Sarcasm detection in natural language processing

Article *in* Materials Today Proceedings · October 2020

DOI: 10.1016/j.matpr.2020.09.124

CITATIONS
10

READS
1,956

6 authors, including:

Ashwitha Anup
M.S. Ramaiah Institute of Technology
14 PUBLICATIONS **28** CITATIONS
SEE PROFILE

Shruthi Gowda
M.S. Ramaiah Institute of Technology
12 PUBLICATIONS **53** CITATIONS
SEE PROFILE

Makarand Upadhyaya
University of Bahrain
70 PUBLICATIONS **493** CITATIONS
SEE PROFILE

Tc Manjunath
Dayananda Sagar College of Engineering
233 PUBLICATIONS **771** CITATIONS
SEE PROFILE

Materials Today: Proceedings 37 (2021) 3324–3331

Contents lists available at ScienceDirect

Materials Today: Proceedings

journal homepage: www.elsevier.com/locate/matpr

Sarcasm detection in natural language processing

Ashwitha A [a,*], Shruthi G [a], Shruthi H R [b], Makarand Upadhyaya [c], Abhra Pratip Ray [d], Manjunath T C [e]

[a] Department of Information Science and Engineering, M.S. Ramaiah Institute of Technology, Bangalore 560054, India
[b] Department of Computer Science and Engineering, Acharya Institute of Technology, Bangalore 560090, India
[c] Department of Management and Marketing, College of Business Administration, University of Bahrain, Bahrain
[d] Department of Physics, Pratibha College of Commerce and Computer Studies, Pune 411019, India
[e] Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, Bengaluru 560078, India

ARTICLE INFO

Article history:
Received 20 August 2020
Accepted 3 September 2020
Available online 31 October 2020

Keywords:
SARCASM detection
Natural language processing and machines and humans interpersonal communications
Kernel matrix
Support vector machine
Random forest

ABSTRACT

The striking property of satire is that it makes it difficult to bridge and bridge the gap between its literal and intended meaning. Identifying sarcastic behavior in the field of online social networks such as Facebook, Twitter, Instagram, surveys, etc. has turned into a fundamental task as they affect social and personal relationships. SARCASM detection is an important processing problem in natural language processing (NLP), which is needed for better understanding to serve as an interface for mutual communication between machines and humans. To understand this is to underline the basic problem behind it - being able to detect the contradiction. For this, a need arose to define contextual understanding and emotion. To accomplish this we need to do two things - gather a stack of target words that display sentiment shifts (sarcastic words) based on context; And with an objective word given an expression, how to naturally identify whether the objective word is used in an exact or sarcastic sense. Collecting information is done by the use of an information retrieval system, for example tweepee. For the latter, some distributed semantic methods are used to convert data into useful information and are then demonstrated using multiple classifier results that is, satire is identified.

Selection and peer-review under responsibility of the scientific committee of the International Conference on Newer Trends and Innovation in Mechanical Engineering: Materials Science.

1. Introduction

Online activities as a sidekick have grown into a huge force in ongoing opportunities for business, government issues, attention diversion, and so on. Social media, for example, Twitter, Facebook, WhatsApp and beyond, has been transformed into a major mechanism as an online partner and is receiving responses from all over the world. These reactions involve one's oblique feeling or feeling towards a particular goal, namely, people, opportunities, subjects, objects, organization, administration and so forth (Fig. 1).

This slant is a person's feeling towards a clear goal. It can be positive or negative. Manual extraction of oppression from online life is a monotonous activity for people or organizations. Along these lines it needs a computerized framework to investigate suppression without human impedance (Fig. 2).

The real difficulties involved with the suppression test are the magnitude of the snide-peek. Because of this, a large proportion of the current frameworks are used for investigation so as to respect the true spirit. Mockery is a specific type of perception that typically flows to the beginning of a conclusion in a given material. According to the Macmillan Dictionary, mockery is known as "the act of saying or making the opposite of what you say" or in the way of talking on the other side that is expected to make someone else feel dumb or display them you are mad [1] (Fig. 3).

