SIGIR 2006 Workshop on

Adversarial Information Retrieval on the Web

AIRWeb 2006

Proceedings of the Second International Workshop on Adversarial Information Retrieval on the Web – AIRWeb 2006

29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 10 August 2006, Seattle, Washington, USA.

Edited by

Brian D. Davison

Department of Computer Science and Engineering, Lehigh University

Marc Najork

Microsoft Research

Tim Converse

Yahoo! Search

Technical Report LU-CSE-06-027, Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA, 18015 USA.

AIRWeb 2006 Program

The Second International Workshop on Adversarial Information Retrieval on the Web
10 August 2006

9:00	Welcome
9:10	Link-Based Characterization and Detection of Web Spam *Luca Becchetti, Carlos Castillo, Debora Donato, Stefano Leonardi, Università di Roma "La Sapienza", and Ricardo Baeza-Yates, Yahoo! Research Barcelona*
9:40	Tracking Web Spam with Hidden Style Similarity *Tanguy Urvoy, Thomas Lavergne and Pascal Filoche, France Telecom R&D*
10:10	Web Spam Detection with Anti-Trust Rank *Vijay Krishnan and Rashmi Raj, Stanford University*
10:30	Break
11:00	Invited Talk on Sponsored Search *Jan Pedersen, Yahoo!*
12:30	Lunch
1:30	Adversarial Information Retrieval Aspects of Sponsored Search *Bernard J. Jansen, Pennsylvania State University*
1:50	Link-Based Similarity Search to Fight Web Spam *András A. Benczúr, Károly Csalogány and Tamás Sarlós, Hungarian Academy of Sciences and Eötvös University*
2:20	Improving Cloaking Detection using Search Query Popularity and Monetizability *Kumar Chellapilla and David Maxwell Chickering, Microsoft Live Labs*
3:00	Break
3:30	Expert Panel on Blog Spam *Tim Converse, Yahoo! Search, Dennis Fetterly, Microsoft Research, Natalie Glance, Nielsen BuzzMetrics, Jeremy Hylton, Google, Greg Linden, Findory, and Paul Querna, Ask*
5:00	Discussion and Final remarks

Contents

Welcome

Dear Participant:

Welcome to the Second International Workshop on Adversarial Information Retrieval on the Web (AIRWeb). This workshop brings together researchers and practitioners that are concerned with the on-going efforts in adversarial information retrieval on the Web, and builds on last year's successful meeting in Chiba, Japan as part of WWW2005.

Out of the thirteen submissions to this year's workshop, we have a total of six peer-reviewed papers to be presented – four research presentations and two synopses of work in progress, conveying the latest results in adversarial web IR. In addition, we are pleased to have an invited talk on sponsored search by Jan Pedersen of Yahoo! and a panel session with experts on blog spam, including: Dennis Fetterly (Microsoft Research), Natalie Glance (Nielsen BuzzMetrics), Jeremy Hylton (Google), Greg Linden (Findory), Andrew Tomkins (Yahoo! Research) and a representative to be determined from Ask.com. Workshop participants are encouraged to raise additional questions of interest to industry experts and researchers.

We extend our thanks to the authors and presenters, to the expert panelists and invited speaker, and to the members of the program committee for their contributions to the material that forms an outstanding workshop. It is our hope that you will find the presented work to be stimulating and that you will ask questions, contribute ideas, and perhaps get involved in future work in this area.

Brian D. Davison, Marc Najork, and Tim Converse
AIRWeb 2006 Program Chairs

Overview

The attraction of hundreds of millions of web searches per day provides significant incentive for many content providers to do whatever is necessary to rank highly in search engine results, while search engine providers want to provide the most accurate results. The conflicting goals of search and content providers are adversarial, and the use of techniques that push rankings higher than they belong is often called search engine spam. Such methods typically include textual as well as link-based techniques, or their combination.

AIRWeb 2006 provides a focused venue for both mature and early-stage work in web-based adversarial IR. The workshop solicited technical papers on any aspect of adversarial information retrieval on the Web. Particular areas of interest included, but were not limited to:

- search engine spam and optimization,
- crawling the web without detection,
- link-bombing (a.k.a. Google-bombing),
- comment spam, referrer spam,
- blog spam (splogs),
- malicious tagging,
- reverse engineering of ranking algorithms,
- advertisement blocking, and
- web content filtering.

Papers addressing higher-level concerns (e.g., whether 'open' algorithms can succeed in an adversarial environment, whether permanent solutions are possible, etc.) were also welcome.

Authors were invited to submit papers and synopses in PDF format. We encouraged submissions presenting novel ideas and work in progress, as well as more mature work. Submissions were reviewed by a program committee of search experts on relevance, significance, originality, clarity, and technical merit and accepted papers cover state-of-the-art research advances to address current problems in web spam.

Workshop Organizers

Program Chairs

Brian D. Davison, *Lehigh University*
Marc Najork, *Microsoft Research*
Tim Converse, *Yahoo! Search*

Program Committee Members

Sibel Adali, *Rensselaer Polytechnic Institute, USA*
Lada Adamic, *University of Michigan, USA*
Einat Amitay, *IBM Research Haifa, Israel*
Andrei Broder, *Yahoo! Research, USA*
Carlos Castillo, *Università di Roma "La Sapienza", Italy*
Abdur Chowdhury, *AOL Search, USA*
Nick Craswell, *Microsoft Research Cambridge, UK*
Matt Cutts, *Google, USA*
Dennis Fetterly, *Microsoft Research, USA*
Zoltan Gyongyi, *Stanford University, USA*
Matthew Hurst, *Nielsen BuzzMetrics, USA*
Mark Manasse, *Microsoft Research, USA*
Jan Pedersen, *Yahoo!, USA*
Bernhard Seefeld, *Switzerland*
Erik Selberg, *Microsoft Search, USA*
Bruce Smith, *Yahoo! Search, USA*
Andrew Tomkins, *Yahoo! Research, USA*
Tao Yang, *Ask.com/Univ. of California-Santa Barbara, USA*

Contributing Authors, Speakers, and Panelists

Ricardo Baeza-Yates, *Yahoo! Research Barcelona*

Luca Becchetti, *Università di Roma "La Sapienza"*

András A. Benczúr, *Hungarian Academy of Sciences and Eötvös University*

Carlos Castillo, *Università di Roma "La Sapienza"*

Kumar Chellapilla, *Microsoft Live Labs*

David Maxwell Chickering, *Microsoft Live Labs*

Tim Converse, *Yahoo! Search*

Károly Csalogány, *Hungarian Academy of Sciences and Eötvös University*

Debora Donato, *Università di Roma "La Sapienza"*

Dennis Fetterly, *Microsoft Research*

Pascal Filoche, *France Telecom R&D*

Natalie Glance, *Nielsen BuzzMetrics*

Jeremy Hylton, *Google*

Bernard J. Jansen, *Pennsylvania State University*

Vijay Krishnan, *Stanford University*

Thomas Lavergne, *France Telecom R&D*

Stefano Leonardi, *Università di Roma "La Sapienza"*

Greg Linden, *Findory*

Jan Pedersen, *Yahoo!*

Rashmi Raj, *Stanford University*

Tamás Sarlós, *Hungarian Academy of Sciences and Eötvös University*

Tanguy Urvoy, *France Telecom R&D*

Link-Based Characterization and Detection of Web Spam*

Luca Becchetti[1] becchett@dis.uniroma1.it
Carlos Castillo[1] castillo@dis.uniroma1.it
Debora Donato[1] donato@dis.uniroma1.it
Stefano Leonardi[1] leon@dis.uniroma1.it
Ricardo Baeza-Yates[2] ricardo@baeza.cl

[1] DIS - Università di Roma "La Sapienza" Rome, Italy

[2] Yahoo! Research Barcelona, Spain & Santiago, Chile

ABSTRACT

We perform a statistical analysis of a large collection of Web pages, focusing on spam detection. We study several metrics such as degree correlations, number of neighbors, rank propagation through links, TrustRank and others to build several automatic web spam classifiers. This paper presents a study of the performance of each of these classifiers alone, as well as their combined performance. Using this approach we are able to detect 80.4% of the Web spam in our sample, with only 1.1% of false positives.

1. INTRODUCTION

The term "**spam**" has been commonly used in the Internet era to refer to *unsolicited (and possibly commercial) bulk messages*. The most common form of electronic spam is **e-mail spam**, but in practice each new communication medium has created a new opportunity for sending unsolicited messages. There are many types of electronic spam nowadays including spam by instant messaging (*spim*), spam by internet telephony (*spit*), spam by mobile phone, by fax, etc. The Web is not absent from this list.

The request-response paradigm of the HTTP protocol makes it impossible for spammers to actually "send" pages directly to the users, so the type of spam that is done on the Web takes a somewhat different form than in other media. What spammers do on the Web is to try to deceive search engines, a technique known as **spamdexing**.

1.1 Web spam

The Web contains numerous profit-seeking ventures that are attracted by the prospect of reaching millions of users at a very low cost. A large fraction of the visits to a Web site originate from search engines, and most of the users click on the first few results in a search engine. Therefore, there is an economic incentive for manipulating search engine's listings by creating pages that score high independently of their real merit. In practice such manipulation is widespread, and in many cases, successful. For instance, the authors of [9] report that "among the top 20 URLs in our 100 million page PageRank calculation (...) 11 were pornographic, and these high positions appear to have all been achieved using the same form of link manipulation".

One suitable way to define Web spam is any attempt to get "an unjustifiably favorable relevance or importance score for some web page, considering the page's true value" [17]. There is a large gray area between "ethical" Search Engine Optimization (SEO) and "unethical" spam. SEO services range from ensuring that Web pages are indexable by Web crawlers, to the creation of thousands or millions of fake pages aimed at deceiving search engine ranking algorithms. Our main criteria to decide in borderline cases is the perceived effort spent by Web authors on providing good content, versus the effort spent on trying to score high in search engines.

In all cases, the relationship between a Web site administrator trying to rank high on a search engine and the search engine administrator is an **adversarial** relationship in a zero-sum game. Every undeserved gain in ranking by the web site is a loss of precision for the search engine. Fortunately, from the point of view of the search engine, "victory does not require perfection, just a rate of detection that alters the economic balance for a would-be spammer" [21].

There are other forms of Web spam that involve search engines. We point out that we do not consider advertising spam, which is also an issue for search engines that involves clicks and ads.

1.2 Topological spam (link spam)

A **spam page or host** is a page or host that is used for spamming or receives a substantial amount of its score from other spam pages. There are many techniques for Web spam [17], and they can be broadly classified into content (or keyword) spam and link spam.

Content spam includes changes in the content of the pages, for instance by inserting a large number of keywords [6, 8]. In [21], it is shown that 82-86% of spam pages of this type can be detected by an automatic classifier. The features used for the classification include, among others: the number of words in the text of the page, the number of hyperlinks, the number of words in the title of the pages, the compressibility (redundancy) of the content, etc.

Unfortunately, it is not always possible to detect spam by content analysis, as some spam pages only differ from nor-

*Supported by the EU Integrated Project AEOLUS (FET-15964).

AIRWEB'06, August 10, 2006, Seattle, Washington, USA.

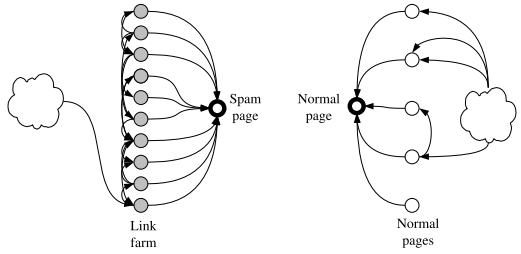


Figure 1: Schematic depiction of the neighborhood of a page participating in a link farm (left) and a normal page (right).

mal pages because of their links, not because of their contents. Many of these pages are used to create **link farms**. A link farm is a densely connected set of pages, created explicitly with the purpose of deceiving a link-based ranking algorithm. Zhang et. al [26] call this **collusion**, and define it as the "manipulation of the link structure by a group of users with the intent of improving the rating of one or more users in the group".

A page that participates in a link farm, such as the one depicted in Figure 1, may have a high in-degree, but little relationship with the rest of the graph. Heuristically, we call spamming achieved by using link farms **topological spamming**. In particular, a topological spammer achieves its goal by means of a link farm that has topological and spectral properties that statistically differ from those exhibited by non spam pages. This definition embraces the cases considered in [13], and their method based on "shingles" can be also applied in detecting some types of link farms (those that are dense graphs).

Link-based and content-based analysis offer two orthogonal approaches. We think that these approaches are not alternative and should probably be used together.

On one hand, in fact, link-based analysis does not capture all possible cases of spamming, since some spam pages appear to have spectral and topological properties that are statistically close to those exhibited by non spam pages. In this case, content-based analysis can prove extremely useful.

On the other hand, content-based analysis seems less resilient to changes in spammers strategies, in much the same way that content-based techniques for detecting email spamming are. For instance, a spammer could copy an entire Web site (creating a set of pages that may be able to pass all tests for content spam detection) and change a few out-links in every page to point to the target page. This may be a relatively inexpensive task to perform in an automatic way, whereas creating, maintaining, reorganizing a link farm, possibly spanning more than one domain, is economically more expensive.

1.3 Our contribution

In [3] we used Truncated PageRank (studied in section 3.4) and probabilistic estimation of the number of neighbors (studied in section 3.5) to build an automatic classifier for link spam using several link-based features. In this paper, we are more focused on investigating **which** (combinations of) features are good for spam detection, and we try to build classifiers that can achieve high precision by using **a small set of features**.

Table 1: Summary of the performance of the different metrics, the ranges in the error rate correspond to a simple classifier with a few rules, and to a more complex (but more precise) classifier.

Section	Metrics	Detection rate	False positives
3.1	Degree (D)	73-74%	2-3%
3.2	D + PageRank (P)	74-77%	2-3%
3.3	D + P + TrustRank	77%	2-3%
3.4	D + P + Trunc. PageRank	77-78%	2%
3.5	D + P + Est. Supporters	78-79%	1-2%
3.6	All attributes	80-81%	1-3%

We are also including several metrics that we have not considered before for this type of classifier: we test in our collection TrustRank [18], and we propose the use of degree-degree correlations, edge-reciprocity and host-based counts of neighbors. The performance of the different classifiers we build in this paper is summarized in Table 1.

The results obtained using all the selected attributes are comparable to those achieved by state-of the art content analysis for Web spam detection [21]. Again, we recall that content-based analysis is orthogonal to the approach we consider, and it is likely that the combination of these techniques might prove effective.

The next section introduces the algorithmic framework and the data set we used. Section 3 presents the different metrics. The last section presents our conclusions.

2. FRAMEWORK

This section describes the type of algorithms we are interested in, and the data set we are using to evaluate the effectiveness of the different metrics for spam detection.

2.1 Web graph algorithms

We view our set of Web pages as a **Web graph**, that is, a graph $G = (V, E)$ in which the set V corresponds to Web pages belonging to a subset of the Web, and every link $(x, y) \in E$ corresponds to a hyperlink from page x to page y in the collection. For concreteness, the total number of nodes $N = |V|$ is in the order of 10^{10} [15], and the typical number of links per Web page is between 20 and 30.

Given the large/huge size of typical data sets used in Web Information Retrieval, complexity issues play a crucial role. These impose severe restrictions on the computational and/or space complexity of viable algorithmic solutions. A first approach to modeling these restrictions may be the **streaming model** of computation [19]. However, the restrictions of the classical stream model are too severe and hardly compatible with the problems we are interested in.

In view of the above remarks, we decided to restrict to algorithmic solutions whose space and time complexity is compatible with the **semi-streaming** model of computation [10, 7]. This implies a semi-external memory constraint [24] and thus reflects many significant constraints arising in practice. In this model, the graph is stored on disk as an adjacency list and no random access is possible, i.e., we only allow sequential access.

In particular, we assume that we have $O(N \log N)$ bits of main (random access) memory, i.e., in general there is

enough memory to store some limited amount of data about each vertex, but not to store the links of the graph in main memory. We impose a futher constraint, i.e., the algorithm should perform a small number of passes over the stream data, at most $O(\log N)$.

We assume no previous knowledge about the graph, so we do not know *a priori* if a particular node is suspicious of being a spam or not. For this reason, there are some semi-streamed algorithms on a Web graph that we cannot use for Web spam detection in our framework. If we have to compute a metric which assigns a value to every vertex, e.g. a score, we cannot of course afford to run this algorithm again for every node in the graph, due to the large size of the data set.

As an example, suppose we want to measure the centrality of nodes. If we use the streamed version of the standard breadth-first search (BFS) algorithm, we are not complying with this requirement, since the outcome would be a BFS tree for a specific node, which is not enough for computing the centrality of all the nodes in the graph. Conversely, an algorithm such as PageRank computes a score for all nodes in the graph at the same time.

The general sketch of the type of semi-streamed graph algorithms we are interested, is shown in Figure 2.

