Analysis and Detection of Web Spam by means of Web Content

Víctor M. Prieto, Manuel Álvarez, Rafael López-García and Fidel Cacheda

Department of Information and Communication Technologies,
University of A Coruña, Campus de Elviña s/n. 15071 A Coruña, Spain
{victor.prieto, manuel.alvarez, rafael.lopez, fidel.cacheda}@udc.es,
WWW home page: http://www.tic.udc.es/

Abstract. Web Spam is one of the main difficulties that crawlers have to overcome. According to Gyöngyi and Garcia-Molina it is defined as "any deliberate human action that is meant to trigger an unjustifiably favourable relevance or importance of some web pages considering the pages' true value". There are several studies on characterising and detecting Web Spam pages. However, none of them deals with all the possible kinds of Web Spam. This paper shows an analysis of different kinds of Web Spam pages and identifies new elements that characterise it. Taking them into account, we propose a new Web Spam detection system called SAAD, which is based on a set of heuristics and their use in a C4.5 classifier. Its results are also improved by means of Bagging and Boosting techniques. We have also tested our system in some well-known Web Spam datasets and we have found it to be very effective.

Keywords: Web characterization, Web Spam, malware, data mining

1 Introduction

Currently, the WWW constitutes the biggest repository ever built, and it is continuously growing. Due to its large size, search engines are essential for users who want to access relevant information. According to a study by Jansen and Spink [16], approximately 80% of search engine users do not take into consideration those entries that are placed beyond the third result page. The increasing use of search engines (Google, Yahoo!, Bing, etc.), has made companies and web developers worry about the ranking of their Web sites. In order to achieve the first positions, it is important to improve the quality of the Web sites and to renew their contents and their "relationships" with web sites about similar topics.

Nevertheless, achieving a good place in the ranking is neither trivial nor inexpensive, so some techniques known as Web Spam [11] appeared with the purpose of getting an unfair relevance of the Web pages or sites. There are many people and organisations that are interested in using Web Spam to damage third parties (usually competitors) or to increase the PageRank in order to achieve a better position and raise their income for publicity or link selling. In fact, there are Web Spam organisations with qualified personnel and the purpose of

earning money illegally, too. Among their most common techniques (which can be combined to make their detection more difficult), we have:

- Content Spam, that is a technique based on the modification of the content or the keywords with the purpose of simulating more relevance to search engines and attract more traffic.
- Cloaking, which consists in dynamically generating different content for certain clients (e.g.: browsers) but not for others (c.g.: crawling systems).
- Redirection Spam, which consists in hiding redirections to pages with different content by means of scripts. Some clients (e.g.: browsers) will follow these redirections, but the rest (e.g.: crawlers) will not detect them, obtaining only the static content that the user will never see.
- Link Spam, which is the creation of Web Spam by means of the addition of evil links between pages with the purpose of raising their popularity. It is also possible to create "link farms", which are pages and sites interconnected among themselves with the same purpose.

Because of all the things we explained before, we should create protection mechanisms to a) the final users who can be cheated and who are wasting their time and money, b) the companies or the owners of the pages that want to get clients by means of ethical methods and c) the companies that provide search engines. The latter are very affected since they do not only lose prestige when they show Web Spam pages among their results, but they are wasting money and resources in analising, indexing and showing results of pages that should not be shown. Not protecting some of these entities means an economic loss.

Taking this into account, we propose SAAD (Spam Analyzer And Detector), a system that uses web content to identify Web Spam pages. A crawling system could incorporate this module in order to avoid Web Spam pages and, more concretely, in order not to analyse, index or show them to the final user.

The structure of this article is as follows. In section 2 we summarize the work on Web Spam techniques and their detection. Section 3 shows the presence of Web Spam on the Web and establishes the need of detecting it. It also discusses the properties we use in our system to characterise Web Spam. Section 4 explains our method for Web Spam detection, which is based on the combination of different heuristics. Section 5 shows the features of the datasets we have chosen to assess the detection methods, as well as the results we have obtained in every one of them. We also compare our results to previous experiments. Finally, in sections 6 and 7 we comment our conclusions and future work respectively.

2 Related Work

Although Web Spam has existed since the Web appeared and it has been growing with the expansion of the Web, its study in the academic sphere is recent.

Some important articles study Web Spam in general, like the one by Henzinger *et al.* [14], who discuss the importance of the phenomenon and the quality of the results that search engines offer. Gyöngyi and Garcia-Molina [11] propose

a taxonomy of Web Spam, too. The survey by Castillo and Davison [4] shows further reference on this topic. However, most of them focus on some of the main Web Spam types: Content Spam, Cloaking, Redirection Spam and Link Spam.