The machine brain has long had a problem understanding its idiapresis and the field of NLP and ML has tried to demolish these walls, albeit unsuccessfully. So here is an effort towards a resolution (Fig. 4).

1.1. Motivation

Online promotion or marketing has gained popularity over the years as social media is the only way to convey the message to the youth and perceived emotion as a person's opinion towards a specific goal. Manually marking sarcastic content on social media can be tedious. But most of the challenges lie in detecting the pres-

* Corresponding author.

https://doi.org/10.1016/j.matpr.2020.09.124
2214-7853/
Selection and peer-review under responsibility of the scientific committee of the International Conference on Newer Trends and Innovation in Mechanical Engineering: Materials Science.

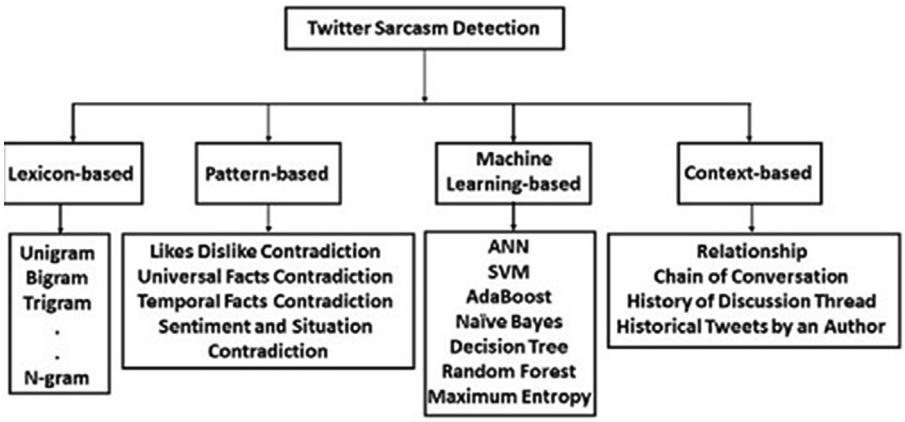

Fig. 1. Approaches of sarcasm detection.

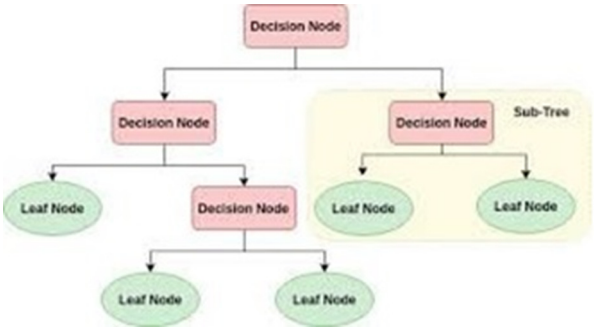

Fig. 2. Structure of a decision tree.

INPUT: S, **where** $S = set\ of\ classified\ instances$
OUTPUT: $Decision\ Tree$
Require: $S \neq \varnothing$, $num\_attributes > 0$

```
procedure BuildTree
    repeat
        maxGain ← 0
        splitA ← null
        e ← Entropy(Attributes)
        for all Attributes a in S do
            gain ← InformationGain(a, e)
            if gain > maxGain then
                maxGain ← gain
                splitA ← a
            end if
        end for
        Partition(S, splitA)
    until all partitions processed
end procedure
```

Fig. 3. Decision tree classifier induction algor.

ence of satire even for automated systems. In view of all this we decided to introduce some machine learning approach to detect sarcasm (Fig. 5).

1.2. Constraints and requirements

The requirements of this project are the successful detection of satire for a precise level of accuracy and also to demonstrate how different models affect accuracy to make it clear that the best option is currently implemented. The main obstacles to the project are mainly with the introduction of the use of memes, new emoticons, etc., new ways of expressing are coming forward each day. Therefore, this project is limited only to the scope of human simplicity (Fig. 6).