Require: graph $G = (V, E)$, score vector S
1: `INITIALIZE`(S)
2: **while not** `CONVERGED` **do**
3: **for** $src : 1 \dots |V|$ **do**
4: **for all** links from src to $dest$ **do**
5: `COMPUTE`($S, src, dest$)
6: **end for**
7: **end for**
8: `POST_PROCESS`(S)
9: **end while**
10: **return** S

Figure 2: Generic link-analysis algorithm using a stream model. The score vector S represents any metric, and it must use $O(N \log N)$ bits. The number of iterations should be $O(\log N)$ in the worst case.

2.2 Data set

We use a set of pages from the `.uk` domain, downloaded in 2002 by the *Dipartimento di Scienze dell'Informazione*, *Università degli studi di Milano*. These collections are publicly available at `http://law.dsi.unimi.it/`.

The collection has 18.5 million pages located on 98,452 different hosts. Due to the large size of this collection, we decided to classify entire hosts instead of individual pages. This increases the coverage of the sample, but introduces errors as there are some hosts that consist of a mixture of spam pages and legitimate contents.

We manually classified a sample of 5,750 hosts (5.9% of the hosts). For every host, we inspected a few pages manually and looked at the list of pages collected by the crawler. Whenever we found a link farm inside the host, we classified the entire host as spam.

As the amount of spam compared to normal hosts is relatively small, and since we want to focus on the most "damaging" types of spam, we biased our sampling towards hosts with high PageRank. This is the same approach taken by other researchers in Web spam detection [4, 18]. In order to do this, our sample includes the top 200 hosts with the higher PageRank in their home page, with the higher overall PageRank and with the larger number of pages. Other hosts were added by classifying all the top 200 pages by hostname length, as several spammers tend to create long names such as "`www.buy-a-used-car-today.example`". For the same reason, we searched for typical spamming terms in the host names, and we classified all the hosts with domain names including keywords such as `mp3`, `mortgage`, `sex`, `casino`, `buy`, `free`, `cheap`, etc.

We discarded from the sample the hosts that no longer were available (about 7%), and classified the rest in one of the following three classes:

Spam (16%): The host is clearly a link farm; or it is spamming by using several keywords in the host, directory or file names; or it includes no content apart from links to a target page or host.

Normal (81%): The host is clearly not a link farm, but a normal site (in the jargon of e-mail spam detection, non-spam items are sometimes called "ham").

Suspicious (3%): Borderline cases, including illegal business (on-line gambling, pharmaceuticals without prescription) and pornography, as they are usual customers of link farms. We also included in this category sites that almost (but not entirely) provide content copied from other sources plus advertising, affiliate networks, advertising servers, and groups of entire sites that share the same template with little added information.

Table 2 shows the number of hosts and pages in each class. Note that as the sample is biased toward spam pages, it cannot be used to infer the overall prevalence of Web spam in this collection. Also, given that we biased our sampling towards hosts with a large number of pages, our sample has only 5.9% of the hosts but covers about 5.8 million pages or 30% of the pages in the collection.

Table 2: Relative sizes of the classes in the manually-classified sample. The last row gives the fraction of classified hosts and pages over the entire collection.

Class	Hosts		Pages	
Spam	840	16%	329 K	6%
Normal	4,333	81%	5,429 K	92%
Suspicious	171	3%	118 K	2%
Total	5,344	(5.8%)	5,877 K	(31.7%)

For the class labels provided to the algorithms in the automatic classification experiments, we adopted a conservative approach and included the suspicious hosts in the normal class.

One final remark about the data set is in order. The Web is a moving target and no spam research paper can have a spam classification whose Web content and structure (links) date to the same time as when the training set classification was done. This can negatively affect the results returned by any classifier for two main reasons:
- A site may not be spam today but it may have been spam in the past. In this case there is the risk of wrong detection of this site as spam and hence the number of false positives will increase.

- A site may be spam today but may not have been in the past. In this case we may not detect the site as spam and hence the number of false negatives will increase.

So, regardless of the technique used, our results may underestimate the false positives and negatives (so in both cases these are lower bounds). This implies that the detection rate we are giving is an upper bound.

2.3 Automatic classification

This paper describes several link-based features that are used to build automatic classifiers. We used the `Weka` [25] implementation of decision trees: binary trees in which each internal node is an inequality (for instance: "if feature A is less than 10, and feature B is greater than 0.2, then the host is spam"). Describing here the algorithm for building automatically these classifiers is not possible due to space limitations, for a description see [25].

The evaluation of the classifiers was performed by a ten-fold cross-validation of the training data. The data is first divided into 10 approximately equal partitions, then each part is held out in turn for testing, and the classifier is trained using the remaining 9 folds. The overall error estimate is the average of the 10 error estimates on the test folds.

We also used boosting [12], which builds 10 different classifiers, assigning a different weight to each element after each classification, depending on whether the element was correctly classified or not. The resulting classifier is a linear combination of the individual weighted classifiers.

For each set of features we build two classifiers. We first limit the number of rules, by imposing a lower bound on the number of hosts in each leaf of the decision tree (this is the parameter M in the implementation of `Weka`). In our case, $M = 30$ hosts, roughly 5% of them. We then build another classifier by using no pruning and generating as many rules as possible as long as there are at least $M = 2$ hosts per leaf.

Evaluation: the error metrics for the evaluation are based on precision and recall [2] for the spam detection task. The main measures we use are:

$$\text{Detection rate} = \frac{\text{\# of spam sites classified as spam}}{\text{\# of spam sites}}$$

$$\text{False positives} = \frac{\text{\# of normal sites classified as spam}}{\text{\# of normal sites}} .$$

The full list of features we used is provided in the appendix. Note that neither the length of the host names, nor their keywords, nor the number of pages were included as features for the automatic classifiers.

3. METRICS

Fetterly et al. [11] hypothesized that studying the distribution of statistics about pages could be a good way of detecting spam pages, as "in a number of these distributions, outlier values are associated with web spam". In this section we consider several link-based metrics, whose computation uses algorithms that are feasible for large-scale Web collections. These are not all possible statistics that can be computed, for a survey of Web metrics, see [5].

From pages to hosts. All the metrics in this section are metrics about individual Web pages. To apply them to sites we measured them for the home page of the site (this is the page in the root directory, or the page with the shortest file name on each host). We computed these metrics for the page with the maximum PageRank. In our sample, these pages are the same in only 38% of the cases, so it is rather common that the highest ranked page on a site is not the home page.

Actually, in 31% of the normal hosts these pages are the same, while for the spam hosts in 77% of the cases the pages are the same. In a normal Web site, the pattern of the linking to the pages in the host are not controlled by its owner, so even if the home page is more "visible", any page has a certain chance of becoming popular. In the case of a spam host, we are assuming that the spammer controls a large fraction of the in-links, so he has an incentive to try to boost its home page instead of an arbitrary page inside the host.

3.1 Degree-based measures

The distribution of in-degree and out-degree can be obtained very easily doing a single pass over the Web graph. In Figure 3 we depict the histogram of this metric over the normal pages and the spam pages. In this section we present several graphs such as Figure 3, in which the histogram is shown with bars for the normal pages and with lines for the spam pages. Both histograms are normalized independently, and the y-axis represents frequencies.

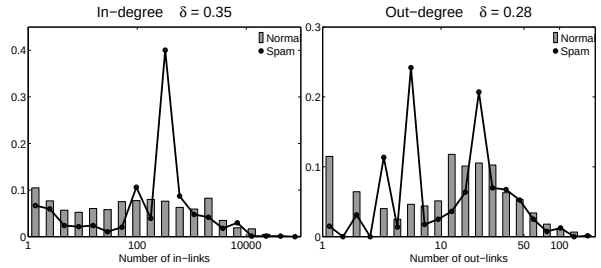


Figure 3: Histogram of the degree of home pages.

We have also included a parameter δ representing how different the histograms are. The value $\delta \in [0, 1]$ is based on the Kolmogorov-Smirnov test to verify if two distributions are the same, and is the maximum difference of the cumulative distribution functions (not shown here due to lack of space). The larger the value, the more different the distributions are.

In the case of in-degree we are using logarithmic binning, and the distribution seems to follow a power-law for normal pages, as the number of elements in each bin are similar. In the case of spam hosts, there is a large group of about 40% of them that have an in-degree in a very narrow interval. Something similar happens in the diagram of out-degree, but the difference between normal and spam pages is not as significant.

Another degree-based metric is the **edge-reciprocity**, that measures how many of the links in the directed Web graph are reciprocal. The edge-reciprocity can be computed easily by simultaneously scanning the graph and its transposed version, and measuring the overlap between the out-neighbors of a page and its in-neighbors.

In Figure 4 (left) we can see that the pages with maximum PageRank of the spam hosts, tend to have abnormally low

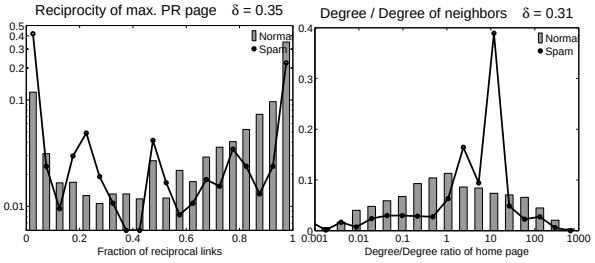


Figure 4: Left: histogram of the edge reciprocity in the page with maximum PageRank. Right: histogram of degree/degree ratio for home pages.

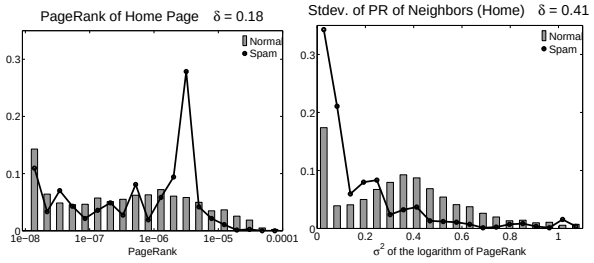


Figure 5: Left: histogram of the PageRank in the home page of hosts. Right: dispersion of PageRank values in the in-neighbors of the home pages.

reciprocity in our sample. In the case of home pages (not shown) the difference is not very large.

The degree of the nodes induces a natural "hierarchy" that can be used to define different classes of nodes. A network in which most nodes are connected to other nodes in the same class (for instance, most of the connections of highly-linked are to other highly-linked nodes) is called "**assortative**" and a network in which the contrary occurs is called "**disassortative**". The distinction is important from the point of view of epidemics [16].

We measured for every host in our sample the ratio between its degree and the average degree of its neighbors (considering both in- and out-links). In Figure 4 (right) we can see that in our collection, there is a mixing of assortative and disassortative behavior. The home pages of the spam hosts tend to be linked to/by pages with relatively lower in-degree. In our case, there is a peak at 10, meaning that for that group, their degree is 10 times larger than the degree of their direct neighbors.

All of the measures in this section can be computed in one or two passes over the Web graph (and the transposed graph). Using only these attributes (17 features in total) we build two spam classifiers, as explained in section 2. Using them we can identify from 72.6% to 74.4% of the spam hosts with a false positive rate from 2.0% to 3.1%.

3.2 PageRank

Let $A_{N\times N}$ be the citation matrix of graph G, that is, $a_{xy} = 1 \iff (x, y) \in E$. Let $P_{N\times N}$ be the row-normalized citation matrix, such that all rows sum up to one, and rows of zeros are replaced by rows of $1/N$. PageRank [22] can be described as a functional ranking [1], that is, a link-based ranking algorithm that computes a scoring vector S of the form:

$$\mathbf{S} = \sum_{t=0}^{\infty} \frac{\text{damping}(t)}{N} \mathbf{P}^t \ .$$

where damping(t) is a decreasing function of t, the lengths of the paths. In particular, for PageRank the damping function is exponentially decreasing, namely, damping$(t) = (1-\alpha)\alpha^t$.

We plot the distribution of the PageRank values of the home pages in Figure 5 (left). We can see a large fraction of pages sharing the same PageRank. This is more or less expected as there is also a large fraction of pages sharing the same in-degree (although these are not equivalent metrics).

An interesting observation we obtained is that in the case of home pages the distribution seems to follow a power-law (in the graph of Figure 5 the bins are logarithmic), while for the pages with maximum PageRank on each host, the distribution seems to be log-normal. This deserves further studying in the future.

Following an idea by Benczúr et al. [4], we studied the PageRank distribution of the pages that contribute to the PageRank of a given page. In [4], this distribution is studied over a sample of the pages that point recursively to the target page (with a strong preference for shorter paths), while here we study the distribution of the PageRank of the direct in-neighborhood of a page only. The result is shown in Figure 5 (right), and it is clear that for most of the spammers in our sample, it is more frequent to have less dispersion in the values of the PageRank of the in-neighbors.

Automatic classifiers built with the attributes we have described in this section (28 features in total), can identify from 74.4% to 77.3% of the spam hosts with a false positive rate of 1.7% to 2.6%.

3.3 TrustRank

In [18] the TrustRank algorithm for trust propagation is described: it starts with a seed of hand-picked **trusted** nodes and then propagates their score by following links. The intuition behind TrustRank is that a page with high PageRank, but without relationship with any of the trusted pages, is suspicious.

The **spam mass** of a page is defined as the amount of PageRank received by that page from spammers. This quantity cannot be calculated in practice, but it can be estimated by measuring the **estimated non-spam mass**, which is the amount of score that a page receives from trusted pages. For the purpose of this paper we refer to this quantity simply as the **TrustRank score** of a page.

For calculating this score, a biased random walk is carried out on the Web graph. With probability α we follow an out-link from a page, and with probability $1 - \alpha$ we go back to one of the trusted nodes picked at random. For the trusted nodes we used data from the Open Directory Project (available at `http://rdf.dmoz.org/`), selecting all the listed hosts inside the `.uk` domain. As of April 2006, this includes over 150,000 different hosts, from which 32,866 were included in our collection. Out of these, we have tagged 2,626 of them as normal hosts and 21 as spam. We removed those spam sites from the seed set (we also made some tests keeping them and the difference was not noticeable).

As shown in Figure 6, the score obtained by the home page of hosts in the normal class and hosts in the spam class is very different. Also, the ratio between the TrustRank score

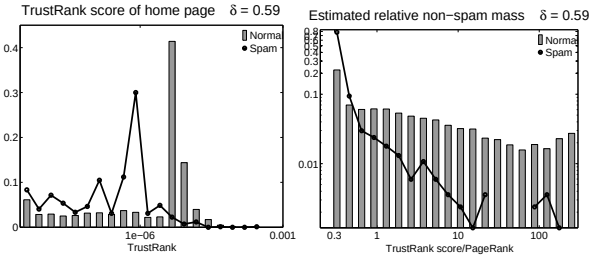


Figure 6: Left: histogram of TrustRank scores of home pages. Right: histogram of the estimated relative non-spam mass.

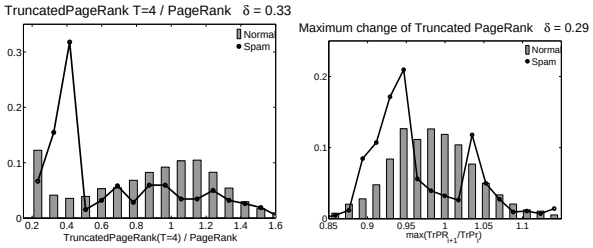


Figure 7: Left: histogram of the ratio between TruncatedPageRank at distance 4 and PageRank in the home page. Right: maximum ratio change of the TruncatedPageRank from distance i to distance $i-1$.

and the PageRank (the estimated relative non-spam mass) is also very effective for separating spam from normal pages.

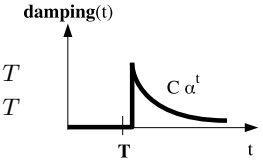
Using degree correlations, PageRank and TrustRank as attributes (35 features in total), we built classifiers with detection rates from 77.0% to 77.3% and 1.8% to 3.0% of false positives.

3.4 Truncated PageRank

In [3] we described Truncated PageRank, a link-based ranking function that decreases the importance of neighbors that are topologically "close" to the target node. In [26] it is shown that spam pages should be very sensitive to changes in the damping factor of the PageRank calculation; in our case with Truncated PageRank we modify not only the damping factor but the whole damping function.

Intuitively, a way of demoting spam pages is to consider a damping function that **removes the direct contribution of the first levels of links**, such as:

$$\text{damping}(t) = \begin{cases} 0 & t \leq T \\ C\alpha^t & t > T \end{cases}$$



Where C is a normalization constant and α is the damping factor used for PageRank. This function penalizes pages that obtain a large share of their PageRank from the first few levels of links; we call the corresponding functional ranking the **Truncated PageRank** of a page. The calculation of Truncated PageRank is described in detail in [3]. There is a very fast method for calculating Truncated PageRank. Given a PageRank computation, we can store "snapshots" of the PageRank values at different iterations and then take the difference and normalize those values at the end of the PageRank computation. Essentially, this means that the Truncated PageRank can be calculated for free during the PageRank iterations.

Note that as the number of indirect neighbors also depends on the number of direct neighbors, reducing the contribution of the first level of links by this method does not mean that we are calculating something completely different from PageRank. In fact, for most pages, both measures are strongly correlated, as shown in [3].

