Spam Detection Using Web Page Content: a New Battleground

Marco Túlio Ribeiro, Pedro H. Calais Guerra, Leonardo Vilela,
Adriano Veloso, Dorgival Guedes*, Wagner Meira Jr.
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, Brazil
{marcotcr,pcalais,vilela,adrianov,dorgival,meira}@dcc.ufmg.br

Marcelo H.P.C Chaves, Klaus Steding-Jessen, Cristine Hoepers
Brazilian Network Information Center (NIC.br)
Sao Paulo, Brazil
{mhp,jessen,cristine}@cert.br

ABSTRACT

Traditional content-based e-mail spam filtering takes into account content of e-mail messages and apply machine learning techniques to infer patterns that discriminate spams from hams. In particular, the use of content-based spam filtering unleashed an unending arms race between spammers and filter developers, given the spammers' ability to continuously change spam message content in ways that might circumvent the current filters. In this paper, we propose to expand the horizons of content-based filters by taking into consideration the content of the Web pages linked by e-mail messages.

We describe a methodology for extracting pages linked by URLs in spam messages and we characterize the relationship between those pages and the messages. We then use a machine learning technique (a lazy associative classifier) to extract classification rules from the web pages that are relevant to spam detection. We demonstrate that the use of information from linked pages can nicely complement current spam classification techniques, as portrayed by SpamAssassin. Our study shows that the pages linked by spams are a very promising battleground.

1. INTRODUCTION

Spam fighting is an "arms race" characterized by an increase in the sophistication adopted by both spam filters and spammers [12, 14]. The co-evolution of spammers and anti-spammers is a remarkable aspect of the anti-spam battle and has motivated a variety of works that devise adversarial strategies to treat spam as a moving target [6, 4].

On the spammers' side, the standard counter-attack strategy to face content-based filters is to obfuscate message content in order to deceive filters. In the early years of the spam arms race, obfuscation techniques were as simple as misspelling Viagra as V1agra, but have evolved to complex HTML-based obfuscations and the use of images to prevent action of text-based filters. However, spammers face a trade-off: their final goal is to motivate a recipient to click on their links; too much obfuscation can lead to lower click rates and reduce spammers' gains [3]. No obfuscation at all, on the other hand, will cause the spam to be easily blocked and few mailboxes will be reached. Therefore, in addition to keeping spam detection rates high, content-based filters caused the positive effect (for the anti-spam side) of making each spam message less attractive and less monetizable – even though spammers have tackled that problem by sending larger volumes of spam.

In this paper, we argue that a fundamental component of spam content has been neglected by content-based spam filters: the content of *web pages* linked by spam messages. We believe web pages can be an useful component added to current spam filtering frameworks for the following reasons:

1. Web page content is an almost unexplored battleground on the spam arms race, in part for the belief that processing those pages would be too expensive in terms of computing resources. Therefore, current spammers may not have major concerns regarding web pages getting identified as spam and so do not implement mechanisms to obfuscate web pages, what would represent an extra cost and might cause their pages to become harder to read. Increasing the cost of the spam activity is one efficient strategy to discourage spammers [16]. In addition to that, although spammers send billion of messages daily, the range of products advertised in web sites are not very diverse; a recent report concluded that 70% of spam advertises pharmaceutical products [9]. A recent work has shown that a few banks are responsible for processing transactions of spam product purchases [18], what is an additional motivation for seeking evidences for spam detection with are closer to the spammer's business and cannot be changed easily.

2. Recently, Thomas et al. have presented Monarch [27], a real-time system to detect spam content in web pages

*Dorgival Guedes is also with the International Computer Science Institute (ICSI), Berkeley:
dorgival@ICSI.Berkeley.EDU

CEAS 2011 - Eighth annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference September 1-2, 2011, Perth, Western Australia, Australia

published in social network sites and in e-mail messages. Their results show that with the current technology it is feasible to collect and process pages as they show up in web sites posts and e-mail messages.

3. In the underground spam market, there is evidence that the spam sender is not always the same agent as the web page owner, making the adaptation of web page content a separate task, harder to coordinate with the spam messages [1]. This is the case when spammers work as business brokers that connect sellers (the web page owner) and buyers (e-mail users targeted with spams) [16].

4. It has been reported that 92% of spams in 2010 contained one or more URLs [17]; previous work reported the presence of URLs in up to 95% of spam campaigns examined [15, 24]. Those numbers indicate that using web page content to improve spam detection is an applicable strategy, as spammers need to use URLs to earn money through advertised web sites.

In this work, we show that web pages can provide precious information about nature of the e-mail message that link to them. Using information from classical spam/ham repositories, we built a new dataset that provides also the information about the web pages mentioned in messages of the set. Using that dataset, we show that by analyzing the web pages pointed by e-mail messages we can complement Spam Assassin regular expressions and blacklist tests providing an improvement of up to 10% in spam classification, without increasing the false positive rate. Our contributions, in summary, are:

- we make available this new dataset, which associates web page content and e-mail message content;
- we propose and evaluate a methodology for e-mail spam detection that considers the content of web pages pointed by messages;
- we show the effectiveness of this methodology for the e-mail spam detection task.

Our results show that considering web pages for spam detection purposes is a promising strategy that creates a new battleground for spam, which has not yet being exploited by current spam filters.

