Sarcasm Detection Using NLP And ML

PES UNIVERSITY EC CAMPUS Hosur Rd, Konappana Agrahara, Electronic City, Bengaluru, Karnataka-560100

Rithika Reddy Kara
Student, Department of CSE
PES University EC Campus
Bengaluru, India
18ritzy@gmail.com

Pawan Prasad P
Student, Department of CSE
PES University EC Campus
Bengaluru, India
pawanssj5@gmail.com

Phani Kumar Vedurumudi
Student, Department of CSE
PES University EC Campus
Bengaluru, India
pvphan.i890@gmail.com

Abstract—**Sarcasm is a kind of slang which is hard to decipher, no matter the language and the grammar. Irony refers to using the word that means it Contrary to what you actually say. It can be offensive, frustrating, or just fun. To understand the sarcasm through captions/text which are generated, the topics of Natural Language Processing and Deep Learning is required. This project aim is to find a more accurate and precise way of predicting whether the sentence is sarcastic or not using ML model and NLP . The Dataset for this project is a news headlines taken from two news websites from the Kaggle source then the Text Preprocessing is done and then feed to the model. This project contains the Embedding layer, GloVe Model and RNN(LSTM) Model which passes the captions and then receive the answer whether the caption/text is sarcastic or not. The GloVe is a Pre-trained model which contains all the vector representation of the words and we have particularly used GloVe twitter model. The LSTM model is used for finding whether the given text is sarcastic or not. This Model shows good results compared to other ML models like SVM, KNN, Naive Bayes and Random Forest. The overall accuracy or prediction rate of our model is more than 88%. This Project can be used further for exploration in the field of Sarcasm.**

Index Terms—**Keywords: Kaggle, Text Preprocessing, Embedding layer, LSTM, GloVe, SVM, KNN, Naive Bayes, Random Forest, Prediction rate.**

I. Introduction

Sarcasm/Irony detection is a very narrow research area of NLP. Specific cases of sentiment analysis. The overall emotion and focus is ironic. Therefore, the role of this field is to detect if a particular text is irony or not. Given a text as input, the output should be whether the given the input is sarcastic or not. This Project consists of analyzing the captions/text retrieved from the dataset from the Kaggle. The Dataset from the Kaggle source consists of news headlines scraped from the internet from two news website such as *TheOnion* and *HuffPost*/. The Dataset consist of three attributes such as **article link**, **headlines** and also target column **is_sarcastic** stating whether the particular news headline is saracastic or not. The Dataset contains the tweets in the form of headlines from the news websites and the dataset requires less data cleaning compared to the twitter dataset which is used in the past analysis for sarcasm detection from the text.

Text Preprocessing is done using NLP techniques for easy processing of the data/text. The Preprocessing consists of the sentences being removed with the emojis and also use Tokenization to convert all the forms of the words into a workable format. The GloVe Pre-trained Model is utilized in the project. To understand the relationships with the words with each other, the GloVe Model is utilized. Long Short Term Memory is a model of recurrent neural network that is utilized in the project.

This Project focuses on the Sentiment Detection of the people through the text. It is one of the fields in the Sentiment Analysis. Understanding is very important when two people are having a conversation. When the person is having a conversation with another person, and they are sarcastic without understanding the context, it gets quite difficult to analyze the spoken word is sarcastic or not. The main aim of this project if the provided input of text is sarcastic or not.

II. Related Work

The study from the previous journal papers regarding this topic shows that authors have used various kinds of models to understand the sarcasm hidden in the text. As known that it is not easy to detect sarcasm from the text, we need proper context and situation on whether the given text is sarcastic or not.

