Characterizing Comment Spam in the Blogosphere through Content Analysis

Archana Bhattarai, Vasile Rus, and Dipankar Dasgupta

***Abstract*—Spams are no longer limited to emails and web-pages. The increasing penetration of spam in the form of comments in blogs and social networks has started becoming a nuisance and potential threat. In this work, we explore the challenges posed by this type of spam in the blogosphere with substantial generalization regarding other social media. Thus, we investigate the characteristics of comment spam in blogs based on their content. The framework uses some of the previously explored methods developed to effectively extract the features of the blog spam and also introduces a novel method of active learning from the raw data without requiring training instances. This makes the approach more flexible and realistic for such applications. We also incorporate the concept of co-training for supervised learning to get accurate results. The preliminary evaluation of the proposed framework shows promising results.**

I. Introduction

Spam has grown to a significant level, polluting the cyberspace in different ways like emails, web-pages, blog posts, blog comments, instant messaging and social network comments. According to MessageLabs[1], the spam rate of emails is around 70.1% of the 2.7 billion messages per day as of September, 2008. Moreover, there are around 3660 malicious websites introduced per day as reported by MessageLabs. Spammers also target social networking sites such as Facebook and Myspace. Such spams, spread in different forms, are being used to unethically advertise products, distribute different malware, spread viruses and steal personal information thus posing a strong threat to the community. Another contributing factor to the increase in number of spam-pages is the use of search engine optimization (SEO). SEO is the process of fabricating or organizing web content across the web to increase its potential relevance to specific keywords on search engines [2]. With the increase in the use of search engines, SEO has become very popular because significant amount of traffic these days results from search engine referrals. Many websites try to manipulate the ranking functions of search engines by using less-ethical gray-hat and black-hat SEO techniques. These are achieved by the creation of extraneous pages, which link to a target page. Due to essential financial benefit of SEO, spam web pages are aggressively attempting to make their sites appear more relevant to search engines. One of the important criteria used to measure the relevance of a page by search engines like Google, Yahoo, MSN, Ask etc is "link weighting"[2]. Link weighting is based on the concept that if a page is referred by many other pages, the relevance of this target page increases. An important aspect of link weighting is that the rank of the page will be higher if it gets a reference from a high ranked page. Spammers try to exploit this property to get better rankings for their sites by utilizing open and easy access to comment sections of blogs by creating a new type of spam known as "comment-spam".

Blogs are gaining constant popularity in recent years and have now become an important genre of web content. A blog is a type of web content which contains a sequence of periodic (dated) articles with user comments and opinions [4]. According to statistics taken in mid 2005, there were 14.2 million blogs existing worldwide, and the number was predicted to double roughly every 5.5 months [4]. Thus, with such a proliferation of blogs, blog spams and comment spams will also increase at least in the same ratio. Moreover, it is even easier to exploit blogs' comment section as they are open by nature to facilitate commentators to write their opinions about the piece of writing. Along with comments, people would also like to link the article with other posts in the blogosphere to express their opinions in relation to these posts. This results in links between pages of a blog or different blogs. While this kind of link can be a legitimate hyperlink to relate two different posts, spammers exploit this concept to increase their link weights by posting random comments and links in blogs.

Various initiatives have been taken to reduce these kinds of comment-spams. For example, many blogs now require users to register before they post any comment. However, this restriction does not seem to hamper spammers completely. Major search engines like Google, Yahoo, MSN, etc. came up with an alternative solution to add an attribute "rel= nofollow" to the hyperlinks that are automatically generated in the pages[3]. But this solution also does not seem to stop spammers from posting spam links in blog comments as the legitimate comments too do not care about the attribute. Other methods are also used to block comment spams using different machine learning algorithms like Bayesian Networks based on the email-spam models and other models[5-7]. However, no spam filtering system has been able to successfully eliminate spam comments. There are fundamental differences between spams in the form of emails and web pages. While email spams are more intended to real users, comment spams mostly are targeted to search engines. Email spams are filtered based on recurrent features such as use of some specific words. However, comment spams may not contain the same kind of features. Spam-comments are

Bhattarai A., Rus V. and Dasgupta D. are with The Department of Computer Science, The University of Memphis, Memphis, TN 38152-3240, USA. e-mail: abhattar@memphis.edu, vrus@memphis.edu, dasgupta@memphis.edu

relatively shorter which make it more difficult to distinguish from legitimate spams. Another important characteristic of blog comments that make it fundamentally different from emails is that blog comments have a strong cohesion with the post and has a gradual build-up on a topic while emails are independent writings in themselves.

The rest of the paper is organized as follows. In section II, we survey the evolution of work done in the area of spam, starting from email, webpage, blog and short text spams like comments spams. We then talk about different features with their complete analysis in section III. In section IV, we present the system framework built, document preprocessing steps, the semi-supervised spam detection technique and the supervised spam detection along with experimental results. We then conclude in section V and give a brief idea of possible future works in section VI.

