A Report on E-Mail Spam Detector (Filter)

Shinde Yash Vikas

Computer Science and Engineering Department,

Lovely Professional University, Phagwara, Punjab.

yashshinde227722@gmail.com

ABSTRACT

The digitalization of the communicating mediums has enhanced the speed of trade and greatly improve the content. Emails, authorities in emails (emails contraction) are increasingly used by individuals and more by businesses due to the fact the 70 billion emails are generated per day. But as far as traditional mails are concerned, users are very rapidly deal with unwanted emails or spams which are mostly quite undesirable. In order to overcome this flood of trash (more than 50% of all email) and not to lose emails that are actually meant to the users , there should is only one practical and feasible solution can be implemented that is to automate the detection and destruction of such type of digital pollution with the risk that a document or misfiled. This paper aims to present the classification of mails as a Spam or Ham (not spam) as well as the progress of spam detection techniques.

Keywords: Classification, E-mail, SPAM, HAM, Tokenization, Stemming, Naïve Bayes, Bag of Words, Term Frequency-Inverse Document Frequency.

INTRODUCTION

Unwanted emails, or spam (trash contraction and emails), represent a very large share of global email traffic. According to various analysis, 5 to 10 billion emails are transfer over the network every day, out of them almost 50% are spam and this number is increasing day by day. Moreover, many of these are spam propagation vectors for some viruses and worms that bring digital security of data stored on our computers to the test. Although it is hard to measure, the cost of digital pollution reflects more than $200 billion per year in the World for Users (lost productivity, connection cost, detection software, etc.). This is enough to understand the importance of application development to fight effectively against this form of pollution.

Difference between Desirable and Undesirable Email: By definition, an email is undesirable when one hand it was not requested and / or content that are either relevant, desired or deemed worthwhile by the user and thus a pollution healthy messages and finished so directly into the trash. But how to judge the relevance of the content of an email? And worse, how to determine that a message is of interest to the user? This touches the heart of email classification problem. By using considerations inherent to the user and the context of use of its email service (work, personal, etc.), it immediately loses the ability to sort messages on strict criteria based on simple characteristics and free from ambiguity. Failing to understand the content of an email, and more importantly, to be able to make a judgment in lieu of the user, it can, in most cases, simply to analyse the structure, origin, destination and edge

effects an email to detect any deviations from a standard derived from the behaviour and context of the user. To detect deviations and suspicious characteristics, the applied techniques will eventually focus on the analysis of user behaviour and corresponding statistical data obtained from the users to perform probabilistic classification. Hence, we can never have absolute certainty about the classification of an email and we always keep to automatically delete junk mail judge.

LITERATURE REVIEW

An implementation on detection of malicious URL in Email is done by Dhanalakshmi R and Chellapan C. considered Age of domain, Host based features, Lexical features and Page rank for analysis of URL to classify into malicious URL and legitimate URL. They have used Bayesian classifier to improve the accuracy by reduced feature sets and considered phishtank dataset, the work was restricted to URL in Email only [17]. Sahami presented a spam classification method using a Bayesian approach. A Bayesian classifier is a statistical classifier works on independence calculation of probability. They have considered content of e-mail with features of domain, shown that accuracy can be increased [18]. V Christina, shown that the need of effective spam filters increases. He discussed spam and spam filtering methods and their correlated problems [19]. Sadeghian A. suggested spam detection based on fuzzy sets. This system gives user more access on categories of spam and grands the comprehension of the spam filter [20]. Congfu Xu derived a feature extraction on Base64 encoding of image with n-gram technique. Effectiveness and efficiency in detecting spam images are shown by these features from legitimate images by training SVM. Experimental results show that it has prominent performance for classification of spam image in terms of accuracy, precision, and recall [21]. Man Qi explored two main semantic methods: Bayesian algorithms and Support Vector Machine (SVM). Recent spam detections are discussed in this paper for classify spam messages which the help of semantic analysis information [22]. Zhan Chuan, LV Xian-liang presented an application to Anti-Spam Email using a new updated Bayesian-based email filter. They have used vector weights for representing word frequency and embrace attribute selection based on word entropy and produced its corresponding formula. It is proved that their filter improves total performances apparently [23]. Holly Esquivel focused on the pre-acceptance altering mechanism IP reputation. They first classify SMTP senders into three main categories: legitimate servers, end-hosts, and spam gangs, and empirically study the limits of effectiveness regarding IP reputation filtering for each category [24]. Georgios Paliouras presented Learning to Filter Spam E-Mail. They investigated the performance of two machine learning algorithm in context of anti-spam filtering by comparison of a Naive Bayesian and a Memory-Based Approach. They have determined the performance on publicly available corpus for naive bayes. And they have compared the performance of the Naïve Bayesian filter to an alternative memory based learning approach so that in both methods accuracy has improved for spam filtering and keyword based filter are used widely for email [25]. Gray Robinson proposed computation of probability of spam mail and legitimate(ham) mail [26].

