Introducing the Webb Spam Corpus: Using Email Spam to Identify Web Spam Automatically

Steve Webb
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
webb@cc.gatech.edu

James Caverlee
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
caverlee@cc.gatech.edu

Calton Pu
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
calton@cc.gatech.edu

ABSTRACT

Just as email spam has negatively impacted the user messaging experience, the rise of *Web spam* is threatening to severely degrade the quality of information on the World Wide Web. Fundamentally, Web spam is designed to pollute search engines and corrupt the user experience by driving traffic to particular spammed Web pages, regardless of the merits of those pages. In this paper, we identify an interesting link between email spam and Web spam, and we use this link to propose a novel technique for extracting large Web spam samples from the Web. Then, we present the Webb Spam Corpus – a first-of-its-kind, large-scale, and publicly available Web spam data set that was created using our automated Web spam collection method. The corpus consists of nearly 350,000 Web spam pages, making it more than two orders of magnitude larger than any other previously cited Web spam data set. Finally, we identify several application areas where the Webb Spam Corpus may be especially helpful. Interestingly, since the Webb Spam Corpus bridges the worlds of email spam and Web spam, we note that it can be used to aid traditional email spam classification algorithms through an analysis of the characteristics of the Web pages referenced by email messages.

1. INTRODUCTION

As the Web grew to become the primary means for sharing information and supporting online commerce, the problems associated with *Web spam* also grew. Web spam is defined as Web pages that are created to manipulate search engines and deceive Web users [12, 13]. Just as email spam has negatively impacted the email user experience, the rise of Web spam is threatening to severely degrade the quality of information on the World Wide Web. Web spam is regarded as one of the most important challenges facing search engines and Web users [12, 14], and recent studies suggest that it accounts for a significant portion of all Web content, including 8% of Web pages [11] and 18% of Web sites [13].

Although the problems posed by Web spam have been widely acknowledged, we believe research progress has been limited by the lack of a publicly available Web spam corpus. In previous Web spam research [2, 6, 7, 9, 10, 11, 13, 21], proposed solutions have been evaluated on relatively small Web spam data sets (on the order of hundreds of Web pages). In many cases, these previous researchers had access to large samples of Web data (on the order of millions of pages); however, the onerous task of hand-labeling each Web page made it impossible for them to evaluate even a small fraction of their data. Given the size and dynamic nature of Web content, a manual approach is neither scalable nor sensible. Additionally, none of the previously cited Web spam data sets have been made publicly available so the reproducibility of current Web spam research results is somewhat limited.

Similar to email spam research on the experimental evaluation of spam filters using large spam corpora such as the Enron [15] and SpamArchive [20] corpora, future Web spam research depends on the availability of large collections of Web spam data. Thus, the first contribution of the paper is a fully automatic Web spam collection technique for extracting large Web spam samples. Our new collection technique is based on the observation that the URLs found in email spam messages are reliable indicators of Web spam pages. Specifically, we extract the URLs from spam messages, cleanse those URLs of false positives (i.e., URLs for legitimate sites), and collect the corresponding Web pages. Given the dynamic nature of the Web, this collection method is extremely useful because it can be configured to maintain up-to-date Web spam data samples.

The second contribution of the paper is the Webb Spam Corpus – a large-scale and publicly available Web spam data set that was created using our automated Web spam collection method[1]. This corpus consists of nearly 350,000 Web spam pages, making it more than two orders of magnitude larger than any other previously cited Web spam data set. We describe interesting characteristics of this corpus, and we encourage Web spam and email spam researchers to use it in their work.

The third part of the paper outlines the usefulness of the Webb Spam Corpus in several application areas. We summarize related research efforts and describe how our Web spam collection technique and corpus could immediately enhance this previous work. Then, we present other interesting application scenarios that we believe could benefit from our approach. One particularly interesting application area is email filtering. Since the Webb Spam Corpus bridges the worlds of email spam and Web spam, we note that it can be

CEAS 2006 - Third Conference on Email and Anti-Spam, July 27-28, 2006, Mountain View, California USA

[1]The Webb Spam Corpus can be found at http://www.webbspamcorpus.org/.

used to aid traditional email spam classification algorithms through an analysis of the characteristics of the Web pages referenced by email messages.

The rest of the paper is organized as follows. Section 2 describes our automated technique for obtaining Web spam pages, and it explains how this technique was used to create the Webb Spam Corpus. Section 3 provides two sample applications that will immediately benefit from our automated technique and corpus: (1) automatic classification and filtering of Web spam pages and (2) identifying link-based Web spam. Section 4 concludes the paper and provides future research directions.

2. THE WEBB SPAM CORPUS

In this section, we describe our method for automatically obtaining Web spam pages, and we present the Webb Spam Corpus – a collection of almost 350,000 Web spam pages that were obtained using our fully automated collection method. First, we explain our general methodology for collecting Web spam. Then, we provide a step-by-step example to clarify the general technique. Finally, we describe the cleansing operations we performed to improve the quality and usefulness of the Webb Spam Corpus.

2.1 Obtaining the Corpus

The motivation for our Web spam collection method comes from the observation that email spammers often include URLs in their spam messages. In fact, in a related paper [17], we show that at least one URL has appeared in between 85% and 95% of SpamArchive spam messages in all but one month over the course of the past three years. We leverage the presence of those URLs in email spam to aid in the collection of Web spam examples.

2.1.1 General Methodology

As previously mentioned, our Web spam collection technique relies on the URLs found in email spam messages. We obtained our email spam messages from the SpamArchive corpora[2]. Between November 2002 and January 2006, SpamArchive collected and published close to two thousand archives (each stored as a gzipped mbox folder), totaling more than 1.4 million spam messages. We used all of these messages to help obtain the Webb Spam Corpus.