The remainder of this work is organized as follows: Section 2 presents related work, while Section 3 go into details of our web page-based spam detection scheme. In Section 4 we describe the dataset we have used in this work and we discuss the experimental evaluation results, which are followed by conclusions and the discussion of future work in Section 5.

2. RELATED WORK

Very few works mention the use of web pages for e-mail spam detection purposes. To our knowledge, the first work to suggest the use of web pages for e-mail spam detection is a proposal of a framework which combines different spam filtering techniques, including a web page-based detection scheme, but the authors did not go into details about the strategy [23].

As previously mentioned, the Monarch system [27] is certainly the one that is more closely related to our proposal. Their work shows the feasibility of collecting and processing web pages in real time for spam analysis with a cost-effective infra-structure, while identifying a large number of attributes that may be used for that goal. Although their goal is mainly to classify the web content itself, the authors also show how information from a Twitter post containing a URL can be used to improve the classification of the web page pointed by it. However, due to the nature of their data they could not explore the relation between e-mail messages and the web pages they link to, since they did not have the original e-mail messages, but only the list of URLs in them. Our datasets make it possible for us to do exactly that.

Obviously, web pages are the basic unity of analysis for the *web spam detection* task, which aims to detect artificially-created pages into the web in order to influence the results from search engines [22]. In that direction, Webb [31] has created a dataset containing web pages crawled from spams from the Spam Archive dataset [13] comprising messages from 2002 and 2006. However, his dataset did not relate web pages with spam messages.

State-of-the art approaches for e-mail spam detection consider message content features and network features [5]. In terms of content-based filtering, a wide range of machine learning techniques such as Bayesian Filters, SVM Classifiers and Decision Trees have being applied over e-mail message content with reasonable success [5, 7, 2]. Although web page content has not being experimented as a spam detection strategy, URLs embedded on e-mail messages are used for phishing detection purposes by considering IP address, *WHOIS* and domain properties and geolocalization of URLs [21, 11]. The web pages linked by e-mail messages have also been used by Spamscatter [1] as a means of identifying spam campaigns by the web pages they linked to. However, their technique was applied only to spam messages already identified as such and required creating and comparing snapshots of the rendered web pages.

3. PROPOSED METHODOLOGY

For each e-mail message processed from the message stream, we download the web pages linked by the URLs contained in the message. Then, content analysis techniques are applied to the web page content — basically, the same approach adopted by filters to analyze the content of a spam message. We assign a spamicity score to the set of web pages linked by a given message and then combine this score with the result of classical spam filtering techniques and the message receives a final score, that takes into account both the message content and the web page content. Figure 1 summarizes the work-flow of our filtering technique, and we discuss the major steps next.

3.1 Web Page Crawling

We begin by extracting the URLs from the body of the messages[1] and use simple regular expressions to remove non-HTML URLs, i.e., URLs that provide links to images or executable files. After that, we download and store the web pages[2]. In the case of spam messages containing multiple URLs, all the web pages are downloaded and stored. Many of the URLs considered lead to redirections before reaching their final page; in that case, we follow all redirects and store

[1]Using the Perl Modules URI::Find and HTML::LinkExtor.
[2]Using the file transfer library *libcurl* [19].

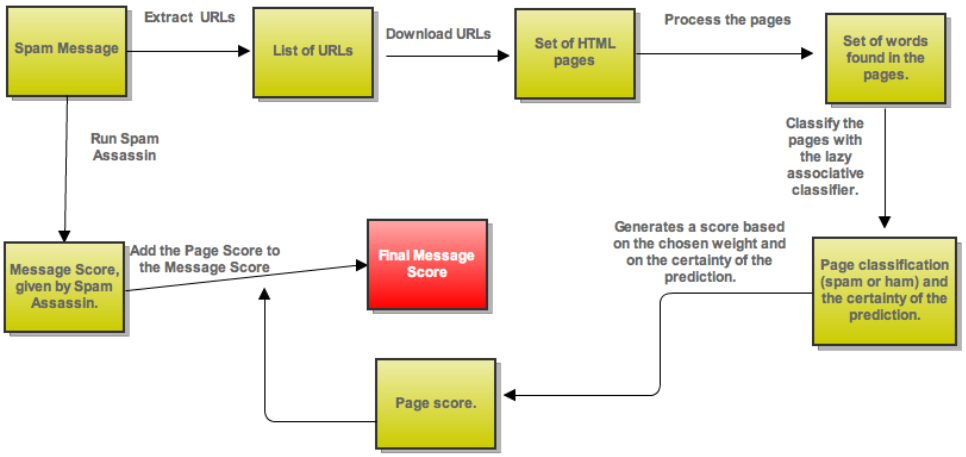


Figure 1: Steps of the Web page-based spam filtering approach. Web pages are crawled, URLs are extracted and a spamicity score is assigned to the set of web pages relative to an e-mail message. Then, web page score is combined with conventional spamicity scores to generate a final spamicity assessment.

the content of the final URL.

After extracting messages' URLs and downloading the web pages linked by them, we use lynx [20], a text-based browser, in order to format the web page's text as users would perceive it (without images). Lynx generates a dump of the web page, already in text mode, removing the non-textual part of the message such as HTML tags and JavaScript code. Textual terms in that dump are then extracted, and the final result is a set of words that are used to determine the spamicity of the web page.

