An Approach to Spam Comment Detection through Domain-independent Features

Jong Myoung Kim, Zae Myung Kim, and Kwangjo Kim
School of Computing, Korea Advanced Institute of Science and Technology (KAIST)
Email: {grayapple, zaemyung, kkj}@kaist.ac.kr

***Abstract*—Previous research in spam detection, especially in email spam filtering, mainly focused on learning a set of discriminative features that are often present in the spam contents. Nowadays, these commercially oriented spams are well detected; the real challenge lies in filtering rather vague spams that do not exhibit distinctive spam keywords. We investigate two ways of detecting such spams: 1) By comparing the similarity between the publisher posts and user comments, and 2) by learning a single representative meta-feature such as user name or ID. The first measure relieves us from repetitively learning a set of domain-dependent spam features, and the second measure enables us to detect potential spam users even before the aggressive actions are performed. Prior to the language model comparison in the first method, we supplement the background information, normalize the text, perform co-reference resolution, and conduct word-to-word similarity measure in hope of enriching the language models to improve the classification accuracy. To evaluate the first measure, experiments on detecting blog-spam comments are conducted. As for the second measure, we employ SVM on the ID space of e-mail data collected by "Apache Spam Assassin".**

Keywords—spam filtering; spam user detection; machine learning;

I. INTRODUCTION

Increased interactivity between users has led to an in-creased amount of unwanted messages posted as comments. Such spam contents affect the user experience, reduce the quality of information provided by the publisher and hence indirectly cause financial losses.

In this paper, two approaches to create a domain-independent spam comment filter are proposed, where we aim to detect vague spam comments that do not contain explicit spam keywords.

The first approach compares the semantic similarity between publisher posts and candidate comments. However, since the size of the comments tend to be significantly shorter than the posts, directly comparing the similarity score would result in a high false-positive rate. This problem is solved by enriching the language model of the post and comment; we fetch background information from Wikipedia, normalize the text, apply co-reference resolution, and finally take similar words into account when measuring the overall similarity of the two models.

The second approach classifies spam users using ID space of e-mail addresses. Many probabilistic language models employ words or phrases separated by white-space as a feature to build a classifier. However these features are not suitable for a classifier that only takes a single attribute. We try to treat this problem using n-gram of characters to represent the attribute.

II. RELATED WORK

A lot of research effort has been put into spam detection over the past decades with considerable work done mainly on spam email classification [4]. However, research on spam comments only started in 2005 and has yet to gain much prominence. In 2005, Mishne et al. [11] used smoothed KL-divergence to compute the similarity difference between the original post and comments, as well as the web pages linked by the comments. Our research follows the pattern of this earlier work. Cormack et al. [16] evaluated the performance of methods commonly used in email spam filtering, when applied to spam comment filtering. He concluded that the short text length of comments makes it difficult to support the bag-of-words model commonly used by spam classifiers. Romero et al. [13] performed a comparative study of four classification techniques (naive Bayes, k-nearest neighbors, neural networks and support vector machines) in blog spam comment filtering. Building a spam classifier using the support vector machine resulted in the highest performance of 84.6%. Huang et al. [8] utilized a cosine similarity measure and KL-divergence to conduct content analysis, and built a heuristic decision tree depending on the length of comments. We attempted to detect spam user using those features: lexical analysis of user name and its past labeled user name.

In addition, many authors have considered the problem of spam detection by employing the meta-data of the posted comment. Such additional information includes, e-mail addresses, click stream patterns, social graph properties, etc. These methods work well when the meta-data are abundant. However, it may not always be like this; we may only have the user's names and the contents available to us. Hence, we turn our attention to the work of Freeman and David [7] where they proposed to filter spam users with just the user's name on LinkedIn website. We adopt this method to another in our second approach to spam detection.

III. SPAM COMMENTS ANALYSIS

Before describing the approach and techniques employed in this work, it is helpful to look at the type of spam comments that we will face. As many blog-hosting websites are equipped with filtering engine, most of spam are in the form of Fig. 1.

> I must thank you for the efforts you have put in writing this website. I really hope to check out the same high-grade blog posts from you later on as well. In fact, your creative writing

Fig. 1. Example of vague and irrelevant spam comments

This kind of comment does not contain explicit links, or commercially oriented keywords. This makes them arguably the most difficult kind of spam to detect. In fact, even a state-of-the-art spam comment filter like Askimet [1] cannot detect them effectively. The goal of this research is to detect this type of spam based on vague comments, as well as the traditional forms of spam comments.

IV. The First Approach: Spam Comments Detection through Language Model Comparison

So how can we distinguish these vague and irrelevant spam from the relevant ham? According to Mishne and Glance [2], ham comments represent about 30% of a blog's content. Therefore, we can consider comparing the similarity between the post and comments. However, as suggested by Cormack et al. [17], a direct comparison would result in a very low similarity score as the lengths of comments are significantly shorter than that of posts. To mitigate this issue, we employ four additional techniques.