In practice, we observe that for the spam hosts in our collection, the Truncated PageRank is smaller than the PageRank, as shown in Figure 7 (left). There is a sharp peak for the spam pages in low values, meaning that many spam pages loose a large part of their PageRank when Truncated PageRank is used. We also found that studying the ratio of Truncated PageRank at distance i versus Truncated PageRank at distance $i-1$ also helps in identifying Web spam, as shown in Figure 7 (right). A classifier using Truncated PageRank, as well as PageRank and degree-based attributes (60 features in total) can identify 76.9% to 78.0% of the spam hosts with 1.6% to 2.5% of false positives.

3.5 Estimation of supporters

Following [4], we call x a **supporter** of page y at distance d, if the shortest path from x to y formed by links in E has length d. The set of supporters of a page are all the other pages that contribute to its link-based ranking.

A natural way of fighting link spam is to count the supporters. The naive approach is to repeat a reverse breadth-first search from each node of the graph, up to a certain depth, and mark nodes as they are visited [20]. Unfortunately, this is infeasible unless a subset of "suspicious" node is known a priori. A method for estimating the number of supporters of each node in the graph is described in [3] which improves [23].

The general algorithm (described in detail in [3]) involves the propagation of a bit mask. We start by assigning a random vector of bits to each page. We then perform an iterative computation: on each iteration of the algorithm, if page y has a link to page x, then the bit vector of page x is updated as $x \leftarrow x$ `OR` y. After d iterations, the bit vector associated to any page x provides information about the number of supporters of x at distance $\leq d$. Intuitively, if a page has a larger number of supporters than another, more `1`s will appear in the final configuration of its bit vector.

The algorithm is described in detail in [3]. In order to have a good estimation, d passes have to be repeated $O(\log N)$ times with different initial values, because the range of the possible values for the number of supporters is very large. We have observed that counting supporters from distances d from 1 to 4 give good results in practice. We measured how the number of supporters change at different distances, by measuring, for instance, the ratio between the number of supporters at distance 4 and the number of supporters at distance 3. The histogram for the minimum and maximum change is shown in Figure 8 (left).

This algorithm can be extended very easily to consider the number of different **hosts** contributing to the ranking of a given host. To do so, in the initialization the bit masks of all the pages in the same host have to be made equal. In Figure 8 (right), we plot the number of supporters at dis-

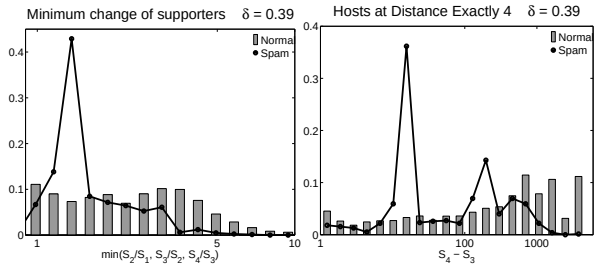


Figure 8: Left: histogram of the minimum change in the size of the neighborhood in the first few levels. Right: number of different hosts at distance 4

tance 4 considering different hosts contributing towards the ranking of the home pages of the marked hosts. We observed anomalies in this distribution for the case of the spam pages, and these anomalies are more evident by counting different hosts than by counting different pages.

Considering distance 4, the estimation of supporters based on pages (62 attributes) yields a classifier with 78.9% to 77.9% of detection rate and 1.4% to 2.5% of false positives. If we base the estimation on hosts (67 attributes, slightly more because in-degree is not the number of neighbors at distance one in this case) allows us to build a classifier for detecting 76.5% to 77.4% of the spam with an error rate from 1.3% to 2.4%.

The detection rate is two to three percentage points lower if distance 2 is considered, with roughly the same false positives ratio.

3.6 Everything

By combining all of the attributes we have discussed so far (163 attributes in total), we obtained a better performance than each of the individual classifiers. The detection rate of the final classifier is between 80.4% and 81.4%, with a false positive rate of 1.1% to 2.8% respectively. The first classifier has 40 rules (which provides a robust classifier), while the second classifier has 175. The performance of our best classifier can be compared with content-based analysis [21], which with an equivalent, unrestricted, boosted classifier, achieves 86.2% of detection rate with 2.2% false positives using content features.

The ten most important attributes in the complete set were obtained by using the attribute selection mechanism of `Weka`, that samples instances and consider the value of each attribute in the nearest same-class and different-class instance:

1. Binary variable indicating if home page is the page with maximum PageRank of the site
2. Edge reciprocity
3. Supporters (different hosts) at distance 4
4. Supporters (different hosts) at distance 3
5. Minimum change of supporters (different hosts)
6. Supporters (different hosts) at distance 2
7. Truncated PageRank at distance 1 divided by PageRank
8. TrustRank score divided by PageRank
9. Supporters (different hosts) at distance 1
10. Truncated PageRank at distance 2 divided by PageRank

4. CONCLUSIONS AND FUTURE WORK

A first criticism of this study can be that the sample is not uniform, but is biased towards large Web sites and highly ranked Web pages. However, a uniform random sample in this case is much harder to obtain, as it requires to inspect a larger set of pages, which we can not do by ourselves at this moment. We are currently collaborating with other researchers in tagging a large uniform sample from a collection of Web pages.

Our host-based approach also has some drawbacks. For instance, hosts can have mixed spam/legitimate content. In any case, we have seen that for several metrics it is important to measure the variables in both the home page of the host and the page with the maximum PageRank. For some metrics only one of the two pages provides useful information for the spam detection technique, and it is not always the same page. Another approach could be to evaluate each metric also by taking its average over each Web host. Finally, a better definition of Web site instead of host would be useful; for instance, considering multi-site hosts such as `geocities.com` as separated entities.

Some authors have hinted that the arms race between search engines and spammers calls for a serious reconsideration of Web search. For instance, Gori and Witten argue that "one might try to address speculative Web visibility scams individually (as search engine companies are no doubt doing); however, the bubble is likely to reappear in other guises" [14]. It would be interesting to try to devise clear rules separating what is allowed and what is not allowed from the point of view of a search engine, instead of continuing playing "hide and seek" with the spammers. In an analogy with sport competitions, this set of rules would define a kind of "*anti-doping rules*" for Web sites. Our work contributes towards this goal by suggesting that it is possible to detect a large fraction of the spammers by analyzing link-based metrics. Following the analogy, this could be used as part of an "*anti-doping test*" for Web pages, which should involve at least both link-based and content-based analysis.

The source code of the implementation of the algorithms presented in this paper will be freely available under a GPL license at `http://www.dis.uniroma1.it/~ae/` for the final version of the paper, along with our data set of hosts and their class labels, for repeatability of these results and further testing of other web spam detection techniques.

5. REFERENCES

[1] R. Baeza-Yates, P. Boldi, and C. Castillo. Generalizing PageRank: Damping functions for link-based ranking algorithms. In *Proceedings of SIGIR*, Seattle, Washington, USA, August 2006. ACM Press.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, May 1999.

[3] L. Becchetti, C. Castillo, D. Donato, S. Leonardi, and R. Baeza-Yates. Using rank propagation and probabilistic counting for link-based spam detection. Technical report, DELIS – Dynamically Evolving, Large-Scale Information Systems, 2006.

[4] A. A. Benczúr, K. Csalogány, T. Sarlós, and M. Uher. Spamrank: fully automatic link spam detection. In *Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web*, Chiba, Japan, May 2005.

[5] L. Costa, F. A. Rodrigues, and G. a. Travieso. Characterization of complex networks: A survey of measurements, Jun 2005.

[6] B. D. Davison. Recognizing nepotistic links on the web. In *Aaai-2000 Workshop On Artificial Intelligence For Web Search*, pages 23–28, Austin, Texas, July 2000. Aaai Press.

[7] C. Demetrescu, I. Finocchi, and A. Ribichini. Trading off space for passes in graph streaming problems. In *Proceedings of the 7th annual ACM-SIAM Symposium on Discrete Algorithms*, 2006.

[8] 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)*, volume 3720 of *Lecture Notes in Artificial Intelligence*, pages 233–243, Porto, Portugal, 2005.

[9] N. Eiron, K. S. Curley, and J. A. Tomlin. Ranking the web frontier. In *Proceedings of the 13th international conference on World Wide Web*, pages 309–318, New York, NY, USA, 2004. ACM Press.

[10] J. Feigenbaum, S. Kannan, M. A. Gregor, S. Suri, and J. Zhang. On graph problems in a semi-streaming model. In *31st International Colloquium on Automata, Languages and Programming*, 2004.

[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 seventh workshop on the Web and databases (WebDB)*, pages 1–6, Paris, France, June 2004.

[12] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996.

[13] D. Gibson, R. Kumar, and A. Tomkins. Discovering large dense subgraphs in massive graphs. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 721–732. VLDB Endowment, 2005.

[14] M. Gori and I. Witten. The bubble of web visibility. *Commun. ACM*, 48(3):115–117, March 2005.

[15] A. Gulli and A. Signorini. The indexable Web is more than 11.5 billion pages. In *Poster proceedings of the 14th international conference on World Wide Web*, pages 902–903, Chiba, Japan, 2005. ACM Press.

[16] S. Gupta, R. M. Anderson, and R. M. May. Networks of sexual contacts: implications for the pattern of spread of hiv. *AIDS*, 3(12):807–817, December 1989.

[17] Z. Gyöngyi and H. Garcia-Molina. Web spam taxonomy. In *First International Workshop on Adversarial Information Retrieval on the Web*, 2005.

[18] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases (VLDB)*, pages 576–587, Toronto, Canada, August 2004. Morgan Kaufmann.

[19] M. R. Henzinger, P. Raghavan, and S. Rajagopalan. Computing on data streams. *Dimacs Series In Discrete Mathematics And Theoretical Computer Science*, pages 107–118, 1999.

[20] R. J. Lipton and J. F. Naughton. Estimating the size of generalized transitive closures. In *VLDB '89: Proceedings of the 15th international conference on Very large data bases*, pages 165–171, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[21] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *Proceedings of the World Wide Web conference*, pages 83–92, Edinburgh, Scotland, May 2006.

[22] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: bringing order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.

[23] C. R. Palmer, P. B. Gibbons, and C. Faloutsos. ANF: a fast and scalable tool for data mining in massive graphs. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 81–90, New York, NY, USA, 2002. ACM Press.

[24] J. S. Vitter. External memory algorithms and data structures. *ACM Computing Surveys*, 33(2):209–271, 2001.

[25] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, October 1999.

[26] H. Zhang, A. Goel, R. Govindan, K. Mason, and B. Van Roy. Making eigenvector-based reputation systems robust to collusion. In *Proceedings of the third Workshop on Web Graphs (WAW)*, volume 3243 of *Lecture Notes in Computer Science*, pages 92–104, Rome, Italy, October 2004. Springer.

Appendix: Full List of Attributes

Included here for repeatability of the results:

Degree-based, 17 features (section 3.1)

All of the following attributes for the home page and the page with the maximum PageRank, plus a binary variable indicating if they are the same page:

- In-degree, out-degree, fraction of reciprocal edges
- Degree divided by degree of direct neighbors
- Average and sum of in-degree of out-neighbors
- Average and sum of out-degree of in-neighbors

PageRank, 28 features (section 3.2)

All of the above, plus the following for the home page and the page with maximum PageRank:

- PageRank, In-degree/PageRank, Out-degree/PageRank
- Standard deviation of PageRank of in-neighbors $= \sigma^2$
- σ^2/PageRank

Plus the PageRank of the home page divided by the PageRank of the page with the maximum PageRank.

TrustRank, 35 features (section 3.3)

PageRank attributes of section 3.2, plus the following for the home page and the page with maximum PageRank:

- TrustRank (estimated absolute non-spam mass)
- TrustRank/PageRank, TrustRank/In-degree

Plus the TrustRank in the home page divided by the TrustRank in the page with the maximum PageRank.

Truncated PageRank, 60 features (section 3.4)

PageRank attributes of section 3.2, plus the following for the home page and the page with maximum PageRank:

- TruncatedPageRank($T = 1 \ldots 4$)
- TruncatedPageRank($T = i$) / TruncatedPageRank($T = i - 1$)
- TruncatedPageRank($T = 1 \ldots 4$) / PageRank
- Min., max. and avg. of TruncatedPageRank($T = i$) / TruncatedPageRank($T = i - 1$)

Plus the TruncatedPageRank($T = 1 \ldots 4$) of the home page divided by the same value in the page with the maximum PageRank.

Estimation of supporters (section 3.5)

PageRank attributes of section 3.2, plus the following for the home page and the page with maximum PageRank:

- Supporters at $2 \ldots 4$ (supporters at 1 is equal to in-degree)
- Supporters at $2 \ldots 4$ / PageRank
- Supporters at i / Supporters at $i - 1$ (for $i = 1..4$)
- Min., max. and avg. of: Supporters at i / Supporters at $i - 1$ (for $i = 1..4$)
- (Supporters at i - Supporters at $i - 1$) / PageRank (for $i = 1..4$). The quantity (Supporters at i - Supporters at $i - 1$) is the number of supporters at distance exactly i.

Plus the number of supporters at distance $2 \ldots 4$ in the home page divided by the same feature in the page with the maximum PageRank.

For the estimation of supporters using hosts, the same attributes but considering that two supporters in the same host count as only one supporter.

Link-Based Similarity Search to Fight Web Spam*

András A. Benczúr Károly Csalogány Tamás Sarlós
Informatics Laboratory
Computer and Automation Research Institute
Hungarian Academy of Sciences
11 Lagymanyosi u, H-1111 Budapest
and
Eötvös University, Budapest
{benczur, cskaresz, stamas}@ilab.sztaki.hu
www.ilab.sztaki.hu/websearch

ABSTRACT

We investigate the usability of similarity search in fighting Web spam based on the assumption that an unknown spam page is more similar to certain known spam pages than to honest pages.

In order to be successful, search engine spam never appears in isolation: we observe link farms and alliances for the sole purpose of search engine ranking manipulation. The artificial nature and strong inside connectedness however gave rise to successful algorithms to identify search engine spam. One example is trust and distrust propagation, an idea originating in recommender systems and P2P networks, that yields spam classificators by spreading information along hyperlinks from white and blacklists. While most previous results use PageRank variants for propagation, we form classifiers by investigating similarity top lists of an unknown page along various measures such as co-citation, companion, nearest neighbors in low dimensional projections and SimRank. We test our method over two data sets previously used to measure spam filtering algorithms.

1. INTRODUCTION

With the advent of search engines web spamming appeared as early as 1996 [7]. Identifying and preventing spam was cited as one of the top challenges in web search engines in a 2002 paper [16]. The birth of the highly successful PageRank algorithm [29] was indeed partially motivated by the easy spammability of the simple in-degree count; its variants [19; 11; 15; 3; 39; 2, and many others] proved successful in fighting search engine spam.

Spam and various means of search engine optimization seriously deteriorate search engine ranking results; as a response, building black and whitelist belongs to the daily routine of search engine operation. Our goal is to extend this invaluable source of human annotation either to automatically demote pages similar to certain known spam pages or to suggest additional pages for the operator to be included in the blacklist.

Recently several results has appeared that apply rank propagation to extend initial trust or distrust judgements over a small set of seed pages or sites to the entire web. These methods are either based on propagating trust forward or distrust backwards along the hyperlinks based on the idea that honest pages predominantly point to honest ones, or, stated the other way, spam pages are backlinked only by spam pages. We argue that compared to unidirectional propagation methods, the initial labels are better utilized if we apply similarity search techniques, which involve a bidirectional backward and forward step.

In this paper we concentrate on spreading trust and distrust information from a seed set with the help of hyperlink based similarity measures. Our main goal is to identify features based on similarities to known honest and spam pages that can be used to classify unknown pages. We demonstrate the usability of co-citation, Companion [8], SimRank [18] and variants [10] as well as the singular value decomposition of the adjacency matrix in supervised spam learning.

Hyperlink based similarity to spam versus honest pages is comparable to trust and distrust propagation while giving a natural combination of backward and forward propagation. Given a link farm alliance [13] with one known target labeled as spam, similarity based features will automatically label other targets as spam as well.

As the main result of our paper, we show that over our data set of the `.de` domain as well as the `.ch` domain data in courtesy of the `search.ch` engine [33] similarity based single features perform better than trust or distrust propagation based single features at higher recall values. Ironically, the easiest-to-manipulate co-citation performs best; as an alternate somewhat more robust against manipulations but performing similarly well we suggest Companion [8]. Our results are complementary to the recent results of [2] based on link structure and of [27] based on content analysis. We leave classification based on the combination of features as future work.

2. RELATED RESULTS

Next we survey related results both for hyperlink based spam detection and similarity search. Recently very large number of results appeared to fight spam; we list just the most relevant ones and point to references therein.

2.1 PageRank based trust and distrust propagation

When using trust and distrust information, we may propagate trust forward to pages pointed by trusted ones or distrust backward to pages that point to spam. In previous results we see all variants: TrustRank [15, 39] propagates trust forward, BadRank [31,

*Support from the NKFP 2005 project MOLINGV and by the Inter-University Center for Telecommunications and Informatics.

AIRWEB'06, August 10, 2006, Seattle, Washington, USA.

9] distrust backward; [38] uses a combination. We describe these important predecessors of our work next.