3.2 Web Page Spamicity Score Computation

Given that we have at our disposal textual terms extracted from the set of web pages linked on each spam message, a range of classifiers could be built based on that information. We chose to use LAC, a demand-driven associative classifier [28, 30]. Associative classifiers are systems which integrates association mining with classification, by mining association rules that correlate features with the classes of interest (e.g., spam or ham), and build a classifier which uses relevant association patterns discovered to predict the class of an object [26]. LAC is a *demand-driven* classifier because it projects/filters the training data according to the features in the test instance, and extracts rules from this projected data. This ensures that only rules that carry information about the test instance are extracted from the training data, drastically bounding the number of possible rules [28].

We have chosen to apply a demand-driven lazy classification algorithm for several reasons: (i) it has a good performance for real-time use, (ii) it generates a readable and interpretable model in the form of association rules (which can be easily transformed into a set of regular expressions and incorporated into SpamAssassin), and (iii) it is well calibrated, which means that it provides accurate estimates of class membership properties. Formally, we state that a classifier is well calibrated when the estimated probability $\hat{p}(c|c)$ is close to $p(c|\hat{p}(c|x))$, which is the true, empirical probability of x being member of c given that the probability estimated by the classifier is $\hat{p}(c|x)$. Therefore, it is possible to know which predictions are more or less accurate and use that information when scoring different pages. For more details on class membership likelihood estimates, please refer to [29].

The demand-driven associative classifier algorithm generates rules in the form $\chi \rightarrow c$, where χ is a set of words and c is a class (spam or ham). One typical rule extracted from e-mail messages would be $buy, viagra \rightarrow spam$. Each one of those rules has a *support* (how often the rule appears in the training data) and a *confidence*, which is given by the number of pages in the training data that are classified correctly by the rule divided by the number of pages that contain the set of terms χ. The final result of each page's classification is a score between 0 and 1 that indicates both the predicted class (spam or ham) and the certainty of the prediction. This score is reliable (as it has been noted that the algorithm is well calibrated), and will be a factor for the final page score.

One of the challenges of spam detection is the asymmetry between the cost associated with classifying a spam message incorrectly and the cost associated with classifying a ham message incorrectly. A false negative might cause slight irritation, as the user sees an undesirable message. A false positive, on the other hand, means that a legitimate message may never reach the user's inbox [10]. Therefore, instead of giving the same importance to false positives and false negatives we build a cost-sensitive classifier [8]. In that scenario, the *cost* of a class measures the cost of incorrectly classifying a message of that class. As it weighs all the rules obtained for a certain message, the algorithm does a weighted sum of the rules, taking into consideration the confidence of each rule and the cost for each class, in order to give higher importance to rules that point to the class that has the higher cost. That implies that as the cost of the ham class grows, the algorithm needs more certainty in order to classify a page as spam.

3.3 Message Scoring

There are several possible ways to use the resulting score

of page classification to classify an e-mail message as spam or ham. Our approach combines the page score S_p with other spamicity scores obtained within the spam message by applying a traditional classifier. The exact formula for S_p will depend on the characteristics of that classifier.

In SpamAssassin, for example, a message is usually considered spam if it reaches 5 or more score points, assigned from a Bayes Filter, regular expressions and blacklists [25]. One way of incorporating our technique to SpamAssassin is to simply assign S_p spamicity score points to a message based on the web content of its linked pages, considering whether our classifier says it is spam ($Is = 1$) or ham ($Is = -1$), weighting the classifier certainty in that prediction (c):

$$S_p = Is * W_p * c \quad (1)$$

Note that if the classifier judges the web page to be ham, S_p will be negative and it will contribute to reduce the overall spamicity of a message. In this way, web pages that are more "spammy" will result in higher scores for their messages; web pages that look more like "ham" will result in lower (negative) scores. This is the strategy we use in this paper. The choice of W_p will influence how much impact the web page classification will have on the final score; we evaluate that for SpamAssassin in Section 4.

Another alternative would to completely eliminate the use of blacklists, and substitute them for our technique, when appropriate. This could be an interesting approach when the network resources are scarce, since both blacklists and our technique demand that a request be made to another server.

Messages that do not have URLs cannot be filtered by our technique, for obvious reasons. Those messages are filtered by conventional spam filtering methods, and their spamicity assessments are not affected by our strategy.

3.4 Illustrative Example

In this section, we present a step-by-step example of the application of our Web page-content based technique. We picked a spam message identified in October of 2010, from the Spam Archive dataset.

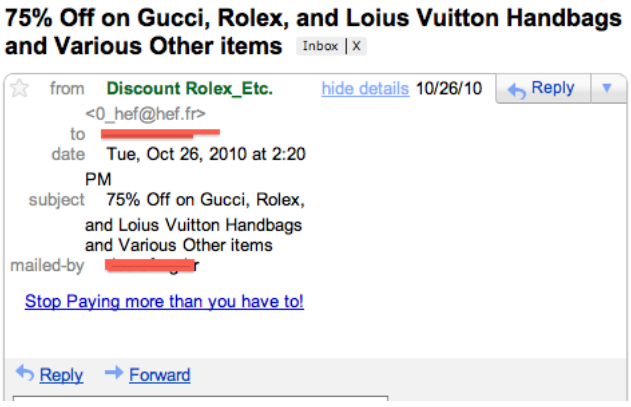


Figure 2: Spam Message extracted from *Spam Archive*. Small textual content poses challenges for analysis of spamicity based on message content.