1.3. Problem statement

SARSCAM is a very important part of human speech. It has existed since time immemorial. For example, a plane missing "What a great day wow!". The true meaning of a sentence will darken its true essence, that is, despair from the shadow of anger. The meaning of this sentence can be expressed only through speech, while writing it may cause confusion. It is just a pebble in the mountain of problems with the increasing use of satire as it is a part of many fields like politics etc. The project aims to solve the problem of satire detection using ML and neural models to increase understanding for it (Fig. 7).

1.4. Scope and objectives

The various difficulties present in understanding sarcastic commentaries refer to many occasions and need to expand a lot of realities, rational know-how, Anupora goals [2], and intelligent thinking. Producers abstain from feature extraction and rely on CNNs to naturally take in highlights from simulated datasets. For classification; the primary purpose of sentiment analysis is to determine whether the message is positive, negative, or neutral. Conversely, the purpose of detecting satire is to determine whether the message is sarcastic or not sarcastic. The main purpose is to change the orientation of any text without manual extraction (Fig. 8).

1.5. Proposal model

In this article, an automated framework for identifying sarcastic comments from Twitter information is proposed. MVME is used for feature extraction of LSTM [3] instead of LSTM, mainly using sequential processing. SVM, linear svc, random forests and decision trees are trained to achieve the objective (Fig. 9).

1.6. Organization of report

To explain the developed system, the following sections are included:

- The literature review studies existing systems and techniques prior to the development of the proposed systems.
- System Analysis and Design, through the software engineering methodology adopted to implement the model, provides an overview of the system and detailed information about the model included in the system.

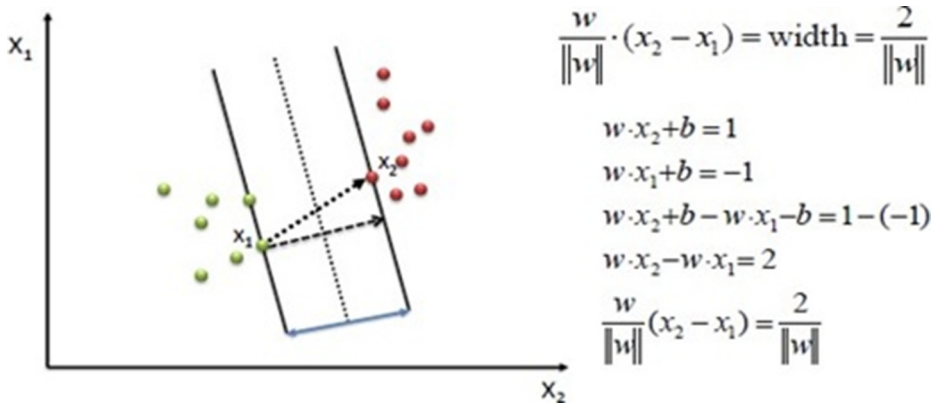

Fig. 4. Linear SVM algorithm.

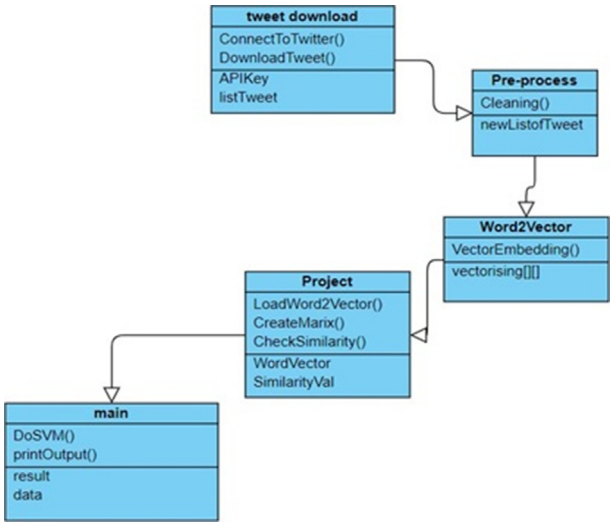

Fig. 5. Class diagram.