First, we downloaded all of the SpamArchive archives that were published up until January 6, 2006, and we gunzipped these archives to obtain their corresponding mbox folders. Then, we parsed the messages in each mbox folder to obtain a list of the unique URLs that were present in the "Subject" headers and bodies of those messages. We extracted URLs from arbitrary text using Perl's URI::Find::Schemeless module, and we used Perl's Html::LinkExtor module to extract URLs from HTML. In total, we extracted almost 1.2 million unique URLs, which we will refer to as the *intended URLs*. Once we had this list of intended URLs, we wrote a custom crawler (based on Perl's LWP::Parallel::UserAgent module) to obtain their corresponding Web pages.

The crawler attempted to access each of the intended URLs; however, many of the URLs returned HTTP-level redirects (i.e., a 3xx HTTP status code). The crawler followed all of these redirects until it finally accessed a URL that did not return a redirect. We refer to this final URL in the redirect chain as an *actual URL*, and it is important to note that all of the intended URLs that were successfully accessed without returning a redirect (i.e., they returned 2xx HTTP status codes) are also considered actual URLs. To illustrate the amount of redirection that occurred, Table 1 shows the number of intended URLs that returned between 0 and 13 redirects. At the top of the table, we observe that the majority (204,077) of the intended URLs were also actual URLs, and at the bottom of the table, we observe that 6 intended URLs forced the crawler to follow 13 redirects before it accessed the actual URL.

Table 1: Number of redirects returned by intended URLs

Number of Redirects	Number of Intended URLs
0	204,077
1	92,952
2	37,789
3	7,230
4	4,358
5	1,120
6	585
7	322
8	117
9	55
10	61
11	193
12	13
13	6

Our crawler obtained two types of information for every successfully accessed URL (including those that returned a redirect): the HTML content of the page identified by the URL and the HTTP session information associated with the page request transaction. The HTTP session information contains a number of headers, but the exact content varies from page to page. The most common headers include the HTTP status code, "Server" (the version of the Web server that served the page), "Content-Type", and "Client-Peer" (the IP address of the machine that served the page). In addition to the standard headers obtained by the crawler, we also added a header for a page's URL using the following format:

URL: <*URL of the page*>

We stored each of these HTTP session headers in a file by using HTML's commenting mechanism (i.e., <!−− HEADER −−>) to ensure each file is a valid HTML document (and parseable by an HTML parser). For example, in the actual corpus files, "Content-Type: text/html" is stored as "<!−− Content-Type: text/html −−>".

For each of the actual URLs, the corresponding HTML content and session information are both stored in a single file that abides by the following naming convention:

<*md5 hash of the page's HTML content*>_*m*,

where m is an integer value used to uniquely identify files that share the same md5 hash value for their HTML content.

[2]SpamArchive's spam corpora can be found at ftp://spamarchive.org/pub/archives/.

For example, MD5_0 contains the first page with MD5 as the md5 hash value for its HTML content, MD5_1 contains the second page with this md5 value, and so on.

For each of the intended URLs that has an associated redirect chain, the HTML content (if any exists) and session information for each link in the redirect chain are both stored in a single file that abides by the following naming convention:

$$<\textit{md5 hash of the page's HTML content}>\_m\_\text{redirect}\_n,$$

where m is the same as above, and n is an integer used to uniquely identify each link in a given redirect chain. For example, MD5_0_redirect_0 contains the original response that was obtained from the intended URL, and MD5_0_redirect_1 contains the response that was obtained from the next link in the chain. This pattern continues for as many links as there were in the redirect chain. As explained above, MD5_0 contains the page that corresponds to the end of the chain (i.e., the response obtained from the actual URL). Also, by extracting the "URL" value from each file's session information, it is possible to traverse the path of links that lead from the intended URL to the actual URL.

We used the md5 hash of each page's HTML content as the primary file name information to facilitate efficient duplicate page detection within the corpus. However, it is important to note that we have not actually removed any duplicate pages from the corpus. Since each of the intended URLs was unique, the duplicate pages in the corpus imply multiple entrances (or gateways) to the same page. This situation is very similar to Web spamming techniques such as link exchanges and link farms, and in Section 3.2, we will investigate this observation further. We believe these types of observations are extremely interesting and quite useful for investigating the techniques that are being used by Web spammers. Thus, we have tried to keep perturbations of the corpus to a minimum. Unfortunately, some corpus cleansing operations were unavoidable, and we describe those operations in Section 2.2.

2.1.2 Illustrative Example

To help clarify our general methodology, we provide a step-by-step explanation (with examples) of our automatic Web spam collection technique. We begin this description with an example of an email spam message that we obtained from SpamArchive. Figure 1 shows the headers and body of this spam message.

Upon obtaining this message, we parsed its "Subject" header and message body text to obtain a list of intended URLs. In this example, two intended URLs were found:

`http://click.recessionspecials.com/sp/t.pl?id=92408:57561182`
and
`mailto:unsub-53821024-5237@recessionspecials.com`.

We rejected the second URL because we only retained URLs with "http" or "https" as their scheme. It is important to note that many URLs were schemeless, and we retained all of those URLs.

