A Novel Approach for Data Scraping and Sentiment Analysis from Twitter using Machine Learning Techniques

Sakshi Singh[*], Alekhya Nandy[*], Panchajanya Das[*], Sudeshna Bhowmik[*], Subham Biswas[*], Rajdeep Sen Gupta[*], Dr. Sandip Mandal[*]
Communication Email: sandy06.gcect@gmail.com
*University of Engineering and Management, Kolkata, India

Abstract
In this paper we propose a method to analyze the sentiments of a user on a particular topic on the basis of their tweets. This method breaks up the texts into words, remove all the unnecessary stop words and then using a dictionary determines the sentiments by grouping certain words together. Using Tokenization it first splits those tweets into smaller units and using lemmatization it reduces the infected words to meaningfulwords before comparing those with the dictionary elements. Our purpose is to establish perceptive links between tweets and public sentiment for a better understanding of the public opinion. In this paper we present two applications of this method: recognizing depression in tweets and recognizing mass opinion. In the first application we use certain keywords to search for tweets that tend towards depression and our program analyzes the exact sentiments of the tweet to confirm the existence of depression. In the second application, tweets regarding a certain topic are scraped and they are analyzed together to give a result that represents the public opinion.

1. Introduction

Microblogging these days has been very popular in the modern era. It has especially paved its way through the young generation.Today's youth share each and every moment of their life in different social media platforms. They tend to help the youngsters to expose their lifestyle, their feelings, their ideas and aspirations and their opinions to the world through these microblogging platforms. Twitter, Facebook, Tumblr and other microblogging sites offer such service to the public. These platforms help the netizens to share their ideas, opinions and views over different issues and discuss about the current issues of concern. Whether be it the product satisfaction or political or religious or social views, these microblogging sites have provided an arena for the ordinary people to voice their thoughts and opinions. This has helped millions and millions of people to share their opinions and concern about any subject. This has helped us to take the data from one of these famous microblogging sites, i.e., Twitter, into a dataset formed of large number of short messages created by the netizens using this platform. The subject of the messages varies from public opinions to personal thoughts.

The data from these sources can be used by various authorities for sentiment analysis and opinion mining. Political parties would be intertest to know what impact their recent decision has on the citizens mind.

Our project "Twitter scraping and sentiment analysis" is a research project which helps to analyse the sentiments of a person based on the text he/she writes. This model displays a bar graph which shows an index of the feelings which the person feels the moment he /she wrote the text. This will help the person to understand the current mental health of him/her.

We chose particularly Twitter for the following reasons:

- Twitter is one of the most used microblogging websites in the recent years with users belonging to a wide range of age from teenagers to the netizens in their fifties.

- Twitter is used by netizens belonging to a wide range of profession, be it celebrities, politicians, doctors, lawyers, corporate workers, civil servants or businessmen.
- Twitter has been a great platform for netizens to share their opinions, sentiments and emotions. Providing us the required diverse dataset for our project.
- Twitter contains a large number of small text posts and the numbers increases day by day. Hence provides us the enormous dataset required.

We have used Program counter to count the elements present in the container. We also used matplotlib for data visualisation and for graphical plotting. NLTK was used to work with human language data for applying statistical NLP. Tokenization was used to extract the tokens from the string of characters. It actually returns the syllables from a single word. Lemmatization was done by calling the lemmatize() function on a single word. Twint is an advanced Twitter scrapping tool written in python that allows for scrapping Tweets from Twitter profiles without using Twitters API.

Contributions

The contributions of our paper are as follows:

1. We present a method to collect a corpus of objective texts from a particular microblogging source. Our method allows to collect objective text such that no human source is needed for classifying the documents. The size of the collected corpora can be arbitrarily large.

2. We perform linguistic analysis of the collected corpus.

3. We use the collected corpora to build a sentiment classification system for microblogging.

Corpus collection

Using Twitter python library twint we collected a corpus of text posts and formed a dataset of three classes: positive sentiments, negative sentiments, and a set of objective texts.

