A Report On
TROLL DETECTION

Submitted in partial fulfilment of the course
CS F469: Information Retrieval

Group Work by -

Madhuresh Bhattacharya	2017H1030139P
Shruti Bansal	2017H1120248P
Sakshi Fuladi	2016A8PS0445P

Birla Institute of Technology and Science, Pilani
November 2018

Acknowledgement

Working on this project titled -'*Troll Detection using Sentiment Analysis*' was a source of immense knowledge to us. We would like to express our sincere gratitude towards Prof. Poonam Goyal for providing such a platform for learning. Also, towards Teaching Assistant Ms. Chandramani Chaudhary for guiding us at all the points wherein we lacked experience. We acknowledge with a deep sense of gratitude, the encouragement and inspiration received from faculty members and colleagues.

Table of Contents

Problem Statement:

For many individuals around the globe web-based social networking destinations are a coordinated piece of their everyday life. There are hundreds of varied social media sites supporting a wide range of practices and interests. Social networks such as Facebook and Twitter have become a source for news, a platform for political and moral debate for a lot of users. Stories with different degrees of truthfulness are spread and little source criticism is applied by regular people, even by journalists. The act of spreading disinformation on social media has developed from being caused by bored youths to being commercialized by organizations and political blocks in the form of troll farms. A troll farm is an organization whose sole purpose is to affect public opinion with the means of social media. It is an association whose sole design is to influence popular supposition with the methods for web-based social networking. A practical implementation of a system or a software that can identify troll farms could be used in order to stop them and therefore avoid the spread of disinformation. Such a usage would intrigue the legislators, media, informal communities or associations that are focused since it could be utilized to clear their names.

The aim of this project is to detect troll farms. The problem at hand is to perform sentiment analysis and thus determine if a text document is a troll or not. The approach will be to study classification algorithms/Python inbuilt package NLTK, Text processing techniques using TextBlob, etc. so as to apply them to a database of comments and analyze them. Therefore, the problem statement boils down to:

- How to detect trolls using sentiment analysis?

Background:

Sentiment Analysis

· Sentiment analysis is a text classification technique which takes a word or sentence and says if the underlying intent is positive, negative or neutral.

· The procedure of computationally recognizing and classifying conclusions communicated in a bit of content, particularly with the end goal to decide if the writer's mentality is towards a specific subject, item, and so on, by calculating its sentiment(+/ -/ neutral)

· Widely used to determine whether the sentiment of mass is towards the subject of interest.

Motivation of the problem

· Working with real life data provides a clear scenario of efficiency, f-measures and other evaluation criteria required for the system to work

· Practical work in this domain being highly promoted in technology across the globe

· An aspect of social media data such as Twitter messages/ Facebook comments is that it includes rich structured information about the individuals involved in the communication.

· It can lead to more accurate tools for extracting semantic information

· It provides means for empirically studying properties of social interactions.

· Freely available annotated corpus, Python NLTK codes used with NLP so as to convey the sentiment in texts being effectively used to catch the culprit

Technical issues

- We faced some difficulty in extracting data from twitter. Tweepy API helped us to obtain data.
- Python 3.5 was used to ensure consistency in creation and loading of pickle files.

Related Work: Literature survey

· **Statistical features-based real-time detection of drifted twitter spam**.

Chen C., Wang Y., Zhang J., Xiang Y., Zhou, W. and Min, G.

* The paper presents classification of tweets into Spam & Non-Spam categories. This project has been built on parallel lines with the paper with categories as Troll and Non-Troll. “Twitter Spam Drift” problem from

* They have collect and labelled a real-world dataset using the Twitter API. Further investigation is supported by both data analysis and experimental evaluation aspects.

* A novel L-fun Algorithm is used to detect Twitter Spam.