II. Related work

Extensive study has been done on different genres of spam like email-spam, web-spam, blog-spam (popularly known as splogs) although not much has been done specifically in the short text type spams such as comment-spam. Classification of email spams started in the early 90s. Sahami et. al.[20] used Naïve Bayes classifier to classify text-based emails. Druker et al.[21] evaluated Support Vector Machines to deal with spams. Later in 2001, Carreras and Marquez[22] showed that AdaBoost is more effective than decision trees and Naïve Bayes. Zhang et al.[23] used link-spam as a feature and compared Naïve Bayes, Support Vector Machines and LogitBoost in their work.

Web-spam filtering is a relatively new field compared to email spams. Davison[24] who was the first to investigate link-based web spam in 2000 built decision tress to identify nepotistic links. Later in 2005, Becchetti et al.[26] again used decision trees to identify link based web-spam with features such as PageRank and trustRank. Drost and Scheffer[27] in 2005 used SVM to classify web spam with content and link based features. Ntoulas et al[14] in 2006 built decision trees to classify web-spam with content based features.

As the blog spam surfaced in 2005, Umbria[18] noted such spams in the blogosphere. Gyongyi and Hector[16] have discussed different types of spams, their techniques of spam generation, techniques of hiding spams and their target algorithms on the web etc. Kolari et al.[17] used SVM models based on local features like bag of words and N-gram features and link-based features to detect splogs. Han et al.[4] proposed a collaborative filtering method for combating link spams. Their idea was to rely on manual identification of spams and share this information through a network of search. Provos et al.[19] identified the mechanisms used to inject malicious content on web pages. They also illustrate the current state of malware on the web showing the requirement for research in the field.

In 2005, Mishne et al.[11] did their work focusing specifically on comment-spam. Their idea was to develop language models for the blog post, blog comment and the pages linked by comments and classify the comments based on the language model disagreement. Later in 2007, Cormack et. al. in [9] conducted a broader analysis on filtering of short messages. They evaluated different content based filtering systems implementing algorithms like Naïve Bayes, Support Vector Machines, Dynamic Markov Compression and Logistic Regression using bag-of-words, orthogonal sparse bigram features and compression model based approach on short text message, blog-spams and email summary information.

Our work focuses on short text type spams, specifically on comment-spam in the blogs. Unlike existing works, we use seven unique aspects to analyze spam and legitimate comments and classify them based on both semi-supervised and supervised learning.

III. Characterizing content spams

In this section, we analyze different features of a spam/legitimate comment, all of them based on the content of the comments and the post, which may prove helpful for the detection of comment spams. Here we define spam-comment as any unsolicited comment sent in response to a blog post which is not a spam, and a legitimate comment is a non-spam comment which is often also termed as ham comment. Most of the time these spam comments are generated automatically with a computer program. Legitimate comments often have a specific pattern whereas spam comments exhibit slightly random characteristic. We developed a system to analyze the comments based on the characteristics mentioned below.

A. Dataset

To analyze the characteristics and evaluate the detection algorithm based on features, we used the corpus created by Mishne and Carmel[8]. The corpus contains around 50 random blog posts with 1024 comments posted to them. All the posts contain a mixture of spam and legitimate comments. Comments in the corpus were manually classified, leading to 332 legitimate comments and remaining being spam. This number itself illustrates the pervasiveness of spam comments in blogs and with the growing popularity of blogs we should expect more spams indicating a serious need of action in the field. All the spam comment excerpts depicted in the following sections are taken from the dataset[8] itself.

B. Post-Comment Similarity

Spammers use automated scripts to produce myriads of spams for the generation of spams. However, these automated spam comments are not related to the context of the post. Thus, the goal is to try to analyze the coherence of the comments compared to the post for which the comments are written. Legitimate comments are expected to have more coherent phrases compared to spam.

Hi, I just wanted to say thank you guys! I really like your site and I hope you'll continue to improving it.

Figure 1: Example of a comment spam which is not related to the subject matter [8]

Figure 1 shows an example of a spam comment which is not related to the subject matter. This kind of spam, which looks very much like a legitimate one, is most difficult to detect as it may be confusing even to a human analyst. It is also to be noted that some legitimate comments too do not have coherent words in them. Hence, just relying on post-comment similarity is not the best idea. However, it can help in the overall prediction of spams when combined with other features.

