessing stage which forms emoticons. Preprocessing includes removal of stopwords, stemming, lemmatization, POS tagging, Expanding abbreviations and chunking. Then feature extraction and selection is explained. Feature extraction consists of unigrams, bigrams, bi-tagged. Also, frequency of features appearing is computed. They also discuss about factors affecting sentiment analysis such as level of analysis, domain dependency, intensifiers, count of emoticons, emoticon positions and classification. Also major issues like Sarcasm, Multilingualism, Dynamic Handling, Acronyms and slangs, Lexical variations are defined. Authors suggest that various machine learning techniques and lexicon based techniques can be combined to form a hybrid approach which may result into more accurate sentiment analysis.

Md Shairi Md Shuaimin et al.[16] say that the challenge of sarcasm detection is more difficult in bilingual texts. Hence, they claims to propose and approach for feature extraction for easing out the difficulty. Features are divided into four categories by author as Lexical, Pragmatic, Prosodic and Syntactic. They also investigated use of idiosyncratic features to capture peculiar and odd comments found in text. SVM(non linear) is used to determine effectiveness. F1 score of 85.2% is claimed to be achieved. They considered Malay and English Languages. The proposed approach comprise of two phases. First they extract lexical, pragmatic and prosodic features in order. Second they translate bilingual data set to English and extract further prosodic features. Working mechanism: Extraction from bilingual corpus which includes preprocessing of corpus and lexical features extraction used Malay and English dictionaries for this, then, pragmatic features extraction is done by recognizing punctuation marks and restricting continuous marks to the maximum length of 3. Prosodic(Malay) feature is extracted in third step which includes the list of interjections. Extraction of features from English translated corpus. Corpus is translated in English using Google Translate. Prosodic (English) features are extracted. Then special features are extracted. Authors selected four POS tags: Noun, Verb, Adjective and Adverb based on Japerson's Theory for ranking content. Then Idiosyncratic features are extracted which means that syntax rules are followed in form of noun-adposition noun. For the taken corpus they get 177 idiosyncratic features identified. For training of classifiers authors took top 25%, 50% and 75% proportion of features. Classification is done with Non linear SVM. The best overall performance was recorded by the combination of the syntactic, pragmatic and prosodic features categories with F measure score of 85.2%. Overall result demonstrated the effectiveness of the use of syntactic feature for sarcasm detection compared to lexical features.

Among numerous researches on the sarcasm detection, none of them is done on the problem of self deprecating sarcasm which is special category of sarcasm mainly used by users to deprecate or criticize themselves. Muhammad Abulaish and Ashraf Kamal [17] claim to propose a novel self deprecating sarcasm detection approach by using amalgamation of rule based and machine learning approach. A total of 11 features are used with three classifiers Decision Tree, Naïve Bayes and Bagging. Dataset is of 107536 tweets. Sarcasm can be divided into seven categories: self depreciating, brooding, deadpan, polite, obnoxious, maniac and raging. Self depreciating is special category that can be defined as sarcasm that plays off an exaggerated sense of worthilessness and inferiority. The proposed approach is a two layer approach: Data Crawling and Preprocessing, Self around tweets detection (Phase 1)

This module is mainly used to filter and its task is to identify self around d tweets that are candidates for self deprecating sarcasms. It is rule based. Feature extraction and classification (Phase 2) Features has 11 types among 2 categories

a. Self depreciating: F1- Interjection followed by "I"
F2- "I" followed by verb and question word
F3- Common depreciating patterns
F4- "I" followed by verb and adverb or adjective
F5- "am" followed by adjective or ad verb
F6- "I" followed by negative modal verb

b. Hyperbolic Feature: F7- Interjection counts
F8- Exclamation marks
F9- Question marks counts
F10- Adverb count
F11- Adjective count

Model learning and Classification: Three classifiers Decision Tree, Naive Bayes and Bagging are used as a component of WEKA tool kit 3.8. Authors experimented on two datasets: dataset 1: (Balanced) and dataset 2 (Unbalanced). Out of all classifiers Decision Tree gave the best result on both data sets by precision 98.40%, recall 91.4%, F score 94.8% and precision 96.9%, recall 92.3%, F score 94.5 for data set 1 and data set 2 respectively.

Shubham Rendalkar and Chaitali Chandankhede [18] aim to develop a system that groups posts on emotions, sentiments and find sarcastic posts. Also the system should be able to determine the emotion of the comments. They used WordNet and WordNet affect libraries. Posts are retrieved with API, Posts are tagged with Stanford POS tagger, Sentiment score are found with WordNet, SentiWord Net, Sentiment score of whole post is calculated, these scores are then provided to sarcasm detection module. The sarcasm detection module has word based detectors, which uses SentiWord Net for positive, Negative and Neutral declaration, Emoticon based detection if emoticon is present, Hybrid sarcas detection approach: Combines the first two approaches, Positive sentiment and negative situation: A contrasting situation with respect to the sentiment, Pattern and Text match based detection: The occurrence of part of speech in the sentence gives possibility of sarcasm, Interjection Word Start: IWS checks for first tag as an interjection word and then subsequently checks for the second tag as an adjective or adverb.The result of experiments shows that hybrid method yields better result as precision 88.57%, recall 93.23% and F score 90.84%.
A cross domain sarcasm detection framework that allows acquisition, storage and processing of tweets to detect sarcastic contents in online reviews texts is proposed by Shrishti Sharma and Shampa Chakraverty [19], they have used Amazon as a source. SVM and Neural Network are used as classifiers using lexical, pragmatic, linguistic incongruity and context incongruity features. They utilize the Amazon review texts as reviews are mostly said in sarcastic manner that make developers hard to understand actual meaning behind the phrases. Various research papers are references in this paper. They (authors) say it is an intricate task in the domain of Natural Language Processing. They have used Machine Learning and precompiled data set for training. Features are divided in four categories: Lexical features: uses N-grams, Pragmatic Features: Number of Capital letters, Emoticons, Slang expressions, punctuation marks, Linguistic Incongruity: Positive polarity for Negative situation, number of sentiment incongruity, count of total number of positive and negative words, largest positive and negative sub sequence, lexical polarity, Context incongruity: Word vector based similarity or discordance is indicative of semantic similarity which in terms is a handle for context incongruity
Un weighted similarity feature, distance weighted similarity features The result of experiments yield precision 72%, recall 93%, F1 Score 81% and precision 76%, recall 83% and F1 score 80% for SVM and Neural Network respectively for tweet corpus. F1 score has also been calculated for different feature categories. For Amazon Product review corpus Precision 50%, Recall 95%, F1 score 66% and precision 51%, recall 88% and F1 score 65% for SVM and Neural Networks respectively.

Joseph Herve Balanke and Haripriya V [20] proposed two sarcasm analysis systems both obtained from the extension of the lexicon algorithm. The First system consists of a combination of lexicon algorithm and a pure sarcasm analysis algorithm. The second system consists of the combination of a lexicon algorithm and sentiment prediction algorithm. They defined types of machine learning i.e. Supervised and Unsupervised. Authors took the problem as lexicon algorithm is insufficient for sarcasm detection as it limits itself to give the polarity of a textual content without being able to specify whether the textual content is sarcastic or not. Proposed work is as follows:

(i) First system: It is the combination of lexicon algorithm and a pure sarcasm analysis algorithm. This system takes textual content as input. Polarity is calculated then parsed into pure sarcasm analysis algorithm. The final output is a list of Sarcastic and non sarcastic lingual content.
(ii) Second system: It is a combination of lexicon algorithm and a sentiment prediction algorithm. Polarity is calculated and also sentiment prediction algorithm is used to predict the sentiment. The result from lexicon and sentiment algorithm are compared, if results are different then sarcastic else non sarcastic.

Naïve Bayes is used for classification purposes. In the first system it has been noticed that the training set of the sarcasm analysis algorithm must be relevant to the actual data that needs to be analyzed in order to contain meaning and improve accuracy. The Second system is a vast area and more works need to be done in order to develop a system that would allow the collection of environment details under which the textual contents would made on Social Media.

The impediment of identifying sarcasm in social media text is discussed by Rapul Bhargava et al [21]. Proposed algorithm analyzes the impact of these features and a combination of them on the review data set in which review had three or more sentences. So the context of sentence is also taken into consideration. The proposed approach is as: Data Set description: Training data contains 1254 Amazon review, 437 ironic and 817 non ironic, Data Preprocessing: Reviews are tokenized. Treebank Word tokenizer for sentence to words, Punkt tokenizer for reviews into sentences, POS Tagging is done, Polarity and Sentiment is attached to each word which can be positive, Negative or Neutral, Feature extraction: following features are used to train model Star Rating, Star Polarity discrepancy, Bag of words, Bag of bigrams, Hyperbole, Scare quote, Ellipsis Plus punctuations, N gram plus punctuations, Emoticons, interjection, Capitalization, Pattern based features. Classifiers used are SVM, Logistic Regression and Naïve Bayes. The performance is measured in F measure. Highest F measure of 80% is given by bag of words. The result given by Naïve Bayes is lower compared to Logistic regression and linear SVM because Naïve Bayes has a higher bias but lower variance. Authors concluded that context matters in training machine for predicting sarcastic sentences. Precision increases by trying different combinations of correct sets of features.
All of the authors we have gone through follow a similar pattern in recognition of Sarcasm in the text. The most common procedure is explained in next section and proposed procedure to detect sarcasm in those text which changes behavior with time is explained in section 4 of this report.

3 Most Common Architecture for Sarcasm Detection (Existing Architectures)

As of the scripts we have read we have come to know about some common steps which are followed by almost every researcher in this field on the topic of Sarcasm Detection. We will be discussing a little bit in brief about the steps involved in the Sarcasm Detection.