In order to collect a corpus of objective posts, we retrieved text messages from Twitter related to a particular individual or trend.

Because each message cannot exceed 140 characters by the rules of the microblogging platform, it is usually composed of a single sentence. Therefore, we assume that an emoticon within a message represents an emotion for the whole message and all the words of the message are related to this emotion. In our research, we use English language.

Vader (Valence Aware Dictionary and sentiment Reasoner)

VADER (Valence Aware Dictionary and sentiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. It is used for sentiment analysis of text which has both the polarities that is positive/negative. VADER is used to quantify how much of positive or negative emotion the corpus has and also the intensity of the emotions.It gives us scores of the following categories:

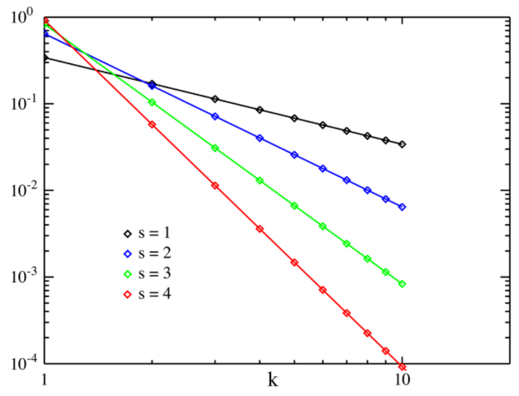
- Positive
- Negative
- Neutral
- Compound

The compound score is the sum of positive, negative & neutral scores which is then normalized between -1(most extreme negative) and +1 (most extreme positive).

The more Compound score closer to +1, the higher the positivity of the text.

Corpus analysis

First, we checked the distribution of words frequencies in the corpus. A plot of word frequencies is presented in the following figure



As we can see from the plot, the distribution of word frequencies follows Zipf's law, which confirms a proper characteristic of the collected corpus. **Zipf's law** is an empirical law formulated using mathematical statistics that refers to the fact that for many types of data studied in the physical and social sciences, the rank-frequency distribution is an inverse relation. The **Zipfian distribution** is one of a family of related discrete power law probability distributions. It is related to the zeta distribution, but is not identical.

Zipf's law was originally formulated in terms of quantitative linguistics, stating that given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.

Training the classifier

Feature extraction

The collected dataset or factually a dictionary is used to extract features that will be used to train our sentiment classifier. The frequency of a keyword's occurrence is a more suitable feature, since the overall sentiment may not necessarily be indicated through the repeated use of keywords.

We have used stopwords to remove the unused words. A stop word is a commonly used word (such as "the", "a", "an", "in") that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. We would not want these words to take upspace in our database, or taking up valuable processing time. For this, we can remove them easily, by storing a list of words that you consider to stop words.

The process of obtaining data from a Twitter post is as follows:

- Filtering – we remove URL links (e.g. http://example.com), Twitter user names (e.g. @alex – with symbol @ indicating a user name), Twitter special words (such as "RT"6), and emoticons.

- Tokenization – we segment text by splitting it by spaces and punctuation marks, and form a bag of words. However, we make sure that short forms such as "don't", "I'll", "she'd" will remain as one word.

- Removing stopwords – We remove all the words that are not being used to analyse the sentiment of the microblogs. Such words includes "i", "me", "my", "myself", "we", "our", "ours", "ourselves", "you", "your", "yours", "yourself", "yourselves", "he", "him", "his", "himself", "she", "her", "hers", "herself", "it", "its", "itself", "they", "them", "their", "theirs", "themselves", "what", "which", "who", "whom", "this", "that", "these" "those", "am", "is", "are", "was", "were", "be", "been", "being", "have", "has", "had", "having", "do","does", "did", "doing", "a", "an", "the", "and", "but", "if", "or", "because", "as", "until", "while", "of", "at", "by", "for", "with", "about", "against", "between", "into", "through", "during", "before" ,"after", "above", "below", "to", "from", "up", "down", "in", "out", "on", "off", "over", "under", "again", "further", "then", "once", "here", "there", "when", "where", "why", "how", "all", "any", "both", "each" ,"few", "more", "most", "other", "some", "such", "no", "nor", "not", "only", "own", "same", "so", "than", "too", "very", "s", "t", "can", "will", "just", "don", "should", "now"

At last we create the list of finals words which will be checked for the analysis.