Regarding Content Spam, Fetterly *et al.* [8], Ntoulas and Manasse [18] and Gonzalez *et al.* [10] highlight the importance of analysing the content and their properties in order to detect Web Spam. Fetterly *et al.* [9] also discuss the special interest of the "cut-and-paste" content between Web Pages in order to detect Web Spam. Hidalgo [15] and Sahami *et al.* [22] propose similar techniques for Email Spam detection, combining content analysis and classifiers (e.g.: C4.5).

With regard to Cloaking, Gyöngyi and Garcia-Molina [11] explain some existing techniques. Wu and Davison [25] propose a detection method which is based on calculating the common words between three copies of a web page.

Link Spam has been studied by Wu and Davison [26] and Gyöngyi and Garcia-Molina [12], who presented some methods for detecting link farms. Amitay *et al.* [1] also analyse the importance of the properties of connectivity among pages in order to identify them. On the other hand, studies like the one by Zhang *et al.* [29] propose methods to prevent link farms from affecting the PageRank. Benczur *et al.* [2] introduce the idea of TrustRank, which consists in starting from a set of "clean" pages that have been analysed previously, extracting links from them and adding the targeted pages with the appropriate level of trust.

Finally, Redirection Spam has been studied by Chellapilla and Maykov [5] and Wu and Davison [25], who presented analyses about the use of this technique.

There are other techniques related to Web Spam that aim to attack browsers by means of evil JavaScript (Malware Spam). These techniques employ illicit actions such as "drive-by-downloads" attacks to take control of the system. Once the machine is in control, the attacker can perform several kind of attacks, like Web Spam, Email Spam or even DDOS (Distributed Denial Of Service) attacks. Cova *et al.* [6] study the most common techniques and their detection.

The articles we have mentioned deal with the different Web Spam techniques and, in some cases, they present their corresponding detection methods. The problem of those studies is that they treat Web Spam detection partially, since they focus exclusively on some of the described types but not in their combination. We propose a new method to detect the different kinds of Web Spam based on existing techniques as well as in the new techniques we have developed.

3 Detection of Spam by Means of Content Analysis

The importance of Web Spam on the Web is obvious. Ntoulas *et al.* [18] have created a random dataset of several domains, deciding manually whether those pages were spam. According to this study, about 70% of the pages of the .biz domain are Web Spam, followed by 35% of the .us domain. Domains like .com, .de and .net have between 15% and 20% of Web Spam pages. Although this amount is lower, it still remains high. On the other hand, we have the .edu domain, whose pages seem to be completely free from Web Spam. The study remarks that a

high percentage of the whole Web is Web Spam, so an appropriate detection that prevents systems from dealing with those pages is important.

In the next sections, we first summarize the set of existing Web Spam detection heuristics. We will discuss both the ones based in content analysis and the others. After this, we will analyse the set of detection heuristics we propose.

3.1 Detection of Some Kinds of Web Spam

Systems as the ones proposed by Ntoulas *et al.* [18] and Gonzalez *et al.* [10] base their detection on content analysis, applying the following heuristics:

- Number of words per page: they are counted and compared to the typical ratios since common words are often added to the page to raise its relevance.
- Number of words in the title: it is the same as the previous one, but applied to the title of the page since it is usually considered a relevant element.
- Word length: it measures the average length of the words, since it has been found that small words are often joint to make longer ones (e.g.: "freemp3").
- Anchor words: it calculates a ratio between the length of the text of the "anchor" tags and the length of the document, since spammers often use this text to hide links that point to spam pages.
- Ratio of the visible content: this method calculates the ratio between the content without HTML tags and the total content. This way we can know if spammers are trying to hide Web Spam in the non-visible texts.
- Compression ratio: the size of the page itself is compared to a compressed version to know whether spammers are trying to repeat certain words.
- Number of common words: spammers usually add common words from queries to capture more traffic. To avoid it, we calculate the ratio between the N most common words and the total number of common words.
- Independent n-gram probability: since gramatical and semantical analysis are very expensive, this heuristic proposes an statistical analysis of the content to detect Web Spam. To that end, we divide the content of each page in n-grams of n consecutive and independent words and compare their probabilities to the ones of Web Spam pages.
- Dependent n-gram probability: the same as the previous one but introducing dependencies between words.

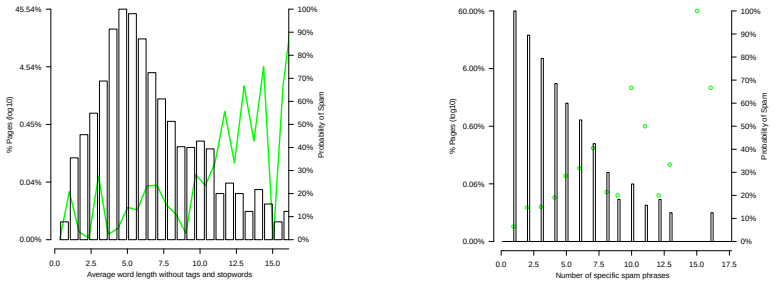
3.2 SAAD Heuristics for Web Spam Detection

The techniques we have explained in the previous section only enable detection of Web Spam in pages which use Content Spam techniques, but our goal is to deal with all the types of spam we commented before (Cloaking, Redirection Spam, Content Spam and Link Spam). With this purpose we have performed an analysis in which we have identified new heuristics.