Several steps which are common to every researcher are:

1. Data Sets: Data sets are of two different types, Training Data and Test data.
   Training Data Set: This is the set of data which is used to train classifiers. More the number of data in Training set, the more accuracy a machine will achieve to classify correctly. Training data consist all types of Sentences including sarcastic and non sarcastic. Manual annotations are done to better train classifiers.
   Test Data Set: Data which are used to test the classifiers are called test are called test data. These data are not the repeated from the training data.
   Some of the researchers also define a third data set which is called Tuning Data Set, which is used after training data set to tune further the classifiers.
2. Data Pre Processing: Raw data fetched from OSMN contain various types of noises and impurities in them. Like non ASCII characters, special characters, lots of punctuations, hashtags, etc. These Noises are removed as they are unnecessary for the data to work, because they carry negligible significance in detecting sarcasm.
   Data Preprocessing contains various steps:
   Noise removal: Noises like mentions, retweet marker, repost marker, shared marker non ASCII characters, Special characters etc are removed from the text.
   Tokenization: Tokenization is the process to mark each word in the text with a token such that they cannot be breached and understood by any third party but only the tokenizer.

Lemmetization: Lemmetization is a process to convert the word to is root word or say the base word which conveys the same meaning. For example “giving” will be converted to its base word “give”, “running” to “run” and so on.

POS tagging: Part of speech tagging is a grammatical association step where every word is tagged with its grammatical Part of Speech. This is also called Semantic analysis part.

3. Feature Extraction: Every researcher have their own considered feature for declaring some text sarcasm. Hence, Feature extraction is the process which remains same, but features and their representation may vary from researcher to researcher. In this step the text is analyzed and features which are programmed to be considered are matched and an output is generated based on the allowed values. For example, 1 for presence of Emoji and 0 no emoji in the text.
4. Classification: Classification is the crucial step as the choice of Classifiers plays an important role in accuracy rate of detection of Sarcasm. Researchers select a suitable Classifier which should be able to handle the type of data being fed into the machine and also the performance measure of the classifier should be monitored.
   There are various types of classifiers available out of which the choice of classifier is based on various factor.
   Classification step is the last step which outputs as if the given text is Sarcasm or a Non Sarcasm.

4 Method to detect sarcasm in the Dynamic Sarcasm/Factual Sarcasm

4.1 Introduction

We have already read about the existing systems and done some practical works in regard with these architectures. While we were going through those papers, a very interesting type of statement we have encountered. That type of statement was not a simply a sarcasm but was able to change its property from simple statement to sarcasm with time.

The most common use of that type of statement is by supporters and people against some leadership. For example suppose Mr. X is a leader and he is contesting to the post of XYZ. Before elections his supporters will flood the internet, Social Media with his promises claiming something like “If Mr. X would be the XYZ, there will be no recession/inflation/crime” but once he become the XYZ and then also he is unable to deliver his promises and all the events take a rise than what they have been promised to diminished. Then person against Mr. X would start posting things like “If Mr. X would be the XYZ, there will be no recession/inflation/crime”. This same statement changed its form from a declarative sentence to a Sarcasm sentence based on the timeline scenarios.

Human mind can easily interpret these changes in the form of sentence as they can relate to the current events easily, while automating this interpretation for machine to do it independently is nowhere to be found openly like other type of sarcasm detection. The field of Sarcasm detection is currently in the evolution stage and has a potential of further developments every day.

4.2 History of Sarcasm detection

Sarcasm detection has attracted many researches throughout the world in recent years. A lot of emphasize has been put into determining the sarcasm in a sentence on Online social Media Network as these statements can affect huge mass.

Automatic detection of Sarcasm has been tried to be carried out by many approaches including Machine Learning, Neural Network and various other methods.

A lot researches are being carried out in this field daily. What actually makes this topic a greater concern is that a news was published on BBC.com in year 2014 about US Secret Service was looking for an automated sarcasm or False positive detection software such that they can analyze the social media data to determining the opinion of Public without even letting them the know that they are being watched. This news can be found on this link https://www.bbc.com/news/technology-27711109

4.3 Developing an approach for Dynamic sarcasm/ Factual Sarcasm detection

This approach will check for possibility of a Statement being Sarcasm after it has been marked non sarcastic by any of the existing approaches. For doing so, we are required to keep some more information about the statement in the memory.

Steps:

1. Modification of existing architecture being used:
   Applying any of the existing approach to the statement to check for sarcasm in it. Just that approach should be modified in such a way that it should fetch following information about the statement before preprocessing:
   a. Location: location will be used in our approach to look for related contents with the statement.
   b. Timestamp: Time stamp is used to match with the fact mentioned in the statement with the existing reality. If the matching will create contradict between mentioned fact and reality then this will be probably a sarcastic statement, otherwise it will be a general statement.
   c. User Demographic: User demographic can be used to compare their information if the statement contains something related to their own self.
2. Run the Sarcasm detection algorithm defined in the architecture being used.
3. If detected sarcasm then Exit. Otherwise go to step 4
4. This step intends to finds some of the keywords in sequential order which may be available in the statement those can make a complete sense completely. For example in statement “If Mr. X would be the XYZ, there will be no recession/inflation/crime”, keywords “X would XYZ” can collectively generate a sense. Also keywords “will no recession/inflation/crime” can collectively makes sense.
   Stanford Core NLP (link https://stanfordnlp.github.io/CoreNLP/) can be used to identify Named Entities(nouns), Dependencies and Coreference very well.
   The named entity will help us to identity the nouns used in the statement like Name of person, place or anything.
   Dependency identification will help in finding which word is closely associated with which word and those words collectively can make a sensible statement to do further processing in our approach.
   The Coreference function will enable us to find the referenced words in the same sentence
   For example “Mr X is a Msc student. He did majors in Computer Science”.
   This statement has:
   Named entities: X, Msc, Computer, Science.
   Dependency: MSc is dependent on Mr X, student is depending on MSc, Science is depending on Computer, etc.
   Coreference: He -> X
   Due to these easily usable features of Stanfore Core NLP, it is most commonly used in this field.

5. As we have already let the existing systems to detect sarcasm in the statement, we will be having the sentiment value of each words in the given sentence. The calculation of word sentiment is majorly done with SentiWordNet library.
6. In this step our system will check for a sequence of words where "If" appears just before any named entity or the coreference and that is followed "would", "will", "can","have","had" or "could" then our system will capture all Named entities in sequence which appear in between "If" and the nearest comma, full stop or stop word. After capturing the sequence, an automated Google Query API will be used, as per my finding I have come across Google's Custom Search JSON API available at https://developers.google.com/custom-search/v1/introduction/?apix=true. This automated Query API will put the sequence of words along with the year of the post in the search portion and make a google search. The automated google search query can be used in http format in this form

   GET
   https://www.googleapis.com/customsearch/v1?key=INSERT_YOUR_API_KEY&cx=[SEARCHENGINE]&q=[QUERY]
7. Google Search will return a list of results.
8. Our system will capture the very first result. More preferably the Wikipedia result if present will be selected and looked for named entity in our search otherwise the very first link will be selected for comparison.
9. A search of all the named entities will be done on the link selected. The first sentence having our all the searched named entities will be capture, if there is no sentence having all these named entities all together then there is greater chance that the fact is mentioned in sentence of our test post is false still for a safer side we will carry out this step number 9 for 10 top google search results. When even the 10^{th} result proves that the fact of test statement is false then Stop and Exit marking it Non Sarcasm, otherwise if any sentence is found to have all of the named entities then go to step 10.
10. The presence of all named entities in the sentence of any result will probably be used to prove that the fact of test statement has become a reality with time then this statement will obviously be marked as sarcasm.

4.4 Importance

An importance of this type of sarcasm detection is going to be for the victim itself that they will come to know what others actually think of him and eventually will come to know about the expectations of the people.

Packages suggested:

Stanford Core NLP, Natural Language Toolkit, Genism

Google Custom Search JSON API

4.5 Disadvantage

In the era of millions of data posted on social media every second, the detection of Dynamic Sarcasm may seem to be a time consuming task. Because no proper work on this topic is still available and daily changing trend of sarcasm also puts an extra hardship for machines. Human minds can intercept easily every sarcasm but a machine must be learnt and the way to detect sarcasm in this world of variable forms and patterns of sarcasm, making machine think about a human being a really tough task.

5 Methodologies

In the area of Sarcasm detection which actually evolves from the area of Sentiment analysis, we have various methodologies. In this section we will discuss about some of those methodology and why they are used and where they are used.

5.1 Stanford Core NLP: https://stanfordnlp.github.io/CoreNLP/

CoreNLP enables users to derive linguistic annotations for text, including token and sentence boundaries, parts of speech, named entities, numeric and time values, dependency and constituency parses, coreference, sentiment, quote attributions, and relations. CoreNLP currently supports 6 languages, including Arabic, Chinese, English, French, German, and Spanish.

5.2 SentiWordNet https://github.com/aesuli/SentiWordNet

https://github.com/aesuli/SentiWordNet/blob/master/papers/LREC06.pdf

SentiWordNet is a lexical resource for opinion mining. SentiWordNet assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity.

5.3 R https://www.r-project.org/

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS.

5.4 Worked out Data fetching and Data Cleaning using R

Different ways to fetch tweets

```
## search for 1000 tweets mentioning Hillary Clinton
hrc <- search_tweets(q = "hillaryclinton", n = 1000)

## data frame where each observation (row) is a different tweet
hrc

## users data also retrieved. can access it via users_data()
users_data(hrc)

## search for 1000 tweets in English
djt <- search_tweets(q = "realdonaldtrump", n = 1000, lang = "en")

## preview tweets data
djt

## preview users data
users_data(djt)

## exclude retweets
rt <- search_tweets("rstats", n = 500, include_rts = FALSE)

## perform search for lots of tweets
rt <- search_tweets( "trump OR president OR potus", n = 100000, retryonratelimit = TRUE )
```

```
## plot time series of tweets frequency ts_plot(rt, by = "mins")

## make multiple independent search queries
ds <- Map( "search_tweets", c("\"data science\"", "rstats OR python"), n = 1000 )

## bind by row whilst preserving users data
ds <- do_call_rbind(ds)

## preview tweets data
ds

## preview users data
users_data(ds)
}
if (FALSE) {

## search using multilple queries
st2 <- search_tweets2( c("\"data science\"", "rstats OR python"), n = 500 )

## preview tweets data
st2

## preview users data
users_data(st2)

## check breakdown of results by search query
table(st2$query) }
if (FALSE) {
hrc users_data(hrc)
djt <- search_tweets(q = "realdonaldtrump", n = 1000, lang = "en")
djt
users_data(djt)
rt <- search_tweets("rstats", n = 500, include_rts = FALSE)
rt <- search_tweets("trump OR president OR potus", n = 100000,
retryonratelimit = TRUE )
ts_plot(rt, by = "mins")
ds <- Map( "search_tweets", c("\"data science\"", "rstats OR python"), n = 1000 )
ds <- do_call_rbind(ds)
ds
users_data(ds)
}
if (FALSE) {
st2 <- search_tweets2( c("\"data science\"", "rstats OR python"), n = 500 )
st2
users_data(st2)
table(st2$query)}
```