- Modeling and implementation provides a deep insight into model work. The various modules and their interactions are represented using relevant descriptive diagrams

1.7. Testing

Testing the model to ensure a bug / error-free model, with the results obtained, then provides detailed analysis on quality assurance measurements. Conclusions regarding the results obtained after successfully running the model and the future scope of the model are highlighted.

2. Literature review

2.1. Sarcasm studies in linguistics

Sarcasm is a type of expressive language where words are not of exact importance, and conversely, understanding is expected. Sarcasm is strongly identified with incongruity - truth be told, a type of anomaly to express that as a system of using coherence by abstract researchers to propose refinements between the real world and the will. In abstract considered by researchers. There are two types of anomalies: oral and constitutional. Verbal dissonance is dissonance that is communicated in words.

2.2. Characteristics of mockery

The satire is accompanied by certain measurements, in particular, naming fizzle, desire, deceit of a calm mind, negative pressure, and an unfortunate accident. As far as expressing this reaction can be portrayed as much joking, sarcastic reactions can be quick, no response, grin, fake (in the counter), difference in subject, strict answer or Nonreactive reactions - a major non-literal response will be spread). As indicated by satire when there is a situational deviation between content and relevant data. For instance, the sentence 'I love proved available' is interpreted as mocking because the difference between logical data is that being disregarded is an annoying circumstance, and that the speaker accepts it in the given sentence.

2.3. Types of mockery

There are four types of mockery

- Propositional: Under such circumstances, the declaration appears, by all accounts, to be a suggestion, but also includes prudence. For example 'Your system looks amazing! If the setting is not understood, the sentence may be counted as nonsnide.
- Embedded: It is a type of joke that consists of words and a transplanted harmony of self. For example 'John has eliminated such a negotiator, that no one pays attention to him.' Confusion in such sentences is established by the importance of the word 'representative' and in the rest of the sentence.
- Like prefixed: 'Like / Us If' are common prefixes to query logical questions. For example "as you wanted". This sentence refers to the reproach of something told by another person.
- Illocutionary: The last type of mockery to consider is known as elocutionary mockery. These are cases where the extension of a joke does not include some suggestion of some component or expression inside the expressed sentence, although the whole redundant function that will attempt the true expression of the key sentence.

2.4. Approaches

- Rule-based approach:The theory-based method attempts to identify spurious through specific evidence. The evidence is captured as a guideline that relies on markers of satire to distinguish satire in similes. [4] Google uses the ٩-step approach to determine how likely the comparison is. The hashtag assumption is a key indicator of this sarcasm. Hashtags are often used by tweet creators to introduce satire, and thus, if the assump-

