Sentiment Analysis for Sarcasm Detection on Streaming Short Text Data

Anukarsh G Prasad; Sanjana S; Skanda M Bhat
Sri Jayachamarajendra College of Engineering
Mysuru Karnataka India
anukarsh123@gmail.com; sanajana.sanj12@gmail.com; bhat.skanda.m@gmail.com

Dr. B S Harish
Sri Jayachamarajendra College of Engineering
Mysuru Karnataka India
bsharish80@gmail.com

Abstract— **The growth of social media has been exponential in the recent years. Immense amount of data is being put out onto the public domain through social media. This huge publicly available data can be used for research and a variety of applications. The objective of this paper is to counter problems with the social media dataset, namely : short text nature - the limited quantity of text data (140 to 160 characters), continuous streaming nature, usage of short forms and modern slangs and increasing use of sarcasm in messages and posts. Sarcastic tweets can mislead data mining activities and result in wrong classification. This paper compares various classification algorithms such as Random Forest, Gradient Boosting, Decision Tree, Adaptive Boost and Logistic Regression to detect sarcasm in tweets from the Twitter Streaming API. The best classifier is chosen and paired with various pre-processing and filtering techniques using emoji and slang dictionary mapping to provide the best possible accuracy. The emoji and slang dictionary being the novel idea introduced in this paper. The obtained results can be used as input to other research and applications.**

Keywords- Social Media dataset; Sentiment Analysis; Streaming data; Emoji and Slang Detection.

I. Introduction

The modern day world can be labelled as a data-driven world. With the invention of mobile devices and advancements in networking technology, there has been an exponential increase in data being generated by a single device in the network. The internet currently has over 3 billion [1] connected devices across the globe. With an enormous number of devices connected to the internet, the amount of data being exchanged is huge. Modern day humans are pioneers in communication; developed countries boast about over 60% of their population owning a smart mobile device connected to the internet. There has been a rise in social media and microblogging sites like Twitter and Facebook in the recent years. These social media websites have provided an open platform of communication in the modern era. Various social media sites have also claimed to have over 1 billion [2] online on a single day. Social media giant Twitter is said to cross over 500 million [3] tweets per day. The amount of data from such social media sites pose an interesting opportunity, it opens up a variant and irregular dataset with variety of information which is available publicly.

There are however, various other challenges that are posed by streaming social media data. The first and foremost is the nature of data; social media data can be referred to as "short text data". The data available is most often very few characters, which makes most text classification algorithms inefficient; as multiple keywords cannot often be derived from such data. Another challenge is posed by the composition of data itself. Recent internet culture has given rise to various slangs and short forms such as "LOL"(Laughing Out Loud) and "TTYL"(Talk To You Later) etc.,. Expression of any emotion is now made through emoticons or emojis. Data acquired from social media sites are often filled with slangs, hashtags and emoticons, thereby making traditional bag-of-words classification algorithms inefficient.

Considering twitter data, the use of sarcasm in tweets is ever increasing; a wide range of sarcastic tweets are being published every second. Such tweets bring down the accuracy of most of the classification algorithms. For example, a simple positive-or-negative tweet classification algorithm will map a tweet saying "I'm so pleased mom woke me up with vacuuming my room this morning. :) #sarcasm" [4] wrongly as a positive tweets because the bag-of-words approach suggests that the tweet is indeed positive. However to the human brain it clearly qualifies as a negative emotion conveyed very sarcastically. This has evolved into a major field of study in computer science and data analytics. Various research articles and publications have proposed a variant set of algorithms for sarcasm detection in twitter data. This paper is also one such effort which proposes a novel idea for enhancing accuracy in sarcasm detection algorithms by suggesting better preprocessing techniques such as emoji and slang dictionaries.

This area of research has multiple applications and use cases in the business world. From basic spam filtering to manufacture product market analysis, sarcasm detection has shown results which has aided in the betterment of understanding and classification of data.

The contents of the paper are divided into sections. The following section talks about the previous works and their shortcomings. Section 3 consists of the proposed system. Section 4 shows experimental results and the paper is concluded in the last section.

