IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 7.39 ⁜ Vol. 11, Issue 9, September 2022

DOI: 10.17148/IJARCCE.2022.11922

Sarcasm Detection Model Building with Vector Visualisation

Adithya Reddy Nalla[1], Ruthvik Varma G[2], Mohammed Shuaib[3]

Department of Electronics and Communication, KL University, Vijayawada, India[1]

Department of Electronics and Communication, KL University, Vijayawada, India [2]

Department of Electronics and Communication, KL University, Vijayawada, India [3]

Abstract: Our work is building a sarcasm detection model which detects the sarcasm in a sentence that may be in news headlines or statements made on social media. There has been an exponential growth of social media in recent years. An immense amount of data is been put into the public domain through social media. To start this work first we have to know what Sarcasm is. When someone says or writes anything that is entirely unrelated to what they actually intended, they are expressing their feelings through sarcasm. Sarcasm is extremely difficult to spot because of how anonymous it is. The accuracy of the sentimental analysis can be increased by performing a perfect analysis and interpretation of sarcastic language. Understanding a person's attitude and opinions is the goal of the sentimental analysis. We attempted to describe the overall architecture of sarcasm, technique, and vector visualization of words using TensorFlow in this work.

Keywords: Sarcasm, Sentimental analysis, TensorFlow, Vector visualization, Social media.

I. INTRODUCTION

The internet currently has 3 billion devices connected to the internet and this is the reason for the high exponential increase in the data. This huge data is being used for research and other applications. There has been a rise in the number of users on social media like Twitter and Facebook and even microblogging is the other reason for the increase in the data. For time being let us have a look at the data of the social media giant which is Twitter, according to the latest information, approximately 500 million Tweets are sent each day on Twitter. The volume of data available on these sites is an opportunity because it creates a varied and erratic dataset with a wide range of publicly accessible information. Taking into account Twitter data, it is clear that more and more caustic tweets are being written and published every second. Every second, tweet of this nature are published. Most categorization algorithms become less accurate as a result of such tweets. A simple algorithm for categorizing tweets as positive or negative, for instance, might incorrectly classify a tweet like "I'm so glad mom woke me up with vacuuming my room this morning #sarcasm" as a positive one because the approach based on the negative connotations of words suggests that the tweet is in fact positive. It obviously qualifies, though, as a negative emotion that is being expressed extremely sarcastically to the human brain. Sentimental analysis's ability to be more effective is more strongly influenced by successful sarcasm analysis. Studies on sarcasm analysis have been conducted in the physical and behavioural sciences, and theories explaining how, when, and why sarcasm is communicated have been developed. Sarcasm detection techniques have been based on these theories. These days giant companies like Twitter, Meta, and Google are mainly working to develop this model in order to avoid commotion among the public. Think about the tweet "I'm so thrilled to squander my three hours on such a crappy movie!" posted on Twitter as an example. Despite the positive connotations of terms like "pleased," the tweet's overall mood is unfavorable. A method for summarizing movie reviews that do not use sarcasm detection would therefore interpret it as a favorable review. A system that can accurately and reliably identify humorous posts across a variety of online social networking platforms is thus now required.

The former research workings in the detection of sarcasm have generally used Rule-based methods as well as statistical methods via (i) Lexical Features and (ii) Pragmatic Features and (iii) the Existence of polarity shifts in sentiments, punctuations, etc. All the deep learning architectures seem to achieve interesting outcomes in Natural Language Processing tasks. A deep neural network (DNN) technique, learns the necessary features without human intervention, despite making use of handcrafted features. An attention function is used to augment the performance of deep neural network models. In this paper, we studied and compared the models available for sarcasm uncovering.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue x, Month 2022