Figure 2 shows the body of the message; it can be noted that the message is very concise, exhibiting very small textual content. SpamAssassin (considering queries to blacklists) yields the rules shown in Table 1.

Table 1: SpamAssassin Rules extracted for Spam Message from Figure 2. SpamAssassin regular expressions and queries to blacklists are not sufficient to classify the message as spam.

rule	description	score
HTML_MESSAGE	HTML included in message	0.001
RCVD_IN_ BRBL_LASTEXT	DNS Blacklist BRBL	1.644
URIBL_BLACK	Contains an URL listed in the URIBL blacklis	1.775

The resulting score considering just spam content is only 0.001. Taking blacklist information into account, the resulting score is 3.4 – still not enough to classify the message as spam. An excerpt from the web page pointed by the URL is shown in Figure 3.

Figure 3: Web page linked by URL present of message from Figure 2. As current filters do not consider web page content, spammers do not need to obfuscate and sacrifice readability.

It can be noted that, in this case, the content of the message and the content of the page are totally different – one seems to be selling watches and bags, the other is selling medicine. The content of the page is then extracted into a set of words (with lynx), and is then delivered as input for the already-trained associative classifier (it was trained with other pages from Spam Archive and from ham pages from SpamAssassin's dataset). The associative classifier finds a list of rules, some of which are listed on Table 2.

After weighting all the rules, the associative classifier yields a result: the page is a spam page, with 90% of certainty (i.e., $c = 0.9$). If we set $W_p = 4.0$, then $S_p = 3.6$. This web page, therefore, would have a 3.6 score. Adding this score to the score obtained by SpamAssassin (Table 1), this message would have a 7.0 score — more than enough to be classified as spam.

Table 2: Rules extracted for Web Page linked by message from Figure 3, unveiling high spamicity terms.

Rule	Support	Confidence
$viagra \rightarrow Spam$	36.70%	99.84%
$levitra \rightarrow Spam$	34.01%	99.90%
$rather \rightarrow Ham$	2.97%	67.30%

4. EXPERIMENTAL EVALUATION

In order to evaluate the applicability of building anti-spam filters using the content of the web pages, we built a dataset with the necessary information. In the period between July and December of 2010, we collected spam messages from the Spam Archive dataset [13]. Spam Archive is update daily, and each day we collected the web pages linked by the new messages included in that day using `wget`, to guarantee we crawled the URLs as soon as they appear. That was essential, since it is well known that spam web pages lifetimes is usually short and few spam-related web pages are online after two weeks from activation [1]. We only included in our dataset the messages for which we could download at least one web page[3].

For each of the 157,114 web pages obtained, we stored two files: one containing the HTML content of the web page and the other containing the HTTP session information. We also associated the downloaded web page with the corresponding message. We decided to evaluate only the unique pages, so that spam campaigns that pointed to the same pages would not introduce bias to our results. Whenever many different spam messages pointed to the same set of web pages, one of them was randomly chosen for the evaluation, in order that only one instance of each page remained. Note that the fact that spammers advertise the same web page multiple times inside a spam campaign (with different message content, though) would actually benefit our approach, but we decided to neutralize this affect to generate results closer to a lower bound.

Table 3: Dataset Description

Spam messages	63,034
Spam web pages	157,114
Pages downloaded per spam msg. (avg.)	2.49
Unique spam web pages	32,929
Chosen spam messages	12,111
Unique spam web pages per msg. (avg.)	2.72
Unique ham messages	4,927
Unique ham web pages	11,134
Unique ham web pages per msg. (avg.)	2.26

We used the same methodology to process ham messages from the SpamAssassin Ham Dataset[4], resulting in 11,134 unique ham pages, linked by 4,927 ham messages. The characteristics of the resulting dataset are summarized in Table 3. It is interesting to notice that the average number of pages per message was not very different between hams and spams. Figure 4 shows the distribution of the number of pages downloaded for each message in the spam dataset.

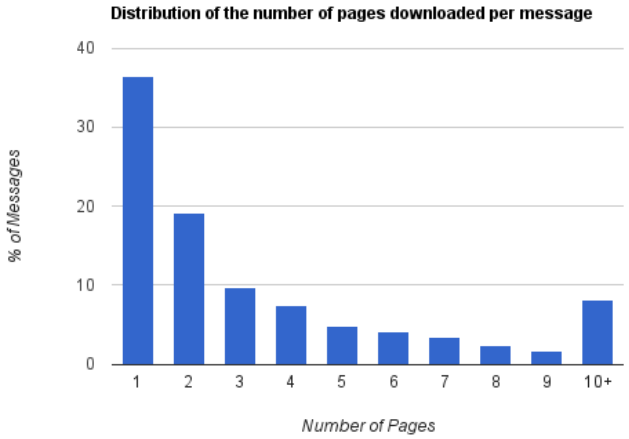


Figure 4: Distribution of the number of pages downloaded per message.

We evaluated our technique using all the resulting unique pages (ham and spam) and the sampled e-mail messages that pointed to them. In all cases, to generated results, we used a standard 5-cross validation strategy. Our technique was used with the SpamAssassin filter, with rules and blacklists activated. We combined the SpamAssassin's score with the associative classifier's score, computed as previously described (Eq. 1), by adding the two values.

