IMPROVING THE PERFORMANCE OF ANTI-SPAM FILTERS USING OUT-OF-VOCABULARY STATISTICS

MEJORA DEL DESEMPEÑO DE FILTROS ANTI-SPAM USANDO ESTADÍSTICAS DE PALABRAS FUERA DE VOCABULARIO

Pablo Daniel Agüero[1] Jorge Castiñeira Moreira[1]
Monica Liberatori[1] Juan Carlos Bonadero[1] Juan Carlos Tulli[1]

Recibido 3 de marzo de 2009, aceptado 6 de agosto de 2009
Received: March 3, 2009 Accepted: August 6, 2009

RESUMEN

Este artículo presenta una característica basada en estadísticas de palabras desconocidas (fuera del vocabulario) que complementa las fuentes de información usadas en la decisión por parte de los filtros anti-spam actuales. Los experimentos incluyeron filtros anti-spam disponibles libremente como referencia: SpamAssassin, Bogofilter, SpamBayes y SpamProbe, así como también un clasificador puramente bayesiano. Los resultados muestran que la decisión basada en la característica propuesta mejora el rendimiento de todos los filtros anti-spam sujetos a estudio.

Palabras clave: Spam, filtrado, palabras desconocidas.

ABSTRACT

This paper presents a feature based on out-of-vocabulary word statistics that complements the information sources used in the decision by state-of-the-art spam filters. The experiments included freely available spam filters as reference, SpamAssassin, Bogofilter, SpamBayes and SpamProbe, as well as a Naive Bayes classifier. The results show that the decision based on the proposed feature improves the performance of all spam filters under study.

Keywords: Spam, filtering, out-of-vocabulary.

INTRODUCTION

Unsolicited commercial email, commonly known as spam, is still an important problem for Internet users. The main annoying effects of spam are decreasing productivity of employees, wasting valuable storage on mail-servers, harming Internet traffic and increasing possible information loss depending on filtering policies. For example, the authors of this paper almost lost a paper-acceptance e-mail due to spam filters.

According to several information sources, such as Softscan [1] or Marshal [2], more than 90% of e-mail traffic is spam, with an increasing volume index during the last years. The highest percentages come from Europe and Asia (65% according to Marshal's statistics) followed by North America (18%).

The spammers use the strategy of sending huge amount of e-mails at virtually no cost, with a high probable profit. If a spammer gets as few as 100 responses for every 1,000,000 e-mail messages, he can make an attractive profit. For further details you may read "Inside the SPAM Cartel: Trade Secrets From the Dark Side" [3].

In the literature we find two main trends on spam filtering techniques: rule-based and learning approaches [4-10].

Rule-based spam filters analyze both header and body of e-mail messages looking for suspicious content. Each activated rule adds up a score to the e-mail. If the score reaches a threshold value, the e-mail is classified as spam.

The rules cover different threats, such as suspicious format (HTML and text parts are different or message body has

[1] Laboratorio de Comunicaciones. Facultad de Ingeniería. Universidad Nacional de Mar del Plata. Juan B. Justo 4302. CP 7600. Mar del Plata, Argentina. E-mail: pdaguero@fi.mdp.edu.ar; casti@fi.mdp.edu.ar; mlibera@fi.mdp.edu.ar; jbona@fi.mdp.edu.ar; jctulli@fi.mdp.edu.ar

80-90% blank lines), weak origin prone to sending spam (sender is confirmed open relay or sender is an open proxy), common words in spam (contains a masked version of *cialis* or *viagra*), illegal encodings or characters (too many raw illegal characters in subject), etc [11].

Another commonly used approach, the learning filters, has become a widely accepted technique for classifying email messages into ham (legitimate e-mails messages) or spam. In some cases, a third category is also included: unsure. The research is focused in several aspects of this approach, such as reliable statistics, techniques to combine different information sources, adaptive algorithms and threshold techniques. The last one is based on maximizing true positives (spam detection) and minimizing false positives (ham filter protection against wrong classification).