From here on, we discuss thoroughly the main heuristics on which SAAD is based. In order to justify the efficacy of each one, we tested them on the dataset we describe in section 5. In each figure, we show the spam probability according

to the values that we have obtained for each technique. We show the ratio of pages that are affected by each value of each heuristic too. In order to improve the presentation of the data, we have used a log10 progression in the Y axis that indicates this ratio, but showing the percentage instead of the absolute value.

– Word length II: we propose an improvement of the method shown in the previous section, which consists in calculating the average length of the content without taking into account the HTML tags or the stop words. We do this because HTML tags are not content that is going to be shown to the user and would introduce noise in the results, and also because we have observed that no-spam pages employ a bigger amount of stop words. This happens because a legitimate page uses a normal quantity of prepositions, conjunctions or articles, but spam pages mainly focus on the insertion of keywords that improve their position in the search rankings. In Figure 1a we can observe that a big portion of the pages contain words with a length between 2.5 and 7.5. We can also say that the probability for a page of being spam is below 25% if its average length (without HTML tags or stop words) is lower than 8. However, if the values are bigger we can see that the probability increases progresively until it reaches near 90%. If we compare to the results obtained by Ntoulas *et al.* [18], we can see that we obtain more conclusive results, since from a value of 8, the progression grows and has better probabilities.

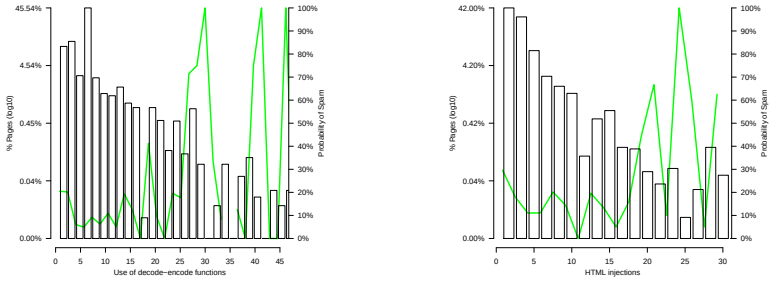


(a) (b)

Fig. 1: Probability of spam relative to the average word length II of the pages (a), and to specific phrases in the page content (b).

– Specific phrases: it is usual that spam pages contain common terms, phrases or queries from search engines. This has also been observed in other areas like Email spam where emails usually contain common sentences or specific spam words. Analysing our dataset, we have created a list of the aforementioned terms, which contains words like “viagra” or “urgent” among others. In Figure 1b we observe the results. On the one hand, there are lots of pages with low number of spam words and a low probability of being Web spam. On the other hand, the number of pages with many spam words is lower and its probability of being Web Spam is, in general, higher.

– Encode/decode functions: spammers often try to hide redirections to other pages, functions or a certain content by codifying it. Hence, the use of functions like escape/unescape is usual. It has also been found a combination of those functions one inside another to make their detection much more difficult. In Figure 2a it can be observed that the spam probability is relatively low if there are between 0 and 17 functions, so it is more probable that those pages are not spam. Nevertheless, between 17 and 22 functions and from 25 on the Web Spam probability grows reaching points with 100%.

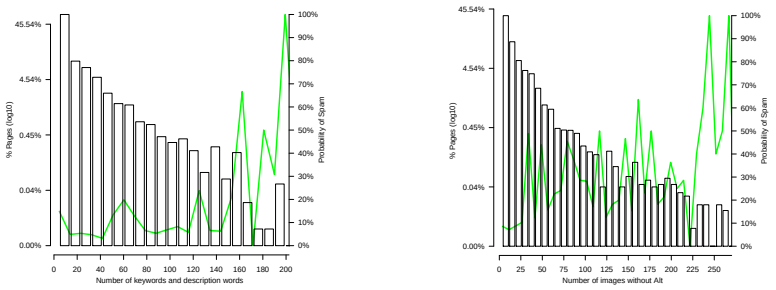


(a) (b)

Fig. 2: Probability of spam relative to encode/decode functions in the page (a), and probability of spam relative to HTML injections in the page (b).