Other method to fetch twitter data

```
library(twitteR)
library(ROAuth)
library(httr)
```

```
# Set API Keys
api_key <- "xxxxxxxxxxxxxxxxxxxx"
api_secret <- "xxxxxxxxxxxxxxxxxxxx"
access_token <- "xxxxxxxxxxxxxxxxxxxx"
access_token_secret <- "xxxxxxxxxxxxxxxxxxxx"
setup_twitter_oauth(api_key, api_secret, access_token, access_token_secret)

# Grab latest tweets
tweets_sanders <- searchTwitter('@BernieSanders', n=1500)

# Loop over tweets and extract text
library(plyr)
feed_sanders = laply(tweets_sanders, function(t) t$getText())

# Read in dictionary of positive and negative works
yay = scan('opinion-lexicon-English/positive-words.txt', what='character', comment.char=';')
boo = scan('opinion-lexicon-English/negative-words.txt', what='character', comment.char=';')

# Add a few twitter-specific negative phrases
bad_text = c(boo, 'wtf', 'epicfail', 'douchebag')
good_text = c(yay, 'upgrade', ':)', '#iVoted', 'voted')
score.sentiment = function(sentences, good_text, bad_text, .progress='none')
{
require(plyr)
require(stringr)

# we got a vector of sentences. plyr will handle a list

# or a vector as an "l" for us

# we want a simple array of scores back, so we use

# "l" + "a" + "ply" = "laply":
scores = laply(sentences, function(sentence, good_text, bad_text) {

# clean up sentences with R's regex-driven global substitute, gsub():
sentence = gsub('[[:punct:]]', '', sentence)
sentence = gsub('[[:cntrl:]]', '', sentence)
sentence = gsub('\d+', '', sentence)

#to remove emojis
sentence <- iconv(sentence, 'UTF-8', 'ASCII')
sentence = tolower(sentence)

# split into words. str_split is in the stringr package
word.list = str_split(sentence, '\s+')

# sometimes a list() is one level of hierarchy too much
words = unlist(word.list)

# compare our words to the dictionaries of positive & negative terms
pos.matches = match(words, good_text)
neg.matches = match(words, bad_text)
```

```
# match() returns the position of the matched term or NA

# we just want a TRUE/FALSE:
pos.matches = !is.na(pos.matches)
neg.matches = !is.na(neg.matches)

# and conveniently enough, TRUE/FALSE will be treated as 1/0 by sum():
score = sum(pos.matches) - sum(neg.matches)
return(score)
}, good_text, bad_text, .progress=.progress )
scores.df = data.frame(score=scores, text=sentences)
return(scores.df)
}

# Call the function and return a data frame
feelthabern <- score.sentiment(feed_sanders, good_text, bad_text, .progress='text')

# Cut the text, just gets in the way
plotdat <- plotdat[c("name", "score")]

# Remove neutral values of 0
plotdat <- plotdat[!plotdat$score == 0, ]

# Nice little quick plot
qplot(factor(score), data=plotdat, geom="bar",
fill=factor(name),
xlab = "Sentiment Score")
doInstall <- TRUE
toInstall <- c("twitteR", "dismo", "maps", "ggplot2")
if(doInstall){install.packages(toInstall, repos = "http://cran.us.r-project.org")}
lapply(toInstall, library, character.only = TRUE)
searchTerm <- "#rstats"
searchResults <- searchTwitter(searchTerm, n = 1000) # Gather Tweets
tweetFrame <- twListToDF(searchResults) # Convert to a nice dF
userInfo <- lookupUsers(tweetFrame$screenName) # Batch lookup of user info
userFrame <- twListToDF(userInfo) # Convert to a nice dF
locatedUsers <- !is.na(userFrame$location) # Keep only users with location info
locations <- geocode(userFrame$location[locatedUsers]) # Use amazing API to guess

# approximate lat/lon from textual location data.
with(locations, plot(lon, lat))
worldMap <- map_data("world") # Easiest way to grab a world map shapefile

zp1 <- ggplot(worldMap)
zp1 <- zp1 + geom_path(aes(x = long, y = lat, group = group), # Draw map
colour = gray(2/3), lwd = 1/3)
zp1 <- zp1 + geom_point(data = locations, # Add points indicating users
aes(x = lon, y = lat),
colour = "RED", alpha = 1/2, size = 1)
zp1 <- zp1 + coord_equal() # Better projections are left for a future post
zp1 <- zp1 + theme_minimal() # Drop background annotations
print(zp1)

install.packages("twitteR")
```

```
install.packages("RCurl")
>library(twitteR)
>library(RCurl)
> require(twitteR)
> require(RCurl)
> consumer_key <- 'fDJjAivdWBzZ34xLYcDSAaSS0'
> consumer_secret <- '9uLBZ2xhBbXJjPhlDGEeN1MSk0DsbhFSKwySyczTX91Li5UoTD'
> access_token <- '1222231309738627072-kqFWA3ijh063LmhlZ8sSSftDYnVdWL'
> access_secret <- 'T7TTqQOqEGhBywNec96ea2OiQphEBfuHcuXG1dW5NNVpq'
>
> setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)
// [1] "Using direct authentication"
Use a local file ('.httr-oauth'), to cache OAuth access credentials between R sessions?
1: Yes
2: No
> LFC_tweets <- searchTwitter("LFC", n=20, lang="en") //To Searc tweets related to LFC
> LFC_tweets //To view the tweets on screen
> str(LFC_tweets) //To check the structure of fetched tweets
> LFC_tweets[1:3] ///For cleaning and mining
> install.packages("tm")
> install.packages("wordcloud")
> require(tm)
> require(wordcloud)
> cricket <- searchTwitter('IPL+2020', lang = "en" ,n=500 ,resultType="recent")
> class(cricket)
> cricket_text <- sapply(cricket, function(x) x$getText())
> str(cricket_text)
> crick_corpus <- Corpus(VectorSource(cricket_text))
> crick_corpus <- VCorpus(VectorSource(cricket_text))
> crick_corpus
> inspect(crick_corpus[1])
> inspect(crick_corpus[100])
> inspect(crick_corpus[10])
> crick_clean <- tm_map(crick_corpus, removePunctuation)
> crick_clean <- tm_map(crick_clean,content_transformer(tolower))
> crick_clean <- tm_map(crick_clean,removeWords, stopwords("english"))
> crick_clean <- tm_map(crick_clean,removeNumbers)
> crick_clean <- tm_map(crick_clean,stripWhitespace)
> crick_clean <- tm_map(crick_clean,removeWords,c("IPL","2020"))
> wordcloud(crick_clean)
> wordcloud(crick_clean, random.order=F)
> wordcloud(crick_clean,random.order=F, scale=c(3,0.5))
> wordcloud(crick_clean,random.order=F, col="red")
> wordcloud(crick_clean,random.order=F, col=rainbow(50))
> wordcloud(crick_clean,random.order=F,max.words=40, colors=rainbow(50))
```

Fetching Twitter Data Using Hashtag :

```
> require(twitteR)
> library(twitteR)
> library(ROAuth)
> install.packages("ROAuth")
> library(ROAuth)
```

```
> require(ROAuth)
> library(ROAuth)
> setup_twitter_oauth('fDJjAivdWBzZ34xLYcDSAaSS0',
'9uLBZ2xhBbXJjPhlDGEeN1MSk0DsbhFSKwySyczTX91Li5UoTD', '1222231309738627072-
kqFWA3ijh063LmhlZ8sSSftDYnVdWL', 'T7TTqQOqEGhBywNec96ea2OiQphEBfuHcuXG1dW5NNVpq')
> 1
> tweets <- searchTwitter("#IPL2020",n=1000)
> View(tweets)
> ipl.df <- twListToDF(tweets)
> View(ipl.df)
> require(xlsx)
> library(xlsx)
> write.xlsx(ipl.df,"F:/R studio/tweets.xlsx") ////To Download/Fetch Tweets in readable form
#Extracting tweets in a list
crowd <- searchTwitter('metro+stampede', lang = "en", since = '2017-10-25', until ='2017-12-05',n= 1000)
```

We will first install the relevant packages that we need. To extract tweets from Twitter, we will need package 'twitteR'.

'Syuzhet' package will be used for sentiment analysis; while 'tm' and 'SnowballC' packages are used for text mining and analysis.

```
# Install Requried Packages
installed.packages("SnowballC")

installed.packages("tm")

installed.packages("twitteR")

installed.packages("syuzhet")
```

```
# Load Requried Packages
library("SnowballC")

library("tm")

library("twitteR")

library("syuzhet")
```

Next, we will invoke Twitter API using the app we have created and using the keys and access tokens we got through the app.

```
# Authorization keys
consumer_key <- 'Your Consumer Key'

consumer_secret <- 'Consumer_secret'

access_token <- 'accesstoken'
```

```
access_secret <- 'access secret'

setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)
```

#Tweets Fetching

```
tweets <- userTimeline("realDonaldTrump", n=200)

n.tweet <- length(tweets)


tweets.df <- twListToDF(tweets)

head(tweets.df)


> head(tweets.df$text)
```

#Removal of some keywords

```
tweets.df2 <- gsub("http.*","",tweets.df$text)

tweets.df2 <- gsub("https.*","", tweets.df2)

tweets.df2 <- gsub("#.*","",tweets.df2)

tweets.df2 <- gsub("@.*","",tweets.df2)


> head(tweets.df2)

word.df <- as.vector(tweets.df2)

emotion.df <- get_nrc_sentiment(word.df)

emotion.df2 <- cbind(tweets.df2, emotion.df)  //Column binding of tweets & emotion

head(emotion.df2)
```

#Most Positive and Negative Line

```
sent.value <- get_sentiment(word.df)

most.positive <- word.df[sent.value == max(sent.value)]

most.positive
```

```
most.negative <- word.df[sent.value <= min(sent.value)]

most.negative

> most.positive

>most.negative

>sent.value

>positive.tweets <- word.df[sent.value > 0]

 > head(positive.tweets)

> negative.tweets <- word.df[sent.value < 0]

 > head(negative.tweets)
```

#For Neutral Tweets

```
> neutral.tweets <- word.df[sent.value == 0]

 > head(neutral.tweets)
```

Alternate way to classify as Positive, Negative or Neutral tweets

```
>category_senti <- ifelse(sent.value < 0, "Negative", ifelse(sent.value > 0, "Positive", "Neutral"))

>head(category_senti)

> category_senti2 <- cbind(tweets.df2,category_senti,sent.value)

> head(category_senti2)

> table(category_senti)     //to show total +ve,-ve & Neutral
```

6 Validation of the newly proposed mechanism for Dynamic Sarcasm Detection in text

To test the processes involved in our model we first have to use some sarcasm detection tool available such that we can use them to carry out the step 1 of our model. Our step 1 is to find out sarcasm in the text if present according to the architecture we are using for it. If classified non Sarcasm then we have to follow all other steps in our model such that we can find if there is any dynamic sarcasm is available in the text.