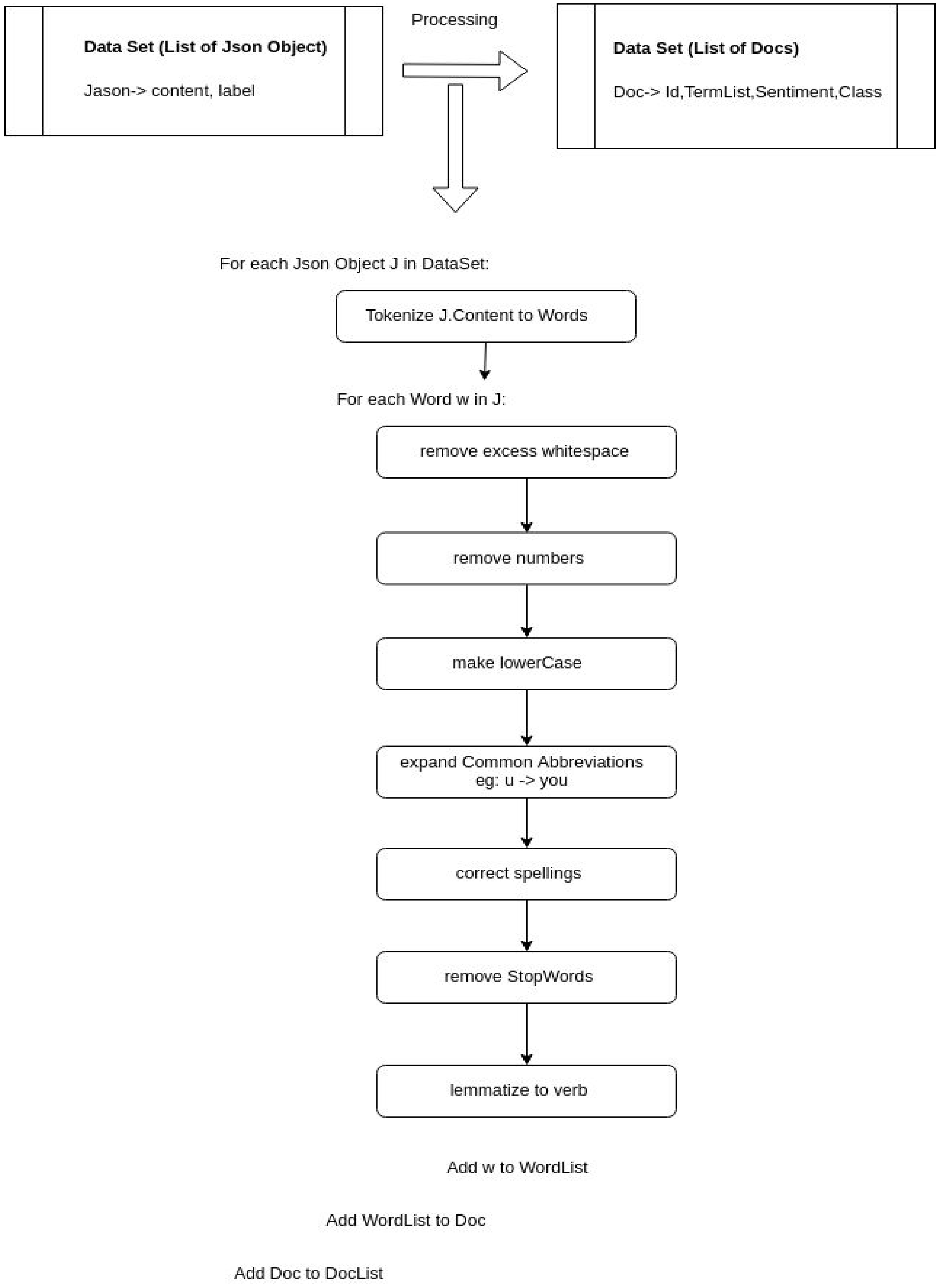
Research Gap

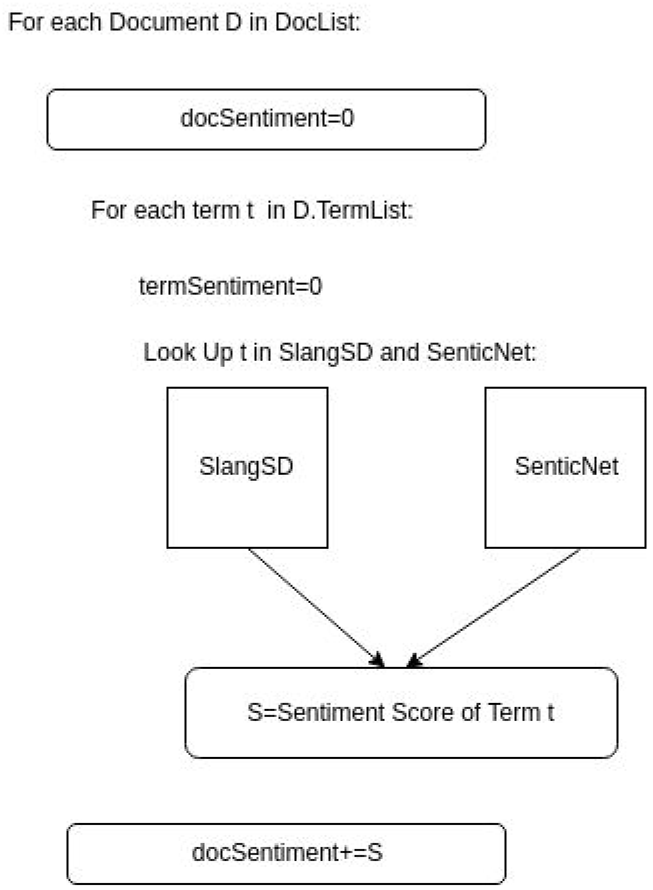
The research gaps observed in the paper are:

- The paper is meant for spam and non-spam detection. We have tried to modify it by performing troll and non-troll classification.
- The paper does not involve the use of sentiments. We have tried to use sentiment as a feature for text classification.