In this paper we show a basic learning filter based on Naive Bayes decision which proves to be competitive in spite of its simplicity. In order to study the proposed filter performance, we compare the results with other state-of-the art spam filters freely available on Internet as open-source software: Spamassassin [12], Bogofilter [13], SpamProbe [14] and SpamBayes [15].

In order to improve the performance of reference systems, we propose the use of out-of-vocabulary statistics. Such feature provides a complementary information source for the decision of the classifier, and was already used by other researchers for text classification algorithms. Bennett, Dumais and Horvitz [16] used out-of-vocabulary words in the calculation of reliability indicator variables, such as *EffectiveUniqueWords* (number of unique words minus the number of unique out-of-vocabulary words) or *PercentOOV* (percentage of the words in a document which weren't seen in the training set). The later is a strong indicator about how likely a classifier is going to incorrectly classify the document into a class due to the lack of information.

The paper is organized as follows. In the next section we briefly describe the different approaches used by the reference filters. After that, we show a simple proposed filter based in Naive Bayes classification and a study of the proposed feature based on out-of-vocabulary statistics. Such feature improves the performance of reference systems, as shown in the experimental results. Finally, the conclusions are stated and future research lines are proposed.

REFERENCE APPROACHES

Bogofilter [13] treats email messages as a bag of tokens. Each token is checked against a word list which is dynamically updated. In this way it maintains counts of the number of times the word has occurred in non-spam and spam mails. These numbers are used to compute an estimate of the probability that a message in which the token occurs is spam. The combination of those probabilities calculated over the words inside a message indicates whether this message is spam or ham. While this method seems quite basic compared to the more usual pattern-matching approaches, it turns out to be extremely effective [18].

Bogofilter substantially improves Graham's proposal by doing smarter lexical analysis performing proper MIME decoding and reasonable HTML parsing. Special kinds of tokens like hostnames and IP addresses are retained as recognition features rather than broken up into sub-domains or numbers, as proposed by Graham in [18]. Various kinds of Message Transfer Agent (MTA) information, such as dates and message-IDs, are ignored to preserve a manageable word list.

Another improvement is the use of Gary Robinson's suggested modifications to the calculations [19]. Robinson et al. have realized that the calculation can be further optimized using Fisher's method. Another advance compensates for token redundancy by applying separate Effective Size Factors (ESF) to spam and non-spam probability calculations [20].

The estimates for the spam probabilities of the individual tokens are combined using the Inverse Chi-Square function. Its value indicates how badly fails the null hypothesis that the message is just a random collection of independent words with probabilities given by our previous estimates. This function is very sensitive to small probabilities (hammish words), but not to high probabilities (spammish words). As a result, the value only indicates strong hammish signs in a message. Then, using inverse probabilities for the tokens, the same computation is done again, giving an indicator that a message looks strongly spammish. Finally, those two indicators are subtracted and scaled into a 0-1 interval. This combined indicator, known as bogosity, is close to 0 if the signs for a hammish message are stronger than for a spammish message and close to 1 if the situation is the other way around. If signs for both are equally strong, the value will be near 0,5. Since those messages don't give a clear indication, there is a tri-state mode in Bogofilter to mark those messages as unsure. The clear messages are marked as spam or ham, respectively. In two-state mode, every message is marked as either spam or ham.

SpamProbe [21], the second reference software, is based on the basic operating principles of Graham [18]. However, it is not a direct implementation of Graham's

algorithm. There are some changes in an effort to improve the algorithm's effectiveness:

- **Use of different tokenizer rules**. The SpamProbe tokenizer attempts to extract terms from emails in an internet savvy way. The tokenizer understands HTML entity references as well as URL encoding.

- **Use of word pairs** as well as single words as terms. In his paper Graham mentioned that accuracy might be improved by scoring word phrases in addition to single words. SpamProbe normally uses word pairs (two word phrases) as terms.

- **Use of within document frequency** to allow frequently appearing words to have greater weight. Within Document Frequency (WDF) means the number of times a term appears in a document. Therefore, emails with higher occurrence of a given term would have a higher score.

- **Removal of HTML tags** to avoid false positives.

