National College of Ireland

Paragraph Vector based Sarcasm Detection in Text

MSc Research Project
Data Analytics

Darshan Diktekoppa Thimmappa
Student ID: x17160936

School of Computing
National College of Ireland

Supervisor: Prof. Noel Cosgrave

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Darshan Diktekoppa Thimmappa
Student ID:	x17160936
Programme:	Data Analytics
Year:	2019
Module:	MSc Research Project
Supervisor:	Prof. Noel Cosgrave
Submission Due Date:	12/08/2019
Project Title:	Paragraph Vector based Sarcasm Detection in Text
Word Count:	6750
Page Count:	18

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	9th August 2019

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Paragraph Vector based Sarcasm Detection in Text

Darshan Diktekoppa Thimmappa
x17160936

Abstract

Sarcasm is one of the never ending challenge in natural language processing and hinders the process of obtaining the true opinion of the people. It is an escape for the people who don't want to share their true opinion. Often it is used to tease others when people like or dislike something. Sentiment analysis will be incomplete without sarcasm detection. Many researchers worked on this problem using different ways like, non machine learning, machine learning and deep learning based techniques using two ways of feature engineering that is manual feature engineering or by using word embedding. In this research a novel adoption of two models of paragraph vectors that is Distributed Bag-of-Words based paragraph vectors (PV-DBOW) and Distributed Memory based paragraph vectors (PV-DM) for sarcasm detection will be made over previously followed techniques like Bag-of-Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF) and Word2Vec considering the fact that these old word embedding techniques cannot capture word order but paragraph vector does. Three different machine learning model Random Forest(RF), Support Vector Classifier (SVC) and Logistic Regression (LR) will be trained and tested for identifying sarcasm. Along with this, manual features will also be taken into consideration to check if the performance of different models vary based on the feature engineering technique involved.

1 Introduction

We are in the era of internet where every individual is connected to each other. Social media plays as important role in this society where people communicate to each other, post photos, comment on the post of various companies, movies, business, politics and lots more. These data are the rich source for analysing people opinion on something. Sentiment analysis can be done in order to get the opinion of the people. These sentiments play a major role in many business, while a company or a person maintains an account on social media or any form of online media it is important to know the opinion of the people about them.

Three types of sentiment people often expresses are positive, negative and neutral. Positive sentiment helps the growth of the business whereas the negative sentiment obstructs it. One of the major issue faced in obtaining the true opinion of the people is sarcasm. Bharti, Naidu and Babu (2017) says, presence of sarcastic sentiment makes sentiment analysis challenging. This is a way in which people tend to act positively in the negative situation and act negatively in the positive situation. This problem needs much attention as we cannot just ignore this. Ignoring sarcastic text leads to misinterpretation of the

sentiments obtained by the people. Sentiment analysis is been hindered by the presence of sarcastic tone in the text (Son et al.; 2019).
Consider the below simple examples to check whether it is sarcastic or not (Sreelakshmi and Rafeeque; 2018):

1. I love being ignored.

2. I hated her enough to be with her till my last breath.

Going through the first statement we can observe that the polarity of the sentence starts positively and ends in a negative way. Similarly, in the second statement, polarity shift from negative to positive can be seen. These are couple of simple examples for understanding sarcasm. In reality, it is going to be a tough task as it is very difficult to identify a sentence as sarcastic. When we encounter these sentences while an attempt is made in order to obtain real opinion of people, if the sarcastic sentences are ignored then we end up in misclassifying the sentences as positive or negative.

Many works have been done in order to identify sarcasm using different approaches. Most of the works are done in extracting various features that a sarcastic sentence consists of. These features include sentiments, pragmatic features like emoticons, hyperbolic features like punctuation marks and capital words and lexical features like ngrams. According to Riloff et al. (2013), sarcasm often occurs when positive sentiment and negative situation words encounters together in a sentence. People use emoticons to express their opinion shown by Jain et al. (2017). Recently using different features like pragmatic, lexical and hyperbolic machine learning algorithms were used to classify a sentence as sarcastic or not (Bharti, Naidu and Babu; 2017). This is one approach where we use manually extracted features as an input for machine learning algorithm. Recently many authors have worked on word embedding, which is also referred to as numerical representation of words. Ren et al. (2018) says "Discrete models require a lot of manual features which can be expensive to obtain" and showed that sarcasm can be identified successfully by using word embedding. These two are the most commonly used ways for getting the input to train machine learning model. But most importantly authors have worked on only one of the approach that is either manual features or by using word embedding as their input to the machine learning models.

One of the recent approach in converting words to word vector is the paragraph vectors, similar to previous what was known as Word2Vec. This word embedding approach is comparatively less explored in terms of use case for natural language processing from previous ways like bag-of-words, TF-IDF, Word2Vec etc and also paragraph vectors are capable of handling word orders by using its two methods, Paragraph Vector based Distributed Memory (PV-DM) and Paragraph Vector based Distributed Bag of Words (PV-DBOW) (Le and Mikolov; 2014).

This research will be focusing on both manual feature and also word embedding based sarcasm detection. This will help in knowing whether type of feature engineering depends on the results, like can the manual features help identify sarcastic text all the time. Consider the following sentence:
"A woman needs a man like a fish needs bicycle."[1] In this example there is no usage of

[1] `https://en.wikipedia.org/wiki/Irina_Dunn`

sentiments, pragmatic or hyperbolic features but it is a sarcastic statement, the fact is that fish does not need a bicycle and hence the sentence tries to target man. Therefore, we can say that sarcasm detection cannot be dependent on manual features alone. Word embedding based approach helps in this scenario as it represents each words by numbers based on the similarities between the words.
Hence from the above discussion, considering the advantages of paragraph vector over other word embedding technique and also to check how results vary in manual and word embedding features based sarcasm detection, the research question will be:

"To what extend paragraph vector based Distributed memory and Distributed Bag of words approach along with machine learning algorithms help detect sarcasm over traditionally followed sarcasm detection by manual feature extraction?"