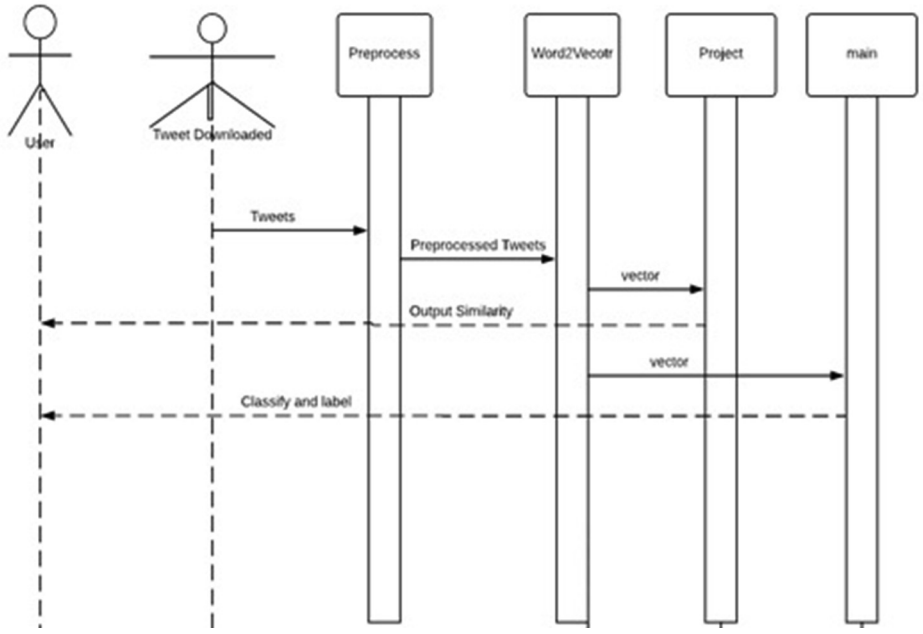

Fig. 6. Sequence diagram.

```
In [1]: runfile('C:/Users/Koli/Documents/Project/FYP/main.py', wdir='C:/Users/Koli/Documents/Project/FYP')
C:\Users\Koli\Anaconda3\lib\site-packages\gensim\utils.py:1197: UserWarning: detected Windows; aliasing chunkize to chunkize_serial
  warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
C:\Users\Koli\Anaconda3\lib\site-packages\smart_open\smart_open_lib.py:253: UserWarning: This function is deprecated, use smart_oper
Technologies/smart_open/blob/master/README.rst#migrating-to-the-new-open-function
  'See the migration notes for details: %s' % _MIGRATION_NOTES_URL

Target Word is  mature

Preprocessing Complete.. ..

Loading vectors pre-trained on Training data
Dimensions in Training Data :  54

Computing Kernel Matirx
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
```

Fig. 7. Results of pre-processing.

```
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

Kernal Matrix computed and saved in csv file

Preparing data for classifiers ...


0
1
2
3
4
5
6
7
8
9
10
11
12
13

---------------------->
Training the SVM model using the kernel matrix
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
  kernel='rbf', max_iter=-1, probability=False, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
Classification Accuracy is  95 %
```

Fig. 8. Results for vector conversion.

tion transmitted by the hashtag does not coincide with the rest of the tweet, the tweet is projected as a snide. Use a hash tag tokenizes to split a hashtag made up of connected words.

- Statistical approach: Statistical methods to deal with sarcastic recognition as far as the various highlights and ups and downs of learning systems take hold. This subsection illustrates

```
3
4
5
6
7
8
9
10
11
12
13

---------------------->
Training the SVM model using the kernel matrix
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
  kernel='rbf', max_iter=-1, probability=False, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
Classification Accuracy is  95 %

---------------------->
Training the RF model using kernel matrix
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
Classification Accuracy is  95 %

---------------------->
Training the LinearSVC model using kernel matrix
LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
     intercept_scaling=1, loss='squared_hinge', max_iter=1000,
     multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
     verbose=0)
Classification Accuracy is  93 %

---------------------->
Training the DT model using kernel matrix
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
Classification Accuracy is  96 %
```

IPython console History log

Fig. 9. Result of iterations.

through various measurable methods for spurious detection. Configuration of the design based on highlights that demonstrates the possibility of discriminating instances as separated by a huge truncated marked corpus. Normalized examples can be viewed by the classifier assuming these pattern-based features to be real dependent on three conditions: fixed match, partial overlap, and no match.

- Pattern based features fill in as their fundamental commitment. He has depicted words in high-repetition words (HFW) and con.
- Feature Sets: This area highlights the arrangement that is responsible for the satire of mediocrity. [5] The pack of words is used as a highlight in most of the functionality. It may be that, despite these, the highlights are responsible for some different arrangements. Content-related highlights are categorized. Logical highlights (i.e., data from previous usage of the content that was highlighted) are depicted in the latter field. Pattern Matching, these pattern-based highlights reflect real-world conditions dependent on three conditions: Exact Match, Half Match and No Match. Different scoring algorithms are used after matching and a conclusion is reached. Some users are AFINN, SentiWordNet and General Inquirer and so on.
- Learning algorithm: Most of the work in sarcastic recognition relies on different types of support vector machines (SVM) regression, with the 2 test used to identify differential features. It is then used to detect sniping.\K

3. System analysis and design

There are mainly four approaches to the above work
1. Lexicon based.
2. Pattern Based.
3. Machine learning based.
4. Context Based.