The paper **Sarcasm Detection of Online Comments Using Emotion Detection** by S. Rendalkar and C. Chandankhede shows the usage of Emotion detection for finding the Sarcastic Comments. The Objective of this paper is to find the Sentiment scores which are used to detect emotions, after filtering out the data for non-English words. After the sentiment scores, word based and emotion based sarcasm detection methodologies are used to detect the sarcasm. Hybrid Sarcasm method is also utilized. Prepossessing is done by fetching of data, parsing using POS Tagger and feature extraction. Emotion and sentiment identification is done by finding the emotions based on words and emoticons. Keeps sentiment scores and emotion of emoticons. After that the Sarcasm is detected from the Online Comments.

Using a single metric to evaluate and search for sarcasm won't do. Multiple approaches need to be taken, as sarcasm is vast and has many forms. The data is obtained from various social media websites as well as public posts using GRAPH API and PBGLA model is utilized. The hybrid combination gives out much better result rather than going with a single approach. Recall, precision, f-Measure and accuracy were used for measuring the results of the model performance.

This paper contains lot of methodologies to detect sarcasm. To have to choose a hybrid method out of all those combinations to give out the best result is tough. The Sarcasm detection is done through word based detection, emotion based detection, hybrid sarcasm detection, pattern and text match based, IWS-Interjection Word Start. They are so many ways to detect the sarcasm and finding the best one is difficult.

The paper **Sarcasm detection of non # tagged statements using MLP-BP** by P. Gidhe and L. Ragha. In this paper the Reddit dataset was used on the project. The Back propagation Neural Networks and MLP-BP classifiers were implemented in the project. Only the sentences which are not tagged with #sarcasm were implemented by the model. To decipher sarcasm through sentiment emotions and semantic similarity through feature extractions. Sigmoid function and Purelin activation functions were used in the hidden layer. Purelin activation function is the last function and after that, it outputs the sentence whether it is sarcastic or not.

The Reddit Dataset goes through the tokenization process. Sentences which are structural and semantic are extracted(Feature Extraction). The pleasantness, imagery, and activation values are extracted from DAL dictionary and sent to classifier. Pointwise Mutual Information (PMI) and Similarity Index are the metrics used to for evaluation. Cosine similarity index.

Sarcasm is not only in the sentence structures but in the emotions. To interpret sarcasm through indirectly portrayed emotions is very tough.

The paper **Automatic Sarcasm detection using feature selection** by Paras Dharwal, Tanupriya Choudhury, Rajat Mittal, Praveen Kumar. The objective of this paper is to automatically detect sarcasm in a given piece of text by building a model through the application of the techniques such as preprocessing, feature engineering,feature selection and training these features on to some classification algorithms such as Logistic Regression,Support Vector Machines and Naive Bayes Classifier. These methods are highly useful when we have more texts from a particular category. E.g., we have more positive texts than negative, and then this is useful.

Explanation:

- Preprocessing includes removal of unwanted symbols,characters,stop words etc from the given piece of text.
- Feature engineering constitutes the following processes of tokenization, stemming, lemmatization and n-grams,in addition to feature extraction.
- Feature selection constitutes selecting the important features from the extracted features by using using the techniques such as TF-IDF and chi-square analysis

Results:

- The n-grams model has given an F-score of around 0.56
- sentiment features have given an F-score around 0.41

Naïve Bayes method does not lead to higher precision, and the chances of poor results are quite high in this,the main issue with SVM is both the speed and the size (both in training and testing), Logistic regression can provide excellent results but not as efficient as SVM.

The paper **Statistical approach to sarcasm detection using Twitter data** by Rahul Gupta, Jitendra Kumar, Harsh Agrawal, Kunal. The objective of this paper is to detect sarcasm in a piece of text using statistical and NLP techniques. The methods use are text preprocessing,feature extraction,feature selection and training of Classification Algorithms such as KNN, Random forest, SVM classifier, Decision Tree voting classifier.

Explanation:

1) Data cleaning and preprocessing includes:-

- removal of Email-ID's
- removal of the links present.
- removing the stop words
- lemmatization to get the root word.