MOTIVATION: Sarcasm detection is often the most interesting and difficult problem in natural language processing. Identifying sarcasm will help anyone who is trying to obtain the true opinion of the customer. Sarcasm detection is one of the essential task in the domain of online media such as online blogs, reviews, Twitter etc for every organization as they influence their business (Bharti, Naidu and Babu; 2017). Consider a product making company maintaining a social media page in order to market their new products. Customers express their opinion in terms of comments or write a product review. But what if a customer expresses their opinion sarcastically. This will lead to a difficult situation where a product company cannot obtain overall positive and negative responses. Only when these are obtained then companies can focus on improvement in the right direction. The application of sarcasm detection is not just applicable to product companies, this is also important for other fields who are dealing with customers.

2 Related Work

In this section we will see all the previous efforts made in identifying sarcasm, their strengths and weaknesses in different scenarios.

Many researchers tried to identify sarcastic sentence in different ways, it can be classified into three main categories without machine learning based, manual features with machine learning and word embedding with machine learning.

2.1 A review on sarcasm detection without machine learning algorithms

Manohar and Kulkarni (2017) says people use social media like twitter to express their thoughts about specific subject. This work showed that sarcasm can be identified successfully by a corpus-based approach. No machine learning algorithm is used instead corpus of action words were created and scoring based classification is done to identify sarcastic sentences. Comparison is made between input text and stored action words and sentiments were calculated, finally sentiment aggregation will tell whether the sentence is sarcastic or not. In a similar way author Rendalkar and Chandankhede (2018) also worked on a scoring-based approach. But instead of making a stored corpus of action words, here

each words in the sentence will be scored and added. The final score will determine the overall sentiment of the sentence. Scores were calculated using SentiWordNet. Different methodologies were used to identify sarcasm such as word and emoticon-based detection, hybrid sarcasm detection which combines both word and emoticon-based approach, positive and negative situation based, pattern based and interjection word starting based sarcasm detection.

According to Bharti et al. (2015) sarcasm is one of the most challenging task in Natural language processing. This work developed two algorithms namely Parsing Based Lexicon Generation algorithm(PBLGA) and Interjection Word Start algorithm(IWSA). Both algorithms are designed differently, first one concentrates more on sentiment and situation contradiction whereas second algorithm can identify sarcastic sentence only when there is an interjection word present in the beginning of the sentence. Data was collected from twitter with hashtags #sarcasm. Each algorithm has its own importance under particular situation, but it is expected to perform poorly when sarcastic sentences contains interjection words in the middle of the sentence. Most of the authors work on tweets which are collected and stored, but Bharti et al. (2016) worked on real time tweets using a Hadoop framework. Different approaches were used to identify sarcasm such as PBLGA, IWSA, positive sentiment with antonym pair (PSWAP), tweets contradicting to universal facts, tweets which are contradicting to time dependant facts. Out of all these the combined (PBLGA, IWSA, PSWAP) approach performed well with the precision score of 0.97.

All the papers discussed shows that sarcasm can be identified without the help of machine learning algorithm. Also, all the works did not consider punctuation feature which is one of the important feature in identifying sarcasm.

2.2 Other works on sarcasm detection

One of the interesting work in sarcasm detection is done by Bharti, Babu and Raman (2017), where author has worked on Hindi tweets, on the other hand most of the researches on sarcasm detection are on English. The concept of the sarcasm remains same, but the language is different. Here in this work a comparison is made between tweets and news of same context. This is done by collecting both tweets and news of same timestamp. Further different algorithms were used to identify the sentiment of the tweet, sentiment in the news and also to identify the contrast between tweet and news. By this way sarcasm is identified if there is a contrast between news context and tweets collected. Similarly, Al-Ghadhban et al. (2017) worked on Arabic tweets and identified sarcastic tweets using sentiment words and hyperbolic features like punctuation marks and quotes.

In the next section works that have implemented machine learning algorithm for sarcasm detection along with manual feature engineering will be discussed.

2.3 A review on sarcasm detection with machine learning algorithms and manual feature engineering

One of the early attempt to identify sarcasm was made by Riloff et al. (2013) where the authors mainly focused on only one possible way sarcasm can occur, that is positive

sentiment contrasted with a negative situation. A novel bootstrapping algorithm was built using a single seed word 'love'which helps in identifying positive sentiment words and negative situation phrases. Using these feature SVM is trained and achieved F-score of 46-48%. The assumption made by the authors will make the algorithm to work poorly when different combination of features occurs like negative sentiment occurring in positive situation or the sarcasm occurring with the help of emoticons. The data was collected from twitter, a of 175000 tweets out of which only 20% were labelled sarcastic. This class imbalance was also not addressed in the paper. This work was one such good early effort made in identifying sarcasm but lacks consideration of various features like pragmatic, hyperbolic and emoticons.

Joshi et al. (2015) worked on a new approach to improve the detection compared to previous work discussed. Along with positive sentiment words and negative situation phrases, this work also added negative sentiment words and positive situation phrases. Other features included in this work are pragmatic, lexical, implicit and explicit incongruity. Explicit incongruity involves the presence of both polarities and implicit incongruity expressing implied sentiment. Combining all these features, SVM with rbf kernel helped achieving F-score of 88%. Even after this due to limited number of learned incongruity expressions this lacks in performance when new implicit expression occurs. Bouazizi and Otsuki Ohtsuki (2016) in their work explains three main purpose behind sarcasm, they are 1. Sarcasm as wit: When people tend to be funny and to exaggerate others. 2.Sarcasm as whimper: To show anger or annoyance for others. 3. Sarcasm as Evasion: To avoid giving clear answer. Machine learning algorithms support vector machine (SVM), Random Forest (RF), k-Nearest Neighbours (kNN) were trained along with four set of features namely sentiments, punctuation, syntactic, semantic and pattern related. Using pattern related features this work got an accuracy of 90% and showed how patterns pay a major role in sarcastic sentences.