DOI: 10.17148/IJARCCE.2022.11xx

II. IDENTIFIED PROBLEM

People often communicate with each other and that may result either in a positive or negative manner. The result of being both at a time is said to result in Sarcasm. A classification job for sarcasm detection can be developed. The issue was categorized as a binary problem. Humans can spot a sarcastic sentiment in the text and reason about what makes it so. In order to avoid misinterpretation of sarcasm in a literal statement, recognizing the sarcasm is the main task for Sarcastic remarks that frequently have an impact on the accuracy and validity of natural language processing models. For example, the sentence "So thrilled to be on call for work the entire weekend!" sounds like a sentence with positive exclamation and sentiment. However, this is because of sarcasm while the actual statement is meant to show negative and dull emotion. All social networking, microblogging, and e-commerce sites frequently use sarcasm. For reliable sentiment analysis and opinion mining, sarcasm detection is crucial.

III.PROPOSED METHODOLOGY

a. Tools and Software Used:

Programming Language: Python 3.8.0, Operating System: Windows 10, Text Editor: Google Colaboratory, TensorFlow, Numpy, JSON, Matplotlib, IO, Keras.

b. Approaching Methodology

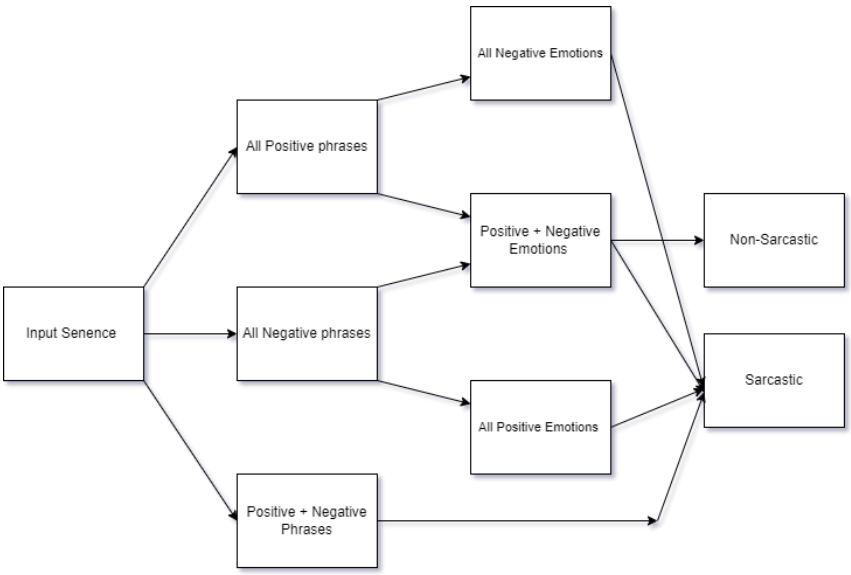

Figure 1 Design Implementation Diagram

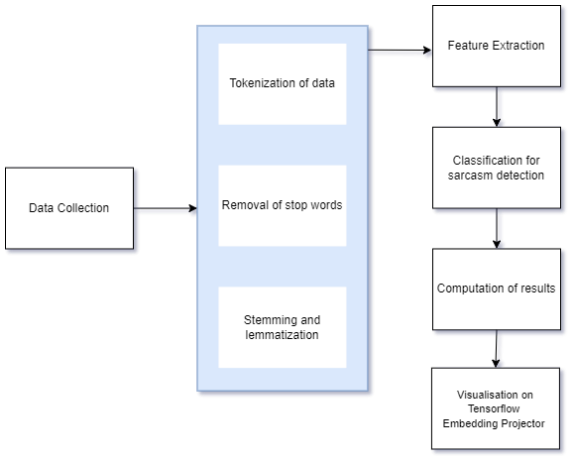

Figure 2 Flowchart Sarcasm Detection

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue x, Month 2022

DOI: 10.17148/IJARCCE.2022.11xx

This approach involves three major stages namely
(a) Data Acquisition (b) Data Pre-processing (c) Feature Extraction.