The post-comment similarity value is calculated taking inner product of all representative words in the post and comment and then normalizing it with the length of comment. The post-comment similarity for a post P_j and comment C_k can be expressed by the following relation:

$$Similarity \quad (P_j, C_k) = \frac{\vec{P}_j . \vec{C}_k}{|P_j||C_k|} = \frac{\sum_{i=1}^{n} w_{i,j} w_{i,k}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2} \sqrt{\sum_{i=1}^{n} w_{i,k}^2}}$$

Here, $w_{i,j}$ is the frequency of the word occurring in the blog post. Since blog comments are relatively short compared to other documents, most of the time the weight is just 1. We plotted a graph to analyze the behavior of spam and legitimate comments based on their similarity to the posts. Figure 2 below shows the distribution of spam and non-spam comments based on post-comment similarity.

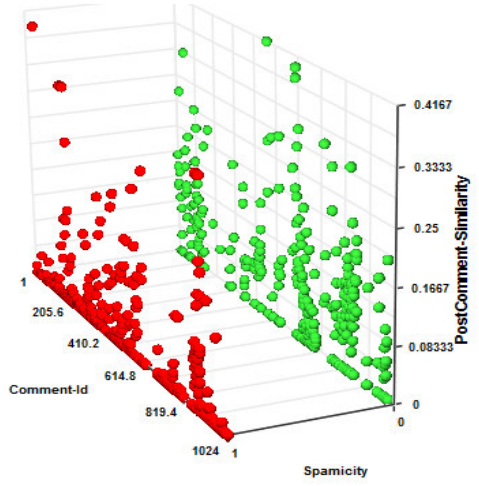

Figure 2: Distribution of spam and legitimate comments based on post-comment (spam comments with spamicity=1)

Here, the front horizontal axis represents the similarity of a comment to a post with scale 0 to 1. Spamicity value of 1 represents a spam comment whereas 0 represents a legitimate comment. Comment Ids are just unique identification numbers. The distribution indicates that most of the comment spams have similarity of less than 0.0833 to the posts, excluding some outliers whereas legitimate comments have higher similarity values.

One of the obvious extensions to our work would be to add synonymous or hyponymous words both in the post and the comment using tools liks WordNet to increase the accuracy of this feature in detecting spam comments. However, we have not included this part in our current work. The average post-comment similarity of spam comments was 0.018 whereas the average post-comment similarity of legitimate comments was found to be 0.073.

C. *Word-duplication*

A careful analysis of spam comments revealed that most of the spam-comments have the same words repeated once and again following a certain pattern very likely to attract search engines, whereas a legitimate comment is often a continuous flow of content related text. As most blog comments are short in nature, the same word rarely repeats in a legitimate comment. Motivated by this idea, we analyzed the behavior of blog comments based on their word repetition pattern.

Word redundancy in our context is defined mathematically as follows:

$$Word\ \text{Re}\ dundancy\ \ Ratio = 1 - \frac{Number\ \ of\ \ unique\ \ \ words\ in\ the\ comment}{total\ \ number\ \ of\ \ words\ in\ the\ comment}$$

Get a Generic Viagra alternative at Cheap Generic ViagraGet a Cheap Generic Viagra alternative at Cheap Generic Viagra

Comments posted by: Cheap Generic Viagra at March 11, 2005 09:27 PM

Figure 3: Example of a spam comment which has redundant words in it [8]

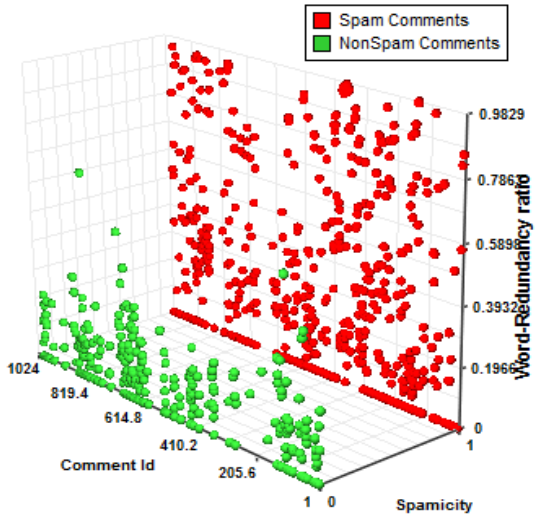

Figure 4: Distribution of spam and non-spam comments based on word redundancy ratio

Figure 3 above shows an example of a comment where same words occur repeatedly in a short comment. Figure 4 depicts the distribution of comments based on word redundancy ratio. The distribution indicates that legitimate comments have fairly low word-redundancy compared to spam

comments which have a redundancy ratio as high as 0.9. The average redundancy ratio for legitimate comments was found to be 0.098 whereas it was 0.265 for spam comments.