- **Folding of 8 bit characters** to avoid foreign character sets, such as Korean, Chinese, Japanese, etc.

- Optional use of a **variable sized terms array** to allow all significant terms to appear in the score.

- **Use of stubborn classification.** SpamProbe attempts to improve recall by classifying each email in a loop. The loop repeatedly adjusts the database counts for terms in the email and then scores the email. If the score shows that the email may not be classified, the process is repeated until ten times.

- **Image Analysis.** Whenever SpamProbe sees an image attachment it processes the image to extract a set of tokens from properties of the image itself as well as any meta data contained in the image file.

The third reference, **SpamBayes** [22-23], uses Robinson's chi-squared probabilities combining scheme. Messages tend to score at the extremes of the range, although difficult to classify messages may fall near the middle.

Tokenizing (i.e. converting the message string to a feature vector) the message has a profound influence on the overall results of the classification engine. During the development of SpamBayes, many ideas for message tokenization have been suggested and tried. However, most schemes have provided no statistically significant results over diverse corpora. SpamBayes allows these ideas to be included as experimental features so that developers can test their effectiveness on their individual corpora. It is important to note that beneficial tokenizing schemes change over time as the nature of spam changes. They must be periodically validated.

Early testing showed that using either character or word n-grams was less effective than simple unigrams. However, in late 2002, Robinson and Robert Woodhead independently came up with an idea for using both unigrams and bigrams with a twist to avoid generating highly correlated clues. This idea was cleaned up and implemented by Peters [24], although it was only added to the SpamBayes code in late 2003. This code is currently in daily use by a number of the SpamBayes group, and is available to users as an experimental option.

The technique mixes single tokens (unigrams) with pairs of adjacent tokens (bigrams). The token stream is tiled into non-overlapping unigrams and bigrams using the strongest tokens. Avoiding overlap is important to prevent a single token from contribution to more than one returned token probability (systematic correlation). Each triplet of tokens in the message generates three unigrams and two bigrams. These are ranked in order of the distance of their score from the neutral 0.5. The strongest is added to the list of tokens used to classify the message, and the remaining tokens that overlap that one are removed. The process is then repeated with the next triplet of tokens.

Finally, the fourth reference is **SpamAssassin** [12]. It is an intelligent e-mail filter which uses a diverse range of tests to identify spam. These tests are applied to email headers and contents to classify e-mail using advanced statistical methods and hand-written rules. In addition, SpamAssassin has a modular architecture that allows other technologies to be quickly wielded against spam and is designed for easy integration into virtually any email system.

PROPOSED APPROACH

In this paper we propose a combination of state-of-the-art spam filters with a decision based on out-of-vocabulary statistics. The employment of out-of-vocabulary (OOV) words complements missing information in certain classifiers on early stages of learning techniques.

Bayes decision

Spam filtering using Bayes's decision rule analyzes the probability of a given e-mail with words w to be classified as spam:

$$P(C = spam / w) = \frac{P(C = spam)P(w / C = spam)}{\sum_k P(C_k)P(w / C_k)} \quad (1)$$

However, as pointed out by several authors, the probability vector *P(w/spam)* is almost impossible to calculate due to sparseness. Instead, the Naive Bayesian classifier makes the assumption that each word is conditionally independent of the other words. As a consequence, the previous equation can be rewritten as:

$$P(C = spam / w) = \frac{P(C = spam)\prod_i P(w_i / C = spam)}{\sum_k P(C_k)\prod_i P(w_i / C_k)} \quad (2)$$

The word statistics are calculated using counts in order to obtain a probability that a given word belongs to ham or spam category (Equations 3 and 4).

$$P(w / C = ham) = \frac{\#_{in\_ham\_emails}}{\#_{in\_ham\_emails} + \#_{in\_spam\_emails}} \quad (3)$$

$$P(w / C = spam) = \frac{\#_{in\_spam\_emails}}{\#_{in\_ham\_emails} + \#_{in\_spam\_emails}} \quad (4)$$