A. Supplementing background information

When a post talks about "John McCain" and his political campaign, a comment which discusses his recent political movement in the "Republican" party would be considered relevant. However, the post may not explicitly contain the word "Republican" and share no words with the comment. In such case, a similarity measure will give a lower score between the post and the candidate comment than it actually deserves.

To alleviate this, we supplement some Wikipedia context of named entities appearing in the post and comments. The Wikipedia article of "John McCain" indeed contains information about his role in the "Republican" party. We add the first paragraph of the wiki entry, which often is a summarization of the entire article.

B. Text normalization

Lexical normalization and lemmatization is used to account for slangs and misspelled words.

C. Co-reference resolution

Co-reference-resolution maps pronouns into their representative entities, making them more frequently appear in the document.

D. Considering word similarity

When computing the similarity measure between the two-term vector models, we extend them to include similar words in the models by calculating their semantic distance using WordNet taxonomy. If their semantic score is above a certain level, we treat the two words the same, and reflect them in our similarity measure. We employ similarity measures by Leacock and Chodorow [9].

E. Similarity measures

After applying the above techniques, we proceed to measure the similarity between the post and comment. Two similarity measures, were employed with some slight modifications in the equations to 1) reflect the differences in size of the vector models of each post and comment, and 2) to incorporate words with high similarity. Loosely speaking, the cosine similarity determines how similar the two-term vector models are, whereas the latter measures their dissimilarity. So in a sense, they are complementary to each other, capturing both sides of the coin.

Skew divergence is a weighted version of the Kullback-Leibler divergence. The skew divergence Sα between two language models l1 and l2, is given by:

$$S_\alpha(l_1||l_2) = KL(l_2||\alpha l_1 + (1-\alpha)l_2) \quad (2)$$

$$KL(l_1||l_2) = \sum_y l_1(y)(log l_1(y) - log l_2(y)) \quad (3)$$

where KL(l1 ‖ l2) is the KL-divergence of language models l1 and l2, y represents each word in l1 and α is the skew divergence constant. We empirically set α to be 0.99. Since skew divergence is asymmetric, we need to calculate both S(lpost ‖ lcomment) and S(lcomment ‖ lpost) and find their mean S as follows:

$$\overline{S} = \frac{S_{0.99}(l_{post}||l_{com}) + S_{0.99}(l_{com}||l_{post})}{2} \quad (4)$$

V. The Second Approach: Spam Users Detection Using Single Representative Attribute

Spam detection methods that use contents as features have a common point that they can only filter the aggressive activities once such activities are performed. To address this limitation, Freeman and David [7] tried to classify spam user with just a single attribute, the name of a user. Most online services require their users' online identity to reflect his or her genuine identity in real life. So they demand genuine user information at registration time. But some users with bad motives intentionally create false identity to engage in abusive activities. For the case that their malicious activities are detected, such users employ programs to create accounts, in which IDs or assumed names are generated from a random string or a dictionary. The approach in represents the users' name by a probabilistic model, and filter spam users with those features. They use the probability that a subsequence of a user name belonging to a spam user.

We apply this method to e-mail account of comment writer. After security of personal information stand out, most of online services demand the least data for registration. In that situation, e-mail is the most representative and important attribute of user.

A. N-gram for feature representation

Most spam classifications use words or phrases as a feature. But in this approach, these features are considered inappropriate, because the ID we will use as a feature in this study has only one attributes. Consequently, we used an n-gram [20] of characters to represent the feature. An n-gram represents a feature by the contiguous sequence of n items from a given sequence of text.

B. Replacement of missing values

We used the ID space of e-mail as a feature of classification. And we set the number of attributes to eight, the average size of the ID. Because every ID has a different length, necessarily missing values occurred. To use SVMs, we had to treat the missing values. We replace them with n-1 gram.

Because an n-gram is a model representing the probability of a gram's occurrence, the probability of an ungrammatical expression and symbol is very low in ham (ham means 'not spam'), and the probability is relatively high in spam. So ungrammatical expressions or symbols can be evidence of spam.

But when we analyzed the data, we found some features. People sometimes use ungrammatical expressions or symbols at the start or end of an ID, such as "lok.tarrrr" or "grayapple★". Words which have "rrr" or "le★" as a substring do not exist. So the occurrence frequency of these words are very low, and the filter regards these as spam. Instead, we wanted to handle these cases as special cases. So we marked the start and end of an ID with the special symbol '/'.

VI. Experiments for the First Approach

The previous research [3] on spam comments filtering used Mishne's spam corpus [11] as a benchmark to illustrate their performance. However, after contacting the author, it was found that the corpus was lost and hence not available to be used in this work. Instead, a collection of spam comments was gathered manually.