– HTML injection: although most of the HTML is generated from user actions, we have observed that it is bigger in pages with spam. Hence, we have analysed the scripts in order to find functions that generate that code, such as: “innertext”, “outerhtml”, “createElement”, “appendChild”, etc. The results are shown in Figure 2b, where we observe that pages with less than 15 HTML injections can be considered as no-spam. However, for values bigger than 15 the probability of being spam rises to 60%, 70% or even 100%.
– Number of Keywords/Description Words: we performed an analysis of the number of words used in the “keywords” and “description” attributes of the META tag, in spam and no-spam Web pages. Results are shown in Figure 3a, where we can see that pages with less than 130 keywords have less of 20% of probability of being spam. When the number rises, the probability of being Web Spam grows as well. Taking this results into account, we decided to use more heuristics. Not only the “keywords” and “description” attributes of the META tag, but also the number of occurrences of these keywords in the attributes and through the page content.
– Images without Alt: since the content of many Web Spam pages is dynamically generated or is not thoroughly detailed, we have analysed the content of the ALT attribute of the images. Figure 3b points out that with values between 0 and 25, the probability of spam is less than 10%. If the number

is bigger, we have peaks up to 60%. When the number is bigger than 220, the probability is bigger than 50%, with some peaks of 100%.



(a) (b)

Fig. 3: Probability of spam relative to number of keywords and description words in the page (a) and to images without ALT tag in the page (b).

Apart from the heuristics we have studied, we have defined another set of heuristics. These are non-determining, but significant for the classification. We highlight the following:

- Ratio of bytes of code and total bytes: during the analysis of the pages, we noticed that the ratio between the size in bytes of the scripting code of a page and the total number of bytes grows according to the probability of spam. The bigger the code in the HTML, the bigger the probability of spam.
- Length of the evaluated code: it has been observed that the evaluated strings are longer than usual in Web Spam pages, since in many cases they contain a large number of escaped variables, functions and data.
- META tag redirection: on the one hand, spammers put content in the current page in order to make crawlers analyse and index it, but, on the other hand, they redirect users to pages with different content. We have also detected that redirections in the META tag of Web Spam pages, often have a delay of less than 5 seconds.
- Orthography: Web Spam pages often generate content automatically or by means of "copy&paste". This is why, as well as in Email Spam, the rate of mistypes is bigger than in a no-spam page.
- Number of images: spam pages are often generated automatically, so it is unusual that they have images or at least they usually do not have a number of images similar to the one of no-spam pages. Taking this into account, we measure the total number of images of each page.
- Size in bytes of the Web page: we have checked that many spam pages have a size lower than the average of normal pages in the studied datasets.
- Number of stop words: a common indexing method for search engines is extracting the stop words from the Web page and then indexing it. Hence,

spammers usually decide to include incoherent content, without conjunctions, articles, prepositions or other common words, this is, without stop words. As a technique to detect this fact, we propose to analyse the amount of stop words in the content (without HTML tags).
- Popular words: in section 3.1 we mentioned the detection of Web Spam by measuring the most common words of the content by means of different methods. We also propose to focus exclusively in the keywords and description, since they are the places where the use of common words is bigger, so spammers use them to get more traffic for their web site.

Finally, we have also taken into account heuristics that are based on theoretical features of the Web Spam pages, such as the presence of a high amount of hiddent text; the massive use of redirections, scripting functions, dynamic invocation of functions, activeX and so on. However, we have not found these heuristics as effective as we expected (and less than the ones we submitted previously), so we will not discuss them more thoroughly.

As we said before, some of the previous techniques can be used for licit purposes, although our idea is to prove that their intensive use and their combination is a determining indicator of Web Spam pages.

4 Method for Web Spam Detection

In this section we describe SAAD, the proposed method for Web Spam detection. As in Ntoulas *et al.* [18] and Gonzalez *et al.* [10], it is based on content analysis. However, we take into account the heuristics we have defined in section 3.2 in order to be able to detect all the known types of Web Spam: Cloaking, Link Spam, Redirection Spam and Malware Spam, as well as Content Spam.

For the appropriate combination of the heuristics, we have tried different classification techniques (decision trees, techniques based on rules, neural nets, etc.). We conclude that we achieve the best results when we use decision trees. More concretely, we have chosen the C4.5 algorithm [21] [20].

In order to improve the results, we have assessed two techniques, "bagging" [3] [19] and "boosting" [19]. These techniques create a set of N classifiers, combine those that obtain the best results and build a composite classifier. The "bagging" technique creates N subsets of n random elements with replacement. This way, N classifiers are obtained. Each Web page that wants to be classified has to be evaluated by each one of the N classifiers. The class in which the Web page will be added (spam or no-spam) depends on the votes of most of the N classifiers. The "boosting" technique works in a similar way. Each item has a weight associated. This weight reflects the probability of occurrence in the set. N classifiers are generated in N iterations, but for each misclassified item its weight is increased. Once again, the final decision of marking it as spam or not will be brought by a majority of results of the N classifiers.