a. Data Acquisition and Analysis

To train the machine learning classifiers more than 25000+ headlines were collected from GitHub and SemEval Task11 datasets. The headlines are in English. The headlines are labeled as “is sarcastic” and “is not sarcastic”. The dataset is unstructured thus data cleaning and data pre-processing must be done.
Fig 1. and Fig 2. represent the design implementation of sarcasm detection and flow chat of our model which is sarcasm detection. The goal is to develop a machine-learning algorithm that will eventually be able to determine if a news headline is ironic or not. Since we don't require the article URL, we modify it to the organization that submitted the story instead because it can help us identify the source of both sarcastic and non-sarcastic news. Additionally, we can observe that there are no null values, demonstrating the completeness of the data. The dataset is divided into two groups, with sarcastic headlines being labeled as category 1 and non-sarcastic headlines as category 0. Nearly 0.44 is the mean value for this data column. This implies that the data isn't significantly distorted and is therefore ideal for our research. Then, we tokenize each headline (a function of the Python framework Keras) and compile the results into a dictionary. There are 29,656 distinct words in the text dictionary, it has been noted (taking all news headlines combined). We exclude the final 9,656 words since they are fewer in quantity and fall beyond the 20,000-word limit that our model (which was randomly started) allows.

b. Data Pre-processing

i. From the data collected links removed that were present.
ii. Stop words were removed.
iii. Padding of the sentences is done in order to obtain the same sentence length for the whole dataset.
iv. Lemmatization was performed to get the root word.

Clean-up was done on the gathered sentences. A dictionary of slang terms was used to standardize the use of slang terminology. e.g., ‘aweeesomeeee’ is standardized to ‘awesome’, and the use of regular expressions to eliminate punctuation and numerical information.

c. Feature Extraction

This is the step wherein we determine the overall accuracy of the project. It is difficult to detect features of sarcasm, as there are no It is difficult to detect features for sarcasm, as there are no specific rules or semantics to deliver sarcasm it changes from person to person. Sarcasm even depends on the situation and it is sometimes time-dependent. So its meaning may change with the situation and time. Sarcasm is also context-based most of the time, so a statement may be considered sarcasm for only those People who understand the background and could even mislead others who do not. It becomes highly challenging to manually or explicitly extract sarcasm-related information due to the contextual, situational, delivery, and time-based aspects of sarcasm. So we needed a technology that could implicitly extract features for sarcasm. Using neural networks solves this problem by implicitly extracting features. One more important benefit of neural networks is that it works on activation function, so the gravitas of activation function changes for each node, and in this way, it learns to detect sarcasm. When using neural networks training your neural network on a variety and feature-rich dataset becomes important.
The whole model is implemented on TensorFlow, a friendly machine learning library that can be easily programmed using the python programming language. The usage of two layers of recurrent neural networks, each made up of 256 LSTM cells, is another crucial component of our model. The calculations are increased, and the accuracy is impacted by the number of neural network layers. Therefore, choosing the right number of neural network layers is crucial.
The following significant element of the model is the creation of the Embedding Matrix. In essence, this is a matrix of vectors that contains the word embeddings for each unique word that the tokenizer selected in the previous step. Stanford's GloVe word The embedding matrix was constructed using word embeddings that had already undergone training. For every 20,000 words in this "txt" file (of 300 features depending on the download performed) word embeddings (each line consisting of one word and the appropriate word embedding) are used to generate the embedding matrix will be created. The resulting embedding matrix has the form that NumPy shape types claim they should have [20000,300].
The network architecture is directly addressed in the next section of the model. Before giving a quick overview of the network architecture, traverse through the embedding matrix starting with the first tokenized headline to collect the properties of each word in the sequence. Next, a convolutional layer, a max pooling layer, and finally a dropout layer are applied to this output. Before anything is put through the LSTM, the first two of these three serve the goal of doing a quick scan of all the words and examining correlations that are significant and irrelevant. Then, this output is sent through a dropout layer, a max pooling layer, and a convolutional layer. Prior to anything being sent through the LSTM, the first two of these three are used to provide a quick scan of all the words and look at correlations that are significant and inconsequential. A bidirectional LSTM receives this output and recognizes the headline's forward and backward

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue x, Month 2022

DOI: 10.17148/IJARCCE.2022.11xx

dependencies. Following flattening, each LSTM sequence is put through a typical neural network layer, and outputs a label, concluding the process like any other standard supervised learning task (what the problem state is after all). In such situations, the normal "binary cross-entropy loss" is utilized together with the Adam optimizer, which is often the first option. The performance was monitored using an accuracy metric.