As the first trust propagation method against link spam, Gyöngyi et al. [15] show that spam sites can be pushed down in PageRank ordering if we personalize on a few trusted hub sites. Their method is semi automatic, the trusted 180 seed pages were carefully hand picked from 1250 good hub pages distilled automatically using Inverse PageRank. Notice that TrustRank requires a very carefully selected seed set that we cannot provide in our experiment. Wu et al. [39] describes an improvement of TrustRank by reducing the bias induced by the seed set. Gyöngyi et al. [12] recognize link spam by comparing the TrustRank and PageRank values.

Trust and distrust propagation in trust networks originates in Guha et al. [11] for trust networks; Wu et al. [38] show its applicability for Web spam classification. As noticed by [11] distrust propagation is more problematic that that of trust. Although for a different data type (trust/distrust among Epinions reviewers), they raise the question of interpreting the distrust of a distrusted party. While [38] emphasizes the difference between identifying preferences of a single user and a global notion of trust over the Web, they also require a combination of trust and distrust propagation to achieve best results.

As an earlier result, [19] EigenTrust is PageRank with weights that are trust values. Another method [25] penalizes the biconnected component of a spam page in a subgraph obtained by backward distrust propagation.

2.2 Similarity search, HITS and spam

Several link-based algorithms were designed to evaluate node-to-node similarities in networks; we refer to [23] for an exhaustive list of the available methods ranging from co-citation to more complex measures such as max-flow/min-cut-based similarities of [24] in the vicinity graph of the query. Closest to our notions of link based similarity is co-citation already used in [11] as an elementary step of trust propagation.

Dean and Henzinger [8] describe the Companion algorithm that is reported to outperform co-citation in finding related pages. Their algorithm computes the authority scores by the HITS algorithm [20] in the vicinity of the query page.

HITS itself is known to be vulnerable to spam and in particular to the so-called tightly knit community (TKC) effect. Vulnerability to spam, however, makes HITS a good candidate to actually detect spam when run in the neighborhood of known spam pages that we explore in our paper. An overview of the theoretical results underlying the TKC effect is given in Section 7 of [22] and the references therein that indicate a very weak TKC-type spam resistance of HITS and a somewhat better but still unsatisfying one of PageRank.

Another example of HITS and spam is the result of Wu and Davison [37]. Unlike our approach of exploiting the spam sensibility of HITS in prediction, they make HITS spam resistant by identifying a seed set of link farm pages based on the observation that the in- and out-neighborhood of link farm pages tend to overlap. Then the seed set of bad pages is iteratively extended to other pages which link to many bad pages; finally the links between bad pages are dropped. Experiments show that a simple weighted in-degree scheme on the modified graph yields significantly better precision for top ten page hit lists than the Bharat-Henzinger [5] HITS variant.

Additionally we mention the first example that gives anecdotal evidence for the usability of similarities in hyperlink structure to identify spam. Amitay et al. [1] extracted features based on the linkage patterns of web sites and trained a decision tree and a Bayesian classifier to classify each site to one of the 8 predefined functional categories. A cosine metric based clustering of the feature space produced a decent amount clusters whose members appeared to belong to the same spam ring. As it was not the original goal of their research, no results were published on classifying sites as spam or non-spam.

Finally we remark that identifying spam pages is somewhat analogous to classifying web documents into multiple topics. Several results [32, and the references therein] demonstrate that classification accuracy can be significantly increased by taking into account the class labels assigned to neighboring nodes. In accordance with our experiments, Qi and Davison [32] found that most of the improvement comes from the neighborhood defined by co-citation.

2.3 Spam data sets and methodology

Before describing our measurements, we elaborate on the hardness of comparing results of different authors and data sets. We show preliminary results indicating the difficulty of correctly labeling spam by human evaluators as well as compare the different availability of data sets.

While we believe that identifying email spam and certain types of web content spam by human inspection is relative easy and automated methods cannot, in any case, perform as good as human judgement. Search engine spam, however, is much harder to identify. Györgyi and Garcia-Molina [14] list a few methods that confuse users including term hiding (background color text); cloaking (different content for browsers and search engine robots) and redirection; some of these techniques can still be found by inspecting the HTML code within the page source. A few examples of the `.de` domain are given in our previous result [3].

In contrast to the hardness of manual spam classification, apart from our previous result [3] we have no knowledge of investigations for the reliability of the manual labels. In our experiment [3] over the `.de` domain we report a very poor pairwise $\kappa = 0.45$ [6] over the 100 pairs of URLs with judgements by two different evaluators. The majority of disagreements could be attributed to different rating of pages in affiliate programs and certain cliques. This shows that assessing link spam is nontrivial task for humans as well. Gyöngyi et al. [15] mention "using an author as an evaluator raises the issue of bias in the results" and emphasize the expertise needed for search engine operators that, in our work, have no access. They also describe the hardness of the task as "manual evaluations took weeks: checking a site involves looking at many of its pages and also the linked sites to determine if there is an intention to deceive search engines."

Results on Web spam are in general based on data that needs careful analysis to replicate and compare. The `.uk` crawl [2] that we plan to use in future work is not yet publicly available. Some of the data such as the MSN crawl [27] is proprietary. Gyöngyi et al. [15] use an AltaVista crawl together with a proprietary tool for contracting pages within the same site to a single node prior to PageRank computation that we only mimic over the `.de` domain. While the Stanford WebBase [39] contains pages that are outdated, manual classification is possible with care through the Wayback Machine [39]. This is also true for our 2004 `.de` crawl [36] even though we use a 2005 manual classification [3].

Various top-level or otherwise selected domains may have different spamming behavior; Ntoulas et al. [27] give an invaluable comparison that show major differences among national domains and languages of the page. For the `.de` domain their findings agree with our 16.5% [3] while for the `.uk` domain together with Becchetti et al. [2] they report approximately 6%; the latter measurement also reports 16% of sites as spam over `.uk`.

We also mention the importance of giving more weight to pages

of high rank. Similar to our method, [15] uses a stratified random sample based on PageRank buckets for evaluation. Notice that a uniform sample would consist of mostly very low rank pages that would give little information about top ranked pages most important in search engine applications.

3. THE SIMILARITY BASED SPAM DETECTION ALGORITHMS

In our experiments we use the four similarity measures co-citation, SimRank [18], Companion [8] and singular vectors and we suggest the applicability of further SimRank variants. In this section we briefly introduce notation and the efficient algorithms [10, 35] that we use. We give special importance to algorithms with modest hardware requirements; our experiments ran on a commodity PC.

Similarity based spam prediction is less straightforward than trust and distrust propagation that directly ranks a page as honest or spam. Before describing the algorithms we hence describe our evaluation method. For a given unknown host u, our algorithm computes the similarity top list of u and makes a prediction based on the known spam and honest hosts in this list. For each similarity measure we extract four different features from the size k similarity top list of u. Let the top list contain h honest and s spam pages of the evaluation sample; in general $h + s < k$. Let the sum of the similarities of these pages be s^* and h^*, respectively. We define our features as follows.

- Spam Ratio (SR): fraction of the number of spam within labeled spam and honest pages, $s/(s+h)$.
- Spam over Non-spam (SON): number of spam divided by number of honest pages in the top list, s/h.
- Spam Value Ratio (SVR): sum of the similarity values of spam pages divided by the total similarity value of labeled spam and honest pages under the appropriate similarity function, $s^*/(s^*+h^*)$.
- Spam Value over Non-spam Value (SVONV): similarity value sum for spam divided by same for honest, s^*/h^*.

Given the above values, we may impose a threshold and predict the unknown input page spam if the measure is above the prescribed threshold. For different thresholds we obtain predictions of different quality; by decreasing its value we increase recall and likely but not necessarily decrease precision. For threshold 0 we predict all pages as spam with recall 1 and precision equal to the spam fraction in the data.

3.1 SimRank

Let us consider the web as a graph over hosts by contracting all individual pages that share a common fully qualified host name into the same vertex as in [15]. Let there be N vertices and let hyperlinks define directed edges E between them. Given node v we denote its in- and out-degree by $d^+(v)$ and $d^-(v)$, respectively.

The *PageRank* vector $p = (p_1, \ldots, p_N)$ is defined as the solution of the following equation [29]

$$p_u = (1-c) \cdot \sum_{(v,u)\in E} p_v/d^+(v) + c \cdot r_u \,, \qquad (1)$$

where $r = (r_1, \ldots, r_N)$ is the teleportation distribution and c is the teleportation probability with a typical value of $c \approx 0.15$. We get the PageRank if we set all r_i to $1/N$; for general r we get PageRank personalized on r.

Jeh and Widom [18] define SimRank by the following equation very similar to the PageRank power iteration: initially $\mathrm{Sim}^{(0)}(u_1,u_2) = 1$ if $u_1 = u_2$ and 0 otherwise and then

$$\mathrm{Sim}^{(i)}(u_1,u_2) = \begin{cases} (1-c)\cdot \frac{\sum \mathrm{Sim}^{(i-1)}(v_1,v_2)}{d^-(u_1)\cdot d^-(u_2)} & \text{if } u_1 \neq u_2 \\ 1 & \text{if } u_1 = u_2, \end{cases} \qquad (2)$$

where the summation is for all pairs $(v_1,u_1) \in E, (v_2,u_2) \in E$. SimRank is the multi-step generalization of co-citation in the same way as PageRank generalizes in-degree.

Given a Web page we predict spam by co-citation and SimRank based on the similarity top-list over the entire host graph. In the case of co-citation, the list includes all pages that have a common back-link; ranking is based on the number of such pages. We compute co-citation by keeping the host graph in internal memory.

SimRank power iterations as in (2) are infeasible since they require quadratic space; we use the algorithm of [35] instead, with additive error $\epsilon = 0.001$ and 10 iterations. We use all non-zeroes as the top list with size k. Since in internal steps the algorithm rounds down to multiples of the error parameter, the choice of ϵ determines the value of k.

Fogaras and Rácz [10] describe two variants PSimRank and XJaccard by modifying similarity propagation in the above equation (2); they give randomized approximation algorithms and measure PSimRank as better predictor of topical similarity than SimRank. Additionally, the algorithm we use for SimRank can very easily be modified to take self-similarities into account by relaxing the condition $\mathrm{Sim}(u_1,u_2) = 1$ and making a page more similar to itself if similar pages point to it. This modified measure may serve well for penalizing nepotism; we plan to test variants of self-similarity and PSimRank in future work.

3.2 Companion and SVD

The Singular Value Decomposition (SVD) of a rank ρ matrix $A \in \mathbb{R}^{m\times n}$ is given by $A = U\Sigma V^T$ with $U \in \mathbb{R}^{m\times\rho}$, $\Sigma \in \mathbb{R}^{\rho\times\rho}$ and $V \in \mathbb{R}^{n\times\rho}$ where Σ is a positive diagonal matrix with the singular values in the diagonal. By the Eckart-Young theorem the best rank-t approximation of A with respect to both the Frobenius and spectral norms is $A_t = U_t\Sigma_t V_t^T$, where $U_t \in \mathbb{R}^{m\times t}$ and $V_t \in \mathbb{R}^{n\times t}$ contain the first t columns of U and V and the diagonal $\Sigma_t \in \mathbb{R}^{t\times t}$ contains first t entries of Σ.

We use SVD for nearest neighbor search after a low dimensional projection of the adjacency matrix of the host graph. We represent host u by row u of $V_t\Sigma_t$ and measure similarity as the Eucledian distance in this t dimensional space. Besides computational advantages, the low dimensional projection also serves noise reduction in a similar way as Latent Semantic Indexing [30] applies to the word–document matrix. We perform brute force nearest neighbor search in the t dimensional space defined by U_t and consider the first 1000 nearest vertices as top list. Given that we use very low values of t, we could replace brute force search by more elaborate data structures [34] or approximation [17]; in our case however the sample was small enough to use the simplest implementation. In the experiments we use the SVDPACK [4] Lanczos implementation for computing the first 10 singular vectors.

HITS [20] authority scores are the coordinates of the first (right) singular vector of the adjacency matrix of the vicinity subgraph. The idea of using more than just the first singular vector appears in several results. The instability of a single authority (or hub) vector and stability of the t dimensional projection U_t is described by [26].

The Companion algorithm [8] builds the 2-step alternating neighborhood of the given vertex; then performs the HITS authority computation and returns the top authorities. We use a simplified

version that excludes steps such as edge weighting, large degree handling and link order considerations. For a query node v we build the vicinity graph by selecting nodes of length two alternating forward-backward of backward-forward paths starting at v. We randomly truncate large neighborhoods to a maximum of 2000 nodes in the first step and to 10 in the second step, as in [8]. We rank by the authority score and use all nodes of the vicinity graph as the top list. HITS is computed by simple power iteration.

4. EXPERIMENTS

4.1 Data sets

We use two data sets, the 31.2 M page crawl of the `.de` domain provided us by Torsten Suel and Yen-Yu Chen and the 20 M page crawl mostly from the Switzerland domain as courtesy of the `search.ch` engine [33]. We apply the evaluation methodologies of [3] for the `.de` and the data of [37] for the Switzerland domain that we review next.

The crawl carried out by the Polybot crawler [36] in April 2004 gives a German graph denser than the usual web graphs with 962 M edges implying an average out-degree of 30.82. Unlike in our previous result on the same data [3] we use the host graph not just because it speeds up experimentation but also because intra-site links that would give trivial similarity within the same host disappear and host level detection forms a more interesting task. When forming the host graph, we are left with a modest 808 K node and 24 M edge graph.

For the `.de` data we manually evaluated a stratified random sample as proposed first in [15]. We ordered the pages according to their PageRank value and assigned them to 20 consecutive buckets such that each bucket contained 5% of the total PageRank sum. As this step was made for the prior experiment, we computed PageRank over the page-level graph instead of the host graph; stratification in the sample selection however has no further effect on our experiments. From each bucket we chose 50 URLs uniformly at random, resulting in a 1000 page sample heavily biased toward pages with high PageRank. The sample was manually classified as described in [3] with judgements reflecting the state as of April 2004. Figure 1 shows[1] the distribution of categories among the *hosts* that slightly differ from the page level distribution [3]. Our prior findings of 16.5% spam among `.de` pages [3] agrees with [27] and our increased 20.9% spam on the host level with the similar findings of [2] for the `.uk` domain.

The `search.ch` data is a 2004 crawl of approximately 20 M pages mostly from the `.ch` domain. We used the domain graph with 300 K nodes and 24 M edges reflecting the connectivity of the two highest levels of the domain hierarchy within this dataset [37].

The 19605 domains appearing in the URL list extracted by Wu et al. [38] from the Switzerland specific ODP [28] topics formed our trusted set. As spam set we used a labeled list of 728 domains provided by `search.ch` [33]. One particular property of this blacklist is that 627 domains share 144 different IP addresses, the remaining 101 could not be resolved in June 2006. Note that the Swiss evaluation sample contains 3.6% spam only.

4.2 Evaluation by cross-validation

We evaluate our methods together with trust and distrust propagation baselines by three-fold cross-validation. We observe very large variance between various random cross-validation splits, a phenomenon that we show for a single feature instance in Fig. 2

[1] Unless stated otherwise all figures in this section refer to the `.de` dataset.

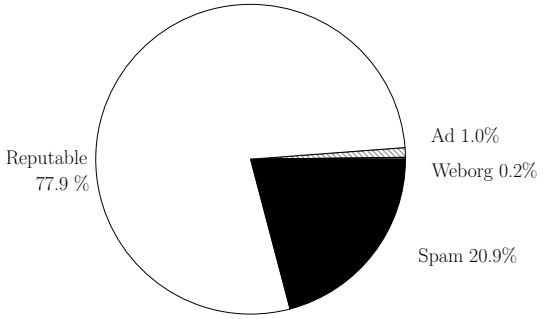


Figure 1: Distribution of categories among the hosts in the evaluation sample

but holds for all features. The explanation likely lies in the small size of our sample: given a query page, the success of spam classification heavily depends on whether those possibly very few pages that contain relevant information are used for test or training.

Given the large variance, we show or measurements by averaging cross-validation results with five independent random splits that altogether correspond to 15 measurements, three for each split. Since in the 15 measurements we will have precision values for different values of recall and we may have several precision values for a given recall, we need special care to average our measurement.

We average our measurements by extrapolating precision for a given recall from measured values. For a single measurement we obtain precision–recall value pairs by imposing different thresholds. By decreasing the threshold we increase recall; increment is however in discrete steps that changes whenever the threshold reaches new hosts. If we reach a single host that is spam, both precision and recall increases by certain amount; we may then linearly extrapolate between the recall at the boundaries. If the new host is honest, we obtain a new, smaller precision value for the previous recall; we average all these values for a single experiment before averaging between measurements. Given ties, we classify more than a single host that makes recall increase and precision change according to the fraction α of spam among the new hosts. For a given intermediate recall we may then interpolate by adding a (possible fractional) number of pages with a fraction α of spam and computing precision. This method reduces to linear interpolation for a single new spam page with $\alpha = 1$ but nonlinear otherwise.