In a piece of text people use sarcasm to flip the orientation of the opinion (Bharti, Naidu and Babu; 2017). 100000 tweets were collected and used as data for this work. Developed a hyperbolic feature extraction algorithm which extracts interjection and intensifier words. These words are used as features for determining sarcasm. Machine learning algorithms such as Nave Bayes, Decision tree, Support Vector Machine, Random Forest and AdaBoost were used for training and predicting. Out of these Random forest showed a good accuracy of 80.79% with hyperbolic features. "Better pre-processing can lead to improved classification accuracy in sarcasm detection" Prasad et al. (2017). Author tried in adding a new emoji and slang dictionary at the pre-processing stage. This helped in boosting up the accuracy by 8%. A total of 22 features were used with machine learning algorithms like Random Forest, Gradient Boosting, Decision Tree, Adaptive Boost, Logistic regression and Gaussian Nave Bayes. Features includes pragmatic, emojis and sentiments. This work failed to include lexical features like unigrams, bigrams like few previous works discussed.

Similar to Riloff et al. (2013) where they have used a seeding word 'love 'to extract positive sentiment words and negative situation phrases Jain et al. (2017) worked in same manner by using seed word 'like 'to get sentiment and situation phrases. Additionally, emoticons and pragmatic features are also included. Unlike other works author built two ensemble models random forest and voted ensemble. These ensemble models showed

advantages in terms of improved precision and recall.

So far, we have seen many works on sarcasm detection based on manual feature extraction, that is extracting features like sentiments, pragmatic, lexical and hyperbolic features from the text and also discussed their strennghts and weakness. Now we will discuss some of works on sarcasm detection based on word embedding.

2.4 A review on sarcasm detection with machine learning algorithms and word embedding

According to Ren et al. (2018) manual features like lexical, pragmatic, sentiment and hyperbolic are expensive to obtain. Hence author worked on word embedding based approach for sarcasm detection. Two convolution neural network were trained using word embedding obtained from Word2Vec tool. Unlike other methodology, in this work target tweet were also considered in order to get the overall picture of the sarcasm. Another way of representing words in numeric form is by global vectors for vector representation (GLoVe). This helps in building semantic word embedding Son et al. (2019). In this work Bidirectional LSTM and Convolution Network is used by combining word embedding as well as punctuation and capitalization features. This method has achieved a very good accuracy of over 97% and showed them capability of deep learning with word embedding. On the other hand, (Ghosh, 2016) using the word embedding trained a combination of CNN, LSTM and Deep neural network and achieved the precision of 91%.

Joshi et al. (2016) experimented on 4 different embedding namely LSA, GloVe, Dependency Weights and Word2Vec. This work mainly focuses on the fact that word embedding will help to capture context incongruity in the absence of sentiment words and achieved a F-score of 80.2% using Word2Vec. Author also showed how feature augmentation help in boosting the performance of machine leaning model. Along with contextual information Amir et al. (2016) used word embedding in order to train neural network model to identify sarcasm in tweets. Author says manual feature engineering are expensive and time consuming.

In the next section paragraph vectors and its application importance is discussed.

2.5 A review on paragraph vectors

Paragraph vectors also referred to as Doc2Vec tool, helps in capturing word ordering and semantics of the words (Le and Mikolov; 2014) and helps to overcome the problem of traditional bag-of-words and bag-of-ngrams. Paragraph vector can be used to create distributed vector representation of text of variable lengths, anything from a single phrase to sentence in a large document. This tool is used in many sentiment analysis tasks before and achieved good results. For example Shuai et al. (2018) in their work for performing sentiment analysis on Chinese Hotel review adopted two models of Doc2vec tool, Distributed Bag of Word model of Paragraph Vector (PV-DBOW) and Distributed memory model of Paragraph vector (PV-DM) along with machine learning classifiers like SVM, Logistic regression and Nave Bayes. This work is one such successful adoption of Doc2vec tool for sentiment analysis. Similarly, Sanguansat (2016) worked on social media for business using Doc2vec tool to obtain word vectors and showed how this tool helps in

increasing the performance by capturing word order compared to other vector models.

When we observe the word embedding based models of different authors, most of the authors have considered word2vec model. According to the Le and Mikolov (2014) paragraph vectors-based word embedding model has advantage over word2vec model as this this helps in storing word order. PV-DM works as a memory and helps in predicting words and captures word order and is better than all other word embeddings. Hence this research makes a novel adoption of paragraph vector for sarcasm detection and use these learned vectors as input to machine learning algorithms like Random forest, Support Vector Machine and Logistic regression. Also, as most of the authors worked on only one way, either manual feature extraction based or word embedding based sarcasm detection. This work also aims to work on both the ways on the same data to check whether the selection of feature engineering methods vary the results in terms of classification accuracy.

3 Methodology

CRISP-DM Process: For any data mining project there should be a road map in order to complete the process. For this reason, Cross Industry Standard Process For Data Mining (CRISP-DM) methodology is chosen as it more suitable for this research. "CRISP-DM is a hierarchical process model which helps the data mining projects to be more manageable, reliable and faster" (Wirth and Hipp; 2000). It has six stages and each stage is modified according to the need of this project.

1. Business understanding: For every company dealing with people and have some sort of online media like Twitter, Facebook, Discussion forms, Customer feedback etc. In order to understand the thoughts of customer, company often perform sentiment analysis. People express their opinion in different ways sometimes sarcastically. This problem needs to be solved as to obtain the true opinion of the customer. Failing to obtain the true opinion impacts the business of the organization. Hence it is important for the business organization to perform sarcasm detection before sentiment analysis.