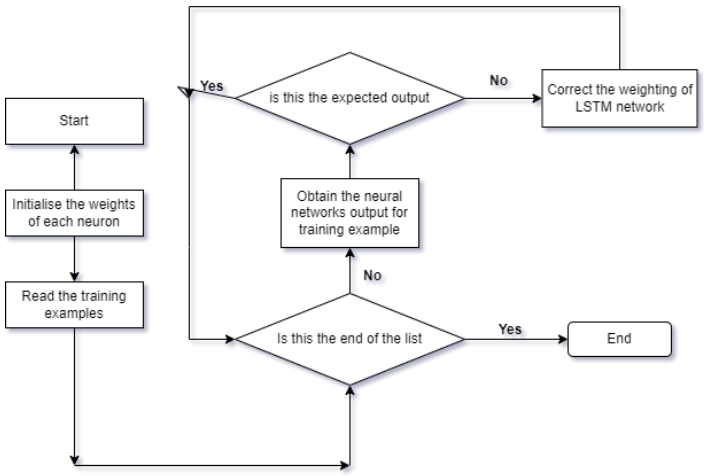

Figure 3 LSTM Flowchart

IV. PERFORMANCE EVALUATION AND RESULT

After evaluating the model on our test dataset after training it, we can see that we were able to obtain 86.52% accuracy with a loss of around 0.392. The following findings are obtained by plotting model accuracy vs model loss: Each dataset's model accuracy is shown in Figure VIL.1 for each epoch, and each dataset's model loss is shown in Figure VIL.2.

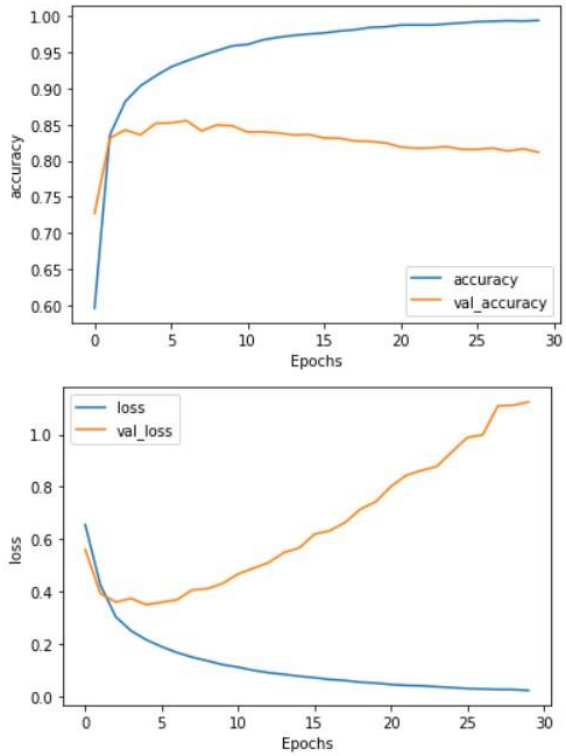

Figure 4 Graphs between Epochs, Model Loss, and Model Accuracy

We calculate the Accuracy, Precision, and F1 – Score metrics using the following formulae:

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue x, Month 2022

DOI: 10.17148/IJARCCE.2022.11xx

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

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

Where TP: True Positives; TN: True Negatives; FP: False Positives; FN: False Negatives. The confusion matrix was obtained after evaluating the parameters (labeling a sarcastic column as I and a non-sarcastic column as 0) is:

	Predicted 1	**Predicted 0**
Labeled 1	2358	358
Labeled 0	436	2572

Figure 5 Confusion Matrix evaluated on the test set of 5724 Data Points

From this table, we get the values of the following as:
Accuracy: 86.13%;
Precision: 0.844;
Recall: 0.868;
F1 Score: 0.856.