System Description:

Block Diagram of the System-




For each Document D in DocList:
docSentiment=0
For each term t in D.TermList:
termSentiment=0
Look Up t in SlangSD and SenticNet:
SlangSD
SenticNet
S=Sentiment Score of Term t
docSentiment+=S

The dataset obtained from twitter containing tweets/comments from online social networking sites is presented in a .json file. The json form of each document (tweet) in the file is (content,label)

On processing these docs individually, each doc and its corresponding doc id, term list, total sentimental and class has to be programmed. This is the output form desired. Now, the processing steps include tokenization of <content> part of each json file into individual constituent terms. Now processing such each term, by removing white spaces within, numbers (as they provide no sentiment), conversion into lower case (as SenticNet Dictionary and Slang SD Dictionary have words in lower case), expanding common abbreviations so as to get more no. of words common in the sentiment dictionary, correcting spelling, removing stop words, and converting using WordNet lemmatizer, every term into verb form. All such words combined form word list and one doc. Such combined docs form the doclist in the dataset.

Sentiment Calculation-

Our assumption was sentiment score would play a big role in troll detection. This is because most troll tweets tend to be negative sentiments. Since the dataset does not come with a sentiment score for learning, we went for unsupervised learning. For this purpose we took help from two sources, first Slang SD dictionary and second the SenticNet Dictionary. Initially the total sentiment of every doc is set to 0(neutral). As the code finds common words from the wordlists in each doc, either in the Slang SD dictionary or in the SenticNet Dictionary, it assigns a score to these words that contributes to the total sentiment of the doc. It was seen that 78% of the total troll tweets had a negative sentiment score.

The features used for text classification is the TF-IDF matrix and the sentiment scores obtained for the bag of words. This matrix is input to the Random Forest Classifier which is a mixture of decision trees. RFC takes the majority of the decision tree results and classifies the document as troll or not-troll. With this majority the documents are classified.

Evaluation Strategy:

The Evaluation strategy is based on the following:

- The extent of testing data which is correctly classified.
- The extent of False Positives in the system.
- The extent of False Negatives in the system.

Experimental Results and evaluation:

Finding trolls is a classification problem. Hence we tried to test our model with testing data and construct the confusion matrix for the results obtained.

Confusion Matrix:

N=4889	PREDICTED: YES	PREDICTED: NO
ACTUAL: YES	1667 (TP)	254 (FN)
ACTUAL: NO	314 (FP)	2654 (TN)

Precision= TP/(TP+FP) = 1667(1667+314) = 0.84

Recall=TP/(TP+FN) = 1667/(1667+254) = 0.86

F1-Measure = 2*P*R/(P+R) = 0.849

The results indicate that the classification is done with high precision and high recall.

Conclusion and future work

Text processing is a difficult task since computers do not understand texts. So, feature extraction is necessary for this purpose. TF-IDF helps to quantify texts numerically by assigning values according to term frequency and document frequency. Sentiments are needed since they indicate whether the tweet is a negative comment, hence a troll. With these features, the random classifier uses multiple decision trees to provide results.

The future work involves using bigram index to compare it with TF-IDF features. This will help to compare the better method of text feature extraction for text classification problem.

References

Paper-

Chen, Chao, Yu Wang, Jun Zhang, Yang Xiang, Wanlei Zhou, and Geyong Min. "Statistical features-based real-time detection of drifted twitter spam." IEEE Transactions on Information Forensics and Security 12, no. 4 (2017): 914-925.

Kaggle Sentiment Dictionary Citation-

Bing Liu, Minqing Hu and Junsheng Cheng. "Opinion Observer: Analyzing and Comparing Opinions on the Web." Proceedings of the 14th International World Wide Web conference (WWW-2005), May 10-14, 2005, Chiba, Japan.

DataSet-
Tweepy API - Extracting data from twitter.

(Kaggle Sentiment Dictionary) for training purpose.

https://www.kaggle.com/dataturks/dataset-for-detection-of-cybertrolls
https://dataturks.com/projects/abhishek.narayanan/Dataset%20for%20Detection%20of%20Cyber-Trolls

Sentiment Dictionary for Slang Words-

The Slang Sentiment Dictionary (SlangSD) includes over 90,000 slang words together with their sentiment scores, facilitating sentiment analysis in user-generated contents.
(https://arxiv.org/abs/1608.05129)
http://slangsd.com/

Citation Paper-

SlangSD: Building and Using a Sentiment Dictionary of Slang Words for Short-Text Sentiment Classification, Liang Wu, Fred Morstatter, Huan Liu

Sentiment Dictionary for Non-slang words-

(Kaggle Sentiment Dictionary)

Kaggle is an online community of data scientists and machine learners, owned by Google, Inc. Kaggle offers a public data platform and a cloud-based workbench for data science.