For the sarcasm detection purpose we are using an online tool which is available on the link http://www.thesarcasmdetector.com/.

We will feed texts into this system and wait for results for sarcasm in it.

If text is sarcastic then we should stop the process and if not then we can go for further processes of our model.

6.1 The first example of our test text of a tweet available on link (figure 16)

https://twitter.com/Azatshatru2010/status/1260970134686126082

Figure 1 test tweet 1

After feeding this text in the available section on the sarcasm detector online tool to test,

We have found that the system identifies the text as non sarcastic by a -19 sarcasm point.

We can find named entities “modi”, “pm”

Now we can see that the text is posted in the year 2020.

Now we will do a google search manually with named entities and year 2020 ie, “modi pm 2020” When we did a search with keyword “modi pm 2020”. Google provides a list of results. We open the first link. Results are shown in figure 18.

After opening the first link we can extract text from the website in figure 19 and can look up for both named entities in a single sentence.
Here the sentence “Prime Minister Narendra Modi said on Friday he is “fully committed” to the National Education Policy (NEP), which was launched last month,

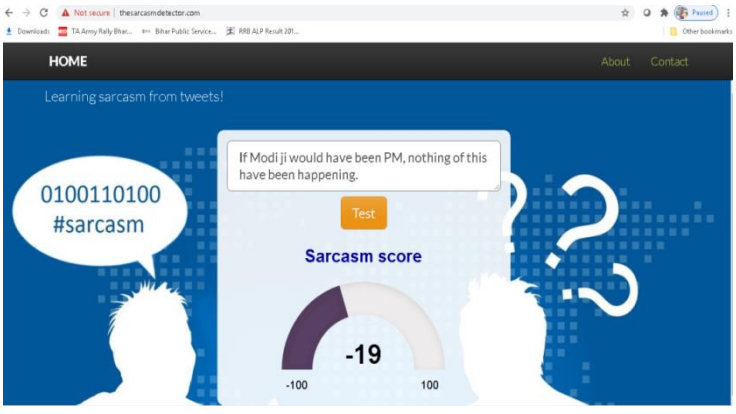


Figure 2 sarcasm value of test tweet 1

as he assured it will be implemented fully" includes both named entities which we have considered.

Now next step is to compare this fetched statement with the statement in the test sentence. By comparing them we can determine that the statement in the test text is factually incorrect. Hence the statement is made with a sense of Sarcasm to the named entity.

Time of the post plays an important role in this as if this post was made before year 2014, the post would be considered general post. But as this post was made in 2020 hence this text has highest probability of being sarcastic in nature.

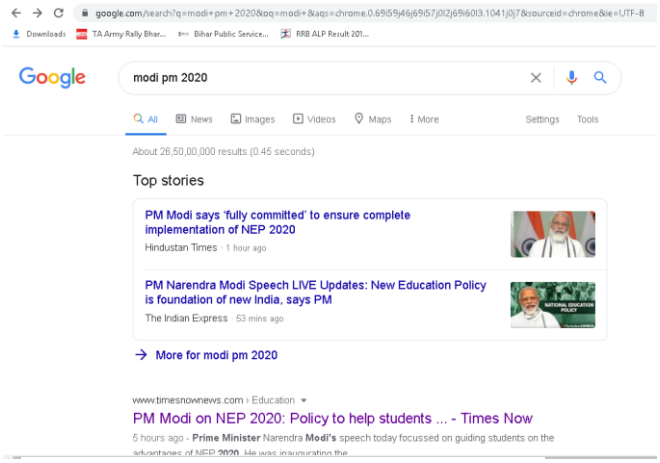


Figure 3 search result for test tweet 1

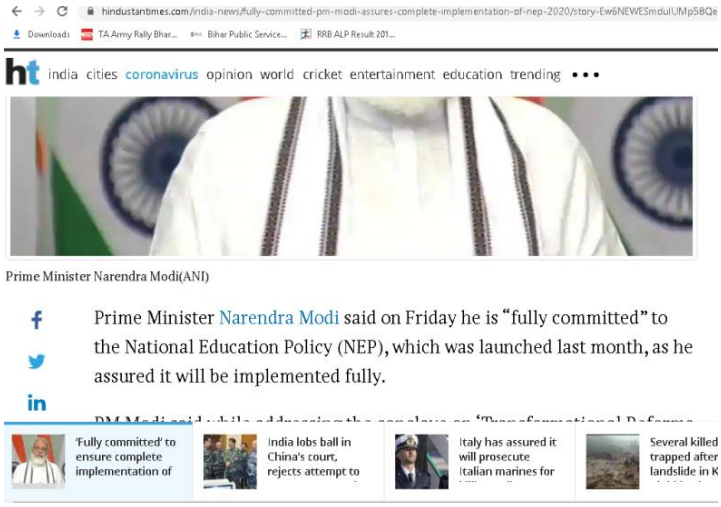


Figure 4 opened first link for test tweet 1

6.2 Second test text link (figure 20)

https://twitter.com/PSHERU/status/1273609914607255557

The sarcasm detection result of this text is shown in figure 22. The result says that this post has -33 chances of being sarcasm.Now we will look for valuable named entities in this text. By using Stanford core NLP we will to know by the coreference feature that in this text "you" referes to "Modi". So we get two names entities of value "modi" and "PM". Also the time of the post can be extracted as year 2020.

Now the google search can be made with "pm modi 2020". The result of this search will be as same as the search result in figure 18. And same from there we can compare statement and find that the fact in the test text is factually incorrect to the real time fact. Hence this will be most probably Sarcasm.