After we parsed `http://click.recessionspecials.com/sp/t.pl?id=92408:57561182` as an intended URL, we used our crawler to obtain its corresponding Web page. However, this URL returned a redirect that directed the crawler to `http://click.recessionspecials.com/`. Figure 2 shows the HTTP session information (it did not have any HTML content) associated with this redirect. As described above in Section 2.1.1, this session information provides valuable information about the HTTP page request transaction. For example, the figure shows the HTTP status code (302), the type of Web server that processed the request (Apache/2.0), and the next URL in the redirect chain (`http://click.recessionspecials.com/`).

```
From nobody Wed May 28 18:53:38 2003
Return-Path: <bounce-53821024-5237@mail22.recessionspecials.com>
Received: from mail22.recessionspecials.com ([65.61.148.12])
    by (InterMail vM.5.01.05.17 201-253-122-126-117-20021021) with SMTP id
    for  Sat, 22 Mar 2003 03:23:44 -0700
Content-Disposition: inline
Content-Transfer-Encoding: 7bit
Content-Type: text/plain; boundary="_----------=_3645302494369200417066"
MIME-Version: 1.0
X-Mailer: MIME::Lite 2.117  (F2.6; B2.12; Q2.03)
Date: Sat, 22 Mar 2003 10:23:43 UT
Subject: You've Won!
X-List-Unsubscribe: <unsub-53821024-5237@recessionspecials.com>
From: "Vicki" <returns-lztfkoskhyzktw@recessionspecials.com>
Reply-To: "Vicki" <returns-lztfkoskhyzktw@recessionspecials.com>
Return-Path: <bounce-53821024-5237@recessionspecials.com>
To: submit@spamarchive.org

You've Won!

Click to see what it is:
http://click.recessionspecials.com/sp/t.pl?id=92408:57561182
-------------------------------------------------

Remove yourself from this recurring list by sending a blank email to
mailto:unsub-53821024-5237@recessionspecials.com
```

Figure 1: Example email spam message obtained from SpamArchive

```
<!-- URL: http://click.recessionspecials.com/sp/t.pl?id=92408:57561182 -->
<!-- HTTP/1.1 302 Found -->
<!-- Connection: close -->
<!-- Date: Fri, 23 Dec 2005 18:11:43 GMT -->
<!-- Location: / -->
<!-- Server: Apache/2.0 -->
<!-- Content-Length: 0 -->
<!-- Content-Type: text/html; charset=UTF-8 -->
<!-- X-Powered-By: PHP/5.0.5 -->
```

Figure 2: Example HTTP session information obtained from an HTTP redirect

Upon receiving the redirect, our crawler attempted to obtain the Web page corresponding to the next URL in the redirect chain. This next URL did not return a redirect (i.e., it is an actual URL) so the crawler successfully obtained the page. Figure 3 shows the HTTP session information and HTML content associated with this Web spam page.

The md5 hash value for this page's HTML content is 25ca3b2835685e7d90697f148a0ae572. Thus, we used this md5 value to name all of the corpus files associated with this page. The data shown in Figure 2 is stored in a file named 25ca3b2835685e7d90697f148a0ae572_0_redirect_0. Similarly, the information shown in Figure 3 is stored in a file named 25ca3b2835685e7d90697f148a0ae572_0.

To investigate this Web spam page further, we rendered its HTML content in a popular Web browser (Firefox). Figure 4(a) shows the browser-rendered view of the HTML file shown in Figure 3. This figure clearly shows that the page is an example of a fake directory (also known as a directory clone [12]) – a seemingly legitimate page that contains a vast number of outgoing links to other pages, grouped into categories. Legitimate directories (e.g., the DMOZ Open Directory) attempt to provide users with an unbiased, categorized view of the Web. Fake directories also provide a categorization of the Web, but it is biased by the motivations of the Web spammer. Figure 4(b) shows the browser-rendered view of the page that is returned after a user clicks on the "Travel" link (located at the top-left of the original

```
<!-- URL: http://click.recessionspecials.com/ -->
<!-- HTTP/1.1 200 OK -->
<!-- Connection: close -->
<!-- Date: Fri, 23 Dec 2005 18:12:19 GMT -->
<!-- Server: Apache/2.0 -->
<!-- Content-Length: 732 -->
<!-- Content-Type: text/html; charset=UTF-8 -->
<!-- Client-Peer: 64.40.102.44:80 -->
<!-- Link: <http://static.hitfarm.com/template/qing/images/qing.ico>;
/="/"; rel="shortcut icon"; type="image/x-icon" -->
<!-- P3P: CP="NOI COR NID ADMa DEVa PSAa PSDa STP NAV DEM STA PRE" -->
<!-- Set-Cookie: source=1; expires=Fri, 23 Dec 2005 20:12:19 GMT -->
<!-- Title: recessionspecials.com -->
<!-- X-Powered-By: PHP/5.0.5 -->

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html lang="en">
<head>
<title>recessionspecials.com</title>
<link rel="shortcut icon" href="http://static.hitfarm.com/template/qing/
images/qing.ico" type="image/x-icon" />
</head>
<frameset cols="1,*" border="0" frameborder="0">
  <frame name="hftop" src="/top.php" scrolling="no" frameborder="0"
    marginwidth="0" marginheight="0" noresize="noresize" />
  <frame name="hfasi" src="http://apps5.oingo.com/apps/domainpark/
    domainpark.cgi?cid=MEDI3409&s=recessionspecials.com&ip=130.207.5.18"
    scrolling="auto" frameborder="0" marginwidth="0" marginheight="0"
    noresize="noresize" />
<noframes>
<body>
<p>This page requires frames</p>
</body>
</noframes>
</frameset>
</html>
```