- Gathering of data using tweetdotcom.
- Preprocessing of data to perform operations.
- Part-of-speech is a process for dividing input texts into atomic words and allocating them with appropriate POS tag data based on setting and engagement with surrounding words in the content.
- Tokenize the input string using TweetTokenizer.
- Facility selection and extraction are the most important undertakings in the ML approach.
- Use of offensive words and intensities to gather a list of features.
- An intensifier is considered an adjective, adverb or a mixture of both.
- Words like wah, aha, nah, yahu and the like are used as offensive words.
- To remove all of these we need to use POS labeling. Classifiers are used to label the data into sarcastic and non-sarcastic categories. SVMs, decision trees, random forests, and linear svc are used to train models for predicting sarcastic tweets.

3.1. Algorithms used

MVME Algorithm:
MVMEwe
1: procedure MVMEwe(vs,vu)
2: vswords ← vs.elements()
3: vuwords ← vu.elements()
4: M[vswords .size(),
5: for k ← 0, vswords .size() do
6: ck ← vswords [k]
7: ∼ck ← getEmbedding(ck)
8: for j ← 0, vuwords .size() do
9: wj ← vuwords [j]
10: ∼wj ← getEmbedding(wj)
11: M[k][j] ← cosine(∼ck, ∼wj)
12: end for
13: end for
14: while True do
15: repeat
16: max ← getM ax(M)
17: Sim ← Sim + max
18: rm, cm ← getRowCol(M, max)
19: Remove rm row and cm column from M
20: remove(M, rm, cm)
21: until max > 0 Or M.size() > 0
22: end while
23: Return Sim
24: end procedure
25: procedure GETEMBEDDING(word)
26: Return wemodel[word]
27: end procedure
28: procedure GETROWCOL(M,max)
29: row, col ← M.indexOf(max)
30: Return row, col 31: end procedure

3.2. SVM algorithm

Support vector machines are used for finding a hyperplane to classify data points among large set of clustered data.

Loss functions of SVM:

$$\min_{w} \lambda \| w \|^2 + \sum_{i=1}^{n} (1 - y_i \langle x_i, w \rangle) +$$

The next word vectors are used for similarities in the words collected from the tweet. This model has used Word2vec to generate words in a bag of words. The kernel matrix is an information structure that includes the distance between each pair of training cases. This kernel matrix is then extended from the Scatic / SKLearn library in Python to a support vector classifier.

3.3. Random forest classification

Random forest is an ensemble based tree based algorithm. Random forest classifier is a group of decision trees from a randomly chosen subset of training sets. This test collects votes from various decision trees to decide the final class of the object.

Types of Random Forest Models:

1. Random forest prediction for classification problem: f (x) = majority vote of all approximate classes on B trees.

2. Random forest prediction for a regression problem:

f (x) = B Sum of all sub-tree forecasts divided on trees.

3.4. Pseudocode of random forest creation

1. Choose "K" features from randomly selected "K" features where K ≪ M.

2. Among the "K" features, compute the node "D" using the best segmentation point.

Divide the node into daughter nodes using the best partition.

4. Repeat steps a to c until the "l" number of nodes is reached.

5. To create an "N" number of trees, create a forest by repeating D for the "N" number bar.

3.5. Pseudo code of prediction using random forest

1. Takes test features and uses the rules of each randomly created decision tree to predict the result and store the predicted result (target)
2. Count the votes for each projected goal
3. Consider the high polling predicted target as the final predictor from the random forest algorithm

3.6. Decision tree

A decision tree is a flowchart such as a tree structure, where each internal node represents a test on an attribute, each branch represents the result of the test, and each leaf node (terminal node) bears a class label.

A tree can be learned by dividing the source set in a subset based on the test value of an attribute. This process is repeated on each derived subgroup in a recursive manner called recursive segmentation. The recursion is complete when the target variable in a subset at a node has the same value, or adds value to the predictions when not partitioned (using information gain, guinea index, etc.). Decision trees can handle high dimensional data. In general, decision tree classifiers have good accuracy.