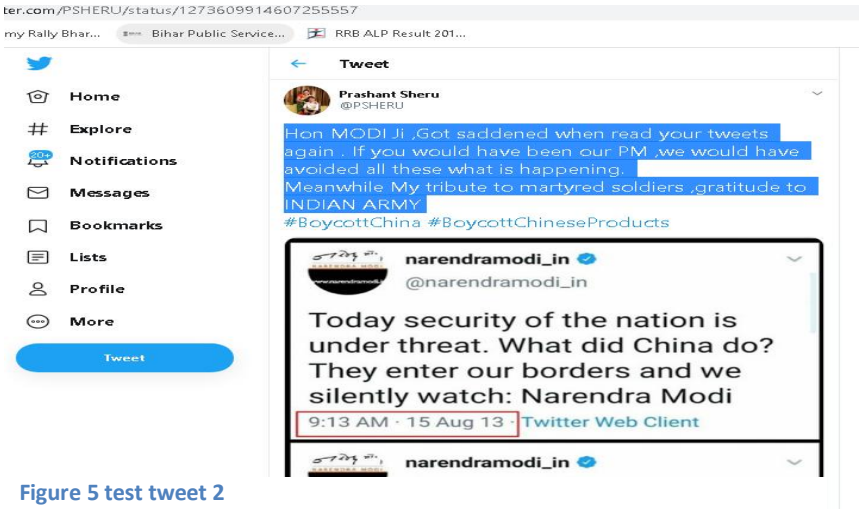


Figure 5 test tweet 2

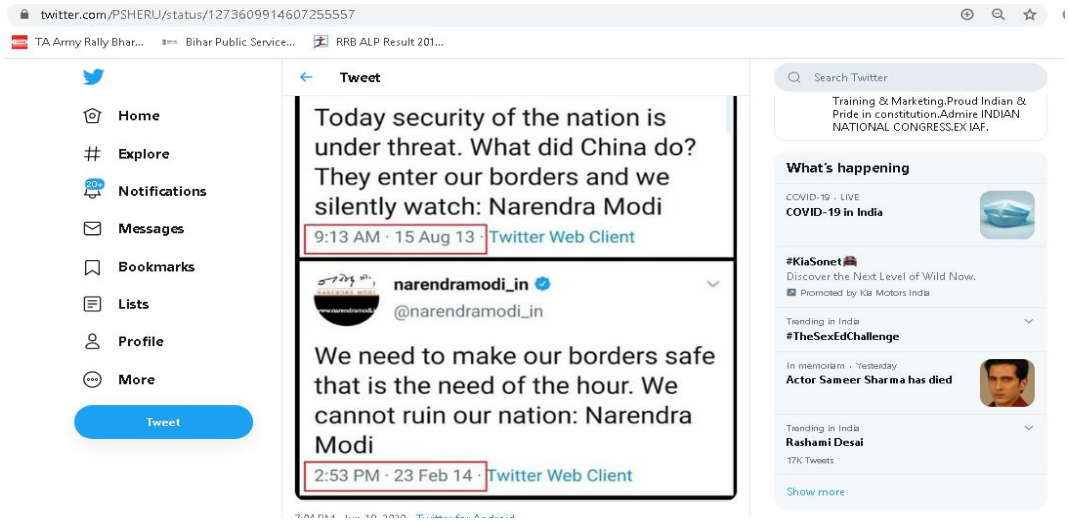


Figure 21 test tweet 2

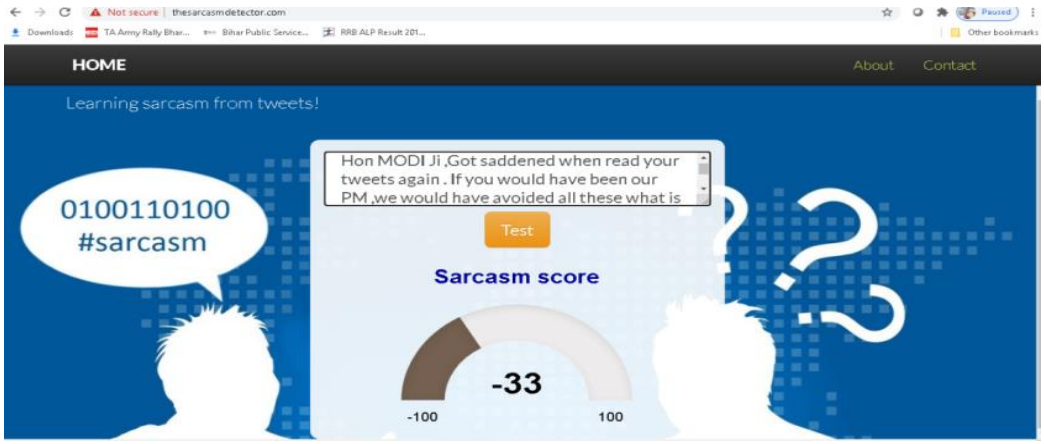


Figure 6 sarcasm value for test tweet 2

6.3 Test Text for third example is from link (figure 23)

https://twitter.com/Samar_Anarya/status/1096002988722667521

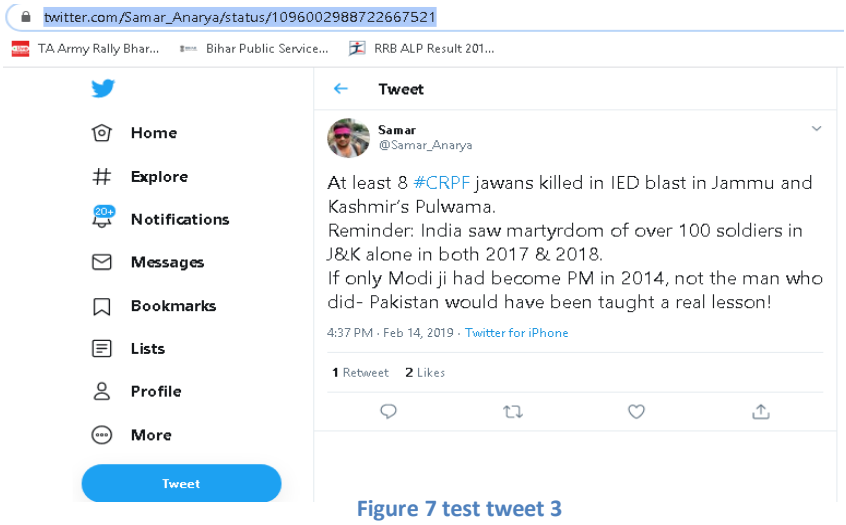


Figure 7 test tweet 3

The result of Sarcasm Detection is as attached in the snap shot in figure 24. Result says that with -19 chances the text is sarcasm, means non sarcastic.

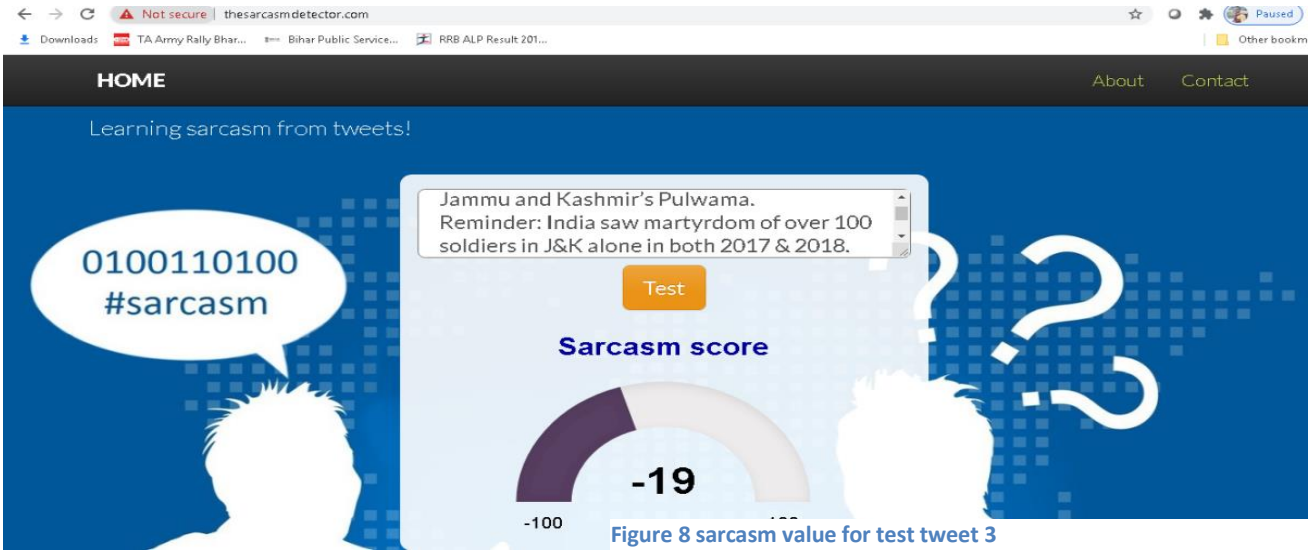


Figure 8 sarcasm value for test tweet 3

Now our model will look for considerable named entities. By the concept of Stanford Core NLP we can assume the named entities to be considered are “modi” “PM”. The time extraction will give year 2019. Google search will be made with “pm modi 2019”. The search result gives following list of links for used keywords. Figure 25

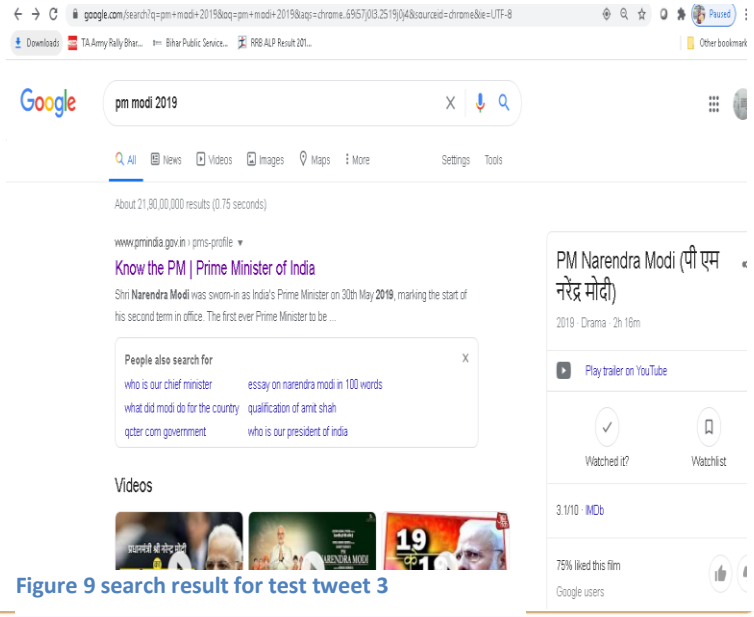


Figure 9 search result for test tweet 3

Opening the first link (figure 26) can provide the model a series of texts and by looking up for our named entities in a single sentence. We can easily find that entities “modi” and “prime minister” are present in the first statement which would be enough to say that the fact mentioned in test text is factually incorrect.

Hence the test text is mostly to be a Sarcasm one.

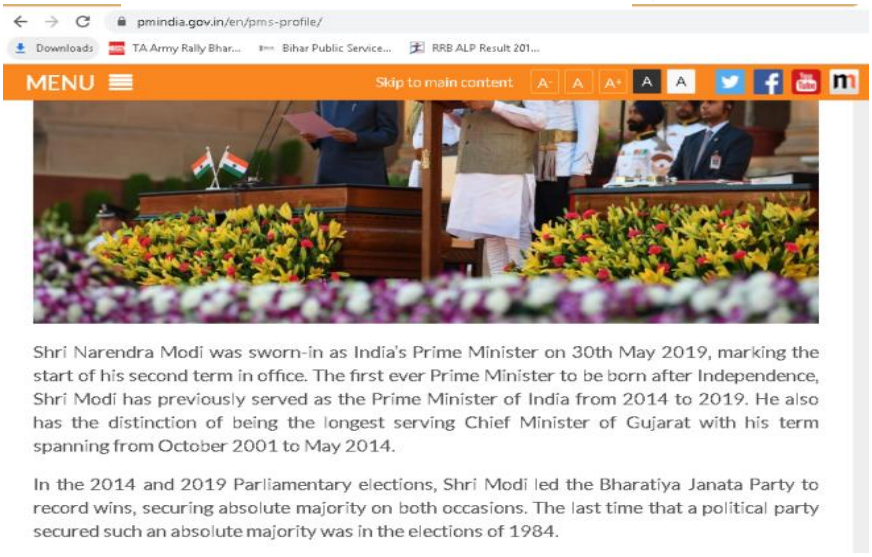


Figure 10 link opened from search result for test tweet 3

7 Conclusion

Sarcasm Detection being a very interesting topic now a day because of almost every service has come up with their online versions, hence reading public reviews about their services and quality plays vital role in further improvement of the organization or the individual. Based on analysis of text if said statements are in a sarcastic way or not could be very useful for self assessment of the products. We have read numerous research papers in the field of Sarcasm Detection. We came to know about various approaches and methodologies to develop a system for doing automatic sarcasm detection either by machine learning or Deep learning or Neural networks or any other method. As we advanced in this field we let our hands in some practical works of the existing systems, also while going through detection methods we came to know about an another type of statement which possess the dynamic behavior that is they changed their property of being a simple statement to being a sarcastic with respect to the time they are posted. That type of sarcasm we have defined as Dynamic Sarcasm or Factual sarcasm, as they tend to conflict with the fact in the statement with the real fact in real time.