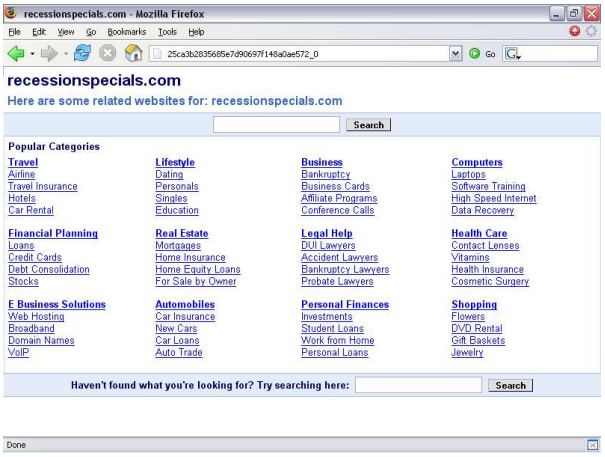
Figure 3: Example HTTP session information and content for a Web spam page

page). This page is filled with travel-related links; however, all of the links are tied to Google's AdSense program. Thus, the Web spammer receives a monetary reward every time a user clicks on one of these links. Also, as is often the case with fake directories, some of these links point to pages that are controlled by the Web spammer. Thus, this spamming technique also generates additional traffic for the spammer's other pages. This example illustrates one of the many interesting observations that can be made with the aid of our technique and corpus. In Section 3, we will investigate other areas where our work can be applied.

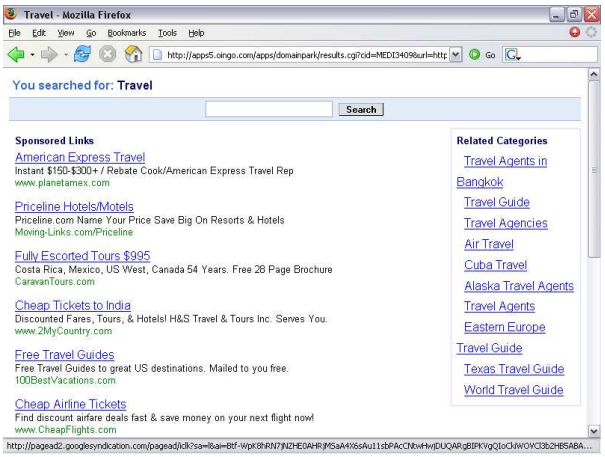
2.2 Cleansing the Corpus

In this section, we describe the cleansing operations we performed to make the Webb Spam Corpus as useful as possible. These cleansing operations fall into two categories: "Content-Type" purification and false positive removal. During the crawling process, we obtained 407,987 files that had a number of different values in their "Content-Type" session headers (e.g., application, audio, image, video, etc.). Since we were only interested in maintaining HTML Web pages in the Webb Spam Corpus, we removed all of the files with non-textual "Content-Type" headers (25,065 content files and their corresponding 33,312 redirect files).

After we removed the non-textual files from the corpus, we began analyzing the remaining 382,922 Web pages. During this analysis, we found a number of duplicates. Specifically, of the 382,922 spam Web pages, 101,453 were duplicates, and 281,469 were unique. Duplicate pages exist in the corpus because a number of intended URLs directed our crawler to the same actual URL. Thus, each of the duplicate pages has a unique redirect chain associated with it. Figure 5 shows the distribution of intended URLs per actual URL (i.e., the number of intended URLs that point to the same actual URL) that we found during our initial analysis. The x-axis shows how many intended URLs mapped to a single actual URL, and the y-axis shows how many of these actual URLs there were. A point at position (x, y) denotes the existence



(a)



(b)

Figure 4: Examples of browser-rendered Web spam pages

of y actual URLs such that each actual URL was pointed to by x intended URLs.

From the figure, we see that 267,533 actual URLs were uniquely pointed to by one intended URL (the point at the top-left of the figure), and 7,628 actual URLs were pointed to by two intended URLs. On the opposite end of the spectrum, one particular actual URL was pointed to by 6,108 intended URLs (the point at the bottom-right of the figure). To further investigate this point and others at the bottom-right of the figure, Table 2 provides the 10 actual URLs that were pointed to by the most intended URLs. In this table, each actual URL is provided along with the number of intended URLs that pointed to it.

This table clearly shows a number of specific Web spam examples; however, it also shows two unexpected URLs:

`http://www.yahoo.com`
and
`http://www.msn.com`.

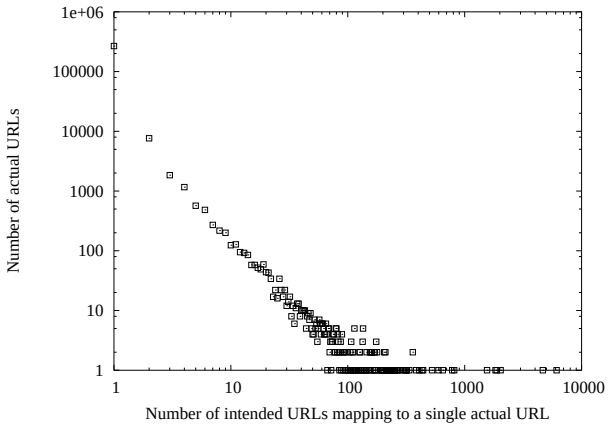


Figure 5: Distribution of the number of intended URLs that point to the same actual URL