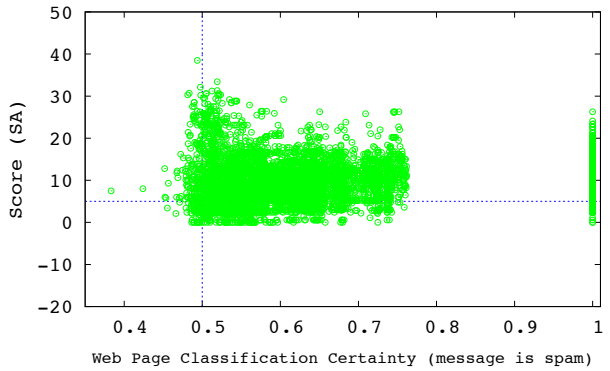
In the next sections we show the relationship between the associative classifier's certainty and the score given by SpamAssassin, and the impact of varying the parameters W_p (web page weight, i.e., the importance of web page content for classification of the e-mail message) and *cost* (the importance of an error in classifying spams and hams).

4.1 Certainty of the classifier's prediction vs SpamAssassin score

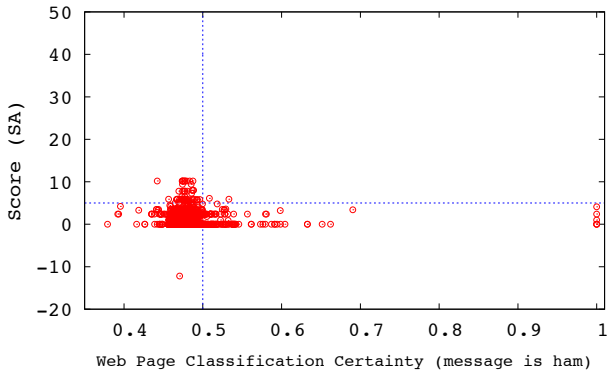
The relationship between the message score given by SpamAssassin and the certainty that the page is spam given by the associative classifier is shown in Figure 5. In this experiment, the cost of each class was the same and the W_p has been set to 4.

In Figure 5, lines show the threshold values that would separate the hams from the spams, as given by SpamAssassin scores (vertical line) and certainty in our technique (horizontal line), representing all four possible situations (spam or ham according to SA X spam or ham according to SA + Web Page approach). Figure 5a represents the spams from Spam Archive, and Figure 5b represents the hams from SpamAssasin Ham Archive. It is worth noticing that there is a large quantity of spams in the bottom right corner of Figure 5a – spams that are weakly identified by SpamAssassin rules, but are linked to pages that are identified as spams by our technique. Also, it can be noted that most hams in the bottom right corner of Figure 5a have a low score given by SpamAssassin, and a low certainty given by our technique, meaning that even if they were misclassified by our technique, they would probably not be marked as spam when both scores were combined.

[3]To obtain a copy of the dataset, please contact the authors.
[4]Available at http://spamassassin.apache.org/publiccorpus/



(a) spams



(b) hams

Figure 5: Comparison between Spam Assassin score and web page classification certainty. It is possible to note in the right bottom quadrant of (a) which spam messages are not caught by the filter, but point to messages that are caught by the web page classifier. In the right bottom quadrant of (b), on the other hand, it is possible to see that the rate of false positives would not increase considerably using our technique, since most messages in that area have a low score given by spam assassin and/or a low certainty given by the web page-content based classifier.

4.2 Impact of the weight parameter in spam detection

The relationship between the rate of false positives and false negatives and different weight values is shown in figure 6. We also show in the same figure the rate of false positives and false negatives obtained using SpamAssassin without our technique, for comparison purposes.

It can be seen that up to a value of 4, the rate of false positives of our technique is equal or lower than the rate of false positives of SpamAssassin, even though the rate of false negatives of our technique is remarkably lower. This is the reason we chose the value 4 for the weight in the other experiments, as it is the value that yields the lowest rate of false negatives while maintaining an acceptable rate of false positives.

4.3 Impact of the cost parameter in spam detection

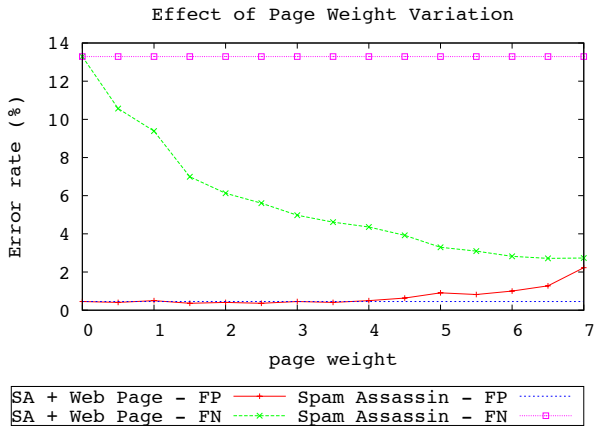


Figure 6: Rate of false positives and false negatives for Spam Assassin with and without web page classification rules, for different values of W_p (page weight). Considering web page content for spam detection reduces the rate of false negatives, without incurring in higher rate of false positives up to $W_p = 4$.