II. Literature survey

The earliest work on sentiment analysis on Twitter data can be traced back to works of Go et al.,(2009) [5] Their approach to sarcasm detection was to classify tweets as positive or negative using Naive Bayes, Maximum Entropy or Support Vector Machines algorithms for classification and unigrams, bigrams, unigrams and bigrams, and unigrams with part of speech tags as feature extractors. The paper proposed a classification model with emoticon detection and analysis for classification and also considered repetitive characters. The algorithm showed an accuracy around 80%. However, the paper didn't explore sarcasm detection, and narrowed down the classification into just positive and negative classes. The next step in this area of research would be making the classification real-time. The real-time nature of tweets have inspired many researches. One among them being the works of Bifet et al., (2011) [6] which uses the Twitter Streaming API to obtain tweets in real-time. The paper shows usage of MOA (Massive Online Analysis); it collects real-time data, uses a collection of algorithms and classifies the tweets into two classes, namely positive and negative tweets. The algorithm proposed uses a feature generation filter which uses a weighting scheme and then performs change detection to arrive at its results. The work, however, is again limited to just two class classification and does not venture into sarcasm detection. Moreover, it uses a predefined dataset for training. Another research which is along the same lines is by Bifet. A. and Frank E. (2010) [7] which again uses the twitter streaming API and a predefined training set to classify tweets. This work evaluates three algorithms namely Multinomial Naive Bayes, Stochastic Gradient Descent and Hoeffding Tree for classification and shows results of up to 82% on the best fit algorithm. However, the proposed work fails to perform sarcasm detection, but uses emoji dictionary to aid in classification.

The introduction to the field of sarcasm detection was mentioned in the works of Buscaldi et al., (2012) [8] which explains in detail what a sarcastic tweet is. The article on the features which make classification possible and give in-depth explanation regarding how the various features contribute to classification as well. There are many publication and research articles which have inspired our work, out of which a few are mentioned here in detail. The works of Barbieri F. and Saggion H., (2014) [9] deal with automated sarcasm detection in twitter data. The paper reports the results of classification of tweets into sarcastic or non-sarcastic classes by performing analysis based on Frequency (gap between rare and common words), Written-Spoken (written-spoken style uses), Intensity (intensity of adverbs and adjectives), Structure (length, punctuation, emoticons, links), Sentiments (gap between positive and negative terms), Synonyms (common vs. rare synonyms use) and Ambiguity (measure of possible ambiguities). The paper proposes an algorithm for classification based on those features mentioned above and claims an accuracy of 71% on irony detection and also mentions the improvements in Information gain. This paper was however bettered by many other who considered more features and better algorithms. The work of Erik F. and Niklas W. (2015)., [10] is the most recent advancement of the sarcasm detection and irony identification on twitter sentiment analysis. It uses existent algorithms such as SVM, Adaboost and Decision tree classifiers and performs Tokenization, Stemming and Lemmatization, Part-Of-Speech (POS)-tagging, Feature selection and then model evaluation. However, the paper does include emoji and slang dictionaries which would improve the accuracy. The proposed model however shows an accuracy of 71% for twitter data. The works of David B. and Noah A. S. (2016) [11] introduces improvements by including the history of tweets and author profiles which will aid in the classification process. The paper presents accuracies ranging from 70% and upwards for different scenarios.

Based on more stringent analysis and referral of various publications, the conclusion obtained was that the research until now, have passed by the fact that emojis and slangs play a huge role in tweets of the modern day. The new trend of tweets which extensively use slangs, short-forms and emojis are throwing present algorithms off-balance as they do not compensate for such changes in the trend of tweets. The paper derives inspiration from various mentioned publications and multiple others in building of the present classifier model. We propose various amends such as inclusion of the slang and emoji dictionaries for classification which may result in more accurate results, which can be observed by the results presented in this paper.

III. Proposed Methodology

The proposed methodology has three major stages : (A) Data pre-processing, (B) Data preparation and (C) Sarcasm detection.

A. Data Pre-processing

Data pre-processing is the most crucial step of the algorithm proposed in this paper. There are three steps in this stage,

namely a) Hashtag identification and replacement b) Slang dictionary mapping and c) emoji dictionary mapping.
The dataset supplied for the proposed model is a 2000 tweets dataset which contains general tweets with sarcastic or non-sarcastic labels. This dataset is a manually classified dataset which also serves as a new introduction in this paper. This collection of tweets consists of many things which need to be processed before moving onto the classification phase. The first step is hashtag identification and replacement. Hashtags are words or phrases preceded by a hash sign (#), used on social media websites and applications, especially Twitter, to identify messages about a specific topic. The hashtags in the dataset are plenty. Some like “#sarcasm” “#irony” etc., are useful for classification but things like “#lisbon” “#entertainment” are not particularly useful. These hashtags are treated as normal tokens and passed to Part of Speech tagging. The weighting is based on the POS tag assigned.