Table 2: Ten actual URLs that were pointed to by the most intended URLs

Actual URL	Count
http://bluerocketonline.TechBuyer.com/landerjsp?referrer=&domain=bluerocketonline.com&cm_mmc=	6,108
http://www.yahoo.com	4,663
http://www.msn.com	2,028
http://yoursmartrewards.com/rd_p?p=99302&c=9479-visa300_emc_d22&a=CD579	1,880
http://click.recessionspecials.com/	1,836
http://migada.com/main_index.html	1,553
http://gmncc.com/main_index.html	813
http://click.beyondspecials.com/	783
http://web.yearendsaver.com/	597
http://mail02a.emailcourrier.com/	526

Most people would agree that neither of these URLs are spam so their presence in the corpus was somewhat confusing. To explain this phenomenon, we investigated the intended URLs that pointed to these two actual URLs and found that they were broken redirects. The following two URLs are examples of the types of broken redirects we found:

`http://drs.yahoo.com/quips/*http://coolguy.biz/sm/chair.php`
and
`http://g.msn.com/8HMBEN/?http://www.all-net-offers.com/jf.`

Email spammers used these URLs in their messages to deceive users. At first glance, the URLs appear to be legitimate Yahoo! and MSN URLs, and as a result, unsuspecting users might click on them. Unfortunately, when our crawler tried to access these URLs, the targets of the redirection were no longer available. Consequently, the legitimate Yahoo! and MSN URLs were inserted into our corpus multiple times.

Upon discovering these anomalous pages, we devised a few heuristics to identify additional false positives (i.e., legitimate pages that were mistakenly included in the corpus). Specifically, we used Alexa's Top 500 list [1] and SiteAdvisor's rating system [19] to compile a list of potential false positives. Then, we manually inspected this list and identified the pages that were clearly false positives. Finally, we removed the false positives that we identified. After we applied this cleansing process to the corpus, we found and removed 34,044 legitimate pages as well as their 44,141 corresponding redirect files, leaving the Webb Spam Corpus with a final total of 348,878 Web spam pages and 223,414 redirect files.

The presence of false positives in our original collection raises an interesting research challenge. Since our approach relies upon the URLs found in email spam messages, we believe it is potentially vulnerable to a new breed of attacks called *legitimate URL attacks*. In these attacks, spammers intentionally include legitimate URLs (e.g., `www.yahoo.com`) in their email spam messages to introduce false positives into our Web spam collection process. This potential attack has not affected the Webb Spam Corpus, but it could impact future efforts to collect Web spam using our methodology as well as other automated spam collection efforts (e.g., SpamArchive). Thus, we present this potential threat as an important direction for future research.

3. SAMPLE APPLICATIONS

In this section, we focus on sample applications that will greatly benefit from our corpus and Web spam collection methodology. Our work is broadly applicable to Web spam research, but we focus our attention on two specific applications: (1) automatic classification and filtering of Web spam pages and (2) identifying link-based Web spam.

3.1 Automatic Classification and Filtering of Web Spam Pages

As long as the Web has been in existence, researchers have been developing techniques to automatically categorize its content. Originally, these categorization efforts were concerned with automatically producing Web directories with minimal human interaction. However, due to the emergence of Web spam, researchers have begun focusing their efforts on automatically classifying Web spam pages to separate them from legitimate Web pages. The evolution of these classification efforts is very similar to the evolution of early email spam classification research. Similar to the limitations faced by email spam researchers before the introduction of the Ling-spam [3], PU123A [4, 5], and Enron [15] corpora, current Web spam classification research is limited by the lack of a publicly available corpus. In the next section, we describe several previous Web spam classification research efforts. Then, we discuss how our corpus and Web spam collection process will contribute to this previous work. Finally, we describe how our work will further future research in this area.

3.1.1 Summary of Related Work

Chandrinos et al. [7] were among the first to investigate automatic filtering of Web spam. Specifically, they presented a method for automatically identifying pornographic content on the Web. First, they trained a Naïve Bayesian classifier to distinguish between obscene and non-obscene Web pages. To train the classifier, they used the textual contents of the page as well as two image attributes: whether or not the page contained at least one suspicious image and whether or not the page contained at least one non-suspicious image. Then, they used this classifier to deter-

mine whether or not a given page was obscene. Using a collection of 849 pages (315 legitimate pages and 534 pornographic pages), their classifier was able to obtain 100% obscene precision and 97.5% obscene recall (with zero obscene false positives).

Amitay et al. [2] presented a unique approach for categorizing Web sites. Instead of focusing on the content of the sites, they only utilized connectivity and structural data to categorize sites into eight pre-defined functionality categories (e.g., corporate sites, search engines, portals, etc.). First, they used their connectivity and structural data to identify 16 distinguishing features. Then, using these features, they utilized two automatic learning techniques to categorize the sites: a decision-rule classifier and a Bayesian classifier. Using a data set of 202 manually tagged Web sites, their decision-rule classifier achieved an average expected error of 45.5%, and their Bayesian classifier achieved an average expected error of 41.0%. Also, although it was not the focus of their research, they proposed using their technique to classify Web spam pages.

Fetterly et al. [11] and Drost and Scheffer [10] both used statistical techniques to identify Web spam pages. Fetterly et al. statistically analyzed two large data sets of Web pages (DS1 and DS2) using properties such as linkage structure, page content, and page evolution. They found that many of the outliers in the statistical distribution of these properties were Web spam, and as a result, they advocated the use of outlier detection for identifying similar pages. Drost and Scheffer trained a Support Vector Machine (SVM) classifier to accurately distinguish between guestbook spam, link farms and link exchange sites, and legitimate sites. In their evaluations, they used 854 DMOZ Open Directory pages as their legitimate sample and 431 manually identified Web spam pages (251 examples of guestbook spam and 180 link farm and link exchange pages). The results of their evaluations showed that the SVM classifier can effectively separate spam and legitimate Web pages, consistently obtaining area under the ROC curve (AUC) values greater than 90%.