D. *Number of AnchorTtexts*

Since most of the comment spams are intended for web crawlers as opposed to humans, these comments try to mention many anchor texts which point to their spam sites, eventually increasing their page rank. Thus, it is evident that spam comments tend to have higher anchor-texts compared to legitimate comments. An example with lots of anchor texts is shown in figure 5 below.

hold em http://texas-hold-em.musicbox1.com/ online poker
online poker http://online-poker.musicbox1.com/ phentermine
phentermine http://phentermine.musicbox1.com/ interest only

Figure 5: Example of a spam comment containing lots of URLs [8]

Our analysis of comments based on anchor text count can be summarized by figure 6 below. It shows that almost all legitimate comments have three or fewer anchor texts whereas spam comments were found to have as many as 233 anchor-texts in the dataset. This clearly indicates that the anchor-text count can be a good feature to detect spam and non-spam comments. The average anchor-text count in spam comments was 6.35 whereas for legitimate comments, it was 0.14.

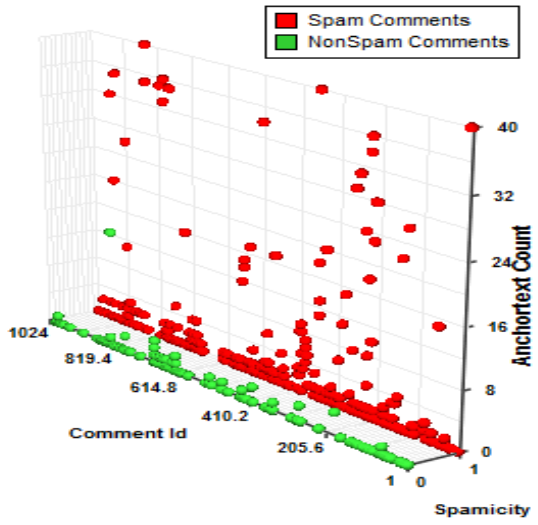

Figure 6: Distribution of spam and non-spam comments based on number of anchor texts

E. *Noun Concentration*

One of the main goals of spam comments is search engine optimization of spam pages. Thus, most auto-generated spam comments are populated either by some keywords in the form of noun-phrase chunks without the formation of a complete sentence or some links to keep the crawler going. These spam comments are expected to have higher concentration of noun phrases compared to other word/phrase categories like verbs, prepositions, etc. However, regular comments are intended more to express an idea using grammatical sentences or phrases. Therefore, the hypothesis here is that spam comments have higher noun concentration whereas non-spam comments do not. Figure 7 shows an example of such concentration of noun in spam comments.

We used OpenNLP [30] tools to extract sentences from the comment and part-of-speech tags for the sentences. Noun-Concentration in our work was calculated using the following fomula:

$$NounConcentration = \frac{No.of\ noun\ phrases\ present\ in\ the\ comment}{total\ \ No.\ of\ words\ in\ the\ comment}$$

Architectural & Garden Asian Antiques Books, Manuscripts Foll
Men's Clothing Coins: US Coins: World Exonumia Animation Art,
Dollhouse Miniatures Dolls Paper Dolls Music Video Games Gen
Books & Manuscripts Business & Industrial Equipment Guitar Ha
Sporting Goods Australia Br. Comm. Other Canada Other Items

Figure 7: Example of a comment spam which has high noun-concentration [8]

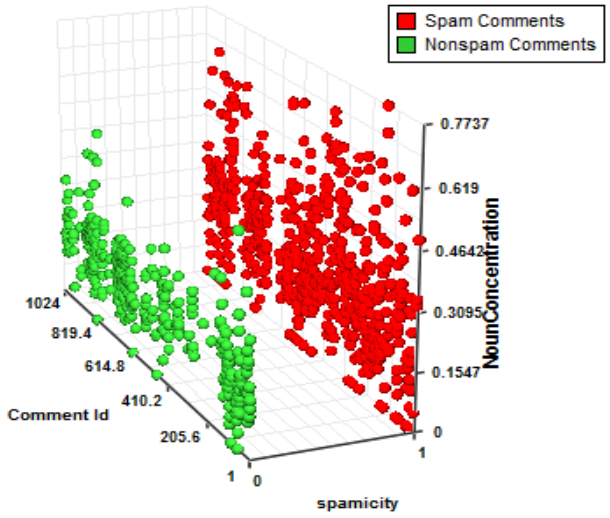

Figure 8: Distribution of spam and non-spam comments based on noun Concentration

Figure 8 depicts the distribution of spam and legitimate comment in blogs. The analysis shows that legitimate comments almost always have noun-phrase concentration of less than 0.4 whereas for spam comments the concentration was found to be as high as 0.7. Although there are spam comments with low noun-phrase concentration, we can always filter a group of spam comments which have high noun-concentration. The average noun-concentration for legitimate comments was found to be 0.196 whereas for spam comments it was around 0.26.

F. *Stopwords Ratio*

Using a similar explanation as noun-phrase concentration, we can hypothesize that legitimate sentences tend to have a fairly balanced stopwords ratio compared to spam comments as shown in figure 9.