Data and methodology

We have tested our classifier on a set of real Twitter posts hand-annotated. We used the same evaluation set as in (Go et al., 2009). The characteristics of the dataset are presented in Table

Sentiment	Number of samples
Positive	108
Negative	75
Neutral	33
Total	216

We compute accuracy (Manning and Schutze, 1999) of the classifier on the whole evaluation dataset, i.e.:

$$\textbf{accuracy} = \frac{\textbf{N(correct classifications)}}{\textbf{N(all classifications)}}$$

$$\textbf{decision} = \frac{\textbf{N(retrieved documents)}}{\textbf{N(all documents)}}$$

The value of the decision shows what part of data was classified by the system.

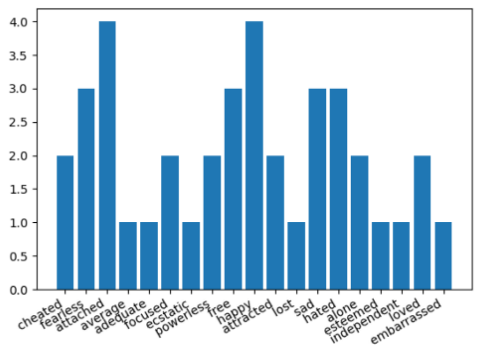
Matplotlib

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to make correlations. They're typically instruments for reasoning about quantitative information. Some of the sample plots are covered here.

We are using Matplotlib here to plot the graph of emotions which we obtain from our code snippet which uses the data obtained from the microblogs from twitter.



Results

Here, first we are obtaining a specific number of microblogs from a specific date about a particular trend or topic or some important names. Then we are stockpiling all the microblogs converting them into lowercase before storing it in a list. We are also using a text file to store all types of words and their respective emotions in the form of a dictionary.

Now, we are importing stopwords from corpus. We checked the distribution of words frequencies in the corpus. Using stopwords we are eliminating all the unnecessary words. Example- is, am, are, he etc.; which are not needed for natural language processing. Then using VADER (Valence Aware Dictionary and sentiment Reasoner) we analyse the obtained text which has both the polarities both negative and positive. If the score of negative is more than that of positive then we observe it to be a negative sentiment else if the score of positive is more than that of negative then we observe it to be a positive sentiment, otherwise we observe it to be neutral sentiment. We are also using tokenizer and lemmatizer to make the word machine understandable so that our code snippet without any flaw. We are also cleaning the text of any fullstops, commas, or any punctuations.Then once the cleaned final text is obtained we send it to thenext section of the code.

Here, we get the desired result in the form of a graphical representation, like if words related to "sadness" are more in the given snippet then a rise will show in that part and vice-versa, as it is evident from the bar graph above.In this project we have learned how to perform basic natural language processing with Python. We used twint to search the data from microblogs and then scraped the results for headlines. We then analysed the headlines for sentiments score and created a dataframe from the results and displayed them in a graph.

DISCUSSION OF CHALLENGES AND DIRECTIONS

After running our program multiple times through numerous blocks of texts we found a few limitations of the dictionaries and the data science used. Sentiment analysis poses a limitation to computers because to understand emotions computers have to be trained like human. We have not yet reached the stage where the mere binary can define the full extent of the varying range of human thoughts and emotions.

Even for humans tone is often a huge problem to decipher verbally let alone via text. Moreover, when the analysis is being conducted on a huge block of text there are many tonal changes across the entire text that makes it more difficult for the analysis. This can be somewhat solved by using a tone detector with a scale that our program can work on.