3.1.2 Benefits of the Webb Spam Corpus

All of this previous research is novel; however, it suffers from two main limitations. First, all of the data sets used in these evaluations are far too small to be considered representative samples of Web spam. These data sets are small because Web spam researchers have been forced to manually identify and tag Web spam examples. This manual labeling process is extremely time-consuming, and as a result, it has forced previous researchers to apply their techniques on limited samples of their Web data. Second, none of the previously cited Web spam data sets have been released into the public domain. Thus, other researchers are currently unable to verify the validity of the claims made by this previous research. In their paper, Drost and Scheffer claim to have a publicly available spam data set; however, the paper does not provide a link, and we have been unable to locate it online.

Our Web spam collection technique and corresponding corpus help solve both of the limitations found in previous research. The Webb Spam Corpus is a very large sample of Web spam (over two orders of magnitude larger than previously cited Web spam data sets). Also, our automated Web spam collection technique allows researchers to quickly and easily obtain even more examples. The main challenge with any automated Web spam classification technique is accurate labeling (as shown by the limited Web spam sample sizes of previous research), and although our approach does not completely eliminate this problem, it does minimize the manual effort required. Researchers simply need to identify a few false positives as opposed to the arduous task of manually searching for a sufficiently large collection of Web spam pages. Additionally, the Webb Spam Corpus is publicly available so researchers can easily publish reproducible results.

3.1.3 New Research Opportunities

In addition to the benefits our approach and corpus offer previous research efforts, we believe this work opens the door for a number of new research opportunities. First, automatic Web spam classification could greatly benefit Web filtering efforts, similar to the way email spam classification has improved email filtering. Specifically, our work could be used to provide more effective parental controls on the Web. The Webb Spam Corpus contains a number of porn-related pages as well as additional content that is not suitable for children. This content provides valuable insight into the characteristics of Web spam pages and allows researchers to build more effective Web content filters.

In addition to its contributions to Web filtering, the Webb Spam Corpus also provides a unique approach to email spam filtering. In another paper [16], we show how Web content can be used to improve the effectiveness of email spam filtering techniques. Specifically, we leverage the Web content that corresponds to the URLs found in email messages to enhance email classification decisions. An email message that points to legitimate Web pages is more likely to be legitimate than an email message that points to suspicious Web pages. By augmenting traditional text-based spam filters with contextual information such as the spamicity of the URLs found within an email message, we can create more sophisticated classification systems. Thus, we can utilize the link between email and the Web to help eliminate spam in both domains.

3.2 Identifying Link-Based Web Spam

One of the most prominent examples of Web spam is the targeted manipulation of search engines to increase the visibility of certain Web spam pages. Since the vast majority of Web users use search engines to access the Web [8, 18], spammers can artificially increase the value of a spam page by effectively deceiving a search engine's core ranking algorithms. Most modern search engines employ link-based ranking algorithms (e.g., Google's PageRank) that evaluate the quality of a Web page based on the number and quality of the other Web pages that point to it. These algorithms rely on a fundamental assumption that a link from one page to another is an authentic conferral of authority by the pointing page to the target page.

Link-based Web spam directly attacks the credibility of such link-based ranking algorithms by inflating the perceived quality of the spammer's target page. The simplest example of link-based Web spam is called a *link exchange* – a scenario in which two or more Web spammers collude to link to each other's pages. A more sophisticated example is the construction of large *link farms* of spammer-controlled Web pages that exist solely to link to a spammer's target page. These link farms make it appear to the link-based

ranking algorithms that the target page is receiving many votes of confidence, and as a result, the target page receives an undeserved boost in its ranking. In contrast to the brute force approach of a link farm (where there are a large number of links from low-quality, spammer-controlled pages), spammers also engage in techniques to acquire links from higher-quality Web pages. For example, a Web spammer can create a seemingly legitimate Web site (called a *honey pot*) to attract links from unsuspecting legitimate Web sites. The honey pot can then pass along its accumulated quality to the target spam page. Spammers can also *hijack* links from legitimate Web sites by inserting links into weblog comments, wikis, Web-based message boards, as well as submitting spam links to legitimate Web directories. From the link-based ranking algorithm's perspective, these hijacked legitimate pages are endorsing the spam page, and as a result, the spam page receives an undeserved ranking boost. Of course, Web spammers may choose to combine these basic link-based Web spam patterns in a number of ways to construct more complicated and less easily detected linking arrangements. For a more detailed discussion of these spamming techniques, please consult [12].

The majority of previous link-based Web spam research has focused on its identification and removal; however, we believe this research has been limited by the absence of a publicly available Web spam corpus. In the next section, we summarize several of these previous efforts and explain their limitations. Then, we explain how our Web spam collection technique and corpus will enhance this previous work and help facilitate future work on the identification of link-based Web spam.

3.2.1 Summary of Related Work

Davison [9] was the first to investigate link-based Web spam, and he studied the identification of *nepotistic links* – "links between pages that are present for reasons other than merit." Specifically, he created decision trees to determine whether or not a given link was nepotistic. His experiments relied on two data sets – 1,536 links that were arbitrarily selected and 750 links that were sampled from a DiscoWeb search engine crawl. This work was extremely valuable because it was the first to use automated learning methods to identify link-based Web spam. However, it also identified two corpora-related problems with Web spam research. First, as the author admits, the two data sets were too small to be considered representative samples. Second, the results obtained with each data set were noticeably different, highlighting the need for a publicly available Web spam corpus to help benchmark research results.