We have tried to develop a model to detect this type of sarcasm in the text with the help of Google Customs search API and Standford Core NLP. But we believe that this model can be more thoroughly studied and there is a lot of space for further improvement in the proposed architecture. We have not tried our model of architecture in any automated form right now. What we have done is all the manual processes till now. Hence future scope of Work can be automating our model using languages like Python and java and developing a platform for this work by using available libraries and APIs. Future Scope also includes feeding huge number of Data into the currently existing systems to make machine learn as much of sarcastic patterns as possible also, the trending sarcastic forms of text can also be fed with manual annotation for training classifiers to gain maximum efficiency and accuracy with highest precision value.

8 Bibliography

1. Online Public Shaming on Twitter: Detection Analysis and Mitigation
   Rajesh Basak, Shamil Sural- Senior Member IEEE, Niloy Ganguly and Soumya K. Ghosh, Member IEEE
   10.1109/TCSS.2019.2895734
   IEEE Transactions on Computational Social Systems

2. Sarcasm Detection in Twitter using Sentiment Analysis
   Bala Durga Dharmavarapu, Jayanag Bayana
   International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, volume-8, Issue- 1S4, June 2019

3. Comparative Study of Different Sarcasm Detection Algorithms based on behavioral approach
   Ravinder Ahuja, Shantanu Bansal, Shuvam Prakash, Karthik Venkataraman and Alisha Banga
   8th International Conference on Advances in Computing and Communication (ICACC-2018)
   ScienceDirect, Procedia Computer Science
   143(2018) 411-418

4. Sentiment Detection in Sentiment Analysis
   Dr. Kalpesh H. Wandra, Mehul Barot
   Technical Research Organization India
   ISSN (PRINT): 2393-8374, (Online): 2394-0697,
   Volume-4 ISSUE-9, 2017

5. Opinion Mining in Twitter- Sarcasm Detection
   Sana Parveen, Sachin N. Deshmukh
   International Research Journal of Engineering and Technology (IRJET)
   e-ISSN: 2395-0056, p-ISSN: 2395-0072, 2017

6. Tweet Sarcasm: Mechanism of Sarcasm Detection in Twitter
   Komalpreet Kaur Bindra, Asst. Prof Ankita Gupta
   International Journal of Computer Science and Information Technologies
   Volume 7(1), 2016, 215-217

7. Sarcasm Detection in Twitter: “All your products are incredibly amazing!!!”- Are They Really?
   Moundher Bouazizi, Tomoaki Ohtsuki
   978-1-4799-5952-5/15/$31.00 2015 IEEE

8. Approaches for Computational Sarcasm Detection: A Survey
   Lakshaya Kumar, Arpan Somani and Pushpak Bhattacharyya

9. Contextualized Sarcasm Detection on Twitter
   David Bamman and Noah A. Smith
   Association for the Advancement of Artificial Intelligence, 2015

10. Investigating the complete corpus of Referendum and Election tweets
    Despoina Antonakaki, Dimitris Spiliotopoilos, Christos V. Samaras, Sotiris Ioannidis, Paraskevi Frahopoulou
    2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)
    August 18-21, 2016, san Francisco, CA, USA
    978-1-5090-2846-7/16/$31.00

11. A Survey of Sarcasm Detection in Social Media
    V. Haripriya, Dr. Poornima G Patil
    International Journal for Research in Applied Science and Engineering Technology (IJRASET)

ISSN: 2321-9653; IC value: 45.98; SJ impact factor: 6.887
Volume 5 Issue XII December 2017

12. Context based Sarcasm Detection in Hindi Tweets
Santosh Kumar Bharti, Korra Sathya Babu, Rahul Raman
978-1-5386-2241-4/17/$31.00
2017 IEEE

13. Hyperbolic feature based Sarcasm Detection in Tweets: A Machine Learning Approach
Santosh Kumar Bharti, Reddy Naidu, Korra Sathya Babu
978-1-5386-4318-1/17/$31.11 2017
2017 IEEE

14. The importance of multimodality in Sarcasm Detection for sentiment Analysis
Md Saifullah Razali, Alfian Abdul Halin, Noris Mohd Norowi, Shyamala C. Doraisamy
978-1-5386-2126-4/17/$31.00
2017 IEEE 15th Student Conference on Research and Development (SCOReD)

15. SentiReview: Sentiment Analysis based on Text and Emoticons
Ms. Payal Yadav, Prof Dhatri Pandya
978-1-5090-5960-7/17/$31.00
International Conference on Innovative Mechanisms for Industry Applications (ICIMIA 2017)

16. Natural Language Processing Based Features for Sarcasm Detection: An Investigation using Bilingual Social Media Texts.
Mohd Shairi Md Suhaimin, Mohd Hanafi Ahmad Hijazi, Raynar Alfred, Frans Coenen
978-1-5090-6332-1/17/$31.00
2017 8th International Conference on Information Technology (ICIT)

17. Self Depreciating Sarcasm Detection: An amalgamation of Rule based and Machine Learning Approach
Muhammad Abulaish, AShraf Kamal
978-1-5386-7325-6/1/$31.00
2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)

18. Sarcasm Detection of Online Comments Using Emotion Detection
Shubham Rendalkar, Chaitali Chandankhede
978-1-5386-2456-2/18/$31.00
Proceedings of the International Conference on Inventive Research in Computing Applications (ICIRA) 2018
IEEE Xplore compliant Part number CEP18N67-ART

19. Sarcasm Detection in Online Review Text
Shrishti Sharma and Shampa Chakraverty
DOI 10.21917/IJSC 2018.0233

20. Extension of the Lexicon Algorithm for Sarcasm Detection
Joseph Herve Balanke, Haripriya V.
978-1-5386-7808-4/19/$31.00
Proceedings of the Third International Conference on Computing Methodologies and Communication(ICCMC 2019)

21. FAID: Feature Aftermath for Irony Discernment
Rupal Bhargava, Ayushi Agarwal and Yashwardhan Sharma
978-1-5386-5933-5/19/$31.00
2019 IEEE

22. SARCASTIC SENTIMENT DETECTION IN TWEETS STREAMED IN REAL TIME : A BIG DATA APPROACH
S.K. BHARTI*, B. VACHHA, R.K. PRADHAN, K.S. BABU, S.K. JENA
Department of Computer Science & Engineering NIT Rourkela, India
JOURNAL (SCIENCE DIRECT)
July 2016

23. A survey of sarcasm detection in social media.
Journal (IJRASET)
December 2017

24. Online Public Shaming on Twitter : Detection, Analysis and Mitigation
Rajesh Basak, Shamik Sural, Niloy Ganguly, and Soumya K. Ghosh
IEEE Transactions

25. http://www.thesarcasmdetector.com/about/

26. https://www.bbc.com/news/technology-27711109

27. https://www.paralleldots.com/sarcasm-detection

28. https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html#:~:text=Lemmatization%20usually%20refers%20to%20doing,is%20known%20as%20the%20lemma%20.

29. http://www.tartarus.org/~martin/PorterStemmer/

30. http://www.cs.waikato.ac.nz/~eibe/stemmers/

31. http://www.comp.lancs.ac.uk/computing/research/stemming/

32. https://expertsystem.com/machine-learning-definition/#:~:text=Machine%20learning%20is%20an%20application,use%20it%20learn%20for%20themselves.

33. https://en.wikipedia.org/wiki/Deep_learning

34. https://en.wikipedia.org/wiki/Statistical_classification

Appendix

```
ackage 'SnowballC' was built under R version 3.6.3
> installed.packages("SnowballC")
     Package LibPath Version Priority Depends Imports LinkingTo Suggests Enhances
     License License_is_FOSS License_restricts_use OS_type Archs MD5sum
     NeedsCompilation Built
> installed.packages("tm")
     Package LibPath Version Priority Depends Imports LinkingTo Suggests Enhances
     License License_is_FOSS License_restricts_use OS_type Archs MD5sum
     NeedsCompilation Built
> installed.packages("twitteR")
     Package LibPath Version Priority Depends Imports LinkingTo Suggests Enhances
     License License_is_FOSS License_restricts_use OS_type Archs MD5sum
     NeedsCompilation Built
> installed.packages("syuzhet")
     Package LibPath Version Priority Depends Imports LinkingTo Suggests Enhances
     License License_is_FOSS License_restricts_use OS_type Archs MD5sum
     NeedsCompilation Built
```

Figure 12 install packages

```
     NeedsCompilation Built
> library("SnowballC")
> library("tm")
Loading required package: NLP
Warning message:
package 'tm' was built under R version 3.6.2
> library("twitteR")
Warning message:
package 'twitteR' was built under R version 3.6.2
> library("syuzhet")
Error in library("syuzhet") : there is no package called 'syuzhet'
> library("tm")
> library("twitteR")
> library("syuzhet")
Error in library("syuzhet") : there is no package called 'syuzhet'
> install.packages("syuzhet")
```

Figure 11 pulling libraries

```
> tweets <- userTimeline("realDonaldTrump", n=200)
Error in get_oauth_sig() : OAuth has not been registered for this session
> consumer_key <- 'fDJjAivdwBzZ34xLYCDSAaSSO'
> consumer_secret <- '9uLBZ2xhBbXJjPhlDGEeN1MSkODsbhFSKwySyczTX91Li5UoTD
+ ''
+ '
Error: unexpected string constant in:
"''
'"
> consumer_secret <- '9uLBZ2xhBbXJjPhlDGEeN1MSkODsbhFSKwySyczTX91Li5UoTD'
> access_token <- '1222231309738627072-kqFWA3ijh063LmhlZ8sSSftDYnvdwL'
> access_secret <- 'T7TTqQOqEGhBywNec96ea2OiQphEBfuHcuXG1dw5NNVpq'
> setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)
[1] "Using direct authentication"
```

Figure 14 keys

```
[1] "Using direct authentication"
> tweets <- userTimeline("realDonaldTrump", n=200)
> n.tweet <- length(tweets)
> tweets.df <- twListToDF(tweets)
```