Many words such as great or disgusting have a solid polarity where their meaning can be very easily distinguishable. But certain words like 'average' or 'not up to the mark' can have varying degrees of meanings. At times the words accompanying these mid – polar words get separated from them in the analysis which further dilutes the meaning of the words.While our program usually is capable of understanding and designating the proper meaning to these mid-polar words a sudden change in the author's style of writing may confuse the program.

In today's generation the usage of sarcasm is highly saturated. Using a compliment to actually express a negative sentiment is hugely popular and also a problem for the computers that analyze these texts. Sarcasm can often lead to a misrepresentation of emotions, usually leading to a higher than the original amount of positive feedback.

The problem of understanding sarcasm can be solved but it requires the API to be able to recognize the context and factors that create emotions. This can be achieved only with a huge dataset that further increases the run time of the program.

Multilingual tweets also pose a challenge to the program. Direct translation to a single language cannot be done because often times the essence of the text is lost during the translation. As direct translation does not solve the problem, the program would need a tagger, lemmatizer and the ability to understand the grammars of various different languages so that the multilingual text could be properly analyzed.

The best way to overcome this is by having different models for different languages and training them separately. Though time consuming this is the best course of action when faced with multiple languages in texts.

Idioms also need special attention when it comes to sentiment analysis. Any idiom will simply confuse the algorithm as it understands everything in a literal fashion. Often times the algorithm may even go so far as to ignore the idiom part completely as it makes no sense to the program in terms of emotion. In other cases, an idiom like "It's raining cats and dogs" will mostly be misunderstood by the algorithm. Moreover, idioms in other languages just adds to the problem.

The neural networks under the emotion mining API must be trained to understand idioms. They are trained by mapping various nouns that depict certain emotions to the idioms. This increases the effectiveness and the accuracy of the analysis.

Reviewers often use twitter to release their opinions. In a review tweet comparativesentences are very common. Comparative sentences can often be confusing as they tend to compare without expressing a certain like or dislike for either product. Forexample a sentence like ‘Car A has a larger base than Car B’ can mean several things depending upon the context.

Here the accuracy of the analysis can be achieved if the algorithm learns the scale of comparison of a certain property. Then it needs to be able to co relate either positive or negative emotion with the said property while fully understanding the context of the text. This needs the artificial intelligence to retrieve information from its database and be able to co relate the properties, sentiments and the words at the same time.

Negations such as ‘I can’t not do this’ can confuse the algorithm. Thought the purpose of the double negative is for emphasis it often confuses the algorithm as sentences are not taken as a whole and words are split up. This may lead to a sentence being misunderstood as negative where its original meaning was positive.

There is a relatively easy fix to this as the algorithm can be trained to understand that double negatives cancel each other out and turn it into a positive. The algorithm just needs to be trained with the negative words so that it can understand all of the combinations.

Conclusion

Microblogging nowadays has become one of the major types of the communication. A recent research has identified it as online word-of-mouth branding.Sentiment analysis or opinion mining is a field of study that analyses people’s sentiments, attitudes, or emotions towards certain entities.Thelarge amount of information contained in microblogging web-sites makes them an attractive source of data for opinion mining and sentiment analysis/ natural language processing. In our research, we have presented a method for an automatic collection of a corpus that can be used to train a sentiment classifier. We observed the difference in distributions among positive, negative and neutral sets. From the observations we conclude that authors use syntactic structures to describe emotions or state facts. Some of them may be strong indicators of emotional text. We used the collected corpus to train a sentiment classifier. Our classifier is able to determine positive, negative and neutral sentiments of documents. Then it plots a bar graph of all types of emotions depicted from the collected data. The classifier is based on a dictionary which contains all types of words and their corresponding emotions.This paper tackles a fundamental problem of sentiment analysis, sentiment polarity categorization.As the future work, we plan to collect a multilingual corpus of Twitter data and compare the characteristics of the corpus across different languages. We plan to use the obtained data to build a multilingual sentiment classifier.