Our spam collection comprises 103 individual spam comments, where each comment only contains the content body and no other information (e.g., name of the spammer, the date of its creation). It is also interesting to note that because these spams got past a very advanced spam filter, almost all of them take the form of the kind of vague spam comments illustrated in Fig. 1 Many of them are either extremely complimentary or utter gibberish, often with bad grammar. Only two comments contained a direct hyperlink in the content.

The experiment was performed by comparing model similarity of a) an article and its set of ham comments, against b) the spam comments. Table 1 illustrates an example.

TABLE I. Comparison of Ham and Spam Comments

Article title	**"McCain tells White House he wants Lieberman for defense secretary [4, 5]"**	
	Ham comments	**Spam comments**
# of comments	9	10
Avg. # of words	71.89	38.70
Avg. cosine similarity	3.08	2.27
Avg. skew divergence	8.25	9.96

The result shows that the average cosine similarity of ham comments to the article is higher than to the spam comments. This means that relevant comments share more similar context with the article. We can also confirm that the skew divergence of the spam comments is larger than the value of the ham comments. This is because the word probability distribution of spam comments differs more significantly from the article's probability distribution.

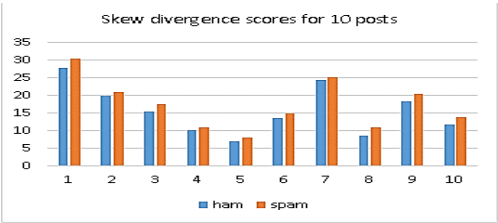

Fig. 2. Cosine similarity scores for the 10 posts

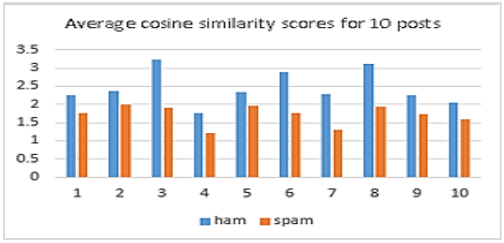

Fig. 3. Skew divergence scores for the 10 posts

The same experiment was conducted using a different set of articles and ham comments gathered from the Washington Post. The top ten most-read articles as of 15 January 2015 were chosen. The results are shown in Fig. 2 and 3.

As depicted in the graphs, further experiments conform to our expectations in the sense that ham comments achieve higher similarity scores and lower divergence values than the spams. However, in the case of the skew divergence, the difference between the two scores is not always significant (e.g., the 7th post shows only 3% difference in the score). We believe such case arises when an original post largely contains simple words that are also abundantly present in the spam comments, reducing the divergence score of spams. Recall that spam comments consist of many general and simple words which can give high word similarity score when compared with another general words in the post. As future work, we will consider employing a scheme to pick out only the "important" words for comparison.

VII. Experiments for the Second Approach

Since a dataset that includes spam comments and corresponding email addresses is hard to achieve due to its scarcity, we instead evaluate the performance of our method on spam e-mail datasets. We used SVM as a classifier and secured 8,846 data which consisted of 6,452 ham mails and 2,394 spam mails to evaluate our approach. We designed the experiment with 10-fold cross validation. We measured the accuracy, precision, recall, F-measure, AUC, and Kappa score.

A. *Data set*

As noted, the data set we used in this experiment consisted of 6,452 ham mails and 2,394 spam mails which were collected between 2002 and 2004. These mails obey the xml form. We extracted the ID space from these mails and used them as a classifier feature. We obtained the dataset from Apache Spam Assassin which is a famous spam filter open source company. They provided the source code of a spam filter and dataset to test.

B. *Performance of rival method*

We set a content based spam filter as a rival method to be compared with our approach. This classifier used the existence of 1,900 words which frequently appeared in pre-investigation as a feature of the classification algorithm, SVMs. The existence of a word is represented by 0 and 1, and the SVM model is a representation of the data as points in hyperspace which has 1,900 axis. This method shows awesome performance. It is presented in Table 2 as follows.

TABLE II. Performance of Basic and Rival Metthods

	Basic	**Rival**
Accuracy	0.729	0.983
Precision	0.0	0.972
Recall	0.0	0.976
F-measure	0.0	0.974
AUC	0.5	0.998

C. Performance of initial version

At first, we evaluated our initial version using only n-gram and SVMs. The result is shown in Table 2. The basic classifier always says "This mail is not spam mail." To improve the performance, we introduced the method of replacing missing values and marking the start and end with the special symbol.

D. Performance of improved version

After the introduction of replacing and marking, there was a tremendous improvement. The proposed method could not overtake the rival method. But it showed great performance, not only in accuracy, but also in precision, recall, F-measure, AUC, and Kappa scores. The detailed results are presented in Tables 3 and 4. RMV means replacement for missing value and MSE means marking start and end with special symbol. The result of the experiment proved this classification algorithm is totally trustworthy. The model which applied 3-gram, RMV and MSE shows the best performance.