Solutions Against Spam: There are two types of email classification software. One is directly from the user and is generally based on their behaviour to classify emails and the other is located directly at the entry

point of emails among email service providers, and that will usually (but not only) rely on the signing of emails and associated with the volume of emails passing through similar service to detect abnormal and massive shipments.

PROPOSED METHODOLOGY

We all face the problem of spams in our inboxes. Let's build a spam classifier program in python which can tell whether a given message is spam or not! We can do this by using a simple, yet powerful theorem from probability theory called as Baye's theorem. It is mathematically expressed as

$$P(A \mid B) = \frac{P(B \mid A)\,P(A)}{P(B)},$$

where A and B are events and $P(B) \neq 0$.

- $P(A)$ and $P(B)$ are the probabilities of observing A and B without regard to each other.
- $P(A \mid B)$, a conditional probability, is the probability of observing event A given that B is true.
- $P(B \mid A)$ is the probability of observing event B given that A is true.

Problem Statement

We have a message m = (w*1*, w*2*, ... , w*n*), where (w*1*, w*2*, ... , w*n*) is a set of unique words contained in the message. We need to find

$$P(spam|w1 \cap w2 \cap \ldots \cap wn) = \frac{P(w1 \cap w2 \cap \ldots \cap wn|spam).P(spam)}{P(w1 \cap w2 \cap \ldots \cap wn)}$$

If we assume that occurrence of a word is independent of all other words, we can simplify the above expression to

$$\frac{P(w1|spam).P(w2|spam)\ldots P(wn|spam).P(spam)}{P(w1).P(w2)..P(wn)}$$

In order to classify we had to determine which is greater

$$P(spam|w1 \cap w2 \cap \ldots \cap wn) \; versus \; P(\sim spam|w1 \cap w2 \cap \ldots \cap wn)$$

1.Loading dependencies

We are going to make use of NLTK for processing the messages, WordCloud and matplotlib for visualization and pandas for loading data, NumPy for generating random probabilities for train-test split.

2. Loading Data

We do not require the columns 'Unnamed: 2', 'Unnamed: 3' and 'Unnamed: 4', so we remove them. We rename the column 'v1' as 'label' and 'v2' as 'message'. 'ham' is replaced by 0 and 'spam' is replaced by 1 in the 'label' column. Finally, we obtain the following dataframe

	message	label
0	Go until jurong point, crazy.. Available only ...	0
1	Ok lar... Joking wif u oni...	0
2	Free entry in 2 a wkly comp to win FA Cup fina...	1
3	U dun say so early hor... U c already then say...	0
4	Nah I don't think he goes to usf, he lives aro...	0

3.Train-Test Split

To test our model we should split the data into train dataset and test dataset. We shall use the train dataset t0 train the model and then it will be tested on the test dataset. We shall use 75% of the dataset as train dataset and the rest as test dataset. Selection of this 75% of the data is uniformly random.

4.Visualizing the data

The most repeated or most frequently words occur in the emails whether it is spam or not can be seen by importing the wordcloud library. The bigger the size of the word, the better the probability to occur and vice-versa.

5. Training the model

We are going to implement two techniques: Bag of words and TF-IDF. I shall explain them one by one. Let us first start off with Bag of words.

***Preprocessing*:** Before starting with training we must preprocess the messages. First of all, we shall make all the character lowercase. This is because 'free' and 'FREE' mean the same and we do not want to treat them as two different words.