> Football betting football betting line online football betting college football betting football betting odds Pro football betting nfl football betting ncaa football

Figure 9: Example of a comment spam where there is no stopword [8]

Stopwords ratio is calculated using the following formula.

$$Stopwords \quad Ratio = \frac{Number \ of \ stopwords \ present \ in \ the \ comment}{total \ number \ of \ words \ in \ the \ comment}$$

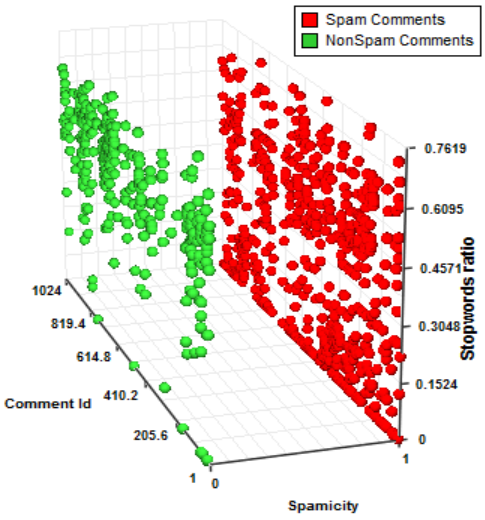

Figure 10: Distribution of spam and non-spam comments based on stopwords ratio

The distribution of spam/legitimate comments based on stopword ratio in figure 10 shows that legitimate comments almost always have a stopwords ratio in the range 0.3 to 0.61 but spam comments have a wide variation in the stopwords ratio. Clearly, as indicated by the graph, comments with less stopwords ratio are more likely to be spams than with high stopwords ratio.

G. *Number of Sentences*

Some spam comments tend to be relatively longer as they try to add a lot of anchor-texts and jargon keywords. Figure 11 shows that the sentence count is not so informative for the dataset we used. Despite this fact, we believe this feature may play an important role in a scenario where there as plentiful spams with relatively high count of sentences.

Thus, the graph in Figure 11 shows the distribution of the number of sentences in spam and legitimate comments. The average number of sentences in spam comments was found to be 5.7 whereas it was 4.08 for legitimate sentences. The corpus had instances which had as high as 233 sentences in them when they were spams while legitimate comments had a maximum of 23 sentences.

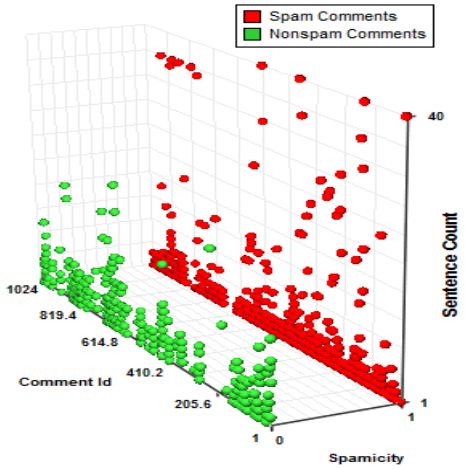

Figure 11: Distribution of spam and non-spam comments based on sentence count

H. *Spam Similarity*

The details of how we compute spam-similarity are explained later in the semi-supervised spam detection section (section IV *B*). Spam similarity of any comment is expressed as

$$SpamSimila\,rity(C_k) = \frac{\sum_{i=1}^{n} w_{k,i}}{i}$$

Here we summarize the distribution of the comments based on their spam-similarity.

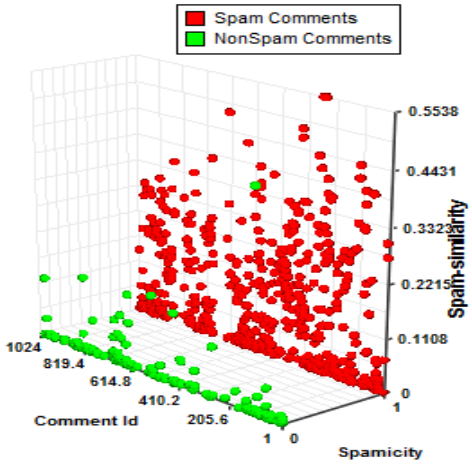

Figure 12: Distribution of spam and non-spam comments based on spam-similarity of the comment

Figure 12 shows that the spam similarity of a legitimate page is almost 0 in most cases whereas spam comments have their similarity as high as 0.55, thus indicating that this feature could be a potential deterministic feature for the detection of spam comments. The average spam-similarity of a legitimate

comment was 0.011714 whereas for spam comments, it was 0.096.