2. Data Understanding: This phase includes data collection and involves various initial tasks in order to understand the data. The main dataset for this work is News Sarcasm Dataset [2] which consists of about 26709 observations with three attributes news link, headline and label that tells whether the sentence is sarcastic or not. The corpus consists of 11725 sarcastic and 14984 non sarcastic records. As there is a slight difference between the count of two labels, dataset is considered as balanced. Initial tasks involve checking the distribution of word count throughout the corpus and developing a word cloud to see most occurring words in sarcastic and non-sarcastic. All these steps will help in getting some insights about the data.

3. Data Preparation: Most of the previous works we discussed in section 2 used twitter as the main source of data and hence lot of cleaning and pre-processing was required in order to remove hashtags, usernames etc. In our case we are using news sarcasm dataset obtained from two newspapers The Onion and HuffPost, hence the amount of noise like misspelling, hashtags and usernames present in the data

[2] `https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection`

are very less compared to twitter data. Unlike other Text classification tasks in which pre-processing involves changing the letter to lower case, remove numbers, punctuation marks and stemming, in sarcasm detection pre-processing involves only removing the numbers, tokenization and parts of speech tagging as capital words and punctuation marks are very essential features. As we are dealing with two ways of sarcasm detection that is manual features and word embedding based features, feature engineering needs to be done in two ways which is discussed next.

4. Feature Engineering

Manual sarcasm detection feature engineering:

Once the pre-processed data is ready, lexical, hyperbolic, pragmatic and sentiment features are extracted from the text.

(a) Lexical features: Contains unigrams and bigrams.

(b) Pragmatic features: Contains Emoticons and smiles.

(c) Hyperbolic features: Contains number of capital words, punctuation marks, interjection and intensifier words.

(d) Sentiments: Contains positive and negative polarity scores.

Word embedding based feature engineering:

In our second approach the pre-processed data will be used to generate word vectors using two models of paragraph vectors, that is PV-DM and PV-DBOW and these vectors will be used as features for next step. Text is often represented in numeric form, these are known as word embedding. Most of the early works in sarcasm detection used one of the word embedding techniques like bag-of-words, TF-IDF and Word2Vec. All these techniques are used for various application dealing with texts. But these lack in capturing word order but paragraph vector model helps in capturing word order and showed improvement in the results of the works dealing with texts (Le and Mikolov; 2014).

(a) Bag-of-words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF): This is one of the feature extraction technique for text. BoW creates a vocabulary of all the words present in the data. TF-IDF is another way of representing text in the form of fraction where term frequency shows how important a word is to the document, that is word importance is directly proportional to the number of times it occurs in the document. IDF does the reverse of it, that is word importance is indirectly proportional to the number of times it appears in the document. In IDF, rare words gets more importance.

(b) Word2Vec: Based on the notion that there is no similarity between words in previous methods discussed Mikolov et al. (2013) in their work introduced word vectors which are capable of capturing semantic and syntactic similarities between the words. This model is built using neural network language model (NNLM). There are two versions of Word2Vec:

i. Continuous Bag-of-Word model: Unlike normal BoW this model uses continuous distributed representation of the context. The vectors created are averaged.

ii. Continuous Skip-gram model: In this model other words in the sentences are used for predicting the current word.

(c) Doc2vec: We saw different models which converts text to numeric format and helps in processing of the textual information. But none of the above models are able to capture the word order like paragraph vector model (Le and Mikolov; 2014). Paragraph vector is an unsupervised algorithm that helps in creation of word vectors on text of various lengths such as sentence, paragraph or documents. Paragraph vector model has two types:

i. Distributed memory model of paragraph vector: Developed on same idea that word vectors helps in the prediction task of the next word. Paragraph vectors also helps predicting next word. It acts as a memory that remembers the missing thing from the context.

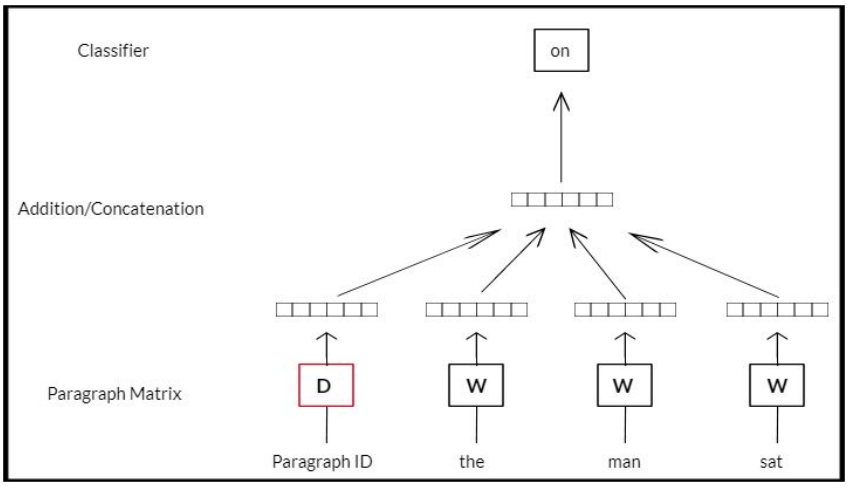

Figure 1: Distributed Memory model of Paragraph Vector

ii. Distributed Bag-of-Word model of paragraph vector: Another architecture under paragraph vector is PV-DBOW. This model is very much similar to Word2Vec skip-gram model Mikolov et al. (2013).