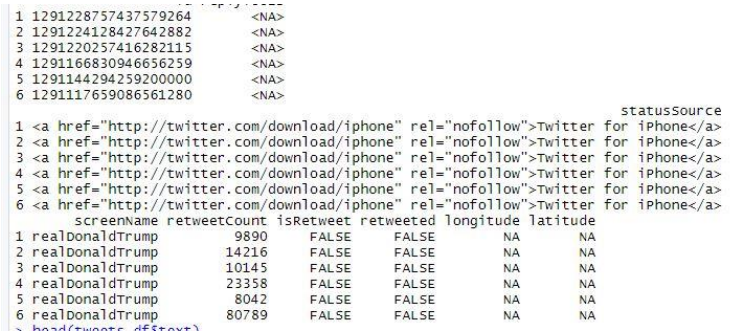
Figure 13 fetching tweets

```
[6] "DRAIN THE SWAMP! https://t.co/68M4SN7LLD
> tweets.df2 <- gsub("http.*","",tweets.df$text)
> tweets.df2 <- gsub("https.*","",tweets.df2
+ )
> tweets.df2 <- gsub("#.*","",tweets.df2)
> tweets.df2 <- gsub("@.*","",tweets.df2)
> head(tweets.df2)
[1] "Two wonderful people! "
[2] "Sally Yates is either lying or grossly incompetent. It is not possible she could have known so little about Dirty C… "
[3] "A very big CONGRATULATIONS to the great "
[4] "\"Cities across the Nation that are run by Democrats are in shambles.\" Matt Walsh  "
[5] "Congressman Scott DesJarlais ("
[6] "DRAIN THE SWAMP! "
```

Figure 16 making dataframe of tweets

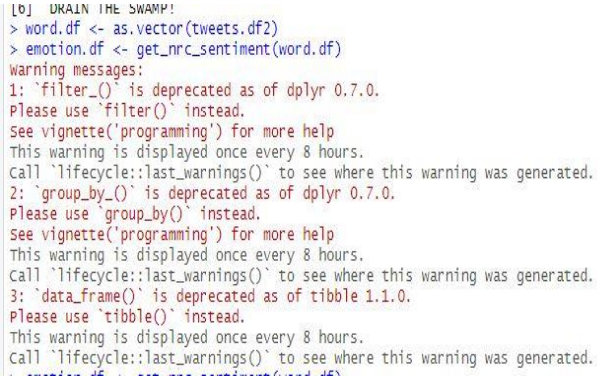
```
> tweets.df <- twListToDF(tweets)
> head(tweets.df)
                                                                                                                    text
1                                                                         Two wonderful people! https://t.co/Fo2CLjc9YZ
2 Sally Yates is either lying or grossly incompetent. It is not possible she could have known so little about Dirty C… https://t.co/TVUDMUOHQB
3 A very big CONGRATULATIONS to the great @seanhannity on having the Number One Book on the Planet, "Live Free Or Die… https://t.co/1FoYp4fJLn
4                                  "Cities across the Nation that are run by Democrats are in shambles." Matt Walsh  @TuckerCarlson
5 Congressman Scott DesJarlais (@tndesjarlais) has been a tremendous advocate for Tennessee! One of my earliest suppo… https://t.co/j1yFGxChTN
6                                                                        DRAIN THE SWAMP! https://t.co/68M4SN7LLD
  favorited favoriteCount replyToSN             created truncated replyToSID
1     FALSE         48011      <NA> 2020-08-06 04:25:00     FALSE       <NA>
2     FALSE         51557      <NA> 2020-08-06 04:06:36      TRUE       <NA>
3     FALSE         49649      <NA> 2020-08-06 03:51:13      TRUE       <NA>
4     FALSE        100725      <NA> 2020-08-06 00:18:55     FALSE       <NA>
5     FALSE         28449      <NA> 2020-08-05 22:49:22      TRUE       <NA>
6     FALSE        192497      <NA> 2020-08-05 21:03:32     FALSE       <NA>
                   id replyToUID
1 1291228757437579264       <NA>
2 1291224128427642882       <NA>
3 1291220257416282115       <NA>
4 1291166830946656259       <NA>
5 1291144294259200000       <NA>
6 1291117659086561280       <NA>
                                                                          statusSource
1 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
2 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
3 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
4 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
5 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
6 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
       screenName retweetCount isRetweet retweeted longitude latitude
1 realDonaldTrump         9890     FALSE     FALSE        NA       NA
2 realDonaldTrump        14216     FALSE     FALSE        NA       NA
3 realDonaldTrump        10145     FALSE     FALSE        NA       NA
4 realDonaldTrump        23358     FALSE     FALSE        NA       NA
5 realDonaldTrump         8042     FALSE     FALSE        NA       NA
6 realDonaldTrump        80789     FALSE     FALSE        NA       NA
```

Figure 15 tweets



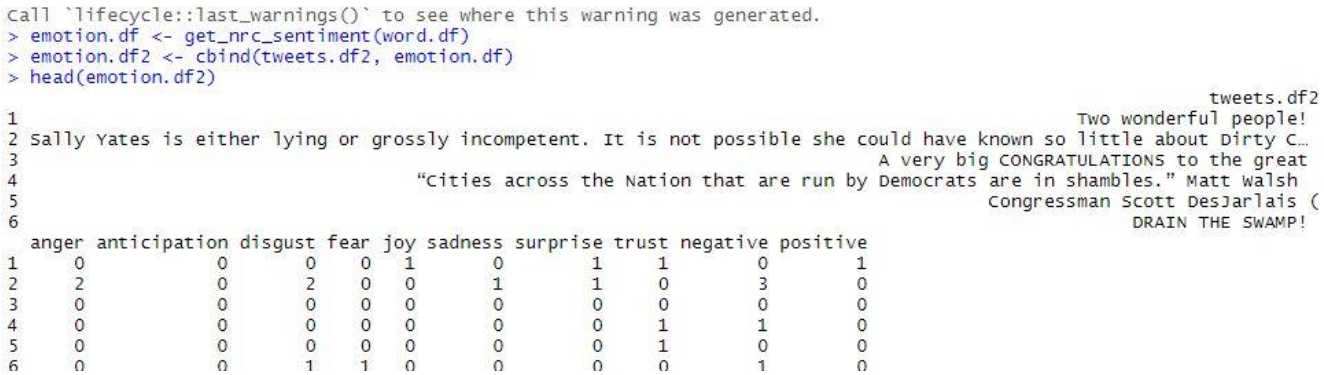
```
1 1291228757437579264        <NA>
2 1291224128427642882        <NA>
3 1291220257416282115        <NA>
4 1291166830946656259        <NA>
5 1291144294259200000        <NA>
6 129111765908656128         <NA>
                                                                    statusSource
1 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
2 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
3 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
4 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
5 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
6 <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
      screenName retweetCount isRetweet retweeted longitude latitude
1 realDonaldTrump         9890     FALSE     FALSE        NA       NA
2 realDonaldTrump        14216     FALSE     FALSE        NA       NA
3 realDonaldTrump        10145     FALSE     FALSE        NA       NA
4 realDonaldTrump        23358     FALSE     FALSE        NA       NA
5 realDonaldTrump         8042     FALSE     FALSE        NA       NA
6 realDonaldTrump        80789     FALSE     FALSE        NA       NA
```

Figure 19 Tweets



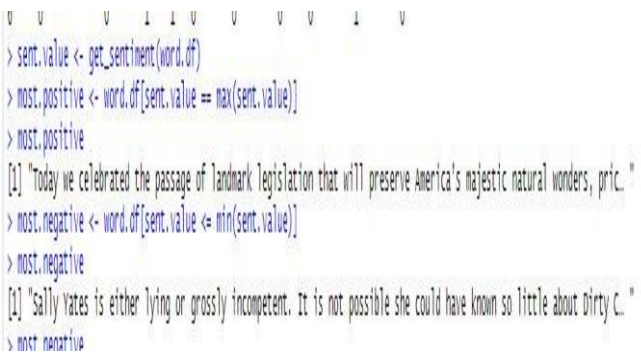
```
[6] "DRAIN THE SWAMP!"
> word.df <- as.vector(tweets.df2)
> emotion.df <- get_nrc_sentiment(word.df)
Warning messages:
1: `filter_()` is deprecated as of dplyr 0.7.0.
Please use `filter()` instead.
See vignette('programming') for more help
This warning is displayed once every 8 hours.
Call `lifecycle::last_warnings()` to see where this warning was generated.
2: `group_by_()` is deprecated as of dplyr 0.7.0.
Please use `group_by()` instead.
See vignette('programming') for more help
This warning is displayed once every 8 hours.
Call `lifecycle::last_warnings()` to see where this warning was generated.
3: `data_frame()` is deprecated as of tibble 1.1.0.
Please use `tibble()` instead.
This warning is displayed once every 8 hours.
Call `lifecycle::last_warnings()` to see where this warning was generated.
```

Figure 18 vector creation from dataframe



```
Call `lifecycle::last_warnings()` to see where this warning was generated.
> emotion.df <- get_nrc_sentiment(word.df)
> emotion.df2 <- cbind(tweets.df2, emotion.df)
> head(emotion.df2)
                                                                                tweets.df2
1                                                                          Two wonderful people!
2 Sally Yates is either lying or grossly incompetent. It is not possible she could have known so little about Dirty C…
3                                                                  A very big CONGRATULATIONS to the great
4                                  “Cities across the Nation that are run by Democrats are in shambles.” Matt Walsh
5                                                                  Congressman Scott DesJarlais (
6                                                                          DRAIN THE SWAMP!
  anger anticipation disgust fear joy sadness surprise trust negative positive
1     0            0       0    0   1       0        1     1        0        1
2     2            0       2    0   0       1        1     0        3        0
3     0            0       0    0   0       0        0     0        0        0
4     0            0       0    0   0       0        0     1        1        0
5     0            0       0    0   0       0        0     1        0        0
6     0            0       1    1   0       0        0     0        1        0
```

Figure 17 showing dataframe



```
> sent.value <- get_sentiment(word.df)
> most.positive <- word.df[sent.value == max(sent.value)]
> most.positive
[1] "Today we celebrated the passage of landmark legislation that will preserve America's majestic natural wonders, pric…"
> most.negative <- word.df[sent.value <= min(sent.value)]
> most.negative
[1] "Sally Yates is either lying or grossly incompetent. It is not possible she could have known so little about Dirty C…"
```