The next step in pre-processing is the emoji dictionary mapping. The popular trend of using emojis in tweets cannot be ignored as it carries a lot of weightage for classification. The emojis in a tweet are identified and then mapped with the manually built emoji dictionary introduced in this paper. The dictionary contains the popular emojis labelled as positive or negative. The same labels replace the emojis in a tweet during this step. Thus all the emojis are replaced by the labels similarly.

The last step is the slang dictionary mapping. This is one more key idea being introduced in this paper. The slang dictionary contains all the popular slangs and their meanings or full forms as key-value pairs. When a tweet is being analysed, if there is any slang that is detected it is immediately mapped to the dictionary and replaced by the appropriate meaning or full form. This will aid in the classification process later on.

B. Data preparation

After the pre-processing steps are completed, the data should be prepared and made ready for the classification phase. The tweets are a collection of sentences which cannot be directly fed into the classifiers. Hence, 5 major steps are performed to prepare the data for the next phase. The stages are (i) Word tokenisation (ii) POS(Parts-Of-Speech) tagging (iii)Stemming and lemmatization (iv) Feature identification and (v) New Representation Creation.

The first step is tokenisation. Tokenisation is performed on tweets to break them down into perfect meaningful modules from a sentence. Sometimes tokens can be in terms of paragraphs or whole sentences but for the proposed model it is a word. The tweet is broken down into words and the keywords which will aid in classification are chosen and stopwords are removed. After the tweets are tokenized, the Part Of Speech tagging is performed. The words in a tweet and their parts of speech play a role in classification. If the person is using a lot of adjectives there is a possibility that he is describing something with too much praise, that hints about it being sarcastic. With this in mind, the proposed model tags the parts of speech for each word. After the tagging is completed, Stemming and Lemmatization is performed. Stemming is built upon the idea that words with the same stem are close in meaning. So the words are stemmed to identify the words which are similar in meaning. After identification particular weights are assigned based on the meaning. Lemmatization is the process of identification of the root word of the various words used in the tweet. For example, words like mice are converted to mouse. Such conversion clarifies the context of usage for the word and makes it easier to map it with its meaning.

After all the necessary preparations are made for the feature identification. The major features considered are a) Blob Polarity b) Blob Subjectivity c) Capitalization d) Positive Sentiment and e) Negative sentiment. Apart from these, there are a few features which are self explanatory, like topic and subject. The first feature is the blob polarity, it refers to sentiment conveyed by the sentence as whole. It tells if the sentence is positive, negative or neutral. This feature is first applied for the whole sentence and then for first and second halves of the sentence. This is done so that if the first part of the sentence is positive and the second is negative, sarcasm is likely to be detected. Based on this a weight between 0 to 1 is assigned as the feature weight. Blob subjectivity is a feature which tells if the sentence actually conveys a sentiment or not and it it does how strongly is the sentiment emphasized. The subjectivity test is also performed for the whole sentence and then for the first and second halves to ensure the weighting is done appropriately. The next feature considered is capitalization. A tweet which contains a lot of capitalized words wants to make a strong point or convey a strong emotion. Thus this feature is set based on the number of capitalized occurrences in the tweet. The next two features are derived from the various tags assigned during the data pre-processing steps. The various emojis and slangs were given sentiment values such as positive and negative. Those tags are considered and the positivity and negativity features are weighted. The same is repeated for first and second halves of the sentence. The another important feature considered is the Parts Of Speech tags. The Part of speech is tagged based on an existent Natural Language ToolKit (NLTK) corpus and then weighted accordingly. Thus, in total 22 features are identified and used for classification.

After all the features are identified and weighted, a new representation is given by using all the features and their respective weights. This representation is fed as input to the classification algorithms for training and testing.