Figure 4 shows a portion of the decision tree generated to classify a page as spam. Each node uses one of the heuristics presented in section 3.2 and,

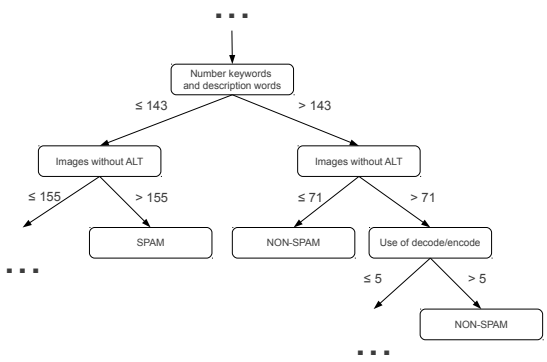


Fig. 4: A portion of decision tree

according to different thresholds, decides or delegates to another node to refine the classification.

5 Experimental Results

In this section we first explain the features of the datasets we used to obtain the results. Then, we explain different matters we have taken into account for the execution and result extraction. In the last two subsections, we thoroughly show and analyse the results we have obtained for each dataset.

5.1 Dataset

In order to test the techniques we have used two well-known Web Spam datasets:

- Webb Spam Corpus: created by Webb *et al.* [23], [24], it is the biggest Web Spam dataset, having more than 350,000 different spam pages. The authors used Email Spam detection techniques to detect spam emails. Then, they followed the links that those emails proposed and they stored the Web pages.
- WEBSPAM-UK2006/7: this dataset was created by Yahoo! [28], whose collection has more than 100 million Web pages from 114,529 hosts, although only 6,479 were manually tagged. The spam percentage is 6%. It was created for the "Web Spam Challenge 2008" [27].

The first dataset only contains Web Spam pages. To add no-spam pages, we have randomly mixed this dataset with no-spam pages from the Yahoo! one. Due to the quality and quantity of the data that these datasets provide, the conclusions that we have obtained are very reliable. Another feature we would like to highlight, apart from the size, is that both datasets are public, in contradistinction to the datasets used in similar articles. This means that any researcher can check and assess the methods and results we show by means of a similar experiment.

5.2 Experimental Setup

In order to perform the experiments, we have developed a tool that allows us to execute and store the result for each heuristic we have explained before and for each Web page we have analysed from the mentioned datasets. This tool also stores the processing time of each one. This data have allowed us to check which heuristics perform better and which ones take less processing time. The tool is also in charge of filtering each page of the datasets, preventing us from analysing pages with no content, minimal content or with only JavaScript redirections.

In order to choose the most appropriate algorithm for our heuristics, we have employed WEKA [13], a tool for automatic learning and data mining, which includes different types of classifiers and different algorithms for each classifier. Once the heuristics had been chosen, our experiment has been executed and we have obtained different results for many classification algorithms. After analysing these results, we have chosen the C4.5 classification algorithm as the best one.

For the evaluation of the classifier we have used "cross validation" [17], which consists in building k data subsets. In each iteration a new model is built and assessed, using one of the sets as "test set" and the rest as 'training set". We have used "ten-fold cross validation" ($k = 10$), since it is a widely used number.

Our experiments have been divided in two types depending on the dataset we have used. First, we have applied the heuristics proposed by Ntoulas *et al.* [18] and Gonzalez *et al.* [10] in the "Webb Spam Corpus" dataset. Then, these results have been compared to the ones obtained by our system and to the ones obtained by extending our heuristics with the ones proposed by Ntoulas *et al.* In the case of the Yahoo! dataset, we have compared the results obtained by our system to the results of the winners of the Web Spam Challenge 2008.

5.3 Results in Web Spam Dataset of Webb

We have evaluated SAAD with several datasets with 22,760, 100,000, 150,000 and 200,000 documents respectively. After this, the results have been used first as an input for the C4.5 algorithm and then they have been applied Bagging and Boosting techniques. Figure 5a shows the results obtained for dataset 1 with 22,760 pages (2,760 spam pages and 20,000 no-spam pages). This size is similar to the one employed by Ntoulas *et al.* [18]. Figure 5b shows the ones obtained for the dataset 2 with 100,000 pages (50,000 spam and 50,000 no-spam). The results we have obtained in the rest of datasets with different sizes are consistent with the ones discussed before, so we do not show them.

For the results we have obtained using only the C4.5 algorithm, we can see that SAAD improves the recall obtained by Gonzalez *et al.* by approximately 10% in both datasets. In the case of Ntoulas *et al.*, SAAD improves their results by 7% in dataset 1 (Figure 5a) and 12.5% in dataset 2 (Figure 5b). If we focus on the union of the heuristics submitted by Ntoulas *et al.* and the ones we have proposed (SAAD and Ntoulas row), we can see that we improve an additional 1.5%. In the case of the precision, we improve the results for dataset 1 by approximately 3% (Figure 5a).