Figure 21 positive and negative

```
[40] -0.65  0.00 -2.75 -0.75  0.00  0.75  1.75 -1.20
> positive.tweets <- word.df[sent.value > 0]
> head(positive.tweets)
[1] "Two wonderful people! "
[2] "A very big CONGRATULATIONS to the great "
[3] "Sally Yates has zero credibility. She was a part of the greatest political crime of the Century, and ObamaBiden kne… "
[4] "MAKE AMERICA GREAT AGAIN!"
[5] "NASA was Closed &amp; Dead until I got it going again. Now it is the most vibrant place of its kind on the Planet...And… "
[6] "A great race run by Roger against a very tough and smart opponent. Roger loves Kansas and will represent it incredi… "
```

Figure 20 positive tweets

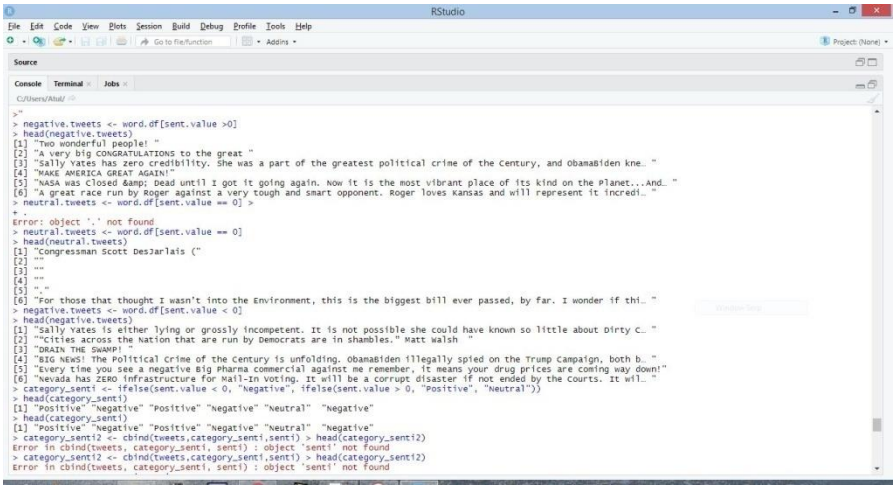


Figure 22 neutral tweets

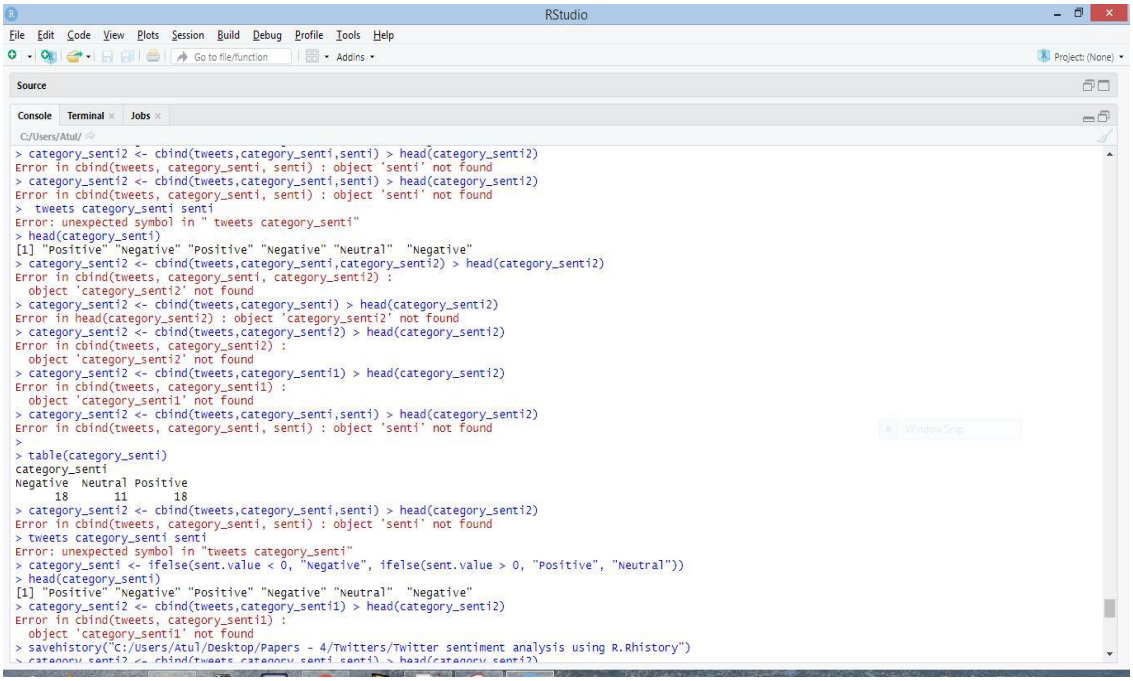


Figure 23 creating table

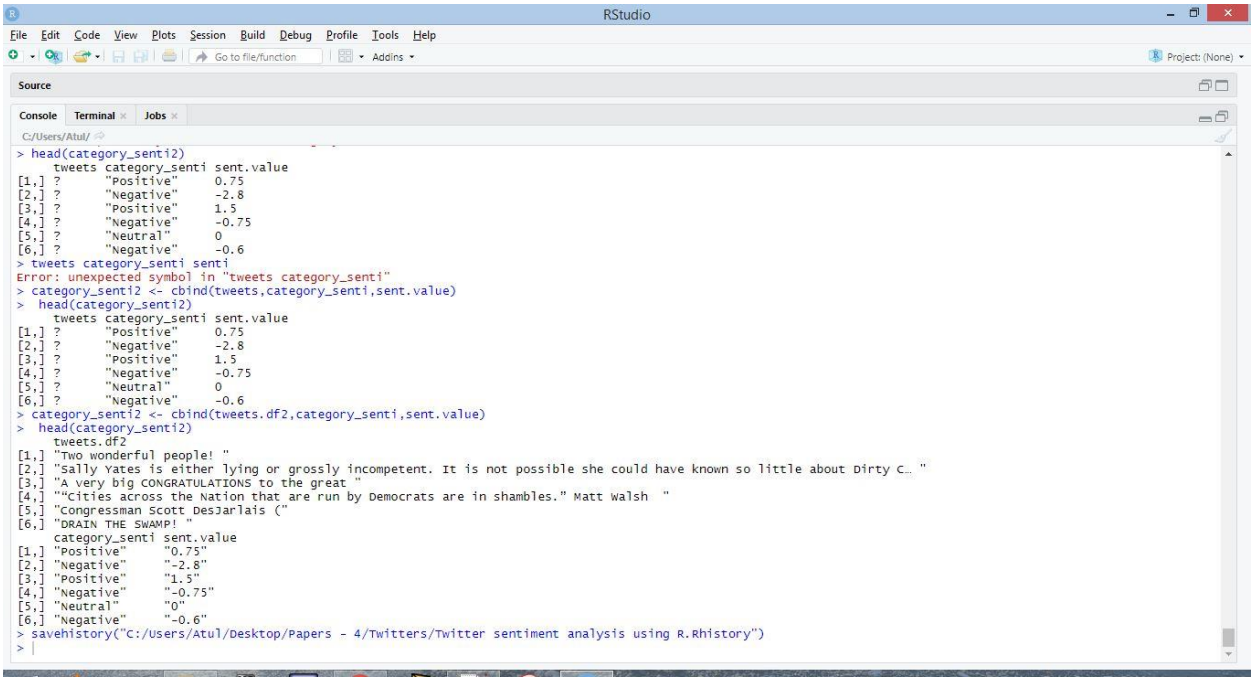


Figure 24 sentiment values

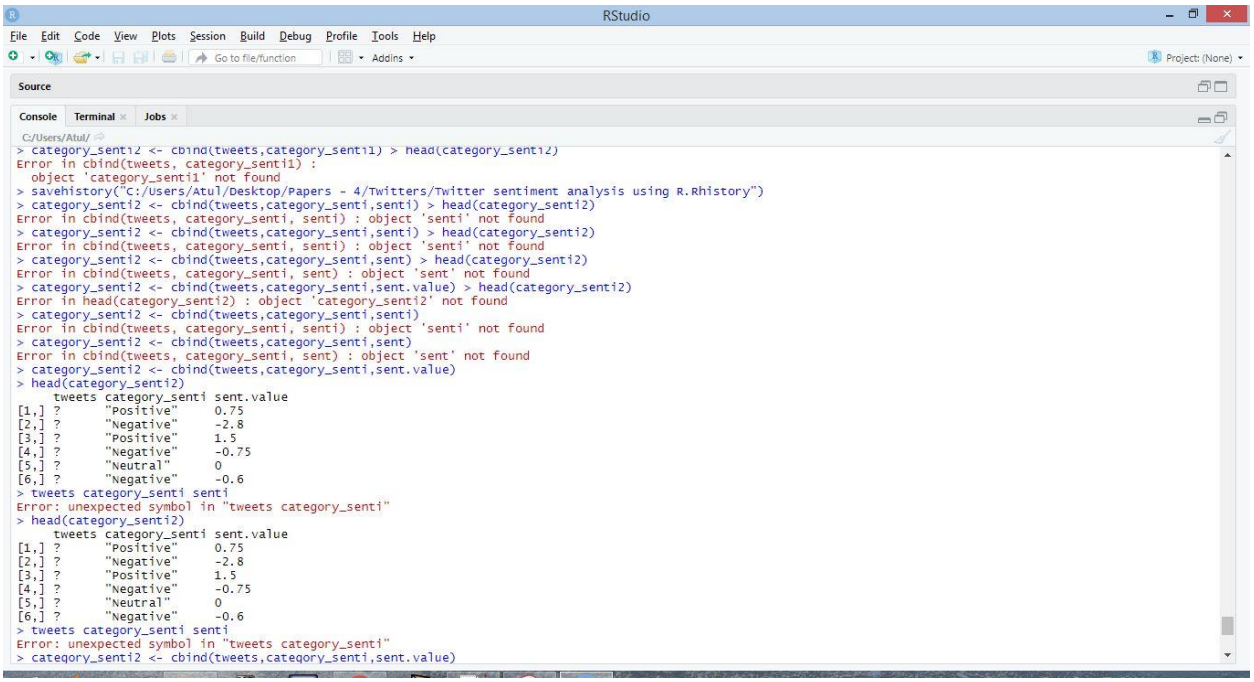


Figure 25 sentiment values