4.3 Baseline results

For baseline experiments we use the trust and distrust propagation measures of Wu et al. [38] by personalizing host based PageRank on known honest vs. spam hosts. We reproduce results of these experiments as Wu et al. [38] choose methods other than precision-recall curves for evaluation. We use the following variants described by [38]. In a single personalized PageRank iteration we may use constant splitting or logarithm splitting instead of equal splitting. We also use the maximum share variant by replacing summation by maximum in the PageRank equation. We leave the maximum parent variant of [38] for future work; we hence test 6 variants, including the original BadRank corresponding to simple

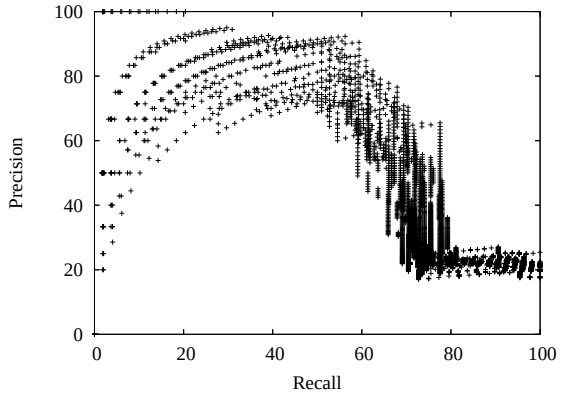


Figure 2: The outcome of five threefold cross-validation results with various random splits, for co-citation with SVR, altogether corresponding to measurement points over 15 precision-recall curves.

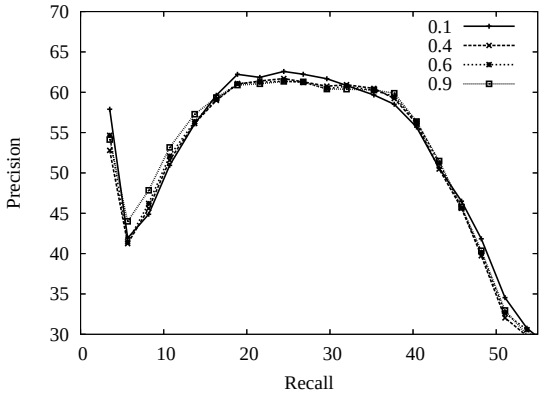


Figure 3: Precision as a function of recall for distrust propagation with logarithm splitting and maximum share. Four different values of teleportation probability c are shown.

summation equal splitting in the terminology of [38].

We experiment with different values of teleportation probability c. This value has negligible effect on the best measures as seen in Fig. 3. Other measures depend more heavily on c and reach best performance in general with low c. Since it has no effect on the comparison of methods we use a uniform $c = 0.1$ afterwards.

Our best trust and distrust propagation measurements are shown in Fig. 4, all with $c = 0.1$. Unmodified BadRank (equal split, summation) performs best at lowest recall but outperformed by equal split maximum share later. Logarithm split maximum share performs slightly worse but still outperforms the remaining three variants. Due to insufficient trust information, trust propagation performs very poor and often even below the random 20.9%, meaning that most spam manages through in cheating our TrustRank. Only the best original TrustRank (equal split, summation) is shown.

As suggested in [38], we improve results by combining trust and distrust propagation. We use linear combinations; surprisingly the best results are achieved by subtracting 0.8 times the trust score from 0.2 times the distrust score. Results for using the previous three best distrust score and the single best TrustRank is shown in Fig. 5. Hence although TrustRank performs bad alone due to insufficient trust information in our `.de` data, still its vote gives significant help to distrust propagation.

Over the `search.ch` dataset unmodified BadRank and logarithm split with simple summation performed best, their graphs are shown in Fig. 11.

4.4 Similarity based features

We use features with abbreviations SR, SON, SVR and SVONV as described in Section 3. For all four Figs. 6–9 we see bad precision at low recall, suggesting that honest pages may also collect high ranked similar spam that may be the result of artificial rank manipulations that is left for future work to verify.

Our methods perform best at relative high recall; finally converges to the random choice of 20.9% spam among hosts for very high recall. We see SR–SVR and SON–SVONV values in pairs performing very close. The first pair performs better at medium recall; the second pair performs poor in general but has a peak at higher recall where outperforms the first pair.

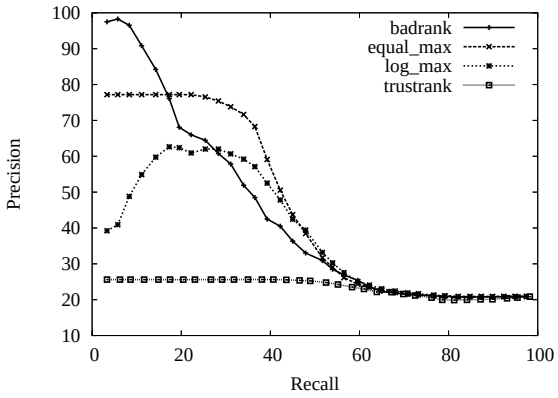


Figure 4: Precision as a function of recall for the best three distrust propagation variants and the single best TrustRank trust propagation.

Co-citation (Fig. 6) turns out best even at relative high recall values with Companion (Fig. 7) as the runner up. Our observations on the relative ordering of individual features and similarity functions hold unchanged over the Swiss dataset as well, hence we report figures only for the German data.

While our most successful candidate, notice the very easy spammability of the co-citation measure. As described by [21] we have to resist both false negative attacks of hiding spam as well as false positive ones that demote the competitor. Co-citation suffers the same vulnerability against spammers as in-degree: a spammer can easily create a large number of honey pot hosts that co-cite quality pages along with the spam target. By adding hosts that point to an honest h and a spam host s, we increase the chance of voting h spam and s honest.

Although SimRank (Fig. 8) performs poorest, it is the most robust measure against manipulations. In order to modify SimRank, the spammer must use a large number of pages that lead to both the spam target s and an honest page h. Depending on the Page-

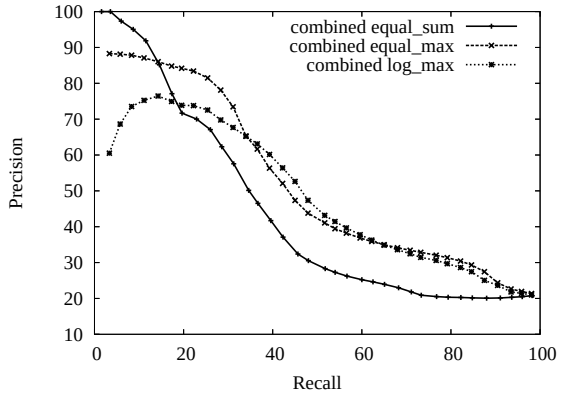


Figure 5: Precision as a function of recall for combined 0.8 times trust and 0.2 times distrust propagation.

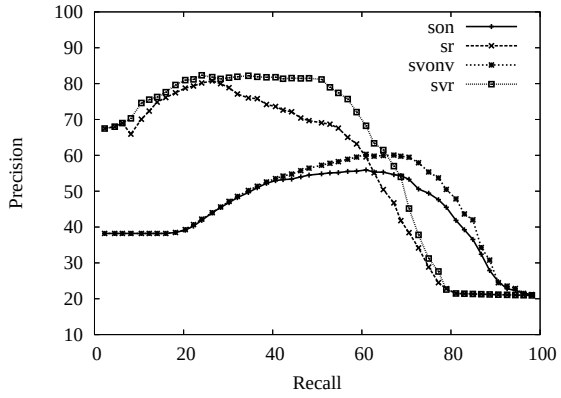


Figure 6: Precision as a function of recall for the four co-citation features.

Rank of h it is very unlikely that paths backward from h meet those from s that would mean high SimRank between h and s. Replacing SimRank with a better performing variant remains future work.

4.5 Comparison of best features

Finally in Fig. 10 we show all features that perform best for certain ranges of recall. BadRank is very effective at penalizing spam only but its recall is very low. Combined 0.2 times distrust minus 0.8 times trust propagation extends BadRank's performance, for the price of slightly decreased precision, to somewhat higher recall. Finally co-citation seems most effective for prediction with high recall. We also show Companion in Fig. 10 as the next best candidate if we disqualify co-citation due to its manipulability.

Turning to the Swiss dataset depicted in Fig. 11 we observe that distrust propagation with the unmodified BadRank algorithm or logarithm split and simple summation performs on par with Companion. As before, co-citation is the most precise measure with the exact feature depending on the level of recall. However overall accuracy is significantly higher than those observed for the `.de` domain. We attribute this to the (undisclosed) method(s) applied

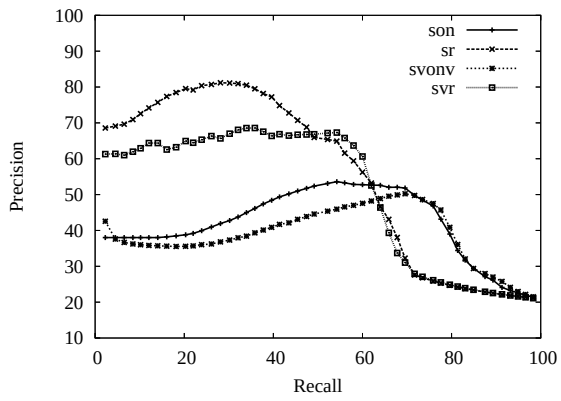


Figure 7: Precision as a function of recall for the four companion features.

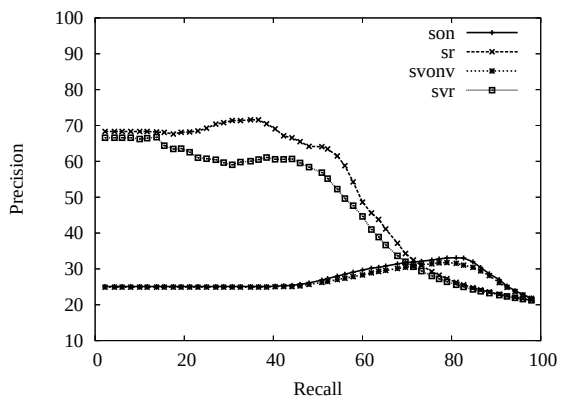


Figure 8: Precision as a function of recall for the four SimRank features.

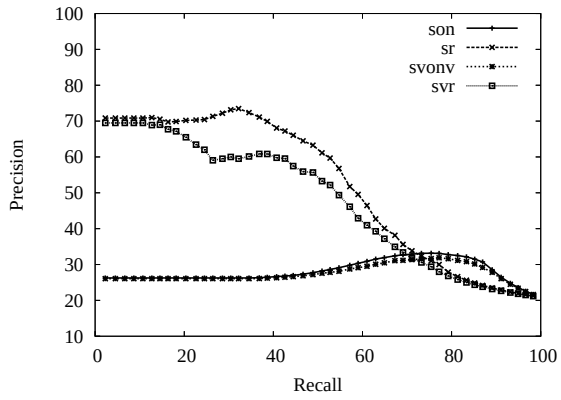


Figure 9: Precision as a function of recall for the four SVD nearest neighbor features.

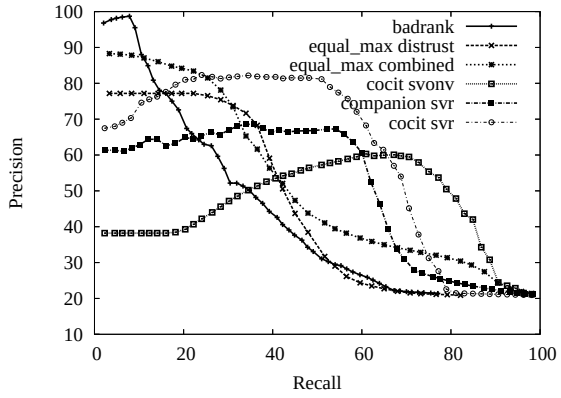


Figure 10: Precision as a function of recall for the best features.

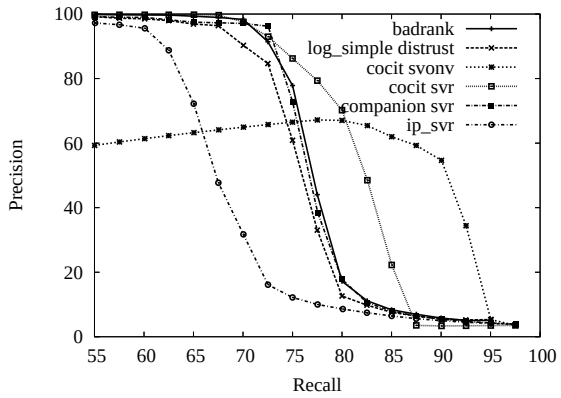


Figure 11: Precision as a function of recall for the best features on the Swiss domain graph.

by `search.ch` [33] to assemble the blacklist. For example, as already noted in Section 4.1, a large number of link farms share the same IP address. Hence a simple similarity measure based on the equality of IP addresses associated with the domains also works reasonably well. As shown on Fig. 12 accuracy decreases if we keep only a single domain for each IP address in the evaluation sample.

5. CONCLUSIONS

We presented hyperlink similarity based single feature classification measurements over a manually classified sample of the `.de` domain and the `search.ch` datasets. Our experiments demonstrated that similarity search based methods are indeed capable of learning the difference between spam and non-spam pages. In further work more SimRank variants and the combination of several features can be measured, including content based statistical features identified by Ntoulas et al. [27]; for combination decision trees as well as SVM should be used. Moreover, akin to [32] it needs to be investigated whether the accuracy of content based spam classifiers can be boosted by incorporating estimates assigned to similar nodes. In addition, the quality of the sample should be

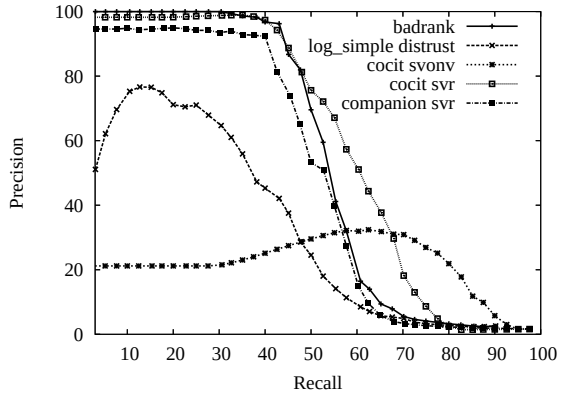


Figure 12: Precision as a function of recall for the best features on the Swiss domain graph with unique IP addresses.

improved by additional manual classification effort as well as other data sets should be involved in the measurement.

6. ACKNOWLEDGEMENT

The authors would like to thank Torsten Suel and Yen-Yu Chen for providing the `.de` web graph and Urban Müller and Baoning Wu and Brian D. Davison for the preprocessed `search.ch` dataset.

7. REFERENCES

[1] E. Amitay, D. Carmel, A. Darlow, R. Lempel, and A. Soffer. The Connectivity Sonar: Detecting site functionality by structural patterns. In *Proceedings of the 14th ACM Conference on Hypertext and Hypermedia (HT)*, pages 38–47, Nottingham, United Kingdom, 2003.

[2] L. Becchetti, C. Castillo, D. Donato, S. Leonardi, and R. Baeza-Yates. Link-based characterization and detection of web spam. In *Proceedings of the 2nd International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2006.

[3] A. A. Benczúr, K. Csalogány, T. Sarlós, and M. Uher. SpamRank – Fully automatic link spam detection. In *Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2005.

[4] M. W. Berry. SVDPACK: A Fortran-77 software library for the sparse singular value decomposition. Technical report, University of Tennessee, Knoxville, TN, USA, 1992.

[5] K. Bharat and M. R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 104–111, Melbourne, AU, 1998.

[6] J. Carletta. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–254, 1996.

[7] C. Chekuri, M. H. Goldwasser, P. Raghavan, and E. Upfal. Web search using automatic classification. In *Proceedings of the 6th International World Wide Web Conference (WWW)*, San Jose, USA, 1997.

[8] J. Dean and M. R. Henzinger. Finding related pages in the World Wide Web. In *Proceedings of the 8th World Wide Web Conference (WWW)*, pages 1467–1479, 1999.

[9] 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)*, volume 3720 of *Lecture Notes in Artificial Intelligence*, pages 233–243, Porto, Portugal, 2005.

[10] D. Fogaras and B. Rácz. Scaling link-based similarity search. In *Proceedings of the 14th World Wide Web Conference (WWW)*, pages 641–650, Chiba, Japan, 2005.

[11] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th International World Wide Web Conference (WWW)*, pages 403–412, 2004.

[12] Z. Gyöngyi, P. Berkhin, H. Garcia-Molina, and J. Pedersen. Link spam detection based on mass estimation. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB)*, Seoul, Korea, 2006.

[13] Z. Gyöngyi and H. Garcia-Molina. Link spam alliances. In *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB)*, Trondheim, Norway, 2005.

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

[15] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with TrustRank. In *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB)*, pages 576–587, Toronto, Canada, 2004.

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

[17] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the 30th ACM Symposium on Theory of Computing (STOC)*, pages 604–613, 1998.

[18] G. Jeh and J. Widom. SimRank: A measure of structural-context similarity. In *Proceedings of the 8th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 538–543, 2002.

[19] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The EigenTrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th International World Wide Web Conference (WWW)*, pages 640–651, New York, NY, USA, 2003. ACM Press.