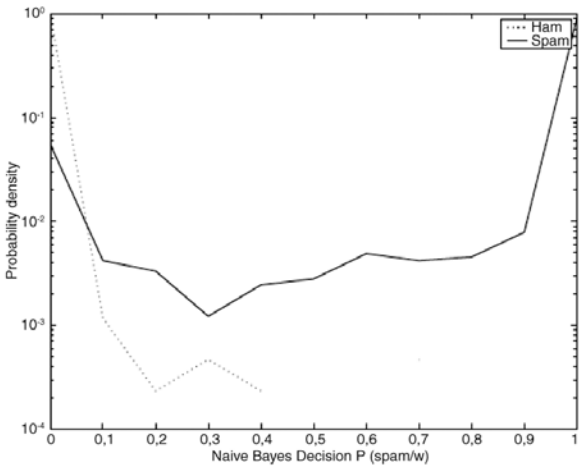

Figure 1. Probability density function of ham and spam scores for training data.

The combination of word probabilities of Eq. 2 is performed using logarithms to avoid an underflow error in the product of probabilities.

Although the simplicity of the score formulation, it is possible to successfully distinguish between ham and spam messages using such score. As shown in Figure 1 a threshold can be selected to minimize the probability of a wrong classification of ham messages, which is a more severe error than spam misclassification.

Decision based on out-of-vocabulary words

Further studies on training data also show that out-of-vocabulary words are an important indicator for e-mail filtering. OOV's are found in spam due to foreign language content, masked version of words (such as "V-I A GRA"), tricky writings (replacing upper-case letter "O" by zero), non-spaced sentences ("itisaofferofVIAGRA"), etc.

An experiment about the percentage of OOV's in a set of 10.000 e-mail messages (57% spam) produces the results in Figure 2. We observe that 35% of spam messages have a 25% of OOV words, while a threshold around 65% produces a negligible classification of a ham e-mail as spam. In this work, we first classify each e-mail using word statistics using the proposed Naive Bayes classifier or the described reference classifiers. Then, e-mails classified as ham are reclassified as spam using OOV statistics when the percentage of OOV words is higher than a predefined threshold. This threshold is set to a high value to minimize false positives.

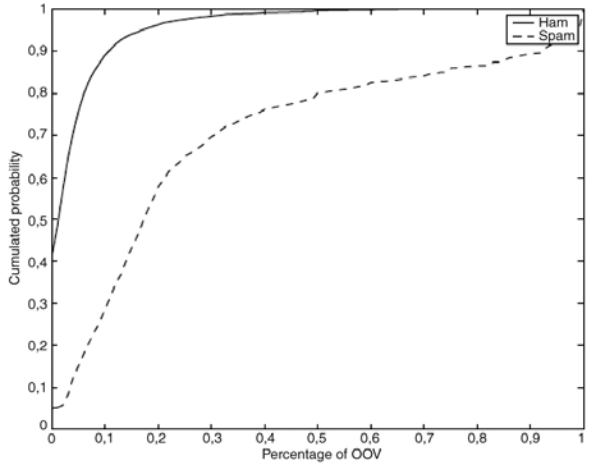

Figure 2. Cumulative probability function of OOV's for ham and spam e-mail.

In the comparative experiments of next section we show the experimental results of the inclusion of OOV statistics in the decision.

EXPERIMENTAL RESULTS

In this paper we use the 2005 TREC Public Spam Corpus (trec05p-1). It contains 92.189 e-mail messages, with a chronological index labelling each as spam or ham: 52.790 messages are labelled spam while 39.399 are labelled ham.

The experiments were performed using a subset of this corpus. We have selected 10.000 e-mail messages (4.279 ham and 5.721 spam) with text and text/html content. This amount of e-mails allows to measure the asymptotic behaviour of all systems under study.

In order to accurately evaluate spam filter's performance, we need to emphasize that labelling a legitimate message as spam (a false positive) is more harm for a user than to classify a spam message as a legitimate one (a false negative).