3.7. Support vector machine classification

The ‘‘support vector machine” (SVM) is a supervised machine learning algorithm that can be used for both classification and regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in the n -dimensional space (where n is the number of features you have) with the value of each attribute being the value of a particular coordinate. Then, we make a classification by detecting the hyper-plane that separates the two classes very well.

3.8. Linear SVM

The SVM algorithm helps find the best line or decision range. This best boundary or region is known as a hyperplane. The SVM algorithm finds the closest point of the lines from both classes. These points are called support vectors. The distance between the vectors and the hyperplane is called the margin. SVM aims to maximize this margin. The hyperplane with the maximum margin is called the optimal hyperplane.

Linear kernel is used when data is linearly separated, that is, it can be separated using a single line. It is one of the most common kernels used. It is mostly used when a particular data set has a large number of attributes. One of the examples where there are too many features is text classification, as each alphabet is a new feature. Therefore we mostly use linear kernel in Text classification.

The red and green circles indicate different classes of data. Hyperplane with the maximum margin is calculated and is known as the optimal hyperplane.

4. Modelling and implementation

Reconstructing satire as a kind of word perception demonetisation issue: given an expression and an objective word, regardless of whether the sense of the objective word is strict or sarcastic. This is called the Literal / Sarcastic Sense Disambiguation (LSSD) task.

Two major challenges must be faced:

1) How to gather lots of target words that may contain a literal or a goofy feeling, contingent on determining what is more,

2) Looking at the expression of an objective word, how can we decide whether to use the word objective in its strict sense, or in a funny sense.

4.1. Collection of words

Those words are used in Twitter messages, to gather learning and test datasets for each of the target words. For mocking sense (S), tweets that contain objective words and are marked with a #sarcasm or #sarcastic tag. For the strict meaning (L), tweets are collected that contain the target word and are not marked with a hashtag of #sarcastic or #sarasm.

Also, think of an unusual case for precise meaning, where tweets are named with hashtags with a positive or negative sense (eg, #happy, #sad).

Accordingly, two LSSD undertakings: S vs. L and S vs. Lesant, intend to gather a decent dataset for each target word.

4.2. Preprocessing

There are three types of tags in the data such as ‘‘sad”, ’sarcastic’ and nothing else. To begin, tweet where the objective word is used in its most regular way, for example in an exact way.

These tweets were not marked by client or hash tag. Next, tweets that use purpose in a funny sense are marked.

These tweets were chosen based on two criteria:

1. They contain objective words and
2. They were named #sarcastic or #sarcasm by the client.

This leads us to the third type of accessible tweet in the dataset. These are tweets that contain objective words and are marked with emotion.

These opinions reflect the use of the word objective in a positive or negative way.

The labels used in the tweet give us an instinct on how to use the word objective in the language.

4.3. Methodology

Paired grouping is used to identify the use of spuriousness by examining the purpose word and its use in tweets. For this reason a classifier model is created. Anyway, the information is set before the information in the classifier is resolved so that the classifier issues can be resolved. For this reason the information is predetermined. In pre-processing, each tweet is cleaned up and creates a list of words for each tweet. So overall, preparation information is a list of records.

4.4. Word2vec

The next word vectors are designed to capture the similarity between the words present in the tweet. We used the Word2vec model executed by Gensim. The rundown of the tweets was preprocessed, which is encouraged in the Word2vec model to be included for every single word caring for the words. This finished model is then set aside for future reason.

4.5. Kernel matrix

The kernel matrix is an information structure in which the piece is different between each pair of training sets. This bit network is then extended from the Scatic / SKLearn library in Python to a support vector classifier. The deleted part between two tweets is determined using the Multi Weld Matrix Matrix (MVME) calculation. The MVME algorithm is used to simulate the word between two tweets to estimate word similarity for each word pair framed by incoming items between tweets

4.6. Classification

After the kernel matrix has been filled in using MVME computation, we extend this matrix to the SVM classifier of the libSVM library and prepare it using the classifier administration learning process. The characterization model was verified with a 5 crease test before being tried. The information for preparing and testing the tee was part of a ratio of 80:20

4.7. Class diagram

The class drawings typically represent the contamination conditions of the project in separate packages and then bring them together in the final stages to represent the complete picture. It also reflects inheritance hence first demarcation ie parent process and later child process. The project is divided into sections

- o Tweet Download: Using Twippy to download tweets.
- o Pre-processing: The received data is cleaned so that punctuation, marks etc. can be removed.
- o Word2Vec: Cleaned data vector to be used at a later stage to conclude in a similar way.
- o Project: Part of the project includes commonly used tasks such as accuracy etc. which are called multiple times for different model iterations.
- o Main: Main is the place where all the functions from the preceding sub block are called and the final program is shaped.