[20] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[21] S. K. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th International World Wide Web Conference (WWW)*, pages 393–402, New York, NY, USA, 2004. ACM Press.

[22] A. N. Langville and C. D. Meyer. Deeper inside PageRank. *Internet Mathematics*, 1(3):335–400, 2004.

[23] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *Proceedings of the 12th Conference on Information and Knowledge Management (CIKM)*, pages 556–559, 2003.

[24] W. Lu, J. Janssen, E. Milios, and N. Japkowicz. Node similarity in networked information spaces. In *Proceedings of the Conference of the Centre for Advanced Studies on Collaborative research*, page 11, 2001.

[25] P. T. Metaxas and J. Destefano. Web spam, propaganda and trust. In *Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web*, 2005.

[26] A. Y. Ng, A. X. Zheng, and M. I. Jordan. Link analysis, eigenvectors and stability. In *Proc. Int. Joint Conf. Artificial Intelligence, Seattle, WA*, August 2001.

[27] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *Proceedings of the 15th International World Wide Web Conference (WWW)*, pages 83–92, Edinburgh, Scotland, 2006.

[28] Open Directory Project (ODP). `http://www.dmoz.org`.

[29] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford University, 1998.

[30] C. H. Papadimitriou, H. Tamaki, P. Raghavan, and S. Vempala. Latent semantic indexing: A probabilistic analysis. In *Proocedings of the ACM Conference on Principles of Database Systems (PODS)*, pages 159–168, 1998.

[31] PR10.info. BadRank as the opposite of PageRank, 2004. `http://en.pr10.info/pagerank0-badrank/` (visited June 27th, 2005).

[32] X. Qi and B. D. Davison. Knowing a web page by the company it keeps. Technical Report LU-CSE-06-011, Lehigh University, 2006.

[33] Räber Information Management GmbH. The Swiss search engine, http://www.search.ch/, 2006.

[34] H. Samet. *The design and analysis of spatial data structures*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.

[35] T. Sarlós, A. A. Benczúr, K. Csalogány, D. Fogaras, and B. Rácz. To randomize or not to randomize: Space optimal summaries for hyperlink analysis. In *Proceedings of the 15th World Wide Web Conference (WWW)*, 2006.

[36] T. Suel and V. Shkapenyuk. Design and implementation of a high-performance distributed web crawler. In *Proceedings of the 18th IEEE International Conference on Data Engineering (ICDE)*, pages 357–368, San Jose, California, USA, 2002.

[37] B. Wu and B. D. Davison. Identifying link farm pages. In *Proceedings of the 14th International World Wide Web Conference (WWW)*, pages 820–829, Chiba, Japan, 2005.

[38] B. Wu, V. Goel, and B. D. Davison. Propagating trust and distrust to demote web spam. In *Workshop on Models of Trust for the Web*, Edinburgh, Scotland, 2006.

[39] B. Wu, V. Goel, and B. D. Davison. Topical TrustRank: Using topicality to combat web spam. In *Proceedings of the 15th International World Wide Web Conference (WWW)*, Edinburgh, Scotland, 2006.

Improving Cloaking Detection Using Search Query Popularity and Monetizability

Kumar Chellapilla
Microsoft Live Labs
One Microsoft Way
Redmond, WA, USA 98052
+1 (425) 707-7575
kumarc@microsoft.com

David Maxwell Chickering
Microsoft Live Labs
One Microsoft Way
Redmond, WA, USA 98052
+1 (425) 703-5426
dmax@microsoft.com

ABSTRACT

Cloaking is a search engine spamming technique used by some Web sites to deliver one page to a search engine for indexing while serving an entirely different page to users browsing the site. In this paper, we show that the degree of cloaking among search results depends on query properties such as popularity and monetizability. We propose estimating query popularity and monetizability by analyzing search engine query logs and online advertising click-through logs, respectively. We also present a new measure for detecting cloaked URLs that uses a normalized term frequency ratio between multiple downloaded copies of Web pages. Experiments are conducted using 10,000 search queries and 3 million associated search result URLs. Experimental results indicate that while only 73.1% of the cloaked popular search URLs are spam, over 98.5% of the cloaked monetizable search URLs are spam. Further, on average, the search results for top 2% most cloaked queries are 10x more likely to be cloaking than those for the bottom 98% of the queries.

1. INTRODUCTION

Cloaking is a hiding technique [11] used by some Web servers to deliver one page to a search engine for indexing while serving an entirely different page to users browsing the site. In short, cloaking is the classic "bait and switch" technique applied to the Web. The motivation behind cloaking is to distort search engine rankings in favor of the cloaked page. Cloaking is commonly used in conjunction with other Web spamming techniques. Spammers can present the ultimately intended content to the Web users (without traces of spam on the page), and, at the same time, send a spammed document to the search engine for indexing.

In order for cloaking to be effective, the Web server must be able to detect Web crawler clients reliably. This is typically achieved by examining the client's: (a) user-agent string, and (b) IP address. A Web server can identify the Web client using the user-agent header in the HTTP request message. A few examples of user-agent strings are:

Internet Explorer:
```
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
```

MSNBot:
```
msnbot/1.0 (+http://search.msn.com/msnbot.htm)
```

GoogleBot:
```
Mozilla/5.0 (compatible; Googlebot/2.1;
+http://www.google.com/bot.html)
```

However, the user-agent strings are not strictly standardized and it is really up to the requesting application what to include in the corresponding message field. For example, it is common for less popular Web browsers to mimic the user-agent strings of the dominant browser to ensure a consistent browsing experience. Nevertheless, search engine crawlers do identify themselves by a name distinct from the ones used by traditional Web browser applications.

A very reliable way of identifying the client requesting a Web page is through its IP address. Some spammers maintain lists of IP addresses used by search engines and identify Web crawlers based on their matching IPs. These IP lists are easily available online and are frequently updated.

The differences between the Web pages served to the search engine crawler vs the user typically include: (a) making some text on the page invisible (e.g. white-on-white, very small font size), (b) using style-sheets to hide text, (c) using javascript to alter page content when loaded in the Web browser, and (d) use of javascript or "meta-refresh" to redirect the user to another page.

Since anyone can be an author on the Web, cloaking practices naturally create a question of information reliability. Users accustomed to trusting print media (newspapers and books) may not be able, prepared or willing to think critically about the information obtained from the Web [10]. As a result, most Web search engines do not approve of cloaking and will permanently ban such sites from their databases.

In this paper, we investigate the distribution of cloaking based Web spam over two different query categories, namely popularity and monetizability. Popularity of a query is proportional to the frequency of occurrence in the search query logs. Monetizability can be defined to be proportional to the number of user clicks or the amount of revenue generated by user clicks on sponsored ads (paid advertisements) served alongside search results. Most major search engines serve online ads and keep track of their usage statistics. We mine these logs to obtain popularity and monetizability scores for search queries. In this paper, these

AIRWeb'06, August 10, 2006, Seattle, Washington, USA.

scores are used only to extract the top N queries from search and ad logs.

Section 2 presents some background on Web spam, online advertising, and cloaking. We argue that attracting online users to commercial Web sites for the purposes of increasing their monetization is a significant source of Web spam. Query popularity and monetizability are introduced in Section 3 along with strategies for combating Web spam. Sections 4 and 5 present cloaking detection experiments and their results. Section 6 concludes with a brief discussion of the experimental results presented in this paper and potential future work.

2. BACKGROUND AND RELATED WORK

2.1 Adversarial Aspects of Web Spam

One common definition of Web spam [9,11] is: "A Web page created for the sole purpose of attracting search engine referrals (to this page or some other "target" page)." Owing to such a broad definition, classifying a Web page as spam is inherently ambiguous. In many cases, determining whether a Web page is spam (or not) is ultimately a judgment call. For example, some Web pages have very little useful content, are badly formatted, and are borderline useless, but are not spam. Some other pages look fine in isolation, but in context are clearly spam.

Several approaches based on statistical analysis and machine learning have been proposed for detecting spam pages [4,9,12, 14,19-20]. However, none of them are guaranteed to succeed against all spammers. When search engines counteract Web spam using these approaches, they simply escalate the arms race: the approaches work for a short period of time while the spammers move to more successful and newer strategies.

Detecting Web spam is inherently an adversarial problem. Static machine learning based approaches do not do well against such adversarial problems. Spam classifiers need to be updated often or should be capable of learning online. Online learning requires a constant source of labeled data which can be expensive. Web spam is similar to other common adversarial problems such as e-mail spam and computer viruses that are contained but are not likely to be solved in the near future. The best systems continuously monitor performance and frequently update themselves.

2.2 Motivation behind Web Spam

Search engine optimization (SEO) is a legitimate way to improve traffic to commercial sites. The preferred approach is to ensure that the search engine spider can find and index the site and to improve overall site quality by offering added value to online users. Online advertising can be used to further improve traffic, but requires extra capital investment.

Most major search engines sell advertising keywords. Vendors can bid directly on these advertising keywords and have their commercial links served alongside search results. When search engines serve sponsored ads, they are clearly marked so that users can tell them apart from search results easily. Web spammers on the other hand act as intermediaries and sell search ranks for specific queries to businesses [12]. These ranks are achieved through various spamming techniques. Since these commercial sites are ranked inline with the other genuine search engine results, online users cannot tell them apart.

The overall motivation for most Web spamming approaches is monetizability. Simple conversion ratios such as impression-to-click and click-to-sale numbers determine how profitable an online business is. Many businesses can increase business revenue simply by increasing traffic to their site (all else being the same). The difference between white hat and black hat SEOs is mostly a difference of means rather than the ends. However, exceptions do exist. For example, Google bombers[1] [6] may not be completely motivated by money. However, we believe that non-monetary motivations for Web spam are secondary and as a result not as wide spread.

Similar monetizability arguments have been a rich source of robust approaches for fighting e-mail spam [3,7]. Understanding the monetization strategies of common Web spammers and designing approaches that increase their operating costs is a promising approach to combating Web spam.

2.3 Internet Advertising and the Generalized Second Price Auction

The search engine is not only a tool for searching the Web, but also an advertising platform for ones business and services of companies. Search engines sell online advertising through an auction process where advertisers bid for specific keywords and phrases. A brief description of the Generalized Second Price (GSP) auction [8] is presented below:

When a Web user enters a search query into a search engine, he gets back a page with results, containing both the links most relevant to the query and the sponsored links, i.e., paid advertisements. The presentation ensures that ads are clearly distinguishable from the actual search results. Different searches yield different sponsored links. Advertisers target their ads based on query keywords and/or phrases. For instance, if a travel agent buys the word "Hawaii," then each time a user performs a search on this word, a link to the travel agent will appear on the search results page. When a user clicks on the sponsored link, he is sent to the advertiser's Web page. The user click constitutes a referral to the advertiser from the search engine. The advertiser then pays the search engine for referring the user, hence the name—"pay-per-click" pricing.

The number of ads that the search engine can show to a user is limited, and different positions on the search results page have different desirabilities for advertisers. Preliminary eye tracking studies indicate a triangular region (Golden Triangle) of maximum visibility on the search results page [21]. The golden triangle is a right angled triangle aligned along the top of the first search result and the left side of the results page. It extends from the left top of the results page over to the top of the first result, then down to a point on the left side about three quarters of the way down the page. Generally, this area includes top sponsored links, top organic results and alternative results, including shopping, news or local suggestions. An ad shown at the top of a page is more likely to be clicked than an ad shown at the bottom.

[1] Search engines associate the anchor text that is used to link to a page with that page. By referring to target pages with anchor terms that have a negative connotation, malicious sites can cause these targets to become search results for negative query terms [6].

Hence, search engines need a system for allocating the positions to advertisers, and auctions are a natural choice. Currently, the mechanisms most widely used by search engines are based on GSP.

In the simplest GSP auction, for a specific keyword, advertisers submit bids indicating the maximum price they are willing to pay. When a user enters a keyword, he receives search results along with sponsored links, the latter shown in decreasing order of bids. In particular, the ad with the highest bid is displayed at the top, the ad with the next highest bid is displayed in the second position, and so on. If a user subsequently clicks on an ad in position *k*, that advertiser is charged by the search engine an amount equal to the next highest bid, i.e., the bid of an advertiser in position $k + 1$. If a search engine offered only one advertisement per result page, this mechanism would be equivalent to the standard second price, or Vickrey-Clarke-Groves (VCG), auction [13]. With multiple positions available, the GSP generalizes the second price auction (hence the name). Here, a winner pays the next highest bidder's bid. Modified versions of GSP are used by Google AdWords[2], Yahoo Search Marketing (SM)[3] and MSN AdCenter[4]. For example, one common modification is to combine the advertisers bid price with the expected click-through-rate (CTR) to compute an expected monetization score. Sponsored links are presented in decreasing order of expected monetization.

2.4 Semantic and Syntactic Cloaking

Cloaking behavior that is aimed at manipulating the search engine is defined as semantic cloaking [19]. The exact definition of semantic cloaking varies from search engine to search engine. On the other hand, syntactic cloaking is a simpler and more basic variant of cloaking. Syntactic cloaking implies that different content is served to automated crawlers vs Web browsers, but not different content to every visitor. Dynamic Web pages that serve different pages to every visitor would not be syntactically cloaking, but could be semantically cloaking. In this paper, our operating definition for cloaking is more than just syntactic cloaking. Syntactic cloaking is definitely cloaking, but dynamic Web pages are also addressed to some extent (see Section 6).

3. POPULARITY AND MONETIZABILITY

Monitoring, evaluating, and understanding user behavior and preferences is crucial for search engine development, deployment, and maintenance. Search engines model and interpret user behavior to improve ranking, click spam detection, Web search personalization, and other tasks [1,2,17]. Further, for billing and reporting purposes every impression, user click, and referral relating to each sponsored link are also logged. We propose mining these logs to determine query popularity and monetizability. Such query categorization has been valuable for improving collaborative Web search [15-17].

[2] http://adwords.google.com/select/

[3] http://searchmarketing.yahoo.com/

[4] http://adcenter.msn.com/

3.1 Query Popularity

We define the popularity of a query to be proportional to the number of times it occurs in the query logs during a specific time period. Using this definition one can compute query lists such as the top 10 popular search queries for a day, a month, or even a year. Most major search engines publish these results online at different granularities. Table 1 presents a list of common sources of popular queries. The list of top 5000 most popular queries was computed from MSN Search query logs. In this paper, we examine the cloaking properties of search results from these top 5000 popular queries from Google[5], MSN Search[6], and Ask.com[7].

Table 1. Common sources of popular queries

Engine	URL
Google Zeitgeist	http://www.google.com/press/zeitgeist.html
Yahoo Buzz Index	http://buzz.yahoo.com/
MSN Search Insider	http://www.imagine-msn.com/insider/
Ask.com IQ	http://sp.ask.com/en/docs/iq/iq.shtml
AOL Hot Searches	http://hotsearches.aol.com/search/hotsearch.jsp
Dogpile Search Spy	http://www.dogpile.com/info.dogpl/searchspy/
Lycos 50	http://50.lycos.com/

3.2 Query Monetizability

Computing the monetizability of a query is not as straight forward as computing its popularity. Advertisers can bid for a single keyword, a keyword and additional search terms, or a phrase. The bidding process can be blind or open, i.e., each bidder's bid price and identity may or may not be disclosed to other bidders[8]. Three different types of matches are typically possible: broad match, phrase match, and exact match. Some providers support negative or excluded keywords also. The advertiser also picks the type of matching done between the user search query and the bids. A broad match occurs when the user query contains all of the keywords (in any order). Bid keywords may be expanded to include plurals and relevant variations. Phrase match occurs when all bid keywords occur in the prescribed order in the search query. Both broad and phrase matches allow extraneous query words. Exact matching occurs only when the search query matches the bid phrase exactly. No extraneous terms are allowed. The occurrence of negative or excluded keywords in the search query suppresses any matching. The matching sponsored links are ranked based on relevance, monetizability (combination of bid price and CTR), and other factors.

In this paper, we define the monetizabilty of a specific query to be proportional to the total revenue generated by sponsored ads

[5] http://www.google.com

[6] http://search.msn.com

[7] http://www.ask.com

[8] For example, Yahoo SM auctions are not blind while Google's AdWords and MSN's AdCenter's auctions are blind. Further, not only the bid-price but the bidding advertiser might be disclosed during these auctions.

served along side the search results (for that query) during a specific time period. The list of top 5000 most monetization queries over a single day were computed from MSN Search's advertisement logs. Note that the ad logs are used only to obtain the top 5000 monetizable queries and their ranks. For simplicity, we do not use their monetization scores. For each of these top 5000 monetizable queries, we examine the cloaking properties of the resulting top 200 search results from Google, MSN Search, and Ask.com.

4. DATA SETS

4.1 Query Data Sets

We use two lists of 5000 queries each in the experiments. The first list is the set of the top 5000 most popular search queries computed over one month. The second list is the set of the top 5000 most monetizable search queries over one day. The former was obtained by processing search query logs, while the latter was obtained by processing ad logs. Both logs were obtained from the MSN search engine. 826 queries (17%) were the same between the two lists.