C. Sarcasm Detection

The sarcasm detection in the proposed model is done using classifiers such as Decision Tree [12], Random Forest [13], Gradient Boosting [14], Adaptive Boosting [15], Logistic Regression [16] and Gaussian Naive Bayes [17]. Decision Tree Classifier is a simple and widely used classification technique. Decision Tree Classifier poses a series of carefully crafted questions about the features that are supplied to the algorithm. Based on the answers received there are more questions are posed and ultimately class labels are assigned based on the cumulative answers. Random Forest Classifier works by building multiple decision trees and obtaining class labels. Gradient boosting generates learners during the learning process. It builds first learner to predict the values/labels of samples, and calculates the loss (the difference between the outcome of the first learner and the real value). It then builds a second learner to predict the loss after the first step. The step continues to learn the third, fourth and so on, until a certain threshold. Adaptive boosting requires users to specify a set of weak learners (alternatively, it will randomly generate a set of weak learners before the real learning process). It will learn the weights of how to add these learners to be a strong learner. The weight of each learner is learnt by whether it predicts a sample correctly or not. If a learner predicts a sample incorrectly, the weight of the learner is reduced a bit. The process is repeated until convergence. Naive bayes and logistic regression both are log-linear models; that is, in both cases the probability of a document belonging to a class is proportional to exp(w·x), where w is a classifier parameter and x is a feature vector for the document. The main difference is that, in naive bayes, the model is specified so that both the data and the labels depend on w, while in logistic regression only the labels depend on w.

Table 1: Results obtained by using different classifiers

	With Emoji and Slang Dictionary			**Without Emoji and Slang Dictionary**		
Classi-fiers	**60:40**	**70:30**	**80:20**	**60:40**	**70:30**	**80:20**
Random Forest	79.44	80.93	77.94	78.82	78.59	79.44
Gradient Boost	81.82	80.60	80.70	79.82	80.93	79.94
Decision Tree	76.06	73.74	73.43	71.05	75.58	71.92
Adaboost	79.82	78.09	81.70	76.06	79.76	80.20
Logistic Regression	40.25	38.33	40.29	32.03	39.89	41.03

These classifiers are given the newly formed dataset as the input. The various features and their weights are considered by the algorithm They are trained for various splits ranging from 60:40, 70:30 and 80:20 of training to testing data. The most accurate algorithm from the best performing split is selected for testing and the same is used for the real-time functioning of the model.

Algorithm 1. Proposed model

```
Data : Manually classified tweet dataset
Result : Binary classification into sarcastic and non-sarcastic
Initialise slangDictionary, emojiDictionary
Import dataset
step 1: Data Pre-processing
        Replace emoji in tweet with emojiDictionary value
        Replace slang in tweet with slangDictionary value
        Hashtag separation
step 2: Data preparation
        Word tokenisation of pre-processed
        Tagging the tokenized words with parts of speech (POS)
Stemming and Lemmatization
        Feature identification and weighting
        Update the dataset by adding the features and respective weights
step 3: Sarcasm extraction
        Classification of prepared data using 5 classifiers
        Choosing the classifier with best accuracy
        Training the model with best classifier
        Testing the model with best classifier
        Real time testing of tweets
```

IV. Experimental Setup

A. Dataset

Twitter produces millions of tweets per hour. It is a difficult task to procure a real-time dataset for training or even a topic specific dataset as it will lead to either too many or too less features for classification. In this paper, we have considered a dataset which is manually prepared, the tweets in the dataset are manually classified as sarcastic or non sarcastic based on human intuition which has staged a very accurate dataset for training. The manually classified dataset is one of the introduction in this paper. The dataset contains a collection of 2000 tweets which have class labels of 1 or 0, where 1 mean sarcastic and 0 means non sarcastic. The dataset taken is that of about 2000 pre-classified tweets. The dataset contains two columns, Tweet and Label. The

Tweet column contains the tweet, and the Label contains a binary label indicating whether the tweet is sarcastic or not.

B. Experimentation

The experimental setup consists of a unix system running a python virtual environment on python 2.7. The setup is on Jupyter, which is an open-source web application that allows us to create and share documents that contain live code, equations, visualizations and explanatory text.

The setup was used to test various splits of data whose results are presented below.

Table 1 shows the results of testing for a split of 60:40, 70:30 and 80:20 with and without the emoji and slang dictionaries. It can be seen that all the accuracies see an improvement when the dictionaries are used. The best accuracy is seen for a 60:40 split with 81.82% which also shows the improvement in accuracy with the novel ideas introduced in this paper. An improvement of 2% can be observed in this case. The highest improvement of accuracy can be is around 8% as seen in the 60:40 split for logistic regression classifier. The testing however is performed by randomly splitting the dataset each time. This will result in a different set on accuracies in any different runs performed. The accuracies also depend upon the strong and weak records. The strong records, when included in training will lead to better accuracies than others. This can be observed as 60:40 split shows better results than higher training splits.