TABLE IV. PERFORMANCE AFTER REPLACEMENT FOR MISSING VALUES

RMV	2-gram	3-gram	4-gram	5-gram	Rival
Accuracy	**0.966**	0.962	0.966	0.904	**0.983**
Precision	**0.969**	0.969	0.969	0.933	**0.972**
Recall	**0.905**	0.889	0.905	0.697	**0.976**
F-measure	**0.936**	0.927	0.936	0.789	**0.974**
AUC	**0.955**	0.943	0.954	0.85	**0.998**
Kappa	**0.913**	0.901	0.913	0.737	**0.961**

TABLE III. PERFORMANCE AFTER REPLACEMENT OF MISSING VALUES AND MARKING START AND END WITH SPECIAL SYMBOL

RMV+MSE	2-gram	3-gram	4-gram	5-gram	Rival
Accuracy	0.967	**0.967**	0.966	0.962	**0.983**
Precision	0.969	**0.969**	0.969	0.969	**0.972**
Recall	0.905	**0.905**	0.905	0.889	**0.976**
F-measure	0.936	**0.936**	0.936	0.927	**0.974**
AUC	0.953	**0.957**	0.954	0.933	**0.998**
Kappa	0.913	**0.913**	0.913	0.901	**0.961**

E. Limitation

Using only a single-attribute and representing attribute with an n-gram is a point of this section. But the method has some limitations. The biggest weakness is unavoidable false positives. A single attribute method cannot handle words from a non-English keyboard or readable symbol expressions. The former issue surface from a user who lived in a non-English country. In a nation which has its own language, I/O devices, such as a keyboard, have to treat both English and the national language. So the keys of a keyboard are used for several letters, and the keyboard has a translation key to change language mode. The users from these countries sometimes type words using their country's language in English mode. Readable symbol expressions are caused by creative users, when letters in words are substituted by symbols, numbers or other letters. For example, most people can read "sk8ter boi", the famous song of Avril Lavign, as "skater boy". But the computer does not have the flexibility to treat that word in that way.

VIII. CONCLUSION

In this paper, we propose an approach to a domain-independent spam filter that detects vague and irrelevant comments by measuring the similarity between the post and comment. In addition, we filter spam users by employing a single representative attribute. To increase the precision of content analysis, techniques such as text normalization, co-reference resolution, word similarity measure and Wikipedia extraction are utilized. Preliminary experiments show promising results.

For further research, we plan to investigate ways to distinguish "important" words from the common words and utilize them in the word-to-word similarity measure.

ACKNOWLEDGMENT

This work was supported by ICT R&D program of MSIP/IITP. [R0101-15-0062, Development of Knowledge Evolutionary WiseQA Platform Technology for Human Knowledge Augmented Services]

REFERENCES

[1] Askimet, "Akismet: Comment spam prevention for your blog," <http://akismet.com/>.

[2] Mishne, Gilad, and Natalie Glance. "Leave a reply: An analysis of weblog comments." Third annual workshop on the Weblogging ecosystem. 2006.

[3] Sculley, D. Advances in online learning-based spam filtering. ProQuest, 2008.

[4] Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., and Spyropoulos, C. D. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages. In Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval (2000), ACM, pp. 160–167.

[5] Cormack, G. V., G´Omez Hidalgo, J. M., and S´Anz, E. P. Spam filtering for short messages. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management (2007), ACM, pp. 313–320.

[6] Cormack, G. V., and Lynam, T. R. Online supervised spam filter evaluation. ACM Transactions on Information Systems (TOIS) 25, 3 (2007), 11.

[7] Freeman, D. M. Using naive bayes to detect spammy names in social networks. In Proceedings of the 2013 ACM workshop on Artificial intelligence and security (2013), ACM, pp. 3–12.

[8] Huang, C., Jiang, Q., and Zhang, Y. Detecting comment spam through content analysis. In Web-Age Information Management. Springer, 2010, pp. 222–233.

[9] Joachims, T. Text categorization with support vector machines: Learning with many relevant features. Springer, 1998.

[10] Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., and Mcclosky, D. The Stanford corenlp natural language processing toolkit. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations (2014), pp. 55–60.

[11] Mishne, G., and Glance, N. Leave a reply: An analysis of weblog comments. In Third annual workshop on the Weblogging ecosystem (2006), Edinburgh, Scotland.

[12] Romero, C., Valdez, M. G., and Alanis, A. A comparative study of machine learning techniques in blog comments spam filtering. In Neural Networks (IJCNN), The 2010 International Joint Conference on (2010), IEEE, pp. 1–7.

.