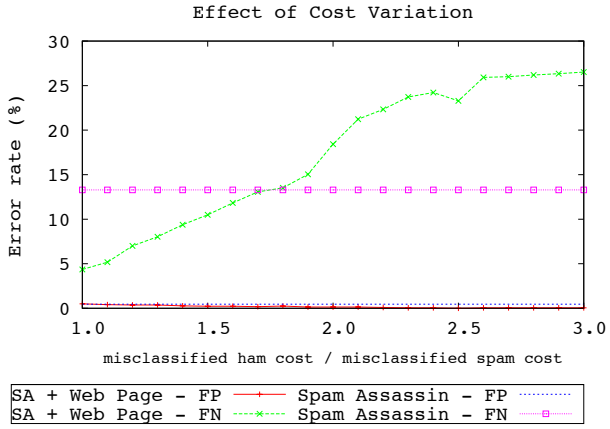


Figure 7: Rate of false positives and false negatives for Spam Assassin with and without web page classification rules, varying the rate between the cost of misclassifying a ham message and the cost of misclassifying a spam message. It can be seen that the rate of false positives and negatives can be adjusted according to the user's necessity concerning the cost of each class.

The relationship between the rate of false positives and false negatives and different cost values is shown in Figure 7. The values in the x axis represent the ratio between the cost of classifying a ham as spam and the cost of classifying a spam as ham. Therefore, if the value in the x axis is 1.5, that means that it is 50% more costly to classify a ham as spam (i.e. a false positive) than to classify a spam as ham (i.e., a false negative). We have not considered ratios lower than 1.0, since that would mean considering that false negatives had a higher impact than false positives. Therefore, it follows that a raise in the relative cost of ham misclassification yields a smaller rate of false positives and a potentially higher rate of false negatives. With a cost for misclassify-

Table 4: Frequency of top 10 terms in web pages in comparison with message content. Frequent terms in web pages are rare in messages, due to spammers' obfuscation efforts.

Word	Support in Web Pages	Support in Messages	Confidence
viagra	0.37	0.14	0.998
cialis	0.35	0.04	0.998
levitra	0.35	0.004	0.999
active	0.34	0.02	0.913
super	0.34	0.05	0.946
professional	0.33	0.05	0.937
propecia	0.32	0.0001	0.999
xenical	0.32	0.00002	0.999
tamiflu	0.31	0.00004	0.999
erectile	0.29	0.005	0.999
dysfunction	0.29	0.004	0.998

ing ham set to 70% higher than that of misclassifying spam, our technique classifies spam with the same accuracy than SpamAssassin, even though we lower the number of false positives to zero. It is worth noticing that this compromise is adjustable in our technique, through the ratio of the cost parameters. It is up to the user to set that ratio according to his situation.

4.4 Robustness of Web Page features

In this section, we assess the difficulty in detecting spams through web pages in comparison with the standard detection approach based on message content. Using the demand-driven associative classifier we presented in Section 3.2, we compute the top 10 most frequent rules the algorithm detected for web page content (relative to the *spam* class), and compare their frequency (support) in message content. Results are shown in Table 4. Note that, in web pages, the most frequent terms are present in a very high fraction of spams; over one third of the web pages contain "popular" spam terms such as *viagra*, *cialis*, *levita* and *active*. On the other hand, the prevalence of those terms in message content is significantly lower: the most popular term in web pages, *viagra*, is observed in no more than 14% of messages. For the remaining popular words in web pages, frequency in messages are lower or equal than 5%, a consequence of spammers' obfuscations in message content and the fact that spam web pages exhibit the full content of the products being advertised, while messages tend to present a summary of the advertised content.

Those numbers indicate that, as spammers currently do not have any concern in obfuscating web page content, a few number of simple rules have a strong impact on detecting spams, and will obligate spammers to react accordingly – increasing the cost of sending spam.

4.5 Operational issues

Three final aspects deserve mention: performance, the effectiveness of the technique for time changing URLs and the adaptation of spammers to our technique. We discuss these issues in this section.

Performance: adding web page analysis to the spam detection pipeline immediately brings to mind the matter of performance. Will such a system be able to keep up with the flow of incoming messages in a busy e-mail server as it crawls the web and processes pages, in addition to the existing spam filter load?

Certainly this solution will require extra processing power, since it adds a new analysis to the anti-spam bag of tricks. There are two major costs to be considered in this scenario: crawling and filtering. Although we have not measured the performance of our crawler, results from the Monarch system show that crawling pages even for a heavy mail server can be done with a median time for page of 5.54 seconds and a throughput of 638,00 URLs per day on a four core 2.8GHz Xeon processor with 8GB of memory [27]. That cost includes the DNS queries, HTTP transfers and page processing. Our feature extraction task is simpler, however, since we only extract the final text from each page. That seems an acceptable delay for an incoming message, in our opinion.

Compared to Monarch, our feature extraction process is simpler, both because we use fewer attributes and because we focus only on the text from HTML documents. One question in this comparison might be the cost of the message filtering itself, since we use a very different approach to classification. We believe that our approach, by using fewer attributes, reduces processing time, but we have no means for a direct comparison. However, in our experiments with a single dual-core Xeon Processor with 4 GB of memory, our classifier showed a throughput of 111 pages per second on average. Since message classification may be done independently for each message, the task can be easily distributed, becoming highly scalable.