IV. Architecture for Spam Detection

Figure 13 below shows the framework we used for the spam detection. Initially, from a blog page, the post and the comments are extracted. Both the post and comments have to be preprocessed before any feature value is extracted. The preprocessing step includes tokenizing, stemming, removing stopwords etc which is performed in block A, as shown in figure 13. After preprocessing, the feature values such as post-comment similarity, word redundancy ratio, number of anchor texts, noun concentration, stopwords ratio and number of sentences are calculated. Some feature extraction such as stopwords ratio requires using raw comments rather than preprocessed comments. After calculating the feature values for each comment, these features are used to extract the spam likelihood data in block C in figure 13. At this step we create a hash with all the bag-of-words features and their frequencies in the probabilistic spam comments as indicated by block B. These values are then used to obtain the spam likelihood of any comment in blocks D and E. Semi-supervised classifier in block E only uses this information to classify the comments whereas the supervised classifier in block D uses this information along with some pre-classified training data to build a model which classifies the comments.

A. *Document Preprocessing*

Blog posts and comments are available in a free form text which has to be preprocessed before features are extracted from them. We have performed word tokenization, stopwords removal, stemming, sentence detection etc before we extract the feature. Grammatical shallow parsing was also done to extract noun-phrases.

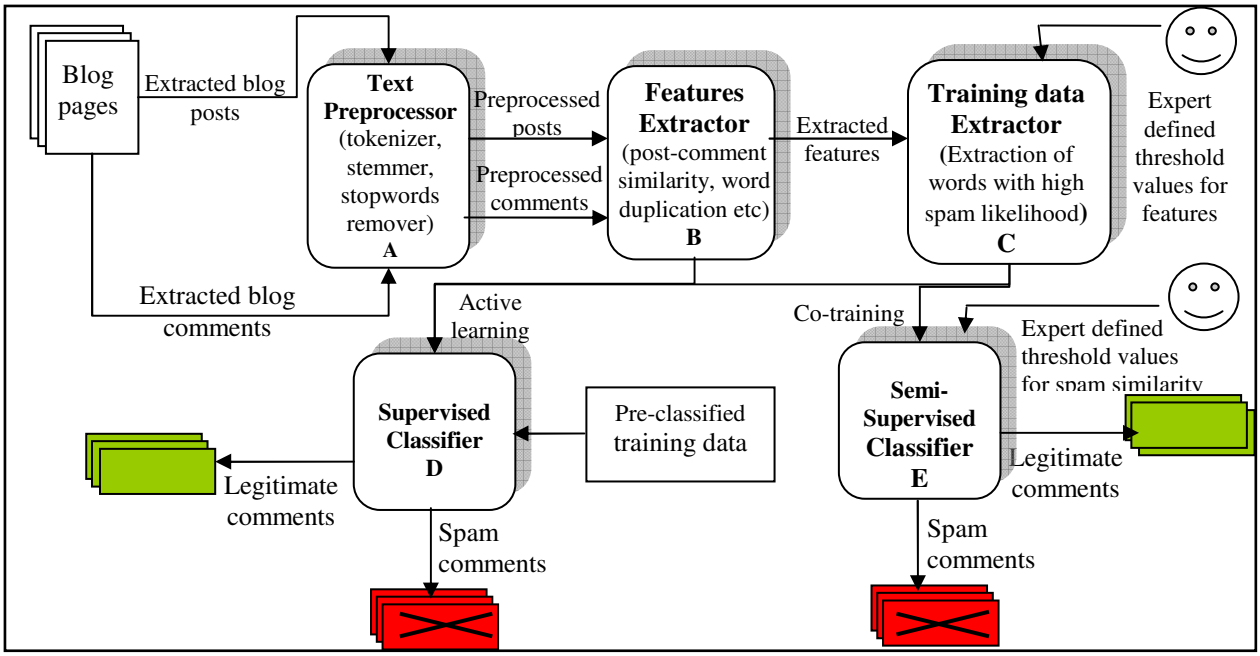

Figure 13: A framework for the spam comment classification system

B. *Semi-supervised Spam detection*

Many spam comments can be blocked just by checking some keywords in the comment and blocking based on the presence of those keywords. The challenge here is to collect the set of keywords which are indicative of spamicity. It may be argued that the set of words can be collected manually or by other means and then used. However, these word sets change dynamically and spammers find new ways to express them thus avoiding the collection. For example, we consider collecting the word "Viagra" as a spam word. But then spammers may start spamming with words like v1agra, vi@gra etc. Even partial words and regular expressions may not be able to capture all the words. Thus, our goal should be to dynamically collect such spam words without any external training data and then use those words to detect spam comments. For this purpose, we used the idea of co-training to actively learn from the given raw data. We adopted the idea of co-training from Blum and Mitchell[29]. They have mentioned that a description of a problem can have multiple distinct views. Following the same idea in our work, we describe a comment in a blog with six different views namely: post-comment similarity, word-duplication, noun-concentration, stopwords-ratio, number of sentences and number of hyperlinks. The whole process can be divided into two steps. In the first step, we collect all the comments which were likely to be spam comments.