The Mandal and Mahto sarcasm detector has an accuracy rate of 86.16%. To identify sarcasm in news headlines, this detector changed into evolved the usage of Deep CNN-LSTM with phrase embeddings. Using split ratios of 50:50, 25:75, and 90:10, Ahuja, Bansal, Prakash, Venkataraman, and Banga implemented 12 classification algorithms. The gradient boosting approach provided the best accuracy in all three split scenarios, i.e., 85.14%, 85.71%, and 85.03%. Nota bene: The same dataset need not be used for testing and training the three models. The models are displayed on the X-axis and their accuracies are plotted on the Y-axis in the comparison of the several sarcasm detection methods that are provided below.
We can infer that the results with incorrect labels can be categorized under those labels in general. For instance, the HuffPost story "kitten courageously attempts high five" may have been classified as ironic by our algorithm. This inaccurate title may be attributable to the fact that HuffPost frequently publishes legitimate, but charming or humorous, items and is not necessarily a serious news source. This could make our model unclear. Similar to this, the Onion publishes articles with headlines that frequently appear serious but are actually satire. For instance, the Onion article "aerosol may shockingly frank about giving you cancer" might pass for a legitimate news headline. It is yet unknown why our approach is able to accurately categorize the majority of headlines despite this uncertainty.
Finally, based on PCA the data is being downloaded in the format of TSV which is integrated into the TensorFlow embedder which gives us the spherical view of words which is shown in 3D axes and separated as ‘is sarcastic’ on the left and ‘is not sarcastic’ on the right side.
The below is the visualisation of the data obtained from the trained model. The data can be downloaded in the csv file format in two categories i.e. meta and vector data. The vector data is responsible for the directional segregation of the data and this directional differentiation factor can be keenly observed when the data is spherized. The other data format is the one which consists of the words which have been categorized whether the data words are sarcastic or not.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue x, Month 2022

DOI: 10.17148/IJARCCE.2022.11xx

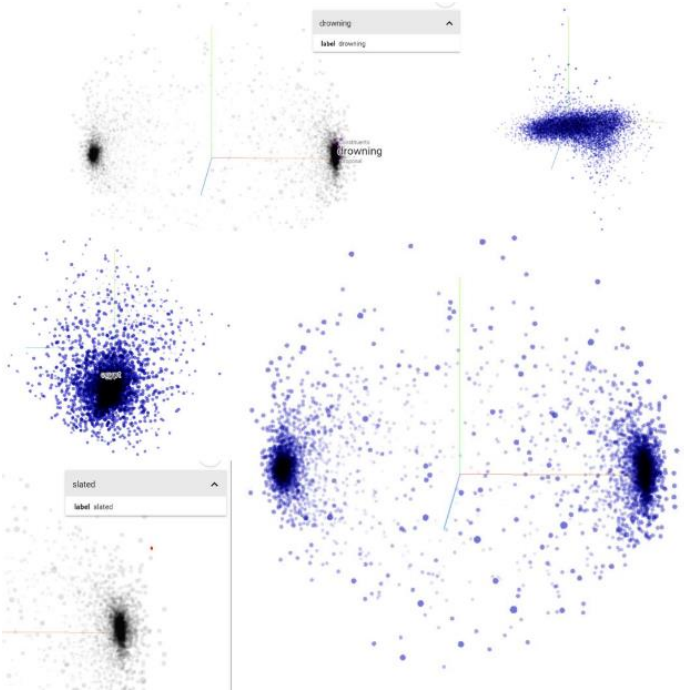

Figure 6 Trained Data Visualisation

V. FUTURE SCOPE

As we know that sarcasm is context-based it becomes difficult to interpret and predict. In areas where huge human-generated data is used like product reviews or company reviews, the sarcasm is misinterpreted by the traditional sentiment analysis tools and software. This results in inaccurate report generation, which might result in the company's reputation being damaged or fraudulent product review numbers. To prevent this, this model may be used as a filter to separate sarcastic from non-sarcastic remarks, giving sentiment analysis tools and software just non-sarcastic utterances. In the future, our focus will be on improving the f-score, accuracy, recall, and precision of the sarcasm detection model. Additionally, we'll strive to put more emphasis on the exaggeration feature and further examine the text's syntactic dependencies.