Time changing URLs: one problem of the solution we proposed, first mentioned by the authors of Monarch, is that spammers might change the content of a web page over time, using dynamic pages. When a message is delivered, the server hosting the spam web pages is configured to return a simple ham page in response to a request. Knowing a server using a methodology like ours would try to verify the pages right after delivery, the server would only start returning the intended spam content some time after the SMTP delivery is completed. That, again, would require more processing by the spammer, and coordination between the spam distributing server and the spam web hosting server, which are not always under the same control [1]. If such practice becomes usual, a possible solution might be to add such tests to the user interface: as soon as an user opened a message with links, they would be collected and analyzed in the background and a message might be shown to the user.

There are other issues to be addressed as future work, as we discuss in the next section.

Spammer adaptation: It is our observation that web pages linked in e-mail messages are currently not explored by spam filters. We also observed that the "spammy" content in such web pages is clear an unobfuscated. However, it is known that spammers adapt to new techniques used by filters [14]. It could be argued that the same obfuscations used in e-mail messages could also be used in these web pages. However, that is not always possible. As with the Time Changing URLs, this would require coordination between the spam distributing server and the spam web hosting server, which are not always under the same control [1]. Not only that, but the spammer would have to sacrifice legi-

bility and the appearance of credibility in order to obfuscate the web page, as is the case with e-mail spam, which results in less profitability from each message. The current state of e-mail spam demonstrates this clearly, as many spam messages being sent are hard to understand, and do not appear credible.

5. CONCLUSIONS AND FUTURE WORK

Web pages linked by spam messages may be a reliable source of evidence of spamicity of e-mail messages. In this work, we propose and evaluate a novel spam detection approach which assigns a spamicity score to web pages linked in e-mail messages. Our approach is suitable to work as a complement to traditional spamicity assessments – such as message content and blacklists.

Our motivation for such proposal is the observation that, so far, web pages linked in e-mail messages are not explored by current spam filters, and, despite that, they offer – currently – clear and unobfuscated content for spamicity assessments. Since spam filters currently do not examine web page content, spammers usually do not obfuscate their advertised sites. Even if spammers begin to obfuscate their web pages, the effort required would serve as an additional disincentive for spam activity.

We evaluate the use of a lazy machine learning algorithm [28] to classify web pages, and propose a simple strategy for aggregating the classification of the pages to the traditional spam message classification, by using SpamAssassin [25]. We show that, by using our technique, it is possible to improve spam filtering accuracy without adding a significant number of false positives. Furthermore, the use of a cost-sensitive classifier allows the adjustment of the false positive cost, allowing users to have better control on the trade-off between false positives and false negatives.

We believe that this work explores a new frontier for spam filtering strategies, introducing a new aspect that has not yet been explored in the literature. In other words, the web pages that are linked by spam messages are a new battleground, one that spammers are not currently worried about, and one that may be explored through many different approaches and algorithms.

As future work, we intend to address issues that may surface once web page analysis becomes integrated into the spam-fighting infra-structure. They relate to the privacy of users, the abuse of the infra-structure and the obfuscation of urls, and we discuss them next.

It is possible that, by crawling a URL in a spam message sent to a user, we may be providing feedback to the spammer. That would be the case if the spammer embedded the message receiver identifier in the URL using some kind of encoding. By collecting that page our system would be confirming to the spammer that such user is active. There is no information on how often such encoded URLs are used, but they certainly demand more processing power from the spammer, what, in turn, may reduce profits.

By implying that all URLs in received messages will result in a crawler execution, this may provide a way for spammers and other individuals to start attacks by abusing such features. A heavy volume of messages crafted by an attacker with multiple URLs might overload the crawler, or be used to turn the crawler into an attacker, requesting a large volume of pages from another site. The later may be avoided just by using DNS and HTTP caches, reducing the load over any specific site. Setting limits on the number or URLs extracted from any message or the concurrent queries to a single server may prevent other overload attacks.

Finally, one countermeasure that spammers could take would be to avoid that a filter could identify the URLs in a message. Although possible, that would imply that the spammer would have to rely on the user to copy — and edit — text from a message in order to turn it into a valid URL. That would again add a level of complexity that would reduce the effect of such spam campaign.

In terms of machine learning challenges, we think the most promising direction is to combine knowledge from message and web page content in order to maximize spam detection accuracy while keeping a satisfactory filter performance: in cases where spam message content is enough to a classifier judge its spamicity, we do not need to pay the cost of crawling and analyzing web page content. Devising strategies that examine web page content only when spam message content is not enough seems to be an interesting approach to lead to robust and efficient spam detection.

Acknowledgments

This work was partially supported by NIC.br, CNPq, CAPES, FAPEMIG, FINEP, by UOL (`www.uol.com.br`) through its Research Scholarship Program, Proc. Number 20110215235100, by the Brazilian National Institute of Science and Technology for the Web, InWeb. CNPq grants no. 573871/2008-6 and 141322/2009-8, and by Movimento Brasil Competitivo, through its Brazil ICSI Visitor Program.

6. REFERENCES

[1] D. S. Anderson, C. Fleizach, S. Savage, and G. M. Voelker. Spamscatter: Characterizing Internet Scam Hosting Infrastructure. In *Proceedings of the 16th IEEE Security Symposium*, pages 135–148, 2007.

[2] I. Androutsopoulos, J. Koutsias, K. Chandrinos, G. Paliouras, and C. D. Spyropoulos. An evaluation of naive bayesian anti-spam filtering. *CoRR*, cs.CL/0006013, 2000.