These corpora-related limitations are also evident in other previous research. For example, the TrustRank algorithm, proposed by Gyöngyi et al. [13], uses a variation of the traditional PageRank algorithm to propagate trust from a seed set of pre-trusted Web pages to the pages that are pointed to by those pre-trusted pages. Intuitively, pages that have many incoming-links from trusted pages will also be trusted. As long as spam pages are relatively distant (in terms of link distance) from trusted pages, the algorithm can yield more spam resilient rankings than PageRank. Unfortunately, although the TrustRank experimental validation had access to a data set of 31 million Web sites that were collected by AltaVista, the actual experiments only used an evaluation sample of 748 manually identified pages (of which, 135 were labeled as spam). The limited size of this evaluation size is another illustration of the difficulty involved with manually identifying Web spam examples.

Wu and Davison [21] proposed a technique that propagates distrust to bad pages. First, their algorithm searches for pages with common nodes in their incoming and outgoing links sets. If the number of common nodes for a given page is above a given threshold (3, in their experiments), that page is marked as bad and placed in a seed set. Then, every page that points to more than a threshold number (3, in their experiments) of pages in the seed set is also marked as bad. Finally, every link involving one of the bad pages is considered spam. In their experiments, Wu and Davison had access to a 20 million page data set that they received from the search.ch search engine; however, their evaluation sample only contained 732 spam sites (due to the manual labeling problem).

Similarly, Benczur et al. [6] proposed the SpamRank algorithm to identify pages with a large amount of undeserved PageRank (i.e., PageRank that was derived from spam pages). First, they identified the major PageRank contributing pages (the "supporters") for every page in their data set. They then penalized pages that had statistically anomalous supporters (in terms of the distribution of their PageRank scores). By incorporating these penalties into the revised PageRank calculation, pages with a large amount of undeserved PageRank were identified and given much lower PageRank values (and correspondingly lower rankings). The authors report experimental results over an evaluation sample containing 910 pages (of which 16.5% were spam) that were taken from a 31 million page data set.

3.2.2 Benefits of the Webb Spam Corpus

Given the interesting research results so far, we believe that rich, new opportunities exist for cross-validating previous results, enhancing the previously proposed link-based Web spam algorithms, and developing more refined algorithms for identifying link-based Web spam. Our new method for obtaining Web spam examples (and the Webb Spam Corpus itself) immediately benefits this previous research because it greatly increases the coverage of Web spam pages. Our corpus already contains over two orders of magnitude more Web spam examples than previous data sets (almost 350,000 pages versus less than 2,000 in each cited case), and unlike those previous data sets, our corpus is publicly available.

With our corpus, researchers can easily benchmark their techniques using a single, publicly available corpus – a luxury currently missing from Web spam research. Additionally, our collection methodology gives researchers a simple technique for automatically obtaining new Web spam examples in the future, a particularly important feature for maintaining the freshness of spam samples in the face of a dynamic and evolving group of spam adversaries. This automatic technique should greatly reduce the workload of Web spam researchers, minimizing the manual Web spam tagging process and allowing them to focus their efforts on developing new solutions.

3.2.3 New Research Opportunities

In addition to providing a standardized corpus for evaluating link-based Web spam identification algorithms, we believe that a careful study of the linking features of the Webb

Spam Corpus may yield new insights into how spammers construct complex linking arrangements and provide new avenues for developing more robust link-based Web spam identification algorithms.

As a first step towards developing new algorithms, we propose the following hypothesis: many of the Web pages corresponding to URLs found in spam messages are also target pages in link-based Web spam. This hypothesis is driven by the observation that email spammers and Web spammers share the same motivations. In email spam, spammers want to promote certain pages so that they receive traffic and ultimately monetary rewards (or private information, in the case of phishers). Similarly, in Web spam, spammers want to promote certain pages for the exact same reasons.

To help investigate this hypothesis, we constructed a host-based connectivity graph to determine the interconnectivity within the Webb Spam Corpus. Initially, we treated each host that appears in the Webb Spam Corpus as a node in a Web graph. Then, we parsed each page in the corpus to obtain the URLs found in its HTML content, only retaining the URLs that pointed to another page within the corpus. Each of these URLs represents a link from one page in the corpus to another. After we had all of these page-level links, we converted them to host-level links (based on the host names in each URL) to make the number of nodes in the Web graph more manageable. Finally, we constructed a host-based connectivity graph.

The host-based connectivity graph contains 70,230 unique hosts and 137,039 unique links (not including self-links). Thus, by simply investigating the hosts within the Webb Spam Corpus, we have already identified a great deal of interlinkage. We have been forced to omit the complete host-based connectivity graph from the paper because it looks like a giant circle of ink (due to the vast number of nodes and interconnections). Instead, we have provided an extremely condensed version of the connectivity graph in Figure 6.

The graph shown in Figure 6 was constructed using the 15 hosts with the most outgoing links (i.e., they linked to the largest number of hosts in the corpus). The figure shows each of the hosts along with their 1-hop (hosts they directly link to) and 2-hop (hosts their direct neighbors link to) neighbors. Even this simple graph, which contains 948 unique hosts and 3,330 unique links, clearly illustrates the interconnectivity of the Web spam hosts within the corpus. We believe this interconnectivity provides preliminary support for our hypothesis, and as a result, we intend to thoroughly investigate this topic in future research.