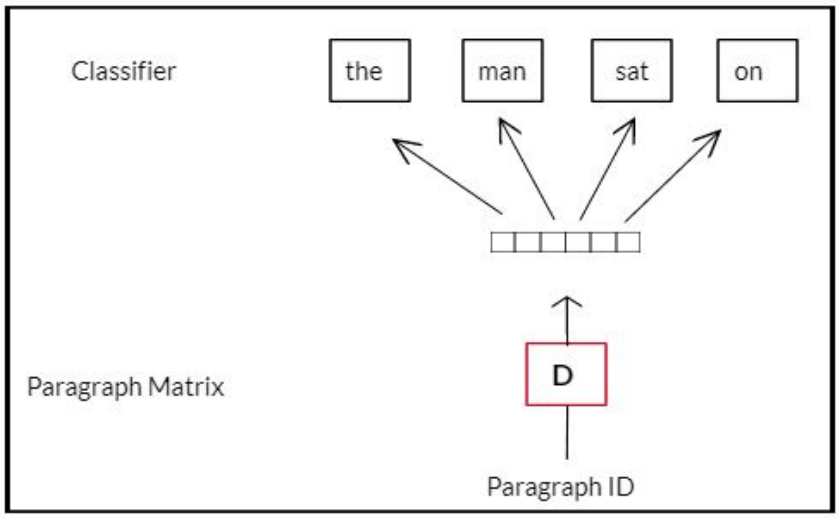

Figure 2: Distributed Bag-of-Word model of Paragraph Vector

During each iteration using paragraph vector, a text window is sampled

and then random word is sampled from text window. This is followed by classification task.

5. Modelling: In this phase using the features created, different models like Support Vector Machine, Random Forest and Logistic Regression will be trained.

6. Evaluation: This phase involves checking the performance of the trained model by using the evaluation matrix like accuracy, precision, recall.

4 Implementation

In this phase we will see how each step defined in methodology is implemented using various technologies will be discussed in detail.

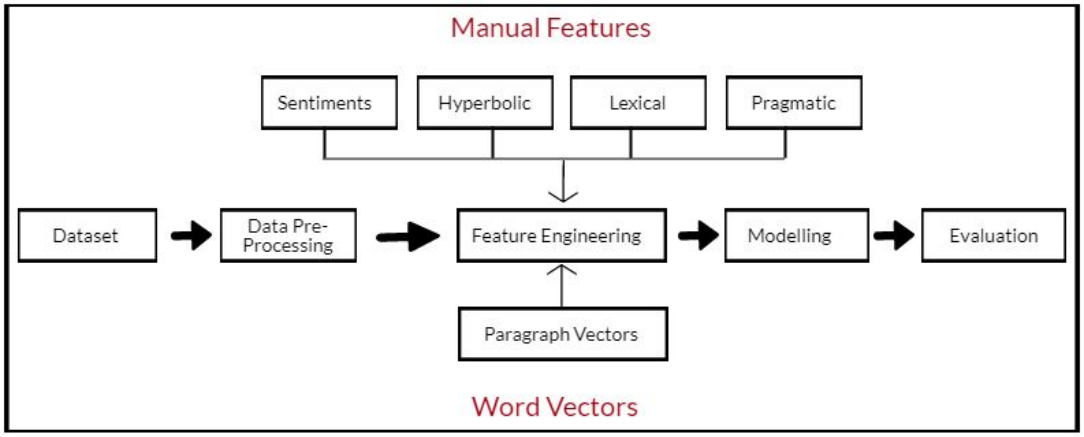

Figure 3: Process Flow Diagram

The above diagram shows the flow of the project, it mainly consists of dataset collection, pre-processing, feature engineering, model training and evaluation. Lets see how each step is performed in the next section.
All the coding is done using python language and Jupyter Notebook.

1. Data Collection: For this research News Sarcasm data set was selected, and it is publicly available in json format. Data consists of an unnecessary column, which is the link to the actual news and this is removed. Sarcastic sentence was labelled as 1 and non-sarcastic sentence was labelled as 0.

2. Data Pre-Processing: As discussed in methodology, for sarcasm detection this stage involves removing of numbers, then text data is tokenized and POS tagged with the help of NLTK[3] library.

   (a) Tokenization: Text analysis starts with splitting the sentences into letters or words called as tokens. This helps in further processes like adding POS tags and also in lexical analysis. For this research tokenization is done using NLTK library which takes sentences as input and provides list of tokens as output.

[3] `https://www.nltk.org/`

(b) Parts-Of-Speech Tagging: Sarcastic sentences mostly rich in adjectives and adverbs(Bouazizi and Otsuki Ohtsuki; 2016). Hence it is important to check how many adjectives and adverbs are used in a sentence. This can be done using POS tagging. A sentence consists of various parts of speech like noun, pronoun, adjective, adverb etc. POS-tagger from NLTK library adds pos tags for each tokens present in a sentence. Using this we can count the parts of speech we are interested in.

3. Feature Engineering:

   Manual Features:

   (a) Sentiments: The likes and dislikes of people about anything is expressed in the form of sentiments, mainly as positive and negative sentiment. As per the knowledge, sarcasm occurs when there is contradicting sentiments in a sentence that is presence of both positive and negative polarity. Most of the previous works used TextBlob for getting polarity of the sentence, TextBlob is a library in python which helps to process textual data and performs tasks such as sentiment analysis, pos tagging, translation etc. In this polarity score varies from -1.0 to +1.0. But in this research vader sentiment module is used as it can easily provides both positive and negative score together for a sentence. On the other hand, TextBlob can only tell either positive or negative. Positive and negative scores obtained are stored in dataframe and use as features for sarcasm detection.

   (b) Lexical Features: Textual features such as like ngrams are referred to as lexical features. Ngrams plays a very important role in sarcasm detection. In this research unigrams and bigrams are used as features. For this purpose, countvectorizer is used from sklearn library[4]. Countvectorizer creates a matrix of word counts, it checks the presence of each word in a document. Number of features needs to be specified or else feature size will be equal to vocabulary size. With the help of parameter ngram range in countvectorizer unigrams, bigrams or ngrams can be specified. For this research 100 unigrams and 50 bigram features are selected.

   (c) Pragmatic Features: Sarcastic sentences are often filled with smiles, emoticons and figurative language, these are called as pragmatic features. According to Jain et al. (2017) emoticons play a major role in expressing sarcasm. Positive smiley with negative situation word makes a sentence sarcastic and vice versa. Hence in this research the usage of smiles is identified and counted in a sentence using regular expression.