[3] B. Biggio, G. Fumera, and F. Roli. Evade hard multiple classifier systems. In O. Okun and G. Valentini, editors, *Supervised and Unsupervised Ensemble Methods and Their Applications*, volume 245, pages 15–38. Springer Berlin / Heidelberg, 2008.

[4] D. Chinavle, P. Kolari, T. Oates, and T. Finin. Ensembles in adversarial classification for spam. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 2015–2018, New York, NY, USA, 2009. ACM.

[5] G. V. Cormack. Email spam filtering: A systematic review. *Found. Trends Inf. Retr.*, 1:335–455, April 2008.

[6] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma. Adversarial classification. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 99–108, New York, NY, USA, 2004. ACM.

[7] H. Drucker, D. Wu, and V. N. Vapnik. Support vector machines for spam categorization. *IEEE TRANSACTIONS ON NEURAL NETWORKS*, 10(5):1048–1054, 1999.

[8] C. Elkan. The foundations of cost-sensitive learning.

In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 973–978, 2001.

[9] eSoft. Pharma-fraud continues to dominate spam. `www.esoft.com/network-security-threat-blog/pharma-fraud-continues-to-dominate-spam/`, 2010.

[10] T. Fawcett. "in vivo" spam filtering: a challenge problem for kdd. *SIGKDD Explor. Newsl.*, 5:140–148, December 2003.

[11] I. Fette, N. Sadeh, and A. Tomasic. Learning to detect phishing emails. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 649–656, New York, NY, USA, 2007. ACM.

[12] J. Goodman, G. V. Cormack, and D. Heckerman. Spam and the ongoing battle for the inbox. *Commun. ACM*, 50(2):24–33, 2007.

[13] B. Guenter. Spam Archive, 2011. `http://untroubled.org/spam/`.

[14] P. H. C. Guerra, D. Guedes, J. Wagner Meira, C. Hoepers, M. H. P. C. Chaves, and K. Steding-Jessen. Exploring the spam arms race to characterize spam evolution. In *Proceedings of the 7th Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS)*, Redmond, WA, 2010.

[15] P. H. C. Guerra, D. Pires, D. Guedes, J. Wagner Meira, C. Hoepers, and K. Steding-Jessen. A campaign-based characterization of spamming strategies. In *Proceedings of the 5th Conference on e-mail and anti-spam (CEAS)*, Mountain View, CA, 2008.

[16] M. Illger, J. Straub, W. Gansterer, and C. Proschinger. The economy of spam. Technical report, Faculty of Computer Science, University of Vienna, 2006.

[17] S. M. Labs. MessageLabs Intelligence: 2010 annual security report. `http://www.messagelabs.com/mlireport/MessageLabsIntelligence_2010_Annual_Report_FINAL.pdf`, 2010.

[18] K. Levchenko, A. Pitsillidis, N. Chachra, B. Enright, M. Felegyhazi, C. Grier, T. Halvorson, C. K. C. Kanich, H. Liu, D. McCoy, N. Weaver, V. P. G. M. Voelker, and S. Savage. Click trajectories: End-to-end analysis of the spam value chain. *Proceedings of the IEEE Symposium on Security and Privacy*, 2011.

[19] Libcurl, 2011. `http://curl.haxx.se/libcurl/`.

[20] Lynx, 2011. `http://lynx.browser.org/`.

[21] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker. Beyond blacklists: learning to detect malicious web sites from suspicious urls. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 1245–1254, New York, NY, USA, 2009. ACM.

[22] A. Ntoulas and M. Manasse. Detecting spam web pages through content analysis. In *In Proceedings of the World Wide Web conference*, pages 83–92. ACM Press, 2006.

[23] C. Pu, S. Member, S. Webb, O. Kolesnikov, W. Lee, and R. Lipton. Towards the integration of diverse spam filtering techniques. In *Proceedings of the IEEE International Conference on Granular Computing (GrC06), Atlanta, GA*, pages 17–20, 2006.

[24] C. Pu and S. Webb. Observed trends in spam construction techniques: a case study of spam evolution. *Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS)*, 2006.

[25] SpamAssassin, 2011. `http://spamassassin.apache.org`.

[26] Y. Sun, A. K. C. Wong, and I. Y. Wang. An overview of associative classifiers, 2006.

[27] K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song. Monarch: Providing real-time URL spam filtering as a service. In *Proceedings of the IEEE Symposium on Security and Privacy*, Los Alamitos, CA, USA, 2011. IEEE Computer Society.

[28] A. Veloso, W. M. Jr., and M. J. Zaki. Lazy associative classification. In *ICDM*, pages 645–654. IEEE Computer Society, 2006.

[29] A. Veloso, W. M. Jr., and M. J. Zaki. Calibrated lazy associative classification. In S. de Amo, editor, *Proceedings of The Brazilian Symposium on Databases (SBBD)*, pages 135–149. SBC, 2008.

[30] A. Veloso and W. Meira Jr. Lazy associative classification for content-based spam detection. In *Proceedings of the Fourth Latin American Web Congress*, pages 154–161, Washington, DC, USA, 2006. IEEE Computer Society.

[31] S. Webb. Introducing the webb spam corpus: Using email spam to identify web spam automatically. In *In Proceedings of the 3rd Conference on Email and AntiSpam (CEAS) (Mountain View)*, 2006.