It is possible to take into account this fact using a performance metric called Weighted Accuracy (w_{acc}) [9]. w_{acc} assigns a weight λ to legitimate messages, treating each one of them as if it were λ messages when counting. The equation for w_{acc} is the following:

$$W_{acc} = \frac{\lambda t_n + t_p}{\lambda(t_n + f_p) + (t_p + f_n)} \quad (5)$$

In the above equation, t_n is the number of correctly classified legitimate messages (true negatives), t_p is the number of correctly classified spam messages (true positives), f_p is the number of incorrectly classified legitimate messages (false positives) and f_n is the number of incorrectly classified spam messages (false negatives). In this paper, we have selected $\lambda = 9$, because it proved to be a good value for this parameter in other works, such as Webb, Chitti and Pu [10].

In Figure 3 we show the experimental results using the corpus of 10,000 messages of TREC 2005. Each filter classifies a message and then updates its parameters. Therefore, the performance of the systems tends to improve after each classification.

The lowest performance corresponds to Bogofilter and SpamBayes. This result is due to the high number of false negatives, 1016 and 750, respectively. However, it is also singular the small number of false positives, two and one, respectively. Therefore, although both systems minimize the number of false positives, it is done at the expense of false negatives.

The Naive Bayes classifier has a similar performance to SpamAssassin and SpamProbe, with a low number of false negatives (385) and false positives (5). Figure 4 shows the relative improvement in the weighted accuracy (I) (compared to the highest achievable performance) due to the inclusion of the decision based on out-of-vocabulary word statistics, as shown in the following equation:

$$I = 100 \frac{W_{acc}^{withOOV} - W_{acc}^{withoutOOV}}{1 - W_{acc}^{withoutOOV}} \quad (6)$$

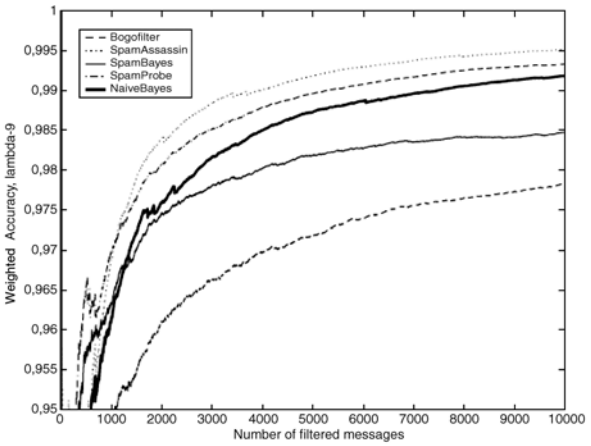

Figure 3. Experimental results using the corpus of 10,000 messages of TREC 2005.

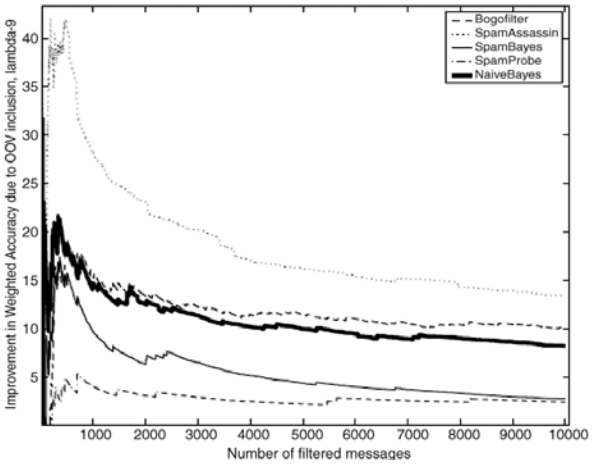

Figure 4. Improvement in Weighted Accuracy due to OOV inclusion.

All systems improve their weighted accuracy due to the use of out-of-vocabulary word statistics in the decision of messages classified as ham by the classifiers under study. This encourages the use of such information source in the decision of spam filters.

The lower improvement corresponds to SpamBayes and SpamProbe (3%), and the highest to SpamAssassin (14%).

CONCLUSIONS

In this paper we present a feature that complements the information used in the decision by state-of-the-art spam filters: out-of-vocabulary word statistics.