2) Feature extraction includes:-

- extracting the sentimental features such as positive word count,negative word count,count of words having high emotional positive and negative content.
- extracting the punctuation related features such as count of capital letters,count of punctuation symbols such as question and exclamation marks.

3) Selection of important features using chi-square analysis.

Results:

The voting classifier gives the best accuracy of around 83.53% as compared to the other classification algorithms

The techniques used for sarcasm detection in this paper are limited to the scope of only twitter comments/posts.

The paper **Sarcasm Detection on Indonesian Twitter Feeds** by D. A. P. Rahayu, S. Kuntur and N. Hayatin. Two different weighting methods and two different classification algorithms like Naive Bayes and KNN were used by combining punctuation and interjection feature extraction methods of the sentences. Preprocessing and stemming algorithms are used instead of translation of sentences to reduce the error count. The dataset is split into 70-30 form. The objective is to find the sentiment of the Indonesian tweet.

Explanation:

- The data set is in the form of crawled tweets which is preprocessed with stemming, case Folding and filtering and a bag of words, and fed into the Naive Bayes Model to obtain the positive tweets. From those positive tweets, again feature extraction is done with interjection and fed to classification using Naive Bayes and KNN for obtaining sarcastic tweets.
- The accuracy of the KNN model is nearly 82%.

KNN method does not lead to higher precision, and the chances of poor results are quite high in this, the values of the classification report is not constant.

The paper **Machine Learning based Sarcasm Detection on Twitter Data** by N. Pawar and S. Bhingarkar. The SVM, KNN and Random Forests are used as the classification of the models. In this project, accuracy, precision, and recall are the metrics which were used to evaluate the quality. The objective is to find the tweets is sarcastic or not based on this classification models.

The dataset is preprocessed by using document which is split based on whitespaces and punctuation and the unnecessary punctuation are removed. The Vocabulary is build and then encoded then the feature extraction is done. They have split the data and passed into various classification models like SVM, Random Forest and KNN and output the result.

Results: Random Forest Model - 81% SVM Model - 74% KNN Model - 58%

The KNN and SVM model are not time efficient, the main issue with SVM is both the speed and the size (both in training and testing).

The best and important papers above were considered as the literature survey for this project, and to dig deeper into solving the problem of identifying whether the input is sarcastic or not.

III. Motivation

- The sentiment analysis is used to identify the sentiments of a person.
- To detect sarcasm is to detect any flaws in the sentence which the person actually doesn't mean.
- Sarcasm is very dependent and highly contextual.
- The novelty is to use various pre-trained model and deep learning techniques to increase the accuracy of this model.

IV. Proposed Methodology

A. Dataset Selection and Required Libraries

The Dataset **News Headlines Dataset For Sarcasm Detection** for this project is collected from a well known source named Kaggle. This Dataset consists of news headlines scraped from the internet from two news website such as *TheOnion* and *HuffPost/*. The Dataset consist of three attributes such as **article link**, **headlines** and also target column **is_sarcastic** stating whether the particular news headline is saracastic or not. The Dataset contains the tweets in the form of headlines from the news websites and the dataset requires less data cleaning compared to the twitter dataset which is used in the past analysis for sarcasm detection from the text. Usually the dataset for the project of sarcastic detection consists of twitter scraped comments. That introduced a lot of noise and a lot to preprocess. Taking it a little bit to the next level, the dataset of news headlines is scraped for the implementation. Obtained from Kaggle platform, the News Headlines Dataset For Sarcasm Detection is a high quality data set for project. The headlines consist of sarcasm as well, improving the scope of finding the sarcasm in the world of non-fiction from a formal environment.

The prepossessing of the is the next step. The emoticons, symbols and pictographs, transport and map symbols are all are substituted with a space of text. To make only the words remain. The grammer is taken into account, with all the short forms and various ways to address a common word, they are all replaced with common word.