	Spam		No-Spam	
	Precision	Recall	Precision	Recall
C4.5				
Gonzalez et al.	0.895	0.715	0.962	0.988
Ntoulas et al.	0.894	0.743	0.965	0.988
SAAD	0.91	0.813	0.975	0.982
SAAD and Ntoulas	0.915	0.826	0.976	0.996
C4.5 - Bagging				
Gonzalez et al.	0.958	0.719	0.962	0.979
Ntoulas et al.	0.954	0.761	0.968	0.995
SAAD	0.969	0.815	0.975	0.996
SAAD and Ntoulas	0.968	0.828	0.977	0.99
C4.5 - Boosting				
Gonzalez et al.	0.932	0.79	0.972	0.992
Ntoulas et al.	0.942	0.791	0.993	0.949
SAAD	0.972	0.85	0.978	0.997
SAAD and Ntoulas	0.982	0.851	0.979	0.998

(a)

	Spam		No-Spam	
	Precision	Recall	Precision	Recall
C4.5				
Gonzalez et al.	0.977	0.66	0.742	0.984
Ntoulas et al.	0.967	0.641	0.705	0.975
SAAD	0.96	0.766	0.797	0.969
SAAD and Ntoulas	0.936	0.715	0.744	0.944
C4.5 - Bagging				
Gonzalez et al.	0.99	0.669	0.749	0.993
Ntoulas et al.	0.981	0.65	0.712	0.986
SAAD	0.982	0.75	0.797	0.986
SAAD and Ntoulas	0.962	0.712	0.747	0.969
C4.5 - Boosting				
Gonzalez et al.	0.985	0.703	0.768	0.989
Ntoulas et al.	0.974	0.657	0.719	0.98
SAAD	0.983	0.767	0.818	0.987
SAAD and Ntoulas	0.949	0.776	0.81	0.958

(b)

Fig. 5: Results for dataset 1 (22,760 pages, 2,760 spam and 20,000 no-spam) (a), and results for dataset 2 (100,000 pages, 50,000 spam and 50,000 no-spam) (b)

After applying Bagging, SAAD improves about 10% the results by Gonzalez *et al.* for dataset 1 and 8.2% for dataset 2 (Figure 5b). SAAD also achieves an improvement of 5.6% in the results by Ntoulas *et al.* for dataset 1 and 10% for dataset 2. In this case, SAAD obtains 4% better results than the union of our heuristics and the ones proposed by Ntoulas *et al.*

The results after applying Boosting are better than the previous ones. Once again, SAAD improves the previous results. For dataset 1, SAAD achieves an improvement of 6% in the results submitted by Ntoulas *et al.* and Gonzalez *et al.* For dataset 2, we improve the results by 6% in the case of Gonzalez *et al.* and approximately by 10% in the case of Ntoulas *et al.* If we focus on the results obtained by joining our heuristics with the ones proposed by Ntoulas *et al.*, we find that we only improve by 1% the results obtained if we only use our system.

In Figure 6 we show the best results we have obtained. With the information from this figure and from Figure 5a and Figure 5b, we can observe two different tendencies. First, the results are better if we use C4.5 and Bagging, and they improve even more if we use Boosting. Second, the best results are obtained when we join the techniques we have proposed before and the ones submitted by Ntoulas *et al.* [23], although in some cases the results are the same or worse than the ones obtained by applying only SAAD. Finally, we have the ones obtained by applying the methods proposed by Ntoulas *et al.* and Gonzalez *et al.* respectively.

5.4 Results in Web Spam Dataset of Yahoo!

In the previous section there were only two classes of pages: spam and no-spam. In the case of Yahoo! dataset, a new "undecided" class has been created. This

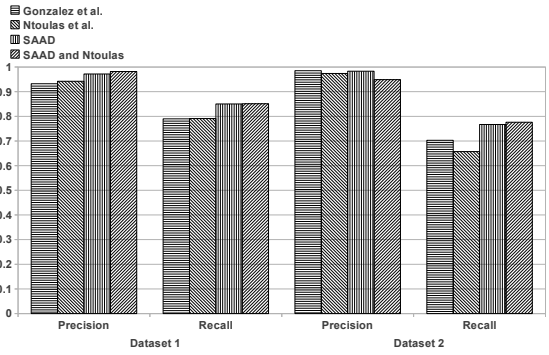


Fig. 6: Summary of results

class contains those pages which are in the limit between spam and no-spam. Before commenting the results, we want to discuss some matters that will help to understand them. First, we will only show the result of the C4.5 algorithm with Boosting, since in the previous section it has been stated that it is the technique that produces the best results. Second, we have not analysed the results obtained by Ntoulas *et al.* and Gonzalez *et al.* in the Web Spam Challenge [27], since it has been proved in the previous section that our methods obtain better results. Finally, we show and compare the ROC area since it is the method used by Yahoo! for showing the results.