4.2 URL Data Sets

For each query, the top 200 search results were obtained from three search engines: Google, MSN Search, and Ask.com. On every search engine, each unique query was looked up only once. Each query produced 600 search result URLs which typically contain several duplicates. Each set of 5000 queries generated 3 million URLs. Overall, the 5000 popular queries generated 1.49 million unique URLs (popular set), and the top 5000 monetizable queries generated 1.28 million unique URLs (monetizable set). Each unique URL was processed only once.

Current search engines already employ numerous but unknown anti-spam mechanisms. In our analysis, we assume that such Web spam filtering/removal techniques are uniformly applied to the 200 search results and the 5000 queries. This is likely the case for automated filtering that is applied to the whole index, for example during the crawling phase. Manual Web spam removal is one special case where this assumption may not be invalid. Given its expense, manual filtering is likely to be limited to the top few (top 10 or top 20) search results for the most popular queries. Since we are looking at the top 200 results for the top 5000 queries, we conjecture that the impact of filtering on the overall results reported in this paper is small.

5. CLOAKING DETECTION RESULTS

We use a modified version of the syntactic cloaking detection algorithm from [19]. For each URL, up to four copies of the Web page, denoted by C_1, B_1, C_2, and B_2, are downloaded and compared. There are several stages where an early out is possible making the modified procedure more efficient. During the download process, many of the non-cloaked pages are detected through simple HTML string comparisons, HTML to text conversion, and text string comparisons. Normalized term frequency difference (NTFD) is subsequently used to compute a cloaking score and used to further reduce the set of possibly cloaked URLs. Finally, using labeled data, a threshold for the cloaking score is chosen to classify remaining URLs. A flow chart depicting the different stages is presented in Figure 1.

5.1.1 Downloading Web Pages

The first copy of the URL (C_1) was obtained by mimicking a popular Web crawler (MSNBot) and the second (B_1) was obtained using a common Web browser's (Internet Explorer) agent string. The *user-agent* strings for MSNBot and Internet Explorer were set to those given in Section 1. These first and second copies were checked for identical HTML content (simple string comparison). If they were identical, the URL was marked as not cloaked. About 70–75% of the URLs fell under this category. The HTML content for the remaining 25–30% was converted to plain text and directly compared (simple string comparison). At this stage, about 13.5% of the URLs produce identical text streams and are marked as not-cloaked. The text streams are tokenized (using white space) and their term frequencies are computed. About 0.5% of the URLs produce identical term frequencies. The remaining URLs (about 12%) with differing text content were downloaded two more times to obtain a third (MSNBot, C_2) and a fourth (Internet Explorer, B_2) copy. These were then converted to text and their term frequencies calculated. Note that at the end of the download process those URLs with only (C_1, B_1) pair of pages are not-cloaked (by definition). The remaining URLs have four copies (C_1, B_1, C_2, and B_2) and need further processing.

Each of the copies (C_1, B_1, C_2, and B_2) was asynchronously crawled using different crawler threads. For example, all C_1 copies were crawled by the first crawler thread. Similarly, all B_1, C_2, and B_2 copies were crawled by the first browser thread, the second crawler thread, and the second browser thread, respectively. The ordering of initiating URLs downloads was the same for all four threads (with the exception of early out scenarios where URLs were skipped by the C_2, and B_2 threads).

In the event of a download failure, the download was reattempted once. URLs that failed download twice were dropped from analysis. For both the popular and monetizable query URL sets, less than 3% of the URLs failed to download. Overall, on average of about 2.1 downloads are done per unique URL.

5.1.2 Normalized Term Frequency Difference

A simple normalized term frequency difference (NTFD) between the four copies was used in computing a cloaking score. Let T_1 and T_2 be sets of terms from two Web pages after conversion and tokenization. Note that T_1 and T_2 may contain repeats. The normalized term frequency difference is computed as

$$D(T_1, T_2) = \frac{\left|(T_1 \setminus T_2) \cup (T_2 \setminus T_1)\right|}{\left|(T_1 \cup T_2)\right|} = 1 - 2\frac{\left|(T_1 \cap T_2)\right|}{\left|(T_1 \cup T_2)\right|}$$

Where |.| is the set cardinality operator and all set operations are extended to work with sets with repeated terms. ($T_1 \setminus T_2$) is the set of terms in the first page but not in the second page, ($T_2 \setminus T_1$) is the set of terms in the second page but not in the first page, and ($T_1 \cup T_2$) is the aggregation of terms in both pages. The normalization by the ($T_1 \cup T_2$) term reduces any bias that stems from the size of the Web page. The NTFD score for any pair of Web pages lies in [0,1]. In essence, for the same $D(T_1,T_2)$, value, larger Web pages are allowed to have more terms that are different between the two pages. We note that the normalized term frequency difference is symmetric, i.e.,

$$D(T_1, T_2) = D(T_2, T_1)$$

The above term-based page-difference score is quite simple and disregards the semantic and layout structure of page content. Further, all sections of the Web page (navigation, header, footer, advertisements, etc) are treated equally[9].

We note that this score differs significantly from that proposed in [19]. Instead of using the cardinality of all the terms in the web pages, a "bag of words" method is used in [19] for analyzing the Web pages. They parse the HTML into terms and only count each unique term once no matter how many times this term appears. Further, they do not normalize the term set difference which could potentially bias the score against large Web pages.

5.1.3 Cloaking Test

As described in Section 5.1.1, many of the URLs are marked as non-cloaking during the download process itself. The remaining URLs end up with four downloaded versions (C_1, B_1, C_2, and B_2). The NTFD score for these four Web page versions is used to obtain a cloaking score, S, given by

$$S = \frac{\Delta_D}{\Delta_S}$$

Where Δ_D is the smaller of the NTFD values for the two cross-pairs of Web pages (C_1,B_1) and (C_2,B_2), and Δ_S is the larger of the NTFD values for the two similar-pairs of Web pages (C_1,C_2) and (B_1,B_2). Mathematically,

$$\Delta_D = \min(D(C_1, B_1), D(C_2, B_2))$$

$$\Delta_S = \max(D(C_1, C_2), D(B_1, B_2))$$

The simple divide-by-zero cases are resolved as follows: (a) If $\Delta_S = 0$ and $\Delta_D = 0$, the URL is marked as non-cloaked ($S = 0$), (b) If $\Delta_S = 0$ and $\Delta_D > 0$, the URL is marked as cloaked ($S = \infty$). At this stage all of the dynamic Web pages are identified using:

$$0 < S < \infty \Rightarrow \text{ dynamic URLs}$$

A subsequent threshold test is used to find cloaked pages:

$$0 < t < S \Rightarrow \text{ cloaking spam}$$

For each of the URL sets (popular and monetizable) 2000 URLs were randomly sampled from the set of dynamic URLs and manually labeled as spam or no-spam.

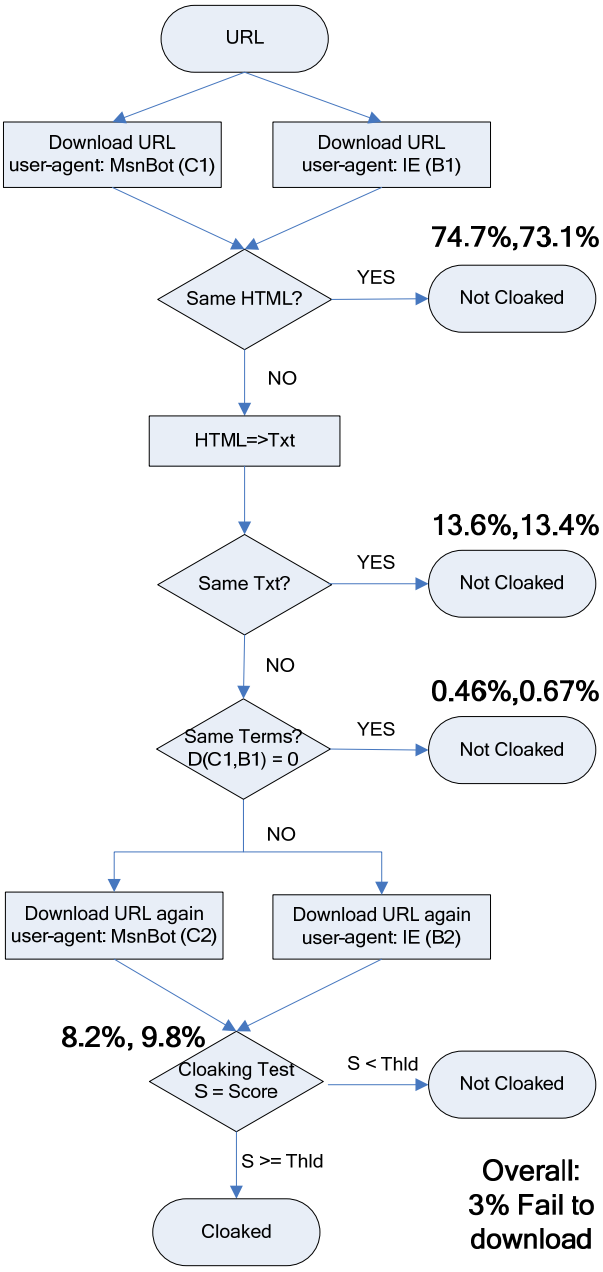


Figure 1. Cloaking detection procedure. The pair of percentages indicate number of classified / unclassified URLs at each stage for the popular and monetizable URLs, respectively.

[9] We plan to explore more advanced methods for computing page difference scores based on page and link content in future work.

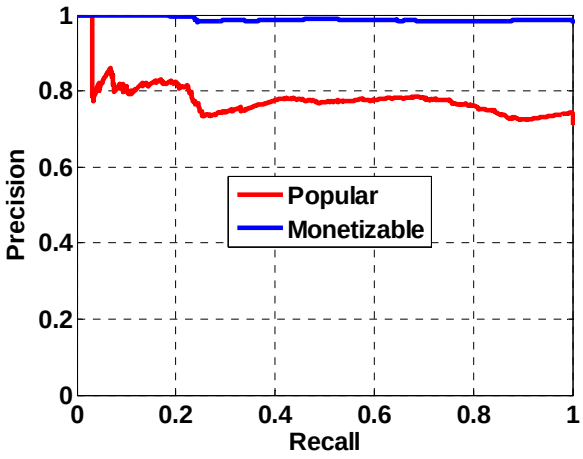


Figure 2. Precision-Recall curve for popular and monetizable URL sets as a function of the cloaking score threshold, *t*.

Figure 2 shows the precision-recall curve for various values of the threshold *t*. The precision and recall values and their associated thresholds are also presented in Table 2. As the value of *t* increases, recall gradually decreases. Precision starts out high at low values of recall and quickly reaches a final value around 75% for popular URLs and a value of 98.5% for monetizable URLs.

All three commonly used F–measures: F_1, $F_{0.5}$, and F_2, reach the highest value at a threshold of 0.0, where the recall is 100% and the precision is 73.12% and 98.54% for popular and monetizable URLs, respectively. This clearly indicates that the cloaking score is a very good indicator of cloaking spam. A threshold of 0 implies that all pages marked as dynamic using

$$0 < S < \infty \Rightarrow \text{dynamic URLs}$$

can be classified as cloaking spam. Overall, we estimate that 5.99% (=8.2*0.731) of popular query results and 9.66% (=9.8*0.985) of all monetizable queries employ cloaking spam.

Table 2. Precision, Recall, and Thresholds for classifying URLs as cloaking spam using their cloaking score

Recall	Precision (threshold, *t*)	
	Popular URLs	Monetizable URLs
10	85.74 (19.93)	100.00 (15.11)
20	81.72 (1.98)	99.91 (1.28)
30	75.33 (1.10)	98.77 (0.97)
40	76.65 (0.94)	98.56 (0.87)
50	77.39 (0.78)	98.79 (0.77)
60	77.81 (0.53)	98.72 (0.56)
70	77.88 (0.27)	98.59 (0.32)
80	75.86 (0.11)	98.34 (0.07)
90	73.26 (0.02)	98.46 (0.004)
100	73.12 (0.00)	98.54 (0.000)

Note that the above percentages are mean values over all 5000 queries. Figure 3 shows the distribution of cloaking spam URLs over different queries. Both popular and monetizable queries were *independently* sorted such that the percentage curves are monotonically decreasing with increasing sorted query rank. Note that these two query sets are not the same. They have only 17% of the queries in common. We note that, on average, the top 100 (2%) most cloaked queries have 10x as many cloaking URLs in their search results than the bottom 4900 queries (98%). This skewed distribution gives an effective way of monitoring and detecting cloaked URLs. By starting with the most cloaked queries once can efficiently and quickly identify cloaked URLs.

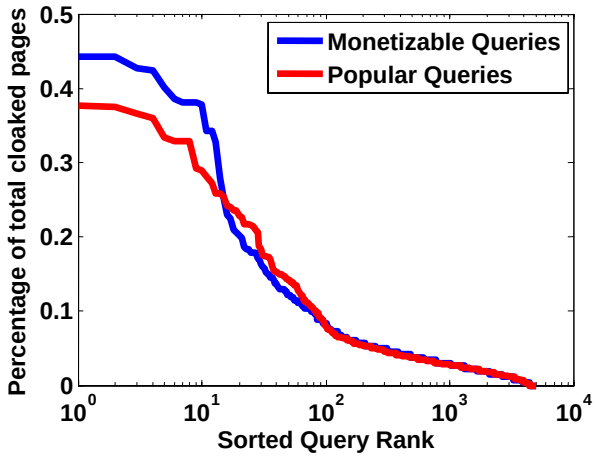


Figure 3. The distribution of cloaking spam URLs over different queries. Both popular and monetizable queries were independently sorted such that the percentage curves are monotonically decreasing with increasing sorted query rank.

6. DISCUSSION AND FUTURE WORK

Cloaking is a search engine spamming technique that reduces the reliability of Web page information that is widely accessible through the search engine. Web sites that deliver one page to a search engine for indexing while serving an entirely different page to users browsing the site inherently hurt the search engine's credibility and waste internet users' time.

In this paper, we showed that the degree of cloaking among search results depends on query properties such as popularity and monetizability. Query popularity and monetizability were estimated based on whether a given query belonged to the popular set of URLs or monetizable set of URLs (or both). We also presented a new cloaking detection algorithm based on normalized term frequency difference scores and demonstrated its effectiveness in identifying cloaking spam pages on a dataset of 3 million URLs obtained using 10,000 search queries.

The proposed cloaking detection algorithm has a very high accuracy in detecting cloaked spam pages in monetizable query results. Moderate accuracy is also achieved for popular queries. By combining a matching model (query ⇔ bid keywords) similar to that used in for serving online advertisements with search

query and advertising logs, one can estimate the popularity and monetizability of arbitrary queries. Such estimates may be valuable in prioritizing URLs to be tested for cloaking spam. We hope to pursue these ideas in our future work.

7. ACKNOWLEDGMENTS

We would like to thank Ahmad Abdulkader and Chris Meek for very useful and productive discussions on Cloaking detection. We would also like to thank Chau Luu for help with our spam labeling efforts.

8. REFERENCES

[1] E. Agichtein, E. Brill, S. Dumais, R. Ragno (2006), "Learning User Interaction Models for Predicting Web Search Result Preferences," To appear in SIGIR'2006: 29th Annual International ACM SIGIR Conference on Research & Development on Information Retrieval, Seattle.

[2] E. Agichtein, E. Brill, S. Dumais (2006), "Improving Web Search Ranking by Incorporating User Behavior," To appear in SIGIR'2006: 29th Annual International ACM SIGIR Conference on Research & Development on Information Retrieval, Seattle.

[3] A. Back (2002), "Hash cash - a denial of service counter-measure," Technical. Report. Available at: http://citeseer.ist.psu.edu/back02hashcash.html

[4] A. Benczúr, K. Csalogány, T. Sarlós and M. Uher (2005), "SpamRank – Fully Automatic Link Spam Detection," In 1st International Workshop on Adversarial Information Retrieval on the Web, May 2005.

[5] B. Davison (2000), "Recognizing Nepotistic Links on the Web," In AAAI-2000 Workshop on Artificial Intelligence for Web Search, July 2000.

[6] I. Drost and T. Scheffer (2005), "Thwarting the negritude ultramarine: Learning to identify link spam." In Proceedings of European Conference on Machine Learning, pages 96-107, Oct. 2005.

[7] C. Dwork, A. Goldberg, and M. Naor (2003), "On Memory-Bound Functions for Fighting Spam," Proceedings of the 23rd Annual International Cryptology Conference (CRYPTO 2003), pages 426-444, Santa Barbara, CA, August 2003.

[8] B. Edelman, M. Ostrovsky, and M. Schwarz (2005), "Internet Advertising and the Generalized Second Price Auction: Selling Billions of Dollars Worth of Keywords," November 2005, NBER Working Paper No. W11765 Available at SSRN: http://ssrn.com/abstract=847037

[9] D. Fetterly, M. Manasse, and M. Najork (2004), "Spam, damn spam, and statistics: Using statistical analysis to locate spam Web pages," In Proceedings of WebDB, pages 1-6, June 2004.