This project uses various Python libraries which are utilize for various different task in the model such as

- numpy
- pandas
- matplotlib (for Data Visualization)
- re (Regular Expression for Data Cleaning)
- tensorflow and keras (for ML Models)
- nltk (for nlp related libraries)
- wordcloud (for generating the visualization of sarcastic word)

B. High Level Design

The High Level Design for this project is shown below.

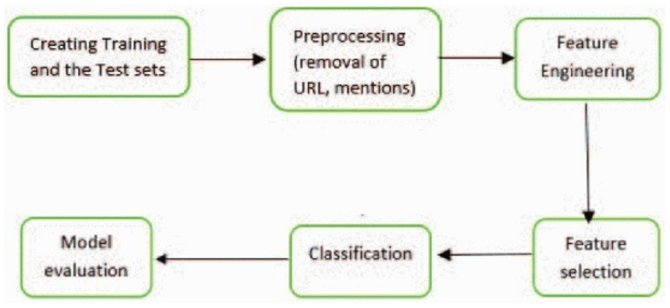
Creating Training and the Test sets
Preprocessing (removal of URL, mentions)
Feature Engineering
Feature selection
Classification
Model evaluation

Fig. 1. High Level Design

C. Text Pre-processing

The data is loaded from the data set using pandas and the text pre-processing is done using various libraries function. The article link attribute from the data set is dropped because the links doesn't provide much information for training the model. The data set is cleaned by using the regular expression library. The Tokenization is performed which is the crucial phase of pre-processing. The removal of the punctuation marks, non-alphabetical characters, and stop words from the headlines. The stop words are words that are common in the text and occur frequently. The stop words are taken from nltk corpus. The headlines need to be clean enough to be analyzed without the above interference. The conversion of all the characters into lower case is done. After the headlines are cleaned and tokenized word are generated then it goes to the embedding layer and GloVe Model for converting the word into its Vector representation.

D. Train-Test Split

In this section the tokens generated after text preprocessing is split into train-test split based on the user requirement. This project have used the 80-20 split in which 80% is used for training the model and other 20% is used for testing. In this

Project the aim is to find whether the whole sentence is sarcastic or not. So, the tokens are formed in sequence by removing all the stopwords which is done in text preprocessing. This uses the concept of padding particularly the post padding is used and then the whole dataset into the training and testing split using the tensorflow functions. It consists of the sarcastic class whether 0 or 1. If 1 then the given text is sarcastic or else 0 if it is not sarcastic. The sarcastic class is the target variable in this project. Then next step is to convert the given word into its word representation using the GloVe model. Then the input layer which is nothing but embedding layer. A new variable is introduced. The max_length. The amount of words required to be in each headline, which would give a good accuracy. For this particular problem statement, max_length=25. It indicates that each headline of 25 words will produce a very good accuracy in identifying whether the headline is sarcastic or not and it is tunable parameter. Next step, keras tokenizers are passed to help make tensors. With the tokenizers, called, the headlines are fed into it. The output will be sequences. The basic approach is to make the words in the dataset into numbers. To models, everything accounts to be numbers. For that to occur, we need sequences, The conversion of words to numbers takes place through the help of vectors. Each sentence is represented as sequences of vectors. The word index needs to be computed in order to compute the vocabulary size. The vocab_size is the variable associated with vocab_size. The word index is the number of unique tokens which are available in the input. The vocab_size=word_index+1. Why is vocab_size one greater than the word index? The count of Word_index only contains the corresponding known words. Vocab_size needs to have a count of 1 unknown word as well. The addition of one represents the unknown word. To pass the headlines of words under 25 in a sentence, padding is required. To hold the binary values of whether the the word is sarcastic or not, the variable sentiment is utilized. The next step is to remove the biases,and the simplest way to do is to remove them is by shuffling. The next part is to have the X_train_pad and y_train, x_test_pad and y_test variables initialized to make the tensors. The target variable is equal to the sentiments whether it is sarcastic or not and it is in the form of tensors.