C. Discussion

The paper presents comparative results of the five algorithms used. The accuracy of the results take a hit due to the amount of data considered for training. The dataset contains 2000 manually classified tweets. The training set has a maximum split ratio of 80%, that is 1600. Thus this factor brings down the accuracy of the algorithms as the dataset has lesser records than expected. The best performing algorithm is the Gradient Boosting algorithm which optimizes the cost function by considering the weak hypothesis which will make the classifier to classify wrongly. This aids in classification of the dataset as it continuously checks the features and modifies the classifier to avoid wrong classification. Random forest also shows good accuracy in some splits as it builds multiple decision trees for classification and divides the tweets based on each and every feature considered. This avoids the classification growing dependent on only a few important features. This is often called as feature bagging, which provide better results. The same technique which is absent in decision trees leads to low accuracy. The Adaptive boosting performs by assigning weights in training and correcting them through one single back iteration. This leads to adaptive boost performing well in certain random splits. This is due to certain strong data items which can sway the classification towards better accuracy, but this is only a corner case. The Logistic regression fails in this classification problem due to the absence of the any trends, as it bases its classification on trends. The dataset however when expanded can be used to procure better results.

V. Conclusion

In this paper, a way of improving the existent sarcasm detection algorithms by including better pre-processing and text mining techniques such as emoji and slang detection are presented . For classifying tweets as sarcastic and non-sarcastic there are various techniques used, many of which are briefed in section 2. However the paper takes up a classification algorithm and suggests various improvements which directly contribute to the improvement of accuracy. The project derived analytical views from a social media dataset i.e., twitter dataset and also filtered out or reverse analysed sarcastic tweets to achieve a comprehensive accuracy in the classification of the data that is presented. The model has been tested in real-time and can capture live streaming tweets by filtering through hashtags and then perform immediate classification.

References

[1] https://en.wikipedia.org/wiki/Global_Internet_usage#Internet_usersJ. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] https://www.facebook.com/zuck/posts/10102329188394581

[3] www.internetlivestats.com/twitter-statistics/

[4] Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva , Nathan Gilbert, Ruihong Huang. "Sarcasm as Contrast between a Positive Sentiment and Negative Situation" (2013)

[5] Go, A., Huang, L., Bhayani, R.: Twitter sentiment classification using distant supervision. In: CS224N Project Report, Stanford (2009)

[6] Bifet, A., Holmes, G., Pfahringer, ., Gavald`a., R. "Detecting Sentiment Change in Twitter Streaming Data", Workshop and Conference Proceedings 17 (2011) 5–11, 2nd Workshop on Applications of Pattern Analysis.

[7] Bifet A., and Frank E. Sentiment knowledge discovery in twitter streaming data. In Discovery Science, pages 1–15, 2010.

[8] Buscaldi, D., Rosso, P, Reyes, A., "From humor recognition to irony detection: The figurative language of social media", Data & Knowledge Engineering, April 2012

[9] Barbieri, F., and Saggion, H. 2014. "Automatic Detection of Irony and Humour in Twitter", In Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics, 56–64. Gothenburg, Sweden: Association for Computational Linguistics.

[10] Erik Forslid and Niklas Wikén., "Automatic irony- and sarcasm detection in Social media", ISSN: 1401-5757, UPTEC F15 045, 2015.

[11] David Bamman and Noah A. Smith., "Contextualized Sarcasm Detection on Twitter", School of Computer Science, Carnegie Mellon University (2016).

[12] http://www.ccs.miami.edu/~hishwaran/papers/decisionTree_intro_IR 2009_EMDM.pdf

[13] https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.ht m

[14] Mason, L.; Baxter, J.; Bartlett, P. L.; Frean, Marcus (1999). "Boosting Algorithms as Gradient Descent" In S.A. Solla and T.K. Leen and K. Müller. Advances in Neural Information Processing Systems 12. MIT Press. pp. 512–518.

[15] Freund, Yoav; Schapire, Robert E (1997). "A decision-theoretic generalization of on-line learning and an application to boosting". Journal of Computer and System Sciences. 55: 119.

[16] Hilbe, Joseph M. (2009). Logistic Regression Models. Chapman & Hall/CRC Press

[17] Gabriel Terejanu, Puneet Singla, Peter D. Scott and Tarunraj Singh "Adaptive Gaussian Sum Filter for Nonlinear Bayesian Estimation"