To explain the concept mathematically, we define any comment instance X in an instance space as

$$X = X_1 \times X_2 .....X_n$$

Here, $X_1 \ldots X_n$ represent unique views such as post-comment similarity, noun-concentration, stopwords ratio etc. In our case, since we only have two classes, namely spam and non-spam, each view is assigned a threshold value heuristically by a domain expert, which separates the boundary between spam and non-spam classes. Thus, at the first stage, any instance X is classified spam if its value is in the spam area in any of the views as described in section 3. Mathematically, it can be expressed as

$$Spamicity(X) = f(X_1) \vee f(X_2) \vee \ldots\ldots\ldots\ldots f(X_n)$$

This step, described as training data extractor in figure 13 is more relaxed as it is the union of all the comments classified as spam in the view space. For example, if a comment has a redundancy ratio less than 0.2, we consider it as spam without caring for its values in other view spaces. It works on each view space.

Now at the second stage, from the set of spam comments obtained in step 1, we extract bag-of-words features after preprocessing the data, i.e. removing stopwords, stemming, tokenizing etc. The weight of each feature is defined by the following relation:

$$W = \frac{frequency\ \ of\ \ word\ \ in\ the\ spam\ corpus}{Maximum\ \ frequency\ \ of\ \ a\ word\ \ in\ the\ spam\ corpus}$$

After calculating the weight of each word in the derived spam corpus from the first stage, we then calculate the similarity of any comment to the spam corpus. A threshold value is assigned heuristically by us, which defines the boundary of spam and non-spam comments. Spam similarity of any comment is expressed as

$$SpamSimila\ \ rity\ (C_k) = \frac{\sum_{i=1}^{n} w_{k,i}}{i}$$

The method described above is semi-supervised as the only external input that needs to be provided to the system is the threshold value for each view space and spam-similarity. Table 1 shows the statistics of threshold values used by us for the experiment.

Stage 1 Classification

Thus, we extract the spam comments which have their corresponding values less than the threshold values as mentioned in table 1. The system performance of the semi-supervised system is shown in table 2 below.

After extracting training instances, we collect all bag-of-words features from the instances. Before collecting bag-of-words features, we remove stopwords and also stem all the words in the instance.

Features	Threshold value
Post-comment Similarity	0.08
Noun-phrase concentration	0.4
Stopwords ratio	0.4
Redundancy ratio	0.2
Anchor text count	4
Sentence count	4

Table 1: Threshold values for different features for training data extraction system

Accuracy	Precision	Recall	F-Measure
72.265625	72.08068	97.0	82.70402

Table 2: Performance measure of the spam corpus extraction system

Then for the weight calculation of each of the features, we calculate the frequency of the word in the collected instances. We normalize the frequency of each word dividing by the maximum frequency in the collection. The higher the frequency, the more likely the word is to be a spam word. This value is then used as the weight for keywords in the spam comments to further classify blog comments. Table 2 shows the performance measure of this stage 2 classifier with different threshold values.

Stage 2 Classification

Table 3 shows the performance measure of the developed semi-supervised classification system at different threshold values of spam similarity. We see a clear tradeoff between recall and precision here. The table shows that the threshold value can be tweaked according to our need. If we need a very precise result at the cost of low recall, we can use a higher threshold value whereas a low threshold value can be used to get better recall at the cost of relatively low precision.

C. Supervised Spam detection

The spam filtering system can be modeled as a binary text classification problem. Many classification methods have been applied to the problem of document classification. These algorithms learn from a set of text that has already been classified (training set) to classify another set of documents (test set). A key difference between these methods is the way the documents are represented by the features selected (most often words or phrases from the text). Here, we attempted to compare some of these supervised algorithms, Naïve Bayes' classifier, Support Vector Machines (SVM), logistic regression, decision trees etc to evaluate the performance of each in detecting spam comments.

Threshold value	Accuracy	Precision	Recall	F-Measure
0.5	32%	100%	28.57%	57%
0.03	66%	94%	53%	68.3%
0.01	74.70%	92.32%	68.71%	78.79%

Table 3: Performance measure of spam-similarity based classifier for spam comments detection

Classifying algorithms	Accuracy	Precision	Recall	F-Measure
Naïve Bayes Classifier	71.94%	94%	62%	75%
Support Vector Machine (SMO)	83%	88%	87%	87%
Neural Network (Multilayer Perceptron)	83%	87%	88%	87%
Logistic Regression	84%	88%	88%	88%
Decision Tree (J48)	86%	90%	88%	89%

Table 4: Performance result of various algorithms for the detection of spam comments based on co-trained data

Table 4 above shows the evaluation result of various algorithms for the detection of spam comments. From the results, we see that Naïve Bayes' algorithm is the best with respect to high precision. However, it has a very low recall. J48 based Decision tree on the other hand exhibits an overall best performance on accuracy, precision, recall and F-measure. The result was obtained with ten-fold cross validation of data.