   (d) Hyperbolic Features: Interjection, intensifier words, punctuation marks fall under hyperbolic features. We have seen papers in which intensifiers and interjection words alone used for detecting sarcasm (Bharti et al.; 2015). A dictionary is created which consists of list of interjection and intensifier words. Using this dictionary, the number of intensifier and interjection words present in the sentence are calculated and used as a feature for sarcasm detection. Number of punctuation marks present in the sentence is calculated by using

[4] `https://scikit-learn.org/stable/`

regular expression which checks the presence of punctuation marks and counts the number of times it is used. As said by Manjusha and Raseek (2018) sarcasm occurs by comparison, a small dictionary is created which consists of words such as like, similar, look like etc. Using this dictionary, the count of comparison words is checked and used as a feature for sarcasm detection.

By the end of this manual feature extraction process, sentiment, pragmatic, hyperbolic and lexical features are used as inputs to the machine learning models.

4. Word Embedding based features: Paragraph vectors is implemented using python genism library[5]. Input to Doc2Vec consists document and a tag associated to it. Tags are added to each document using TaggedDocument module. These documents are passed to Doc2Vec module which generates word vectors. Parameter needs to be set in order to specify which model to use between PV-DM and PV-DBOW. The vectors generated by these models are passed as input to the machine learning classifiers. The dimentionality of the feature vector is set to 300 and the window size in checked between 5 to 15 as mentioned in (Lau and Baldwin; 2016).

5. Parameter Tuning: It is very important to consider tuning of the parameters of the model as default values cannot provide good results. For this reason GridSearchCV and RandomizedSearchCV methods were chosen initially, after considering the fact that GridSearchCV performs as exhaustive search of the provided parameter, RandomizedSearchCV is chosen over GridSearchCV as it is inefficient and time consuming to perform on each combination of the parameters and check the results.

6. Implementation of Models: The data that we have in this project has labels associated with it, hence this is a supervised machine learning problem. Previous researches used many different machine learning algorithms, and few worked with deep learning. Based on the literature survey and the characteristics of the algorithms, for this research three algorithms Support Vector machine (SVM), Random Forest (RF) and Logistic Regression (LR) will be trained for sarcasm detection.

   (a) Support Vector Machine (SVM) : This algorithm helps in creating a boundary called as hyperplane between the data points. Hyperplane can be visualized as a flat surface. SVM falls under supervised machine learning techniques. This algorithm deals with linearly separable and linearly non separable data. The algorithm aims to maximize the distance between the data points of two classes. Implementation of this is done using python library sklearn. This provides Support Vector Classifier (SVC) which provides various parameters like kernel, gamma, penalty (C) etc. Parameters are not set to default values and parameter tuning is done using RandomizedSearchCV, this is a technique which helps to find the optimal values in order to get highest accuracy. Kernel was set to linear and rbf, penalty varying from 1 to 10 and gamma value from 0.1 to 0.9.

   (b) Random Forest (RF): Another supervised learning algorithm that works in ensemble way. This is also called as decision tree forest as the algorithm ensembles only decision trees. Random forest has random feature selection, it

[5] `https://radimrehurek.com/gensim/models/doc2vec.html`

selects only a portion of all the features and helps to deal with large dataset with large number of features. Using RandomizedSearchCV various parameters of the algorithm are tuned. Implementation of RF is done using sklearn library in python.

(c) Logistic Regression: This is considered as the powerful algorithm for a binary classification problem. Logistic regression is a probabilistic model. Similar to previous models RandomizedSearchCV is used for parameter tuning. This is implemented using sklearn library in python.

5 Evaluation

In this stage all the models built will be evaluated in order to see if the results obtained are good enough for sarcasm detection and also to check which approach is more suitable to classify a text as sarcastic. For the evaluation purpose accuracy will be used as main metric. Accuracy in this case refers to how many sarcastic text are correctly labelled as sarcastic out of all text. Other evaluation metric like precision and recall could be used for sarcasm detection depending on the topics underlying in the data. Below table shows the results of different models.

Model	Manual	PV-DBOW	PV-DM
SVC	83.61	91.26	85.11
RF	86.79	88.45	79.99
LR	77.08	81.39	77.07

Table 1: Accuracy of the algorithm with manual and word vector features

5.1 Evaluation of Models built using manual features like sentiments, lexical, pragmatic and hyperbolic

1. Model 1: Support Vector Classifier:
   Based on the results obtained from the parameter tuning using RandomizedSearchCV, SVC kernel was selected as rbf with the gamma value of 0.2 and penalty of 8. Also 5-fold cross validation is done to obtain the best parameters. With these values obtained for manual features, SVC achieved an accuracy of 83.61%. Precision and recall values are 0.79 and 0.85 respectively.

2. Model 2: Random Forest Classifier:
   Using the manual features RF was able to classify text as sarcastic or not with an accuracy of 86.79%, also precision and recall of 0.85. Based on parameter tuning technique, the total number of trees in the forest also called n_estimators are set to 100, max depth of the tree as 100, max features selected as sqrt and the criterion that measure the quality if the split is selected as entropy. 5-fold cross validation is done in similar way as previous model.

3. Model 3: Logistic Regression:
   The overall accuracy obtained by using this classier is 77.08%. The precision value obtained is 0.75 and recall of 0.72. Based on RandomizedSearchCV the penalty

norm was set to l2 with newton-cg solver and max iteration 150 along with 5-fold cross validation.

Out of the above three model for manual feature-based sarcasm detection random forest performed best with the highest accuracy of 86.79% and logistic regression was not good enough as other two models.

5.2 Evaluation of Models built using word vectors obtained from PV-DBOW

1. Model 1: Support Vector Classifier: With the help of word vectors obtained from Distributed Bag of word-based paragraph vector, SVC has achieved a very good accuracy of 91.26% using rbf kernel with the gamma value of 0.5 and penalty of 5. The precision value obtained is 0.9 and recall 0.89.