OOV statistics compensate the lack of information about new vocabulary in e-mail messages which harms the performance of spam filters based only on word statistics. This second information source is introduced due to the observation of a high number of OOV words in spam messages compared to legitimate e-mails.

This additional feature does not introduce any substantial storage or processing load. The higher load corresponds to word statistics.

Experimental results show that the additional decision based on OOV statistics improves the performance of all state-of-the-art spam filters.

In the future, further work will be dedicated to incorporate more information sources and combining them.

REFERENCES

[1] "Virus and spam stats". 2009. Date of visit: August 3, 2009. URL: http://www.softscan.co.uk/content/us/support/virus_and_spam_stats

[2] "Spam statistics". 2009. Date of visit: August 3, 2009. URL: http://www.marshal8e6.com/TRACE/spam_statistics.asp

[3] "Spammer-X: Inside the spam cartel: trade secrets from the dark". Editor: Syngress Publishing, Inc. 2004.

[4] P. Pantel and D. Lin. "Spamcop: A spam classification & organization program". Proceedings of AAAI Workshop on Learning for Text Categorization. 1998.

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

[6] H. Drucker, D. Wu and V. Vapnik. "Support vector machines for spam categorization". IEEE Transactions on Neural Networks. Vol. 10 Nº 5. 1999.

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

[8] I. Androutsopoulos. "An evaluation of naive bayesian antispam filtering". Proceedings of the Workshop on Machine Learning in the New Information Age, 11th European Conf. on Machine Learning. 2000.

[9] I. Androutsopoulos, G. Paliouras and E. Michelakis. "Learning to filter unsolicited commercial e-mail". National Center for Scientific Research Demokritos, Tech. Rep. 2004/2. 2004.

[10] S. Webb, S. Chitti and C. Pu. "An experimental evaluation of spam filter performance and robustness against attack". International Conference on Collaborative Computing: Networking, Applications and Worksharing. 2005.

[11] SpamAssassin. "The Apache SpamAssassin Project: Tests Performed". Date of visit: August 3, 2009. URL: http://spamassassin.apache.org/tests.html

[12] SpamAssassin. "The Apache SpamAssassin Project". Date of visit: August 3, 2009. URL: http://spamassassin.apache.org/

[13] Bogofilter. "Bogofilter Project". Date of visit: August 3, 2009. URL: http://bogofilter.sourceforge.net/

[14] SpamProbe. "SpamProbe Project". Date of visit: August 3, 2009. URL: http://spamprobe.sourceforge.net/

[15] SpamBayes. "SpamBayes Project". Date of visit: August 3, 2009. URL: http://spambayes.sourceforge.net/

[16] P. Bennett, S. Dumais and E. Horvitz. "The Combination of Text Classifiers using Reliability Indicators". Information Retrieval. Vol. 8 Nº 1. 2005.

[17] P. Graham. “A plan for spam”. 2002. Date of visit: August 3, 2009. URL: http://www.paulgraham.com/spam.html

[18] P. Graham. “Better bayesian filtering”. 2003. Date of visit: August 3, 2009. URL: http://www.paulgraham.com/better.html

[19] G. Robinson. “A statistical approach to the spam problem”. Linux Journal. Issue 107. 2003.

[20] G. Robinson. “Handling redundancy in email token probabilities”. 2004. Date of visit: August 3, 2009. URL: http://www.garyrobinson.net/2004/04/improved_chi.html

[21] B. Burton. “SpamProbe - Bayesian Spam Filtering Tweaks”. 2002. Date of visit: August 3, 2009. URL: http://spamprobe.sourceforge.net/paper.html

[22] T.A. Meyer and B. Whateley. “SpamBayes: effective opensource, bayesian based, email classification system”. First Conference on Email and Anti-Spam. 2004.

[23] T.A. Meyer. “SpamBayes: TREC 2005 Spam Track Notebook”. Text Retrieval Conference. 2005.

[24] T. Peters. “The first trustworthy wink GBayes results”. 2002. Accessed: August 3, 2009. URL: http://mail.python.org/pipermail/python-dev/2002-September/028513.html