E. GloVe Model and Embedding Layer

The GloVe Model is a vector representation of the words. This project requires it because the Machine Learning model works only on the numeric data not on the text data. The GloVe(Global Vectors) is a way of representing words in the form of vectors. GloVe (Global Vectors) is an unsupervised learning algorithm. To get a vector representation of a word. Training is Aggregate global words Perform with word match Representation of statistics and results from the corpus to display interesting linear substructures of word vectors space. This Project has utilized the twitter GloVe model which consists of 2 Billion tweets, 27 Billion tokens, 1.2 Million vocabulary and also 25 dimension, 50 dimension, 100 dimension and 200 dimension vectors representation of this 27 Billion tokens. Embedding Layer is used as the input to Deep Learning Model which converts the text into its vector form. In this project the text which is converted into tokens in previous step of text preprocessing using the GloVe Model.Data Visualization. The embedding matrix needs to created. The size of the embedding matrix is equal to the number of vector dimensions. The matrix's element is a 1, if the word is constrained in the vector representation. If not, then a 0 is placed.

F. Data Visualization

To interpret the frequency and the importance of the words, visualization is required. The Dataset is visualized by using the wordcloud library. In this project the most occuring word used in the sarcastic class is identified so that visualization of most frequently occuring word can be seen by using the wordcloud function. The higher the word used in the sarcastic the huge the size is seen in the (fig.2)

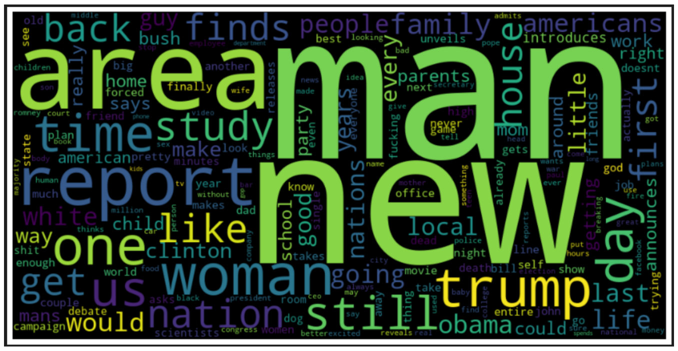

Fig. 2. Visualizing the most occuring word in sarcastic class

G. Model Building

In this project the type of RNN Model is used particularly **LSTM**(Long Short Term Memory) Model. This Model contains the Cell State which remember the previous state of the model and stores the information. The keras library is used to build a Sequential model. The sequential model is used, along with the embedding layer being the first which is a input layer to the LSTM Model. After, the LSTM's cell are added. For the particular model 64 LSTM's are added. The dropout and the recurrent dropout values are the hyper parameters to tune the model to obtain higher accuracy, and they are assigned with a value of 0.2. Adding a dense layer with the sigmoid function as the activation function because of the binary value (binary classification) of the target variable. The model uses the loss function as **binary cross entropy** and the best optimizer **Adam**. The model uses the accuracy metric to calculate the performance of the model. The accuracy of both the training and testing is done. The loss function for this particular use case is binary cross entropy, as the task is of binary classification. The formula of the binary cross entropy :

$$H(P,Q) = -sumxinXP(x) * log(Q(x)) \qquad (1)$$

where, P and Q is the probabilities of the events.

H. Experimental Results

The results of the experiment in this project is based on the accuracy of the model on both the training and testing data. The accuracy of the training data is 92% which is the highest and testing shows 91% accuracy. The loss and accuracy of the model can be visualized using the graph plotted for the training data and testing(validation) to the no. of epochs(iteration). The validation loss is nothing but testing loss. The Accuracy and Loss of the Model is shown for the model in fig.3 and fig.4 respectively.