We have created two subsets from the Yahoo! dataset, the first one with 20,000 Web pages and the second one with 50,000. Both datasets have been created by means of a random selection and always maintaining the 6% of spam that Yahoo! indicated.

Dataset	**20000**				**50000**			
	Class	Precision	Recall	ROC	Class	Precision	Recall	ROC
C4.5 - Boosting	Spam	0.992	0.966	0.997	Spam	0.999	0.971	0.996
	No-Spam	0.99	0.999	0.991	No-Spam	0.996	0.999	0.996
	Undecided	0.985	0.87	0.989	Undecided	0.986	0.953	0.995

Fig. 7: Results on Yahoo! datasets

In Figure 7 we can observe the results obtained for both datasets. The first, with 20,000 pages, has 17,679 no-spam pages, 1,270 spam pages and 1,043 Undecided pages. The second dataset we show has 50,000 pages, containing 43,584 no-spam, 3,368 spam and 3048 Undecided. The results we have obtained are very good, since in the spam detection SAAD obtains a ROC area between 0.996 and 0.997. Similar results, near 0.99, have been obtained for the no-spam and Undecided classes. Figure 8 shows clearly the improvements for the results we have obtained with respect to the results obtained in Yahoo!'s Web Spam Challenge.

In spam detection, SAAD has achieved an improvement of 15% in the worst case and near 27% in the best case. The results for no-spam detection or Undecided pages have not been shown by Yahoo!, so we cannot make a comparison.

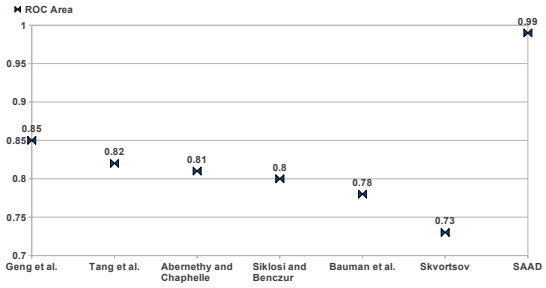


Fig. 8: Comparison of results by ROC area

6 Conclusions

We have proposed a wide number of properties that help us to detect Web Spam and that improve and increase the ones that have been discussed in other articles. Our heuristics do not deal with the relationships between Web pages or sites, but we analyse a lot of features of the Web content to try to detect Web Spam.

Each technique we have presented could identify by itself a Web Spam page, but legitimate ones could be also classified as spam. In the same way, if we trusted in the techniques independently, spammers could avoid them easily. For this reasons, we have proposed the union of all the heuristics in order to create a more robust detector. This improves a lot the probability of success when we are detecting Web Spam and makes it more difficult to avoid all techniques.

Our methods have been tested with two public datasets. First, we have used one created from the Email Spam (Webb Spam Corpus), which exclusively contains Web Spam pages. Then, we hvae used a Yahoo! dataset, manually tagged.

At the same time we have joined the methods, we have created a tool that generates the results. These results have been used as the input of several classifiers. We have proved that the best results are obtained by means of a C4.5 classification algorithm improved by applying Boosting. With this, we improve from 6% to 10% the results submitted by Ntoulas *et al.* and Gonzalez *et al.*

In the case of the Yahoo! dataset [28], we have achieved a ROC area up to 0.99, this is, between 15% and 27% better than the results achieved by the winners of the Web Spam Challenge 2008 [27].

To the best of our knowledge, these results are the best we have found in similar studies, so the use of the proposed system in crawlers would generate a safer client environment, saving resources since they would not waste them in analysing, indexing or even showing Web Spam.

7 Future Work

We contemplate a future analysis of the profits and efficiency of including a module in web browsers, in order to alert the user when risky pages are being shown. We also think about using it in a crawling system, so it will improve the resource use by not wasting them in analysing and indexing pages that are potential Web Spam. With this, we would improve the ranking results shown to the user. We would also like to discuss the possibility of generalising the Web Spam detector to a distributed architecture, based in the Map-Reduce technique[7], which is currently the most used in crawling environments.

Another complementary study would be the analysis of the computing times of each method and the possibility of combining them in two or more stages, separating those which have lower resource use and computing time, and those which are more effective in the detection but they are also very resource demanding. This way, the system would be more efficient, since if a first stage with a low resource use marks a page as Web Spam, it would not be necessary to execute other resource demanding modules.