V. CONCLUSION

In this work, we have introduced a framework for content based spam detection without requiring any extra information. Apart from the post-comment similarity, other methods can be equally applied to any form of short message spams such as instant messaging spams, social network spams, etc. Although our system does not have a perfect detection mechanism, the undetected ones are more in the grey area even to a human analyst. The dataset used for analysis was not a large dataset. Thus, our future work would be to evaluate our methods on a larger dataset. We also plan to improve our post-comment similarity value by representative keyword expansion using tools like WordNet[28]. The evaluation of the analyzed features on other type of short spams such as instant messaging spams, short messaging service spams and other comment spams is also a possible extension.

REFERENCES

[1] Messagelabs. http://messagelabs.com.

[2] Wikipedia. http://en.wikipedia.org/wiki/Search_engine_optimization.

[3] Nofollow. http://googleblog.blogspot.com/2005/01/preventingcomment-spam.html.

[4] Han, S., Yong Y. A., Sue, J. M. and Hawoong., Collaborative Blog Spam Filtering Using Adaptive Percolation Search. WWW2006, Edinburgh, UK, 2006.

[5] Raymond, E. S., Relson, D., Andree, M. and Louis, G. , http://bogofilter.sourceforge.net, 2004.

[6] Assis, F., A text classification module for Lua – the importance of the training method. In Fifteenth Text Retrieval conference, Gaithersburg, MD, 2006.

[7] Bratko, A., Cormack, G. V., Filipi˘c, B., Lynam, T. R. and Zupan, B., Spam filtering using statistical data compression models. Journal of Machine Learning Research , 2006.

[8] Mishne, G., and Carmel, D. , Blocking blog spam with language model disagreement, Proceedings of AIRWeb, 2005.

[9] Cormack, G.V., Gomez, J. M. and Sanz, E. P., Spam Filtering for short messages, 2007.

[10] Hearst, M.A., Hurst, M. and Dumais, S. T., What Should Blog Search Look Like, SSM, Napa Valley, California, USA, 2008.

[11] Mishne, G., Multiple Ranking Strategies for Opinion Retrieval in Blogs, 2006.

[12] Ku, L., Liang, Y. and Chen, H., Opinion Extraction, Summarization and Tracking, American Association for Artificial Intelligence, 2006.

[13] Webb, S., Caverlee, J. and Pu, C., Characterizing Web Spam Using Content and http session Analysis, CEAS, Fourth Conference on Email and Anti-Spam, 2007

[14] Ntoulas, A., Najork, M., Manassee, M. and Fetterly, D., Detecting Spam Web Pages through Content Analysis, WWW, Edinburgh, Scotland, 2006.

[15] Castillo, C., Donato, D., Becchetti, L., Boldi, P., Leonardi, S., Santini, M., Vigna, S., A reference collection for web spam, SIGIR forum, 2006.

[16] Gyongyi, Z. and Hector, G, Web Spam Taxonomy, 2005, Adversarial Information Retrieval on the web, AIRWeb

[17] Kolari P., Java A., Finin T., Oates T. and Joshi A., Detecting Spam Blogs: A Machine Learning Approach, AAAI 2006

[18] Umbria. Spam in the blogosphere http://www.umbrialistens.com/consumer/showWhitePaper, 2005

[19] Provos N., McNamee D., Mavrommatis P., Wang K.. and Modadugu N., The Ghost In The Browser Analysis of Web-based Malware, Google, 2007

[20] Sahami M., Dumais, S. D., Heckerman, E. and Horvitz, A Bayesian approach to filtering junk e-mail. AAAI Workshop on Learning for Text Categorization, 1998

[21] Drucker, H., Wu, D., and Vapnik, V., Support vector machines for spam categorization. IEEE Transactions on Neural Networks, 1999

[22] Carreras, X., and Marquez, L., Boosting trees for anti-spam email filtering. In Proceedings of RANLP-01, 4th International Conference on Recent Advances in Natural Language Processing, 2001

[23] Zhang L., Zhu J, and Yao T.. An evaluation of statistical spam filtering techniques ACM Transactions on Asian Language Information Processing, 2004.

[24] Davison B. D., Recognizing Nepotistic Links on the Web, 2000

[25] Davison B.D. and Wu B., Identifying link farm pages. In Proceedings of the 14th International World Wide Web Conference (WWW), Chiba, Japan, 2005.

[26] Becchetti, L., Castillo, C., Donato, D., Leonardi, S. and BaezaYates, R., Link- based Characterization and Detection of Web Spam, 2005

[27] I. Drost and T. Scheffer., Thwarting the nigritude ultramarine: Learning to identify link spam. In Proc. ECML,, 2005.

[28] Miller, G., WordNet: a lexical database of English. Communications of the ACM, 1995.

[29] Blum A., Mitchell T., Combining Labeled and Unlabeled Data with Co-Training, ACM, 1998

[30] OpenNLP : http://opennlp.sourceforge.net/