2. Model 2: Random Forest: The accuracy obtained by random forest classifier using PV-DBOW is 88.45%, this is comparatively better than the accuracy obtained using manual features. Parameter tuning is done separately for this model. The n_estimators was set to 1000 with max depth of 50. Precision and recall values are 0.89 and 0.84 respectively.

3. Model 3: Logistic Regression: There is a lot of improvement in the accuracy of the logistic regression model built using PV-DBOW. The overall accuracy of the model is 81.39% which is nearly 4% more than the logistic regression model built using manual features. Precision obtained was 0.8 and recall 0.77. The solver used for this case is lbfgs with l2 penalty and max iteration of 200.

In this experiment of using the PV-DBOW, the highest accuracy was achieved by support vector classifier whereas in the previous approach, that is using manual features random forest achieved the best accuracy.

5.3 Evaluation of Models built using word vectors obtained from PV-DM

1. Model 1: Support Vector Classifier: Using the paragraph vector method which is also known as memory-based approach, the accuracy of the SVC model is 85.11% which is better than the manual feature based result. The optimal parameters are obtained using RandomizedSearchCV with 5-fold cross validation. The kernel used was rbf with gamma of 0.2 and penalty of 6. The precision and recall obtained are 0.84 and 0.82 respectively.

2. Model 2: Random Forest: Similar to the decrease in the accuracy of the SVC model compared to PV-DBOW based SVC, random forest model also able to classify a text as sarcastic or not with the accuracy of 79.99%. This is the least value obtained for this model compared to the model built using manual features and PV-DBOW vector. Based on parameter tuning the number of trees used are 1000 with max depth of 100. The model attained the precision of 0.82 and recall of 0.7.

3. Model 3: Logistic Regression: The last model of the experiment gained the accuracy of 77.07% with l1 penalty and max iteration of 60. Precision and recall obtained are 0.76 and 0.7 respectively. Logistic regression performed poorly in all the three experiments.

Out of all the three ways of sarcasm detection, SVC classifier which is trained using word vectors obtained from PV-DBOW model showed the best accuracy score. In the other words when compared to other word embedding techniques discussed in literature paragraph vector has great potential in its application to sarcasm detection.

5.4 Discussion

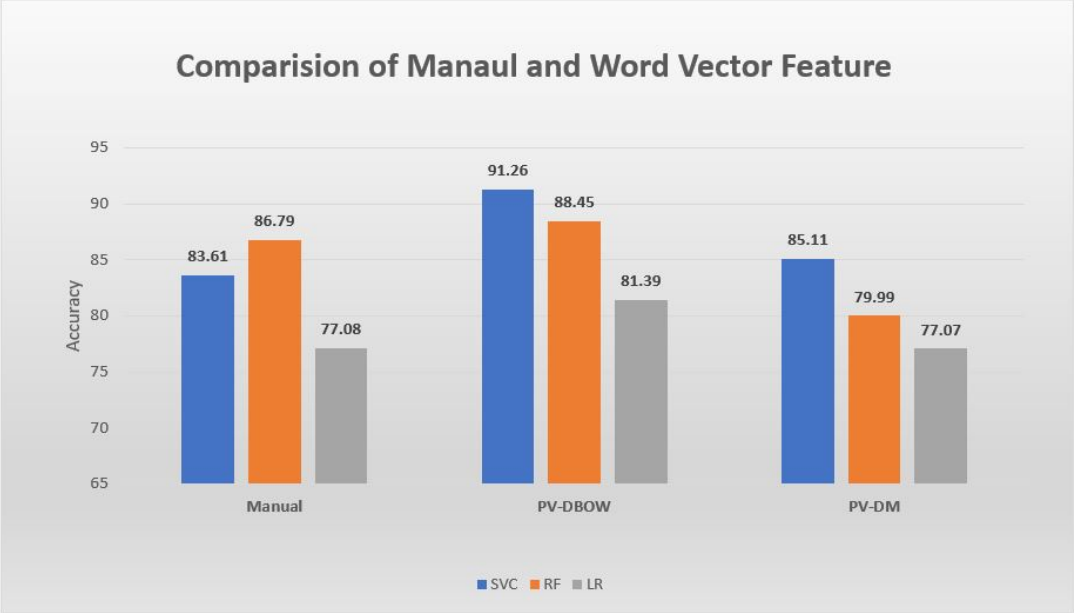

Figure 4: Results

The results of all the models that was implemented is shown in the above chart. From all the above models it is observed that paragraph vector based distributed bag of word (PV-DBOW) features has provided the highest accuracy in classifying a text as sarcastic or not. The main intention of this research was to check the extent to which the selection of paragraph vector over other word embedding technique like Word2Vec in sarcasm detection and also to compare how manual and word vector features help identify sarcasm.

By the results it is evident that the word vectors are more suitable for sarcasm detection in terms of classification accuracy. Also obtaining manual features is quite expensive in terms of time and effort that is needed. Features like sentiments, pragmatic, hyperbolic and lexical will help only if they are present in the data, but paragraph vectors learn the word ordering, semantic and syntactical features just by presence and usage of the words. So, whenever data which consists of less manual features, it is good to select paragraph vector for feature engineering and then use this to train the machine learning models.

When comparing the performance between PV-DBOW and PV-DM, it is observed that

distributed bag of word approach outperforms memory model, so for sarcasm detection the word vectors obtained by PV-DOW are more suitable. The machine learning model SVC performed well with word vectors as features where as with manual features random forest classier outperforms other two models. It also observed that PV-DBOW performed better with a bigger window size compared to the window size of PV-DM.

During various experiments it is observed stop words play a role in sarcasm detection, it is noticed that the usage of stop words has impact on the sarcastic sentences. Generally, in other text classification tasks like news article classification, stop words dont play a major role as most of the text classification problems remove stop words. But here in sarcasm detection a decrease in the accuracy was observed when stop words are removed from the text. Hence it is important to check the impact of removing of stop words in sentiment analysis tasks and also in sarcasm detection.
One of the drawback of this work is that, as the dataset size is not very big the models trained on the dataset that is used in this work may not produce the same result when tested on some random data, because the learned word vectors or manual features may not be same as this dataset.