[10] L. Graham and P. T. Metaxas (2003). "Of course it's true; I saw it on the internet!": Critical thinking in the internet era. Commun. ACM, 46(5): 70–75, 2003.

[11] Z. Gyöngyi and H. Garcia-Molina (2005), "Web spam taxonomy," In First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb'05), Chiba, Japan, 2005.

[12] Z. Gyöngyi, H. Garcia-Molina and J. Pedersen, "Combating Web Spam with TrustRank," In 30th International Conference on Very Large Data Bases, Aug. 2004.

[13] V. Krishna (2002). *Auction Theory*, Academic Press, 2002.

[14] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly (2006), "Detecting Spam Web Pages through Content Analysis," In Proceedings of the World Wide Web Conference 2006 (WWW'06). pp. 83-92, Edinburgh, United Kingdom, May 23-26, 2006.

[15] D. Shen, J-T. Sun, Q. Yang, Z. Chen (2006), "Building Bridges for Web Query Classification," In Proceedings of the 29th ACM International Conference on Research and Development in Information Retrieval (SIGIR'06). Seattle, USA, August 6-11, 2006.

[16] D. Shen, R. Pan, J-T. Sun, J. J. Pan, K. Wu, J. Yin and Q. Yang (2005), "Q2C@UST: Our Winning Solution to Query Classification in KDD Cup 2005," SIGKDD Explorations. Volume 7, Issue 2, December 2005.

[17] C. Silverstein, H. Marais, M. Henzinger, and M. Moricz (1999), "Analysis of a Very Large Web Search Engine Query Log. SIGIR Forum," 33(1):6--12, 1999.

[18] J-T. Sun, X. Wang, D. Shen, H-J. Zeng, Z. Chen (2006), "Mining Clickthrough Data for Collaborative Web Search," In Proceedings of the World Wide Web Conference 2006 (WWW'06). pp. 947-948, Edinburgh, United Kingdom, May 23-26, 2006.

[19] B. Wu and B. D. Davison (2005) "Cloaking and Redirection: A Preliminary Study," In Proceedings of AIRWeb'05, May 10, 2005, Chiba, Japan.

[20] Baoning Wu and Brian D. Davison. (2006), "Detecting Semantic Cloaking on the Web," Accepted in 15th International World Wide Web Conference, Industrial Track, Edinburgh, Scotland, May 22-26, 2006.

[21] "Did-it, Enquiro, and Eyetools Uncover Google's Golden Triangle: *New Eye Tracking Study verifies the importance of page position and rank in both Organic and PPC search results for visibility and click through in Google*." Available at http://www.prweb.com/releases/2005/3/prweb213516.htm

Tracking Web Spam with Hidden Style Similarity

Tanguy Urvoy, Thomas Lavergne, Pascal Filoche

France Telecom R&D*

{tanguy.urvoy,thomas.lavergne,pascal.filoche}@orange-ft.com

ABSTRACT

Automatically generated content is ubiquitous in the web: dynamic sites built using the three-tier paradigm are good examples (e.g. commercial sites, blogs and other sites powered by a web authoring software), as well as less legitimous spamdexing attempts (e.g. link farms, faked directories...).

Those pages built using the same generating method (template or script) share a common "look and feel" that is not easily detected by common text classification methods, but is more related to stylometry.

In this paper, we present a (hidden) style similarity measure based on extra-textual features in HTML source code. We also describe a method to clusterize a large collection of documents according to this measure. The clustering algorithm being based on fingerprints, we also give some recalls about fingerprinting.

By conveniently sorting the generated clusters, one can efficiently track back instances of a particular automatic content generation method among web pages collected using a crawler. This is particularly useful to detect pages across different sites sharing the same design — this is often a good hint of either spamdexing attempt or mirrored content.

1. INTRODUCTION

Automatically generated content is nowadays ubiquitous on the web, especially with the advent of professional web sites and popular three-tier architectures such as "LAMP" (Linux Apache Mysql Php). Generation of these pages using such architecture involves:

- a scripting component;
- a page template ("skeleton" of the site pages);
- content (e.g. product catalog, articles repository...), usually stored in databases.

When summoned, the scripting component combines the page template with information from the database to generate an HTML page, having no difference with a static HTML page from a robot crawler point of view (shall robots have point of view).

*Thomas Lavergne also ENST Paris

AIRWEB'06, August 10, 2006, Seattle, Washington, USA.

1.1 Spamdexing and Generated Content

By analogy with e-mail spam, the word *spamdexing* designates the techniques used to reach a web site to a higher-than-deserved rank in search engines response lists. For instance, one well known strategy to mislead search engines ranking algorithms consists of generating a maze of fake web pages called *link farm*.

Apart from the common dynamic web sites practice, the ability to automatically generate a large amount of web pages is also appealing to web spammers. Indeed [3] points out that *"the only way to effectively create a very large number of spam pages is to generate them automatically"*.

When those pages are all hosted under a few domains, the detection of those domains can be a sufficient countermeasure for a search engine, but this is not an option when the link farm spans hundreds or thousands of different hosts — for instance using word stuffed new domain names, or buying expired ones [6].

One would like to be able to detect all pages generated using the same method once a spam page is detected in a particular search engine response list. One direct application of such a process would be to enhance the efficiency of search engines blacklist databases by "spreading" detected spam information to find affiliate domains (following the philosophy of [7]).

1.2 Detecting Generated Pages

We see the problem of spam detection in a search engine back office process as two-fold:

- detecting new instances of already encountered spam (through editorial review or automatic methods);
- pinpointing dubious sets of pages in a large uncategorised corpus.

The first side of the problem relates to supervised classification and textual similarity, while the second is more of the unsupervised clustering kind.

1.2.1 Detecting Similarity With Known Spam

Text similarity detection usually involves word-based features, such as in e-mail Bayesian filtering. This is not always relevant in our case, because though those pages share the same generation method, they rarely share the same vocabulary [15] (apart from the web spam specifically involving adult content) — hence using common text filtering methods with this kind of web spam would miss a lot of positive instances. For example exiled presidents and energising sex drugs are recurrent topics in e-mail spam, but link farm

automatically generated pages tend to rather use large dictionary in order to span a lot of different possible requests [6].

To detect similarity based on pages generation method, one needs to use features more closely related to the internal structure of the HTML document. For instance, [13] proposed to use HTML specific features along with text and word statistics to build a classifier for genre of web documents.

1.2.2 Stylometry and HTML

In fact, what best describes the relationship between those pages generated using the same template or method seems to be more on a *style* ground than a *topical* one. This would relate our problem with the *stylometry* area. Up to now, stylometry was more generally associated with authorship identification, to deal with problems such as attributing plays to the right Shakespeare, or to detect computer software plagiarism [5]. Usual metrics in stylometry are mainly based on word counts [12], but also sometimes non-alphabetic features such as punctuation. In the area of web spam detection, [15] and [11] propose to use lexicometric features to classify the part of web spam that does not follow regular language metrics.

1.2.3 Overview of This Paper

We first give some recalls about similarity, fingerprints and clustering in section 2. We then detail the specificities of the *"Hidden Style Similarity"* algorithm in section 3. The experimental results are described in section 4.

2. SIMILARITY AND CLUSTERING

2.1 Similarity Measure

The first step before comparing documents is to extract their (interesting) content: this is what we call *preprocessing*. The second step is to transform this content into a model suitable for comparison (except for string edition based distances like Levenshtein and its derivatives where this intermediate model is not mandatory). For frequencies based distances the second step consists of splitting up the documents into multi-sets of parts (frequencies vectors). For set intersection based distances, the split is done into sets of parts. Depending on the granularity expected, these parts may be sequences of letters (n-grams), words, sequences of words, sentences or paragraphs. The parts may overlap or not.

There are many flavors of similarity measure [14, 16]. The most used measure in stylometry is the Jaccard similarity index. For two sets of parts D_1, D_2:

$$Jaccard(D_1, D_2) = \frac{|D_1 \cap D_2|}{|D_1 \cup D_2|} \ .$$

Variants may be used for the normalizing factor, such as in the Dice index:

$$Dice(D_1, D_2) = 2 \cdot \frac{|D_1 \cap D_2|}{|D_1| + |D_2|} \ .$$

Whatever kind of normalisation used for $|D_1 \cap D_2|$, the most important ingredients for the quality of comparison are the preprocessing step and the parts granularity.

The one-to-one calculus of similarities is interesting for fine comparison into a small set of documents, but the quadratic explosion induced by such a brute-force approach is unacceptable at the scale of a web search engine. To circumvent this explosion, more tricky methods are required. These methods are described in the next three sections.

2.2 Fingerprints

The technique of documents fingerprinting and its application for similarity clustering is a kind of *locality sensitive hashing* [9]. We mainly based our work on the papers [8], [2] and [1], where the pratical use of minsampling (instead of random sampling) and its leverage effect on similarity estimation is well described. We also noticed a phrase level use of fingerprints to track search engine spam in [4].

2.2.1 Minsampling

Each document is split up into parts. Let us call P the set of all possible parts. The main principle of *minsampling* over P is to fix at random a linear ordering on P (call it $\prec$) and represent each document $D \subseteq P$ by its m lowest elements according to $\prec$ (we denote this set $Min_{\prec,m}(D)$).

If $\prec$ is chosen at random over all permutations over P then for two random documents $D_1, D_2 \subseteq P$ and for m growing, it is shown in [1] that

$$\frac{|Min_{\prec,m}(D_1) \cap Min_{\prec,m}(D_2) \cap Min_{\prec,m}(D_1 \cup D_2)|}{|Min_{\prec,m}(D_1 \cup D_2)|}$$

is a non biased estimator of $Jaccard(D_1, D_2)$.

2.2.2 Fingerprints and Index Storage

With this fingerprinting method, the fingerprint of a document D is stored as a sorted list of m integers (which are hashing keys of the elements of $Min_{\prec,m}(D)$). The experiments of [8] show that a value around $m = 100$ is reasonable for the detection of similar documents in a large database. The drawback of this method it that it does not provide a real vector space structure to the fingerprints. A vector space structure is more convenient, for instance to build indexes in a database to later fetch near duplicates of a particular document.

2.2.3 Optimization of Minsampling

An improvement of the model is to use m independent linear orderings over P, let us call them $\prec_i$ for $i \in [m]$, and use these ordering to select one minimun element by ordering. The result is a real vector of m independant integers and the similarity measure becomes:

$$Sim_a(D_1, D_2) = \frac{\sum_{i=0}^{m} |Min_{\prec_i}(D_1) \cap Min_{\prec_i}(D_2)|}{m}$$

which is a correct estimator of the Dice similarity.

2.3 Clustering

When working on large volume of documents, one would like to group together the documents which are similar enough according to the chosen similarity. This is especially useful when no specific paragon to look for is known in advance.

If D is the set of documents, we want to compute a mapping $Cluster : D \to D$ associating to each element x its class representative $Cluster(x)$, with $Cluster(x) = Cluster(y)$ if and only if $sim(x, y)$ is lower than a given threshold.

2.3.1 Clustering with Fingerprints

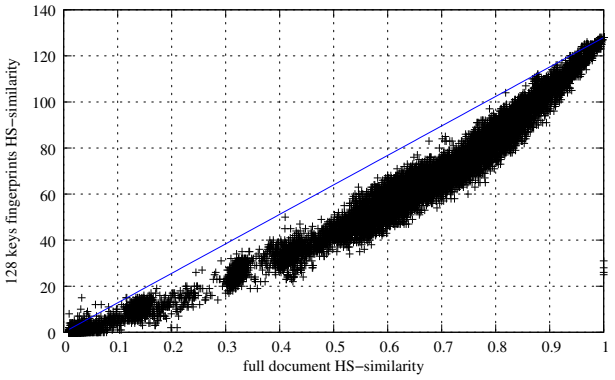


Figure 1: The rate of matched dimensions according to the full document HS-similarity (one-to-one comparison between 10000 HTML files).

The first benefit of using fingerprints is to reduce the size of documents representatives, allowing to perform all computation in memory. As shown in figure 1, this reduction by sampling is at the cost of a little loss of quality in the similarity estimation.

Another important benefit of fingerprints is to give a low dimension representation of documents. It becomes possible to compare only the documents that match at least on some dimensions. This is a way to build the sparse similarity matrix with some control over the quadratic explosion induced by the biggest clusters [2].

3. THE HSS ALGORITHM

To capture similarity based on pages generation method, we propose to use a specific document preprocessing excluding all alpha-numeric characters, and keeping into account the remaining characters through the use of n-grams. By analysing usually neglected features of HTML texts like extra-spaces, lines feed or tags, we are able to modelize the "style" of HTML documents. This model enables us to compare and group together documents sharing many hidden features.

We consider both the one-to-one full document HS-similarity and the global HS-clustering of several documents. These two aspects of the HSS algorithm are described in figure 2.

As a side effect, the algorithm is efficient to characterize HTML documents coming from the same web site without information about the host or URL, but the most interesting results are similarity classes containing pages across many differents domains yet with a high HS-similarity.

3.1 Preprocessing

The usual setup procedure to compare two documents is to first remove everything that do not reflect their content. In the case of HTML documents, this preprocessing step may include removing of tags, of extra spaces and of stop words. It may also include normalization of words by capitalization or stemming.

The originality of our approach is to do exactly the opposite: we keep only the "noisy" parts of HTML documents by removing any alphanumeric character. For example, applying such a preprocessing to a relatively human readable HTML code like:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xht...
```

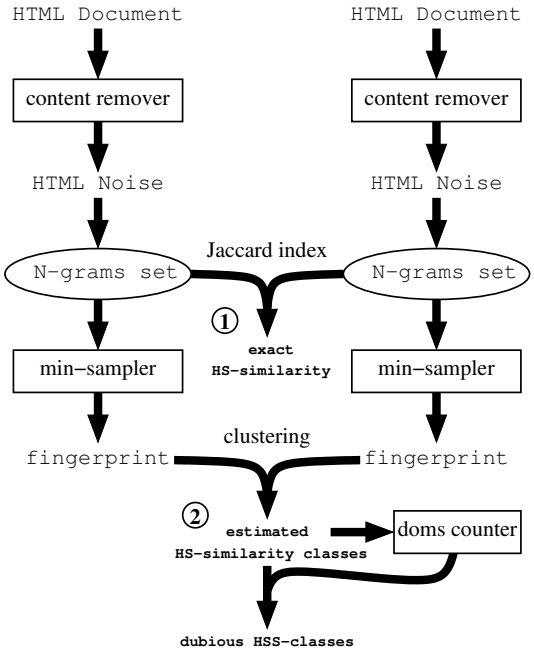


Figure 2: The HSS algorithm: step (1) describes one-to-one full document HS-similarity computation, step (2) describes large scale similarity classes calculus and link farm detection.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en"><head><title>Th...

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="Keywords" content="GNU, FSF, Free Software Foundation, Linux, Em...
<meta http-equiv="Description" content="Since 1983, developing the free UNIX style...
<link rev="made" href="mailto:webmasters@gnu.org">
<link rel="stylesheet" type="text/css" href="gnu_fichiers/gnu.css">
```

gives an HTML noise such as:

```
<!   "-////  . //" "://..////-.">
< ="://..//" :="" =""><><>   </>

< -="-" ="/; =-">
< -="" =", ,   , , , , ,  , , , ,  ">
< -="" =" ,        ,              .">
< ="" =":.">
< ="" ="/" ="/.">
```

Using non-alphanumeric characters – in the case of standard text, these are punctuation signs – as features to classify text is not completely unusual ([10], [13]), though most use it only as a complementary hint. Since HTML syntax includes a lot of non-alphanumeric characters, they happen to be very relevant in our case. Because it is straightforward, this filtering of HTML text is also extremly fastly computed.

3.2 Similarity

For scalability reasons, we chose to use a set intersection based distance with overlaping n-grams on the preprocessing output as parts. A simple one-to-one style similarity measure can then be computed using formula from 2.1 But as said earlier, using this one-to-one similarity measure doesn't fit for large scale clustering. Using fingerprints on the preprocessing output is required to address the scalability issue.

3.3 Fingerprints

We chose to use minsampling with m independant orderings, with another (speedup) improvement which consists of using a *pre-hashing* function to select in advance which dimensions of the final vector are concerned by a given part

of the document. Formally, if $(C_0, \ldots, C_{m-1})$ is the partition of P induced by the pre-hashing function, we have the following similarity measure:

$$Sim_b(D_1, D_2) = \frac{\sum_{i=0}^{m} |Min_{\prec_i}(D_1 \cap C_i) \cap Min_{\prec_i}(D_2 \cap C_i)|}{m}$$

This pre-hashing avoids the heavy calculus of m linear orderings for each considered part, and also avoids to fill two dimensions of the vector with the same part of a document. The drawback is that small documents may not contain enough parts to fill all dimensions of their fingerprint vector.

We chose to ignore these empty dimensions in the counting of matched dimensions, thus lowering drastically the similarity estimation for small documents. This side effect is not critical, HS-similarity diagnostic being by essence unreliable for small documents. Figure 1, shows a comparison between exact Dice measure and Sim_b measure on fingerprints (with $m = 128$).

3.3.1 Implementation

Each document is preprocessed on the fly. The parts we used are overlapping n-grams hashed into 64 bits