The sarcasm detection model can also be implemented as an application programming interface (API). API makes access to this model really easy by-passing sentences as input and returning the output as sarcasm or non-sarcasm. Future work might also include improving language models and investigating feature engineering with a conceptual foundation.

VI. CONCLUSION

The core of sarcasm may be observed in words that start off positively and then turn negative in the second part. In this research, a method for enhancing the current sarcasm detection algorithms is provided, which includes enhanced pre-processing and text mining methods like emoji and slang identification. There are several methods used to categorize tweets as sarcastic or not. However, the paper maintains the class set of rules and proposes numerous improvements, which immediately make a contribution to the development of accuracy.

The mission derives its analytical views from the SemEval Task11 dataset. As we've visible in this article, detecting irony is one of the major demanding situations in sentiment analysis. With new phrases and idioms constantly emerging due to the expansion and impact of social media platforms, it can still be difficult to recognize exact sarcasm. Sarcasm is difficult to identify when interjections are present. Additionally, the intricacy is increased by the new similes and acronyms that are acquired daily. Since certain new words are not being uploaded to Sentiwordnet, it is difficult to tell whether words truly express true feelings. Future research can successfully discern sarcasm by taking into account the POS location in phrases.

REFERENCES

[1]. Mandal, Paul & Mahto, Rakeshkumar. (2019). Deep CNN-LSTM with Word Embeddings for News Headline Sarcasm Detection. 10.1007/978-3-030 14070-0 69.

[2]. R. Ahuja, S. Bansal, S. Prakash, K. Venkataraman and A. Banga, "Comparative Study of Different Sarcasm Detection Algorithms Based On Behavioral Approach". [Online].

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue x, Month 2022

DOI: 10.17148/IJARCCE.2022.11xx

[3]. D. Maynard and M. Greenwood, "Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis" in Language Resources and Evaluation Conference (LREC), 2014.
[4]. Da Silva, Nadia FF, Eduardo R. Hruschka, and Estevam R. Hruschka. "Tweet sentiment analysis with classifier ensembles." Decision Support Systems 66 (2014): 170-179.
[5]. https://en.wikipedia.org/wiki/Global_Internet_usage#Internet_users
[6]. Maynard, D.G. and Greenwood, M.A., 2014, March. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In LREC 2014 Proceedings. ELR
[7]. Mondher Bouazizi, Tomoaki ohtsuki," Opinion Mining in twitter-How to make use of sarcasm to enhance sentiment analysis", IEEE/ACM International conference on Advances Social Networks and Mining (ASONAM), 25-28 August 2015, Paris, France.
[8]. https://monkeylearn.com/sentiment-analysis/
[9]. https://www.analyticsvidhya.com/blog/2017/01/ultimate-guide-to
[10]. Jihad aboobaker, E. Ilavarasan. "A Survey on Sarcasm detection and challenges", 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), 2020.
[11]. Adarsh MJ, Pushpa Ravikumar. "Sarcasm detection in Text Data to bring out genuine sentiments for Sentimental Analysis", 2019 1st International Conference on Advances in Information Technology (ICAIT), 2019
[12]. Nishan. A.H*, Dr. Joy Winnie Wise.D.C, Malaiarasan. S, Dr. Gopala Krishnan.C. "Sarcastic Detection of Twitter Comments using Python", International Journal of Innovative Technology and Exploring Engineering, 2020.