8 Acknowledgments

This work was partialy supported by the Spanish government, under project TIN 2009-14203.

References

1. Amitay, E., Carmel, D., Darlow, A., Lempel, R., Soffer, A.: The connectivity sonar: detecting site functionality by structural patterns. In: In Proceedings of the Fourteenth ACM Conference on Hypertext and Hypermedia. pp. 38–47. ACM Press (2003)
2. Benczur, A.A., Csalogany, K., Sarlos, T., Uher, M., Uher, M.: Spamrank - fully automatic link spam detection. In: In Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb (2005)
3. Breiman, L., Breiman, L.: Bagging predictors. In: Machine Learning. pp. 123–140 (1996)
4. Castillo, C., Davison, B.D.: Adversarial Web Search 4(5), 377–486 (2010)
5. Chellapilla, K., Maykov, A.: A taxonomy of javascript redirection spam. In: Proceedings of the 3rd international workshop on Adversarial information retrieval on the web. pp. 81–88. AIRWeb '07, ACM, New York, NY, USA (2007)
6. Cova, M., Kruegel, C., Vigna, G.: Detection and analysis of drive-by-download attacks and malicious javascript code. In: Proceedings of the 19th international conference on World wide web. pp. 281–290. WWW '10, ACM, New York, NY, USA (2010)
7. Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. Commun. ACM 51, 107–113 (January 2008)
8. Fetterly, D., Manasse, M., Najork, M.: Spam, damn spam, and statistics: using statistical analysis to locate spam web pages. In: Proceedings of the 7th International Workshop on the Web and Databases: colocated with ACM SIGMOD/PODS 2004. pp. 1–6. WebDB '04, ACM, New York, NY, USA (2004)

9. Fetterly, D., Manasse, M., Najork, M.: Detecting phrase-level duplication on the world wide web. In: In Proceedings of the 28th Annual International ACM SIGIR Conference on Research & Development in Information Retrieval. pp. 170–177. ACM Press (2005)
10. Gonzalez Jesus, B.W., Cristina, A.: Implementacion y evaluacion de un detector masivo de web spam (2009)
11. Gyongyi, Z., Garcia-Molina, H.: Web spam taxonomy. Technical Report 2004-25, Stanford InfoLab (March 2004)
12. Gyöngyi, Z., Garcia-Molina, H.: Link spam alliances. In: Proceedings of the 31st international conference on Very large data bases. pp. 517–528. VLDB '05, VLDB Endowment (2005)
13. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. 11, 10–18 (November 2009)
14. Henzinger, M.R., Motwani, R., Silverstein, C.: Challenges in web search engines. SIGIR Forum 36, 11–22 (September 2002)
15. Hidalgo, J.M.G.: Evaluating cost-sensitive unsolicited bulk email categorization (2002)
16. Jansen, B.J., Spink, A.: An analysis of web documents retrieved and viewed (2003)
17. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. pp. 1137–1143. Morgan Kaufmann (1995)
18. Ntoulas, A., Manasse, M.: Detecting spam web pages through content analysis. In: In Proceedings of the World Wide Web conference. pp. 83–92. ACM Press (2006)
19. Quinlan, J.R.: Bagging, boosting, and c4.5. In: In Proceedings of the Thirteenth National Conference on Artificial Intelligence. pp. 725–730. AAAI Press (1996)
20. Quinlan, J.R.: Improved use of continuous attributes in c4.5. Journal of Artificial Intelligence Research 4, 77–90 (1996)
21. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1993)
22. Sahami, M., Dumais, S., Heckerman, D., Horvitz, E.: A bayesian approach to filtering junk e-mail (1998)
23. Webb, S.: Introducing the webb spam corpus: Using email spam to identify web spam automatically. In: In Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS) (Mountain View (2006)
24. Webb, S.: Webb Spam Corpus. `http://www.cc.gatech.edu/projects/doi/WebbSpamCorpus.html` (2011)
25. Wu, B., Davison, B.D.: Cloaking and redirection: A preliminary study (2005)
26. Wu, B., Davison, B.D.: Identifying link farm spam pages. In: Special interest tracks and posters of the 14th international conference on World Wide Web. pp. 820–829. WWW '05, ACM, New York, NY, USA (2005)
27. Yahoo!: Web spam challenge (2011), `http://webspam.lip6.fr/wiki/pmwiki.php`
28. Yahoo!: Web Spam Detection - Resources for Research on Web Spam. `http://barcelona.research.yahoo.net/webspam/` (2011)
29. Zhang, H., Goel, A., Govindan, R., Mason, K., Van Roy, B.: Making Eigenvector-Based Reputation Systems Robust to Collusion, Lecture Notes in Computer Science, vol. 3243/2004, pp. 92–104. Springer Berlin / Heidelberg (2004)