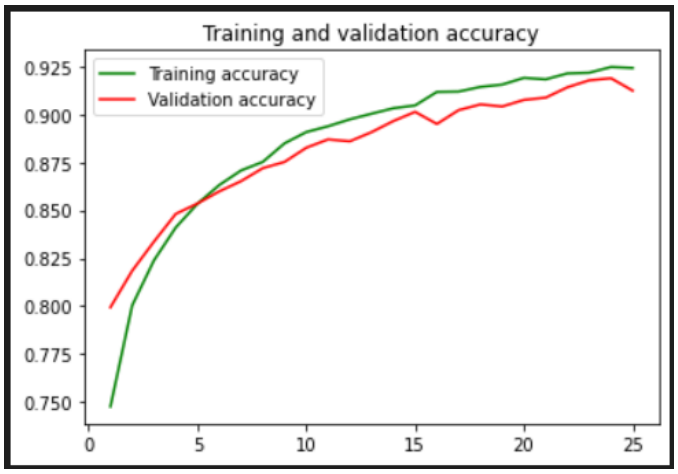

Fig. 3. Training and Validation Accuracy

The Accuracy of the model on both Training and testing is shown in the above figure.

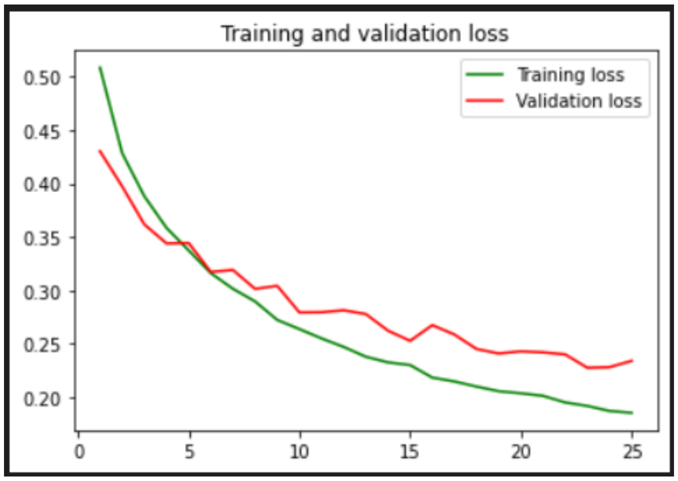

Fig. 4. Training and Validation Loss

The Loss of the model on both Training and testing is shown in the above figure.

CONCLUSION

This Project is used to determine whether the given text or sentence is sarcastic or not. The dataset used in this project is high quality dataset for sarcasm detection. A basic detector which is used to find whether a text is sarcastic or not based on the sarcastic word present in the text. The two main models used in this project are GloVe Model and LSTM Model for sarcasm detection. The project aim is to find a more accurate and precise way of predicting whether the sentence is sarcastic or not using ML model. The main aim is to increase the accuracy rate in the prediction. The overall accuracy or prediction rate of our model is more than 88%. The further study/research on the sarcasm detection can be done by utilizing the large dataset for sarcasm detection available online to find various sarcastic text by using the various Deep learning model and NLP techniques.

REFERENCES

[1] S. Rendalkar and C. Chandankhede, ”Sarcasm Detection of Online Comments Using Emotion Detection,” 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), 2018.

[2] P. Gidhe and L. Ragha, ”Sarcasm detection of non # tagged statements using MLP-BP,” 2017 International Conference on Advances in Computing, Communication and Control (ICAC3), 2017.

[3] B. Venkatesh and H. N. Vishwas, ”Real Time Sarcasm Detection on Twitter using Ensemble Methods,” 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA), 2021.

[4] R. Gupta, J. Kumar, H. Agrawal and Kunal, ”A Statistical Approach for Sarcasm Detection Using Twitter Data,” 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), 2020.

[5] D. A. P. Rahayu, S. Kuntur and N. Hayatin, ”Sarcasm Detection on Indonesian Twitter Feeds,” 2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), 2018.

[6] N. Pawar and S. Bhingarkar, ”Machine Learning based Sarcasm Detection on Twitter Data,” 2020 5th International Conference on Communication and Electronics Systems (ICCES), 2020.