Then we tokenize each message in the dataset. Tokenization is the task of splitting up a message into pieces and throwing away the punctuation characters. For eg :

Input: Friends, Romans, Countrymen, lend me your ears;

Output: | Friends | Romans | Countrymen | lend | me | your | ears |

The words like 'go', 'goes', 'going' indicate the same activity. We can replace all these words by a single word 'go'. This is called stemming. We are going to use porter stemmer, which is a famous stemming algorithm.

Sample text: Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

Porter stemmer: such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

We then move on to remove the stop words. Stop words are those words which occur extremely frequently in any text. For example, words like 'the', 'a', 'an', 'is', 'to' etc. These words do not give us any information about the content of the text. Thus, it should not matter if we remove these words for the text.

Bag of Words: In Bag of words model we find the 'term frequency', i.e. number of occurrences of each word in the dataset. Thus, for word w,

$$P(w) = \frac{Total\ number\ of\ occurrences\ of\ w\ in\ dataset}{Total\ number\ of\ words\ in\ dataset}$$

and

$$P(w|spam) = \frac{Total\ number\ of\ occurrences\ of\ w\ in\ spam\ messages}{Total\ number\ of\ words\ in\ spam\ messages}$$

TF-IDF: TF-IDF stands for Term Frequency-Inverse Document Frequency. In addition to Term Frequency we compute Inverse document frequency.

$$IDF(w) = \log \frac{Total\ number\ of\ messages}{Total\ number\ of\ messages\ containing\ w}$$

For example, there are two messages in the dataset. ‘hello world’ and ‘hello foo bar’. TF(‘hello’) is 2. IDF(‘hello’) is log(2/2). If a word occurs a lot, it means that the word gives less information.

In this model each word has a score, which is TF(w)*IDF(w). Probability of each word is counted as:

$$P(w) = \frac{TF(w) * IDF(w)}{\sum_{\forall\ words\ x \in train\ dataset} TF(x) * IDF(x)}$$

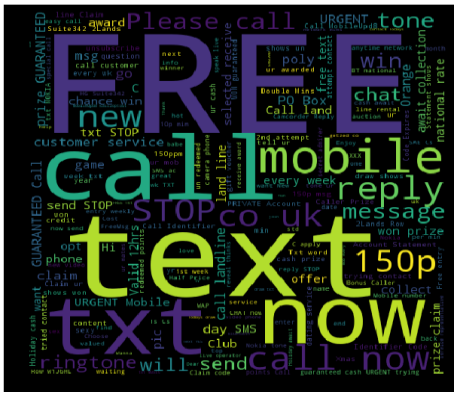
$$P(w|spam) = \frac{TF(w|spam) * IDF(w)}{\sum_{\forall\ words\ x \in train\ dataset} TF(x|spam) * IDF(x)}$$

6. Classification

For classifying a given message, first we preprocess it. For each word w in the processed messaged we find a product of P(w|spam). If w does not exist in the train dataset we take TF(w) as 0 and find P(w|spam) using above formula. We multiply this product with P(spam) The resultant product is the P(spam|message). Similarly, we find P(ham|message). Whichever probability among these two is greater, the corresponding tag (spam or ham) is assigned to the input message. Note than we are not dividing by P(w) as given in the formula. This is because both the numbers will be divided by that and it would not affect the comparison between the two.

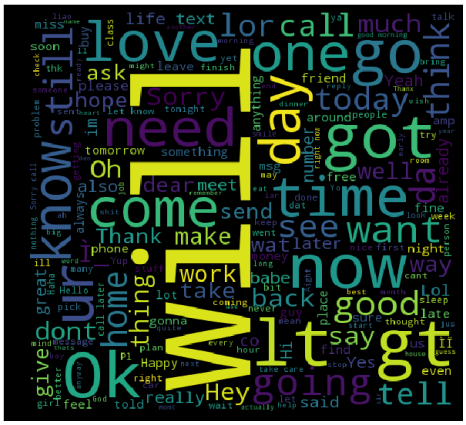
RESULTS AND DISCUSSION

WordCloud for SPAM messages:

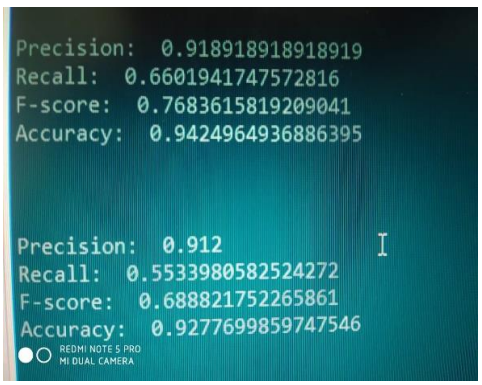


As expected, these messages mostly contain the words like ‘FREE’, ‘call’, ‘text’, ‘ringtone’, ‘prize claim’ etc.

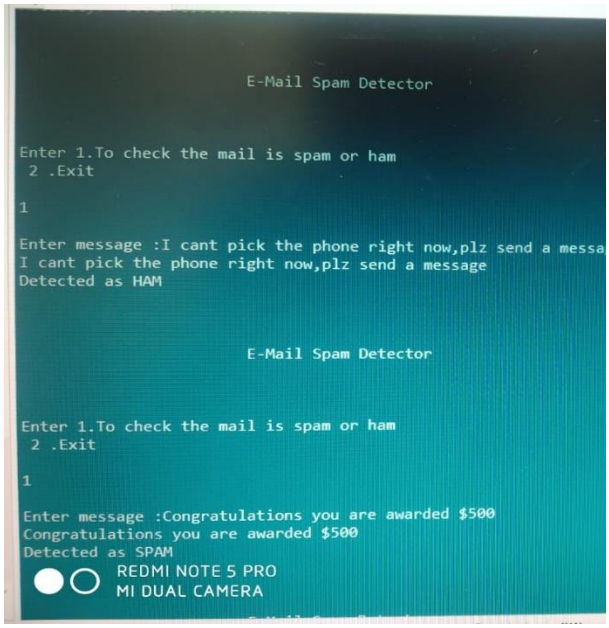
WordCloud for HAM messages:



Confusion Metrics for **Term Frequency-Inverse Document** Frequency and **Bag Of Words** algorithm respectively.



Spam Detector (filter) working as a bot.



CONCLUSION

Whatever the technique we have seen that it is not possible to obtain a correct automatic classification to 100%. However, regardless of the technique used, it is seen that the solutions reach a correct classification rate and especially the False Positive rate quite correct or even well-priced for some. But anyway, do not lose because the choice of technique to be used and the expected performance levels are strongly related to the context of use and the requirements associated with it. For a company, it is not acceptable to lose (by misclassification) a valid email while for a particular this can be much less severe. According to his expectations, so we choose to favour a False Positive rates low or high overall rate of correct classification. Similarly, the choice of solution must take into account the volume of emails to deal with and the sacrifices we are willing to concede. Finally, we must not forget that it is often possible - and preferable - to combine different techniques to achieve the expected performance.

REFERENCES

1. Dataset: https://www.kaggle.com/uciml/sms-spam-collection-dataset
2. http://machinelearning.wustl.edu/mlpapers/paper_files/icml2003_RennieSTK03.pdf
3. Tokenization: https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html
4. Stemming: https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html
5. Stop words: https://nlp.stanford.edu/IR-book/html/htmledition/dropping-common-terms-stop-words-1.html

[6] Dhanalakshmi Ranganayakulu and Chellappan C., "Detecting malicious URLs in E-Mail - An implementation", AASRl Conference on intelligent Systems and Control, Vol. 4 ,2013,pg. 125-131.

[7] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, "A Bayesian approach to filtering junk email", AAAI Tech. Rep.WS-98-05. pp. 55-62, 1998.

[8] V Christina., "A study on email spam filtering techniques", International Journal of Computer Applications, Vol. 12- No.1, 2010.

[9] Sadeghian, A and Ariaeinejad, R., "Spam detection system: A new approach based on interval type-2 fuzzy sets", iEEE CCECE -000379, 2011.

[10] Congfu Xu, Yafang Chen, Kevin Chiew, "An approach to image spam filtering based on base64 encoding and N-Gram feature extraction", IEEE international Conference on Tools with

Artificial intelligence, DOI I0.1109/ICTAl.2010.31,2010.

[11] Zhan Chuan, LU Xian-Iiang, ZHOU Xu, HOU Mengshu, "An Improved Bayesian with Application to Anti-Spam Email ", Journal of Electronic Science and Technology of China, Mar. 2005, Vol.3 No.1.

[12] G. Robinson., "A statistical approach to the spam problem", October 2014.