6 Conclusion and Future Work

From all the experiments we observed PV-DOW method of paragraph vectors has great potential in sarcasm detection. The combination of SVM and PV-DOW method achieved an accuracy of 91.26%. With the manual feature also the machine learning models, SVC and RF performed well. As per the results we are successful in knowing the capability of paragraph vector and also from the result it is evident that the data set which is used is suitable for both manual feature and word embedding based approach of sarcasm detection. As the dataset which is used in this research is different from that of the works which are discussed in section 2, it is quite difficult to make a comparison of the results but the methods that was implemented have shown good results for this dataset.

In future large sarcasm datasets can be used. Also, cross domain experiment needs to be done to explore how model performs in such situation. Sarcasm is a never-ending problem in obtaining true opinion. Hence data needs to be collected from various different sources and make use of word embedding as well as manual features whenever applicable. For manual feature the dictionaries can be improved by adding more words and also polarity score of the emoticons can included as in this research only the presence of the emoticons is taken into consideration. Along with paragraph vectors, deep learning techniques can be used in future.

Acknowledgement I would like to thank my supervisor Prof. Noel Cosgrave for his continuous support and guidance provided throughout my research work.

References

Al-Ghadhban, D., Alnkhilan, E., Tatwany, L. and Alrazgan, M. (2017). Arabic sarcasm detection in twitter, *2017 International Conference on Engineering MIS (ICEMIS)*, pp. 1–7.

Amir, S., Wallace, B. C., Lyu, H. and Silva, P. C. M. J. (2016). Modelling context with user embeddings for sarcasm detection in social media, *arXiv preprint arXiv:1607.00976* .

Bharti, S. K., Babu, K. S. and Jena, S. K. (2015). Parsing-based sarcasm sentiment recognition in twitter data, *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 1373–1380.

Bharti, S. K., Babu, K. S. and Raman, R. (2017). Context-based sarcasm detection in hindi tweets, *2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR)*, pp. 1–6.

Bharti, S. K., Naidu, R. and Babu, K. S. (2017). Hyperbolic feature-based sarcasm detection in tweets: A machine learning approach, *2017 14th IEEE India Council International Conference (INDICON)*, pp. 1–6.

Bharti, S., Vachha, B., Pradhan, R., Babu, K. and Jena, S. (2016). Sarcastic sentiment detection in tweets streamed in real time: a big data approach, *Digital Communications and Networks* **2**(3): 108 – 121. Advances in Big Data.
URL: *http://www.sciencedirect.com/science/article/pii/S235286481630027X*

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

Jain, T., Agrawal, N., Goyal, G. and Aggrawal, N. (2017). Sarcasm detection of tweets: A comparative study, *2017 Tenth International Conference on Contemporary Computing (IC3)*, pp. 1–6.

Joshi, A., Sharma, V. and Bhattacharyya, P. (2015). Harnessing context incongruity for sarcasm detection, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Association for Computational Linguistics, Beijing, China, pp. 757–762.
URL: *https://www.aclweb.org/anthology/P15-2124*

Joshi, A., Tripathi, V., Patel, K., Bhattacharyya, P. and Carman, M. (2016). Are word embedding-based features useful for sarcasm detection?, *arXiv preprint arXiv:1610.00883* .

Lau, J. H. and Baldwin, T. (2016). An empirical evaluation of doc2vec with practical insights into document embedding generation, *arXiv preprint arXiv:1607.05368* .

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents, *International conference on machine learning*, pp. 1188–1196.

Manjusha, P. D. and Raseek, C. (2018). Convolutional neural network based simile classification system, *2018 International Conference on Emerging Trends and Innovations In Engineering And Technological Research (ICETIETR)*, pp. 1–5.

Manohar, M. Y. and Kulkarni, P. (2017). Improvement sarcasm analysis using nlp and corpus based approach, *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 618–622.

Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013). Efficient estimation of word representations in vector space, *arXiv preprint arXiv:1301.3781* .

Prasad, A. G., Sanjana, S., Bhat, S. M. and Harish, B. S. (2017). Sentiment analysis for sarcasm detection on streaming short text data, *2017 2nd International Conference on Knowledge Engineering and Applications (ICKEA)*, pp. 1–5.

Ren, Y., Ji, D. and Ren, H. (2018). Context-augmented convolutional neural networks for twitter sarcasm detection, *Neurocomputing* **308**: 1 – 7.
URL: *http://www.sciencedirect.com/science/article/pii/S0925231218304284*

Rendalkar, S. and Chandankhede, C. (2018). Sarcasm detection of online comments using emotion detection, pp. 1244–1249.

Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N. and Huang, R. (2013). Sarcasm as contrast between a positive sentiment and negative situation, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Seattle, Washington, USA, pp. 704–714.
URL: *https://www.aclweb.org/anthology/D13-1066*

Sanguansat, P. (2016). Paragraph2vec-based sentiment analysis on social media for business in thailand, *2016 8th International Conference on Knowledge and Smart Technology (KST)*, pp. 175–178.

Shuai, Q., Huang, Y., Jin, L. and Pang, L. (2018). Sentiment analysis on chinese hotel reviews with doc2vec and classifiers, *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 1171–1174.

Son, L. H., Kumar, A., Sangwan, S. R., Arora, A., Nayyar, A. and Abdel-Basset, M. (2019). Sarcasm detection using soft attention-based bidirectional long short-term memory model with convolution network, *IEEE Access* **7**: 23319–23328.

Sreelakshmi, K. and Rafeeque, P. C. (2018). An effective approach for detection of sarcasm in tweets, *2018 International CET Conference on Control, Communication, and Computing (IC4)*, pp. 377–382.

Wirth, R. and Hipp, J. (2000). Crisp-dm: Towards a standard process model for data mining, *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, Citeseer, pp. 29–39.