4.8. Sequence diagrams

Sequence diagrams represent the flow of information. It depicts the various stages of a project and at which stage it sends information from which flow is being shown. Downloaded Tweets in Diagram -> Pre-process-> Word2Vector-> Project -> Main. Finally the key sends the output to the user as well as the parity report.

5. Testing, result and discussion

5.1. Testing

It has used 20% of the original dataset for testing data. The Maturetrain dataset and Joytrain are used. In mature trains the hashtag is depressed, satirical and with data in common. The hash tags in Joy train are data with pleasure, satire and general data.

5.2. Result

The first diagram shows the pre-processing and word results for vector conversion. It also shows the result of iterations. The accuracy rate is between 90 and 96 percent.

5.3. Discussion

Replication of the input data degrades the classification accuracy for the model. This is an expected output because feature extraction exposes the same output multiple times for the training set, and is overfed for information preparation. Experimentation with Bayesian methods for classifying words establishes the same baseline as pattern and thesaurus looking methods. Both configurations for the respective models show an iterative increase in mass processing.

5.4. Conclusion and future work

The data stream for sarcasm does not exhibit any static structure. Due to this, the prediction of sarcasm in Twitter (or any semi-structured information format) that leverages Machine Intelligence is a challenging task. Compared with other heuristics employing pattern match or context based, this proves to be a troublesome assignment, albeit a more comprehensive one.

The project aims to convey the use of modern technology to combat social problems and constructs that hinder free communication. This is established by the use of classification methods for detection and classification of tweets. This has been done using a hyperbolic feature set.

Future work for this project includes resolution of semantic disambiguation using a progressive Recurrent Neural Network model. To employ this, use the features and metadata generated from the current model as input for the network. Bi-directional LSTMs can be used for context detection and a comprehensive sentiment search can be carried out using the VADER library.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Tomas Ptacek, Ivan Habernal and Jun Hong, "Sarcasm Detection on Czech and English Twitter", Proceedings of 25th International Conference on Computational Linguistics, pp. 213-223, 2014.
[2] Chinmay Jain, Sarcasm Detection in Twitter Data using SVM Classification, IEEE Access, 2010.
[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, Efficient estimation of word representations in vector space, IEEE Access, 2013.
[4] R. Rehurek, P. Sojka, Software framework for topic modeling with large corpora. proceedings of the LREC 2010 workshop on new challenges for NLP frameworks, IEEE Access, 2010.
[5] Scikit-learn: machine learning in Python-scikit-learn 0.16.1 documentation http://scikit- l earn.org/stable.

Further Reading

[1] Debanjan Ghosh, Weiwei Guo, Sman Muresan, Sarcastic or not: word embeddings to predict the literal or sarcastic meaning of words, IEEE Access (2015).
[2] Regina Barzilay, Kathleen R McKeown, Extracting paraphrases from a parallel corpus, IEEE Access (2001).
[3] Lakshya Kumar, Arpan Somani, Approaches for computational sarcasm detection: A survey, IEEE Access (2011).
[4] Colin Bannard and Chris Callison-Burch, Paraphrasing with bilingual parallel corpora. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, IEEE Access, 2005.
[5] Mohit Bansal, Kevin Gimpel, Karen Livescu, Tailoring continuous word representations for dependency parsing, IEEE Access (2014).
[6] Kenneth Ward Church and Patrick Hanks, Word association norms, mutual information, and lexicography. Computational linguistics, IEEE Access, 1994.