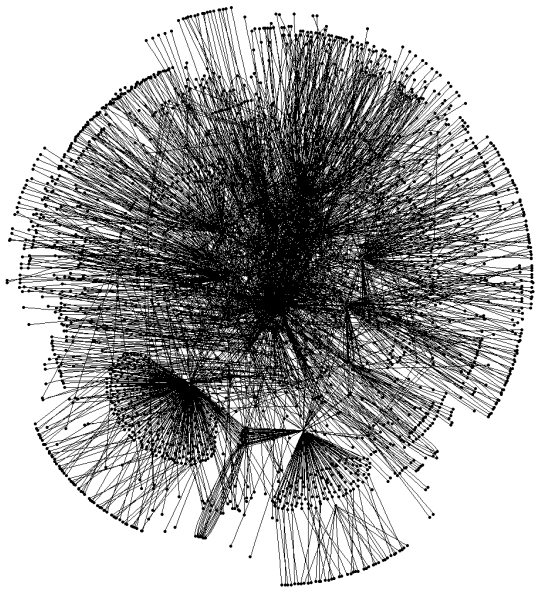


Figure 6: Host-based connectivity graph

4. CONCLUSIONS AND FUTURE WORK

As the problems posed by Web spam continue to grow in severity, it is imperative that the research community follow the best practices that have already been established in similar domains (e.g., email spam research). Of these best practices, one of the most important is the use of large, publicly available corpora. In this paper, we have taken the initial step towards applying these best practices to the Web spam domain. We have provided a novel method for automatically obtaining Web spam pages, and we have also presented the Webb Spam Corpus – a publicly available corpus of almost 350,000 Web spam pages that were obtained using our automated method.

The Webb Spam Corpus is the first public data set of its kind, and it is more than two orders of magnitude larger than previously cited Web spam data sets. In all research domains, the lack of publicly available corpora severely impedes research progress; thus, by presenting our approach to Web spam collection and the Webb Spam Corpus, we hope to fuel significant research efforts.

In the future, we plan to investigate techniques to make our collection process more robust against legitimate URL attacks. As this problem applies to all spam-related corpora, we believe it is an interesting area of research, and we encourage others to follow suit. Additionally, we are currently investigating the statistical properties of the Webb Spam Corpus, identifying characteristics that uniquely distinguish its pages from legitimate Web pages.

5. ACKNOWLEDGMENTS

This work was partially supported by NSF under the ITR (grants CCR-0121643 and CCR-0219902) and CyberTrust/DAS (grant IIS-0242397) programs, an IBM SUR grant, and Hewlett-Packard.

6. REFERENCES

[1] Alexa. Alexa web search – top 500. `http://www.alexa.com/site/ds/top_sites?ts_mode=global&lang=none`, 2006.

[2] E. Amitay et al. The connectivity sonar: Detecting site functionality by structural patterns. In *Proceedings of the 14th ACM Conference on Hypertext and Hypermedia (HYPERTEXT '03)*, pages 38–47, 2003.

[3] I. Androutsopoulos et al. An evaluation of naive bayesian anti-spam filtering. In *Proceedings of the Workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning*, pages 9–17, 2000.

[4] I. Androutsopoulos et al. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with encrypted personal e-mail messages. In

Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 160–167, 2000.

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

[6] A. A. Benczur, K. Csalogany, T. Sarlos, and M. Uher. Spamrank - fully automatic link spam detection. In *Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb '05)*, 2005.

[7] K. V. Chandrinos et al. Automatic web rating: Filtering obscene content on the web. In *Proceedings of the 4th European Conference on Research and Advanced Technology for Digital Libraries (ECDL '00)*, pages 403–406, 2000.

[8] J. Cho and S. Roy. Impact of web search engines on page popularity. In *Proceedings of the 13th International World Wide Web Conference (WWW '04)*, 2004.

[9] B. D. Davison. Recognizing nepotistic links on the web. In *Proceedings of the AAAI-2000 Workshop on Artificial Intelligence for Web Search*, 2000.

[10] I. Drost and T. Scheffer. Thwarting the nigritude ultramarine: Learning to identify link spam. In *Proceedings of the 16th European Conference on Machine Learning (ECML '05)*, 2005.

[11] D. Fetterly, M. Manasse, and M. Najork. 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 (WebDB '04)*, pages 1–6, 2004.

[12] Z. Gyöngyi and H. Garcia-Molina. Web spam taxonomy. In *Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web (AIRWeb '05)*, 2005.

[13] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. In *Proceedings of the 30th International Conference on Very Large Databases (VLDB '04)*, 2004.

[14] M. R. Henzinger, R. Motwani, and C. Silverstein. Challenges in web search engines. *SIGIR Forum*, 36(2):11–22, 2002.

[15] B. Klimt and Y. Yang. Introducing the enron corpus. In *Proceedings of the 1st Conference on Email and Anti-Spam (CEAS '04)*, 2004.

[16] C. Pu et al. Towards the integration of diverse spam filtering techniques. 2006. Proceedings of the IEEE International Conference on Granular Computing (GrC '06).

[17] C. Pu and S. Webb. Observed trends in spam construction techniques: A case study of spam evolution. 2006. Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS '06).

[18] F. Qiu, Z. Liu, and J. Cho. Analysis of user web traffic with a focus on search activities. In *Proceedings of the 8th International Workshop on the Web and Databases (WebDB '05)*, 2005.

[19] SiteAdvisor. Protection from spyware, spam, viruses and online scams — siteadvisor. `http://www.siteadvisor.com/`, 2006.

[20] SpamArchive. Spamarchive.org - donate your spam to science. `http://www.spamarchive.org/`, 2006.

[21] B. Wu and B. D. Davison. Identifying link farm spam pages. In *Proceedings of the 14th International World Wide Web Conference (WWW '05)*, 2005.