2016 International Conference on Collaboration Technologies and Systems

Content Based Spam E-mail Filtering

Pingchuan Liu and Teng-Sheng Moh
Department of Computer Science
San Jose State University
San Jose, CA, USA

ABSTRACT - Currently, E-mail is one of the most important methods of communication. However, the increasing of spam e-mails causes traffic congestion, decreasing productivity, phishing, which has become a serious problem for our society. And the number of spam e-mail is increasing every year. Therefore, spam e-mail filtering is an important, meaningful and challenging topic. The aim of this research is to find an effective solution to filter possible spam e-mails. And as we know, in recent days, there are many techniques that spammers use to avoid spam-detection such as obfuscation techniques. In this case, the following proposed approach uses email content only to build keyword corpus, together with some text processing to handle obfuscation technique. The algorithm was evaluated using the CSDMC2010 SPAM corpus dataset that contained 4327 emails in the training dataset and 4292 emails in the testing dataset. The experimental results show that the proposed algorithm has 92.8% accuracy.

Keywords - Spam E-mail, Ham E-mail, Keyword Corpus, Spam E-mail Filtering

1. INTRODUCTION

Currently, E-mail is one of the most important methods of communication. However, the increasing number of spam e-mails causes traffic congestion, decreases productivity, and phishing; which has become a serious problem for our society. Based on a recent Internet Security Threat Report, published by Symantec Corporation [1], in 2012 and 2013, the estimated Global Email Spam Volume per day is about 30 and 29 Billion, and the global average spam rate was 69% and 66%, respectfully. The real cost of spam emails is more than one can imagine. According to a paper [2], it cites the real cost of spam emails could at least add up to $20 billion annually to American firms and consumers, and the cost could be much higher. Therefore, spam e-mail filtering is an important and meaningful topic.

Typically, E-mails are mainly consisted of two parts: header and body. In the header section, it contains many fields that can be categorized into two main types: mandatory and optional. Mandatory fields includes “From”, “To and Cc”, “Sender” etc. Optional fields contains “Subject”, “Message-ID,” etc. Both mandatory and optional fields provide valuable information, like sender’s email address, number of recipients, and subject to help us to classify spam email.

In practice, spam email filtering methods can be categorized into several categories. For example, blacklists and whitelists, IP blocking, header-based filtering and content-based filtering approach. Blacklists, whitelists and IP blocking are relatively the fast way, as compared to other detection approaches, to identify spammers. However, blacklists and whitelists or IP blocking have potential issues that the spammer could change current email account(s) or one IP to another one, in order to escape detection. In this case, normal methods could not easily filter these spam emails. Poor performance and low accuracy are the result of using these approaches.

This paper proposes a content-based spam email filtering approach. The proposed algorithm contains two main phases: training phase and classification phase. In the training phase, individual users’ emails’ are extracted from training datasets. After the email content is collected, the next step was to build a spam and ham keywords corpus that was used to compare with those keywords that were extracted from individual users’ email. Before comparing those extracted words with the spam and ham keywords corpus, in order to improve the accuracy and handle more possible spam techniques, some content processing methods are applied to handle obfuscation techniques that spammer(s) intentionally apply to elude keyword detection; for example, HTML tags removing, insignificant words, and infrequent words filtering. Beside those approaches mentioned above, a weighed scheme for keyword detection is applied in order to improve the accuracy of classification. According to the experimental results, the proposed approach has 92.8% accuracy rate.

The rest of the paper is organized as follows. In section 2, a brief review of present related works of spam email filtering approaches. Section 3 describes a proposed algorithm for content-based spam email filtering. Section 4 presents the experimental results of this work. Section 5 discusses the conclusion of this paper, the future work in spam filtering. Finally, Section 6 lists the references of this paper.

2. RELATED WORK

Compared to other spam email filtering approaches, header-based spam email filtering methods are more straightforward and lightweight, whereas, header-based approaches collect individual senders’ information from

978-1-5090-2300-4/16 $31.00 © 2016 IEEE
DOI 10.1109/CTS.2016.50

email header fields to classify email as spam or not. Aziz, Ismail and Mahdi [3] proposed a method that was based on statistical header feature and sender behavior to identify spam email. They utilized the email header fields to select and extract features, and then used the extracted features to build a feature vector space. Finally, the feature vector space is used to classify spam emails in the classification process.

They used the Trust Value Prediction mechanism as a combined method. The idea of the trust value is also based on the header part. For each sender, the reputation of the sender can be achieved based on his or her historical behavior. If a sender sent a huge amount of emails in the past, then the sender is more likely to be a spammer. On the contrary, if a sender never sent a spam email, then the sender is more likely to be a normal email user rather than a spammer. In the end, they used different machine learning techniques to compare the experimental results. Using the feature extraction and trust value together, the experimental results show that among different machine learning techniques such as Random Forest, Random Tree, C4.5 Decision Tree, Naïve Bayes etc., random forest has the best performance with accuracy, precision, recall and F-Measure of 99.10%, 99.30%, 99.40% and 99.30%. One of the disadvantages of this proposed work is that the behavioral mail header feature still has possible limitation. As we known, there are many ways that spammers may use to elude whitelist or blacklist detection. For example, faking the senders' email addresses. If one of the email addresses is marked as spammer address, the spammer can easily change the current email address to another one. And in the future, the proposed feature could get poor classification results if the spammers keep changing his or her email address.

Pattaraporn and Pattarasinee [4] proposed a pre-send detection or client-side filtering; an e-mail authentication system. The main difference between pre-send and post-send filtering is that pre-send filtering classifies an email as spam or ham before the email is sent to its recipient. Compared to post-send filtering, efficiency within the pre-send method is a major advantage as it reduces the network consumption and the workload of the email server. Once an email is classified as spam, it is no longer needs to respond to the recipient. In this paper, the email authentication system is consisted of two parts; authentication agent and faithful sender agent. The authentication agent is used to generate user profiles for each individual user. The user's profile is treated as an identification of each user. Before the user sends an email, the system will use the profile to identify the current sender. For faithful sender agents, it contains two steps. The first step is to check the application profile. Since, based on the fact, emails are more likely to be legitimate if they are created by some legitimate email applications, instead of some automatic email generators. The second step is to check the spam corpus. Comparing the words extracted from email(s) with the spam keyword corpus. As mentioned before, the major benefit of this approach is that it filters spam emails out before emails can reach to the server, making the workload for email server largely reduced. However, this approach still has its limitations. In the spam corpus processing step, the proposed work used a spam keyword corpus to compare with words were that extracted from individual users' email. The problem faced was that it is entirely possible that spammers could use several techniques to obfuscate the contents in the emails, such as; add some html tags, change some characters in some specific spam keywords or store those email contents into an image. Therefore, a simply keyword corpus is not enough to handle these obfuscation approaches.

Khurum and Asim [5] presented an automatic statistical method to classify spam email without users' feedback through significant word modeling. The significant word model is based on words that were extracted from emails and their frequencies. This algorithm contains two steps. The first step is called the "training step". The algorithm built a statistical model of ham and spam keywords using the training dataset. The second step is called the "specialization step." In the second step, the statistical model of ham and spam keywords is used to classify individual users' email. After classification, the statistical model will be updated. With several experiments in which was used in comparison, we found out that as they keep updating the statistical model more times, the performance improves in some extent. They achieved an optimal accuracy from 82.29% to 99.89%. One of the advantages of this proposed work is that after several updating processes, the performance of the statistical model will be increased significantly. The implementation for this algorithm is comparative easy. However, this proposed work suffers one limitation in that it needs to calculate weighted score for each incoming email word(s) and keeps updating the statistical model after each classification phase in order to get better classification performance. As a result, heavy computations are needed in order to support the proposed algorithm.

Lixin and Geetha [6] presented a hybrid approach that combines local whitelist, keyword processing, and Naïve Bayesian filtering. This approach is also called three-layer spam email filtering. The filter will first check the sender's email address to verify whether the current sender is in the whitelist or not when an email arrives in a mailbox. If the sender's email address is actually in the whitelist, this email will be directly sent to inbox. Otherwise, a keyword processing will find out every significant word and count every keyword in the email to calculate the probability of spam and ham. After this process, if the probability of spam is higher than ham, then this email is classified as a spam email. However, if the probability of spam is lower than ham, an additional process is applied to find out if any special character is significant word matched with the spam keyword. The significant benefit of this hybrid approach is that it used multi-anti-spamming methods to handle more spamming techniques and increases the accuracy of spam filtering. The performance of local whitelist feature in this proposed algorithm is quite limited. Since, as we mentioned before, the spammer could easily change the email address or IP address to avoid address detection. During the keyword-processing step, it is required to calculate the probability of spam and ham for each significant keyword.

Hence, the probability calculation is dependent on the keyword corpus. If some spam keywords are not stored in the keyword corpus, then the classification performance may be affected by this reason.

3. PROPOSED WORK

We proposed a personalized spam filtering based on email content. As shown in Fig. 1, the proposed architecture has two major phases: one that separate email contents from training email dataset, preprocess those contents and then build spam and ham keyword corpus based on the contents. Another phase is called classification phase that used to classifies the testing email dataset using the keyword corpus built from the training section.

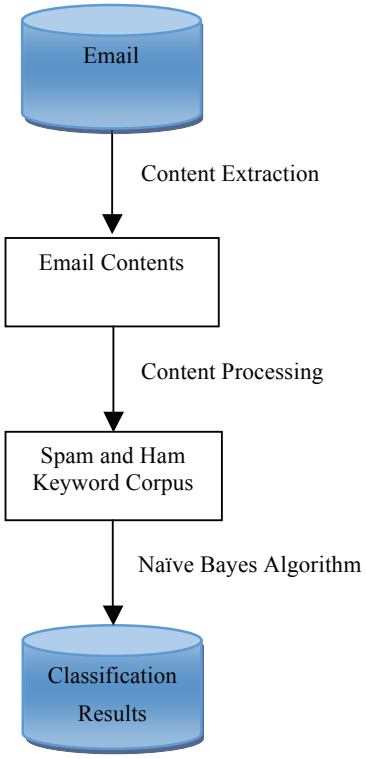

Figure 1. System Architecture

In the training step, the major phase can be divided into several small processes. At the beginning, the process starts to extract email contents from training dataset. Fig. 2 illustrates the processes to build the spam and ham keyword corpus.

After we get the email contents from the training dataset, the next process is to remove HTML tags from the extracted content(s). As we know, sometimes, the extracted email content contains some HTML tags that comes from emails themselves, such as HTML formatted emails, or created by spammers intentionally trying to obfuscate the email content in order to elude keyword based anti-spam detection. As a result, normal keyword-based anti-spam filtering may not be able to detect those obfuscated words. Therefore, the classification result will be somehow affected. In our proposed algorithm, first of all, we tried to find out all existing HTML tags and other special characters among email contents after we got emails from the extracted dataset. If the email contains any special characters, we replace the original characters with graphically closest lower-case letters. For example, '@' would be replaced by 'a' and '\ /' is treated as character 'V' etc. Moreover, digits from a word that do not represent any meaning are considered as null characters and will be removed from the string, for example, passw0rd.

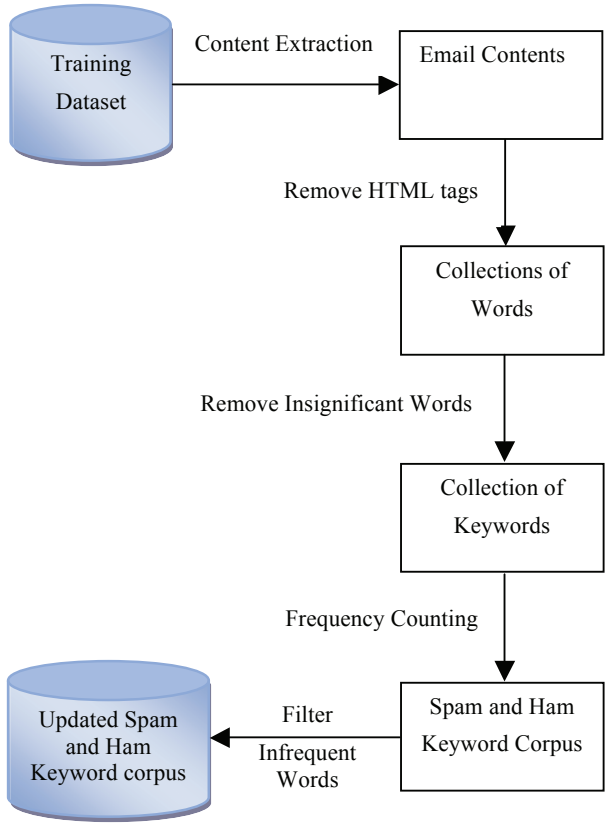

Figure 2. Processes to build Keyword corpus

Hence, we can establish that our email contents are safe to use in the following text-processing steps and avoid the obfuscation that intentionally was created by spammers in some extent. We did the same process for each email and save the words. Last, we arrive with a collection of words.

The next process is to filter some insignificant words such as "I", "You", "She", "He", "the", "a", "an", "am" etc. For every email we used daily, there are some words that have no significant use like "I", "You", "am", "an", "the" etc. Those words cannot be treated as spam keywords and would increase the overall computation time, if we saved them for future text processing. Therefore, we can simply ignore those words for further processing, as the overall processing time will be reduced.

After we removed the HTML tags, insignificant words, and updated collection of keywords; the next step was to build the spam and ham keyword corpus, and count the frequency for each word extracted from contents. The goal for the spam and ham keyword corpus is to keep track of how many times each word appeared in spam keyword corpus or ham keyword corpus. Later, during the classification phase, we can calculate the possibility of spam and ham for all words, of each email based on the frequency.

Initially, we will iterate the collection of keywords for each email and counted the frequency. If the current email is an spam email, and the current keyword is the first time that it appears in the keyword corpus, then we will update the spam keyword corpus and set the frequency count to 1. On the contrary, if current email is a ham email, we can simply update the frequency count in the ham keyword corpus. We would keep updating the frequency count if the same keyword repeats itself again in the future. By the end of this step, we will get the complete spam and ham keyword corpus.

We can get the complete spam and ham keyword corpus after the first three text-processing steps. The goal of this processing is to filter some keywords that are less frequent in the spam and ham keyword corpus. The reason we want to get rid of some infrequent words, is that it is entirely possible that there are some words that appear only one or two times among hundreds or thousands of emails. In this situation, those words are less significant than any other words. Therefore, we want to filter these infrequent words before we use the spam and ham keyword corpus for the classification step.

In our proposed algorithm, we used a threshold to filter some of those insignificant words and kept updating the spam and ham keyword corpus. Basically, we simply iterate all the keywords in both spam and ham keyword corpus and checked the frequency count for each keyword. If the frequency count of any keyword is less than the threshold, we can decide 'if" the keyword is less likely significant in the training dataset, and then delete the keyword and its frequency count, from both spam and ham keyword corpus. On the contrary, if the frequency count is greater than the threshold then we can keep it for future classification step. By the end of this step, after all the text-processing steps mentioned above, now we have the complete updated spam and ham keyword corpus. Fig.3 presents our proposed algorithm for the whole training phase.

After all the processes mentioned before, now we can use the updated spam and ham keyword corpus to classify the emails that were extracted from each testing dataset.

In the classification step, given a testing email dataset, like we did before, the first process still is to extract contents and remove HTML tags. And then filter some insignificant words from the collection of words. The following process is to use both the spam and ham keyword corpus to calculate the possibility of spam or ham email after we get the collection of keywords for each testing email.

The intuitive idea for the spam and ham keyword corpus is to compare the possibility of spam or ham email based on the word frequency count. For example, word "A" shows 15 times among 20 spam emails. On the same time, word "A" shows only one time among 20 ham emails. In this case, word "A" is more likely to be a spam keyword since the possibility of spam word is higher than the possibility of ham word.

Based on this idea, for each email, we can simply calculate the possibility of spam and ham for each word and put the results together. In the end, for an email, if the possibility of spam is greater than possibility of ham then the email is more likely to be a spam email. Otherwise, the email is more likely to be a ham email. The algorithm of our proposed work for testing phase is given in Fig. 4.

4. EXPERIMENT

For the purpose of this section is to present the results of my experiment. Our proposed algorithm is implemented in Python. The data structure we use in python code contains list, set and dictionary. Python dictionary provides an efficient and convenient way to store value with a key and access the value given the key. In our case, we can store each email keyword as the key and the frequency of the keyword as the value. Later on, during the testing phase, we can easily get the frequency of each keyword through keyword lookup. All experiments were performed on 1.3 GHz Intel Core i5 with 4GB memory, running on OS X Yosemite.

Training Phase

```
Input: Training dataset, filter_threshold
Output: Spam and ham keyword corpus
for each email from training dataset:
    Extract email content only and store the content
    Remove html tags and insignificant words from content
    for each word from updated content:
    ignore the case sensitive of word
    if current email is a spam email:
        spam_num++
        if word in spam keyword corpus:
            update the count in spam keyword corpus
        else:
            add the word to spam keyword corpus
            set the count to be 1
    else:
        ham_num++
        if word in ham keyword corpus:
            update the count in ham keyword corpus
        else:
```

```
        add the word to ham keyword corpus
        set the count to be 1
for each word, count in spam keyword corpus:
   if count < filter_threshold:
      remove word from spam keyword corpus
for each word, count in ham keyword corpus:
   if count < filter_threshold:
      remove word from ham keyword corpus
```

Figure 3. Our proposed content based spam email filtering algorithm for Training Phase

A. Dataset

The experiments in this paper use the CSDMC2010 SPAM corpus that is used for the data mining competitions associated with ICONIP 2010. The corpus contains two parts: one is the training dataset that has 4327 email messages consists 1378 spam messages and 2949 ham messages, another part is the testing dataset that has 4292 email messages. The dataset is available at the website [8]. In this research, we take the training dataset and use the first 2327 emails as the training dataset, and the rest, 2000 emails, as the testing dataset.

B. Evaluation Metric

The proposed algorithm is evaluated based on the following metrics:

Accuracy - stands for the fraction that both spam and ham emails are classified correctly among all the emails.

Recall – denotes the fraction of spam emails are founded

Precision – represents the fraction of spam emails classified as spam email.

False Positive Rate - stands for the fraction that all ham emails classified as spam emails.

False Negative Rate - denotes the fraction that all spam emails classified as ham emails.

F-Measure – represents the harmonic mean of recall and precision.

```
Testing Phase
Input: Spam and ham keyword corpus, threshold
Output: Classification result
for each email from testing dataset:
   Extract email content only and store the content
   Remove html tags and insignificant words from content

   prob_spam = 1.0
   prob_ham = 1.0
   for each word from updated content:
      ignore the case sensitive of word
      if word both in spam and ham keyword corpus:
         spam_count = count of word in spam keyword
corpus
         ham_count = count of word in ham keyword corpus
         p1 = (spam_count / spam_num)  /
              ((spam_count / spam_num) + (ham_count
ham_num))
         p2 = 1 - p1
         prob_spam = prob_spam * p1
         prob_ham = prob_ham * p2
         if prob_spam - prob_ham > threshold:

            classify as spam email
         else:
            classify as ham email
```

Figure 4. Our proposed content based spam email filtering algorithm for Testing Phase

C. Experimental Result

Table I presents the experimental result of our proposed algorithm.

We evaluated the performance of each process in order to prove whether each process can improve the accuracy of an entirely proposed work. As we can see in the table, the accuracy of our proposed algorithm without any processes is about 76.4%. And the recall, precision, false positive rate, false negative and F-Measure are 79.9%, 59.2%, 25.2%, 20.1% and 68% respectively. If we used F2 module, which is collecting all the words from email contents, the result is largely increased, the accuracy is increased from 76.4% to 90.3%. And the recall, precision, false positive rate, false negative and F-Measure are 99.0%, 76.7%, 13.8%, 1% and 86.4% respectively. And there are 277 more emails are classified correctly compares to original method. When we combined F1 module that removing html tags from email contents and F2 module together, we can found that the overall classification result is better than F2 module only. About 15 more emails are classified correctly than F2 module. Moreover, when we tested the performance of F2 module and F3 module that used to filter some infrequent words using threshold together, the result is also better than F2 module only. The accuracy and F-Measure is increased about 2%. And 38 more emails are classified correctly compares to F2 module only.

TABLE I. EXPERIMENT RESULTS OF THE PROPOSED ALGORITHM

{}: Keyword only algorithm without F1, F2 and F3

F1: Remove Html Tags

F2: Collecting all the words from Emails

F3: Filter some infrequent words using threshold

	{}	F1 + F2	**F2**	F2 + F3	**F1 + F2 + F3**
Accuracy	0.764	0.91	**0.903**	0.924	**0.928**
Recall	0.799	0.968	0.990	0.984	0.939
Precision	0.592	0.792	0.767	0.812	0.846
False Positive Rate	0.252	0.117	0.138	0.10	0.078
False Negative Rate	0.201	0.032	0.01	0.02	0.061
F-Measure	0.68	0.871	**0.864**	0.889	**0.891**
Diff_Num	0	+292	+277	+319	+327

TABLE II. VERIFICATION OF EXPERIMENT RESULTS OF THE PROPOSED ALGORITHM

	F1 +F2	F2	F2 + F3	F1 + F2 + F3
Diff_Num From Original	480	481	485	497
Ham To Spam Num	200	222	199	173
Spam To Ham Num	280	259	286	324
Correct Num	386	379	402	412
Incorrect Num	94	102	83	85
Subtraction	+292	+277	+319	+327
Correct Fraction	0.804	0.788	0.829	0.829

In the end, when we compared the classification result of F1 module, F2 module and F3 module together, we can found out that the accuracy of our proposed algorithm with all these processes together can reach up to 92.8%. And the recall, precision, false positive rate, false negative and F-Measure are 93.9%, 84.6%, 7.8%, 6.1% and 89.1% respectively. Compared to F2 module only, 50 more emails are classified correctly. The classification result is best among all these experiments.

In order to verify the results we got from Table I, we did some other experiments. Table II presents the verification of experiment results of our proposed algorithm. First of all, if we looked at results of F2 module only, we found out that there were 481 emails moved to another category as compared to the original method. Among those emails, about 222 emails were moved from ham email to spam email, the resulting 259 emails were shifted from spam email to ham email, 379 emails were shifted to the right category, and 102 emails were transferred to the wrong category. Last, there were 277 more emails classified correctly than with the origin method. This number matched the "diff_num" we got in Table I. The results we arrive with here were all the same as we had in the previous experimental results. Therefore, we can make the conclusion that the experimental results in our proposed algorithm in Table I are correct.

TABLE III. EXPERIMENT RESULTS OF THE KEYWORDS ONLY ALGORITHM

F1: Remove Html Tags

F21: Collecting keywords only from Emails

F3: Filter some infrequent words using threshold

	F21	F1 + F21	F21 + F3	**F1 + F21 + F3**
Accuracy	0.764	0.761	0.766	**0.791**
Recall	0.799	0.777	0.795	0.583
Precision	0.592	0.59	0.595	0.701
False Positive Rate	0.252	0.247	0.248	0.114
False Negative Rate	0.201	0.223	0.205	0.417
F-Measure	0.68	0.671	0.680	**0.637**
Diff_Num	0	-7	+3	+54

Beside that, there are some papers that use some specific keywords to build the keyword corpus. Based on this, in our experiments, we tried to test the performance of both these two approaches by using some specific keywords, and collecting all the words extracted from emails. Table III, lists the experiment results of keywords only algorithm. As the results show in Table III, we found out that by using some specific keywords to build the spam and ham keyword corpus and then classify new incoming emails, the overall performance is worse than using all the words that were extracted from emails. The result of F21 module that was used to collect keywords only from email contents presented that the accuracy, recall, precision, false positive rate, false negative and F-Measure are 76.4%, 79.9%, 59.2%, 25.2%, 20.1% and 68% respectively. When we applied F1 module and F21 module together, the classification result is even worse than the origin method. However, if we combined all these three modules together, the classification result was increased from 76.4% to 79.1%. And the recall, precision, false positive rate, false negative and F-Measure are 58.3%, 70.1%, 11.4%, 41.7% and 63.7% respectively. About 54 more emails were classified correctly, as compared to the original method.

TABLE IV. VERIFICATION OF EXPERIMENT RESULTS OF THE KEYWORDS ONLY ALGORITHM

	F1 + F21	F21 + F3	F1 + F21+ F3
Diff_Num From Original	119	59	410

Ham To Spam Num	49	25	42
Spam To Ham Num	70	34	368
Correct Num	56	31	232
Incorrect Num	63	28	85
Subtraction	-7	+3	+54
Correct Fraction	0.471	0.525	0.566

Like we did before, we verified the experiment result of keywords only algorithm. And Table IV presents the verification of experiment result of keywords only algorithm.

Based on those experimental results, we found that the module of collecting all the words from email plays an extremely important role in our proposed algorithm and can largely improve the overall classification results. And if we put all three modules together, the overall accuracy of our proposed algorithm can reach its maximum.

5. CONCLUSION AND FUTURE WORK

In this paper, we proposed a content-based spam email filtering approach. The system uses keyword-based corpus that were built from training datasets to classify new incoming email message. In order to improve the accuracy of our algorithm, we came up with some different processes to handle obfuscated, insignificant, or infrequent words. We performed some experiments to evaluate our proposed work. As can be seen from our results, our proposed algorithm with an accuracy, recall, false positive rate, false negative rate and f-measure of 92.8%, 93.9%, 84.6%, 7.8%, 6.1% and 89.1% respectively.

However, there are still some fields we can improve in the future. For example, currently we are only focus on the email content, however, there are some useful information we can use in the email header part such as sender email address and IP address, email subject, number of recipients or even time. Beside this, users' choice is also a good feature help to detect spam emails. In some cases, even for the best algorithm, the filter can still somehow misclassify some emails. Therefore, the email receivers can get a chance to solve this problem by identifying the email through them. Later on anti-spam system can keep updating the keyword corpus or filter strategies based on the feedbacks that collect from users.

Furthermore, currently, most anti-spam email approaches are client-side based filtering approaches. Therefore, all the emails are classified after the email has already been sent to the recipient. The sending and delivering process already wastes the networks and servers efficiency. Therefore, if the email can be classified before it is sent to a receiver, it can help to reduce the workload of both networks and servers. For instance, a rating system can be applied to determine if user is spammer or not based on user historical behavior. The rating system keeps track of user behavior and set a threshold that how many emails are classified as spam in given amount of time. If the number of spam emails reach or exceed the threshold, system can automatically either send a warning to customer or freeze this account. Then the workload of networks or servers can be reduced from spams.

REFERENCES

[1] 2014 Internet Security Threat Report, Volume 19; Available from:http://www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_v19_21291018.en-us.pdf.

[2] The economic of Spam, 2012. Available from: http://pubs.aeaweb.org/doi/pdfplus/10.1257/jep.26.3.87.

[3] A. Qaroush, I. M.khater, M. Washaha, "Identifying spam e-mail based-on statistical header features and sender behavior," Proceedings of the CUBE International Information Technology Conference, pp. 771–778, 2012.

[4] P. Klangpraphant, P. Bhattarakosol, "E-mail authentication system: a spam filtering for smart senders," Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human, pp. 534–538, 2009.

[5] K. N. Junejo, A. Karim, "Automatic Personalized Spam Filtering through Significant Word Modeling," Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence, Vol.2, pp.291–298, 2007.

[6] L. Fu, G. Gali, "Classification Algorithm For Filter Email Spams," Recent Progress in Data Engineering and Internet Technology Lecture Notes in Electrical Engineering, Vol 157, pp.149, Springer, 2012.

[7] L. Firte, C. Lemnaru, R. Potolea, "Spam Detection Filter using KNN Algorithm and Resampling," Intelligent Computer Communication and Processing (ICCP), 2010 IEEE International Conference on, pp. 27–33, 26-28 Aug 2010.

[8] http://csmining.org/index.php/spam-email-datasets-.html. Nov 2014.

[9] A. S. Ali, Y. Xiang, "Spam Classification Using Adaptive Boosting Algorithm," Computer and Information Science, 2007. ICIS 2007. 6th IEEE/ACIS International Conference on, pp. 972–976, 11-13 July 2007.

[10] M. Sasaki, H. Shinnou, "Spam Detection Using Text Clustering," Cyberworlds, 2005. International Conference on, pp. 319, 23-25 Nov 2005.

[11] S. B. A. Razak, A F. B. Mohamad, "Identification of Spam Email Based on Information from Email Header," Intelligent Systems Design and Applications (ISDA), 2013 13th International Conference on, pp. 347–353, 8-10 Dec 2013.

[12] D. Tran, W. L. Ma, D. Sharma, T. Nguyen, "Possibility Theory-Based Approach to Spam Email Detection," Granular Computing, 2007. GRC 2007. IEEE International Conference on, pp. 571, 2-4 Nov 2007.

[13] R Bocu, "IP Pooling-Based Email Systems Reputation Assurance," Roedunet International Conference, 2011 10th, pp. 1–6, 23-25 June 2011.

[14] S. Dhanaraj, Dr. V. Karthikeyani, "A Study on Email Image Spam Filtering Techniques," Proceedings of the 2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering (PRIME), pp. 49-55, 21-22 Feb 2013.

[15] S. Youn, D. McLeod, "Improved Spam Filtering by Extraction of Information from Text Embedded Image E-mail," Proceedings of the 2009 ACM Symposium on Applied Computing, pp. 1754-1755, 2009.

[16] A. Han, H. J. Kim, I. Ha, G. S. Jo, "Semantic Analysis of User Behaviors for Detecting Spam Mail," Semantic Computing and Applications, 2008. IWSCA '08. IEEE International Workshop on, pp. 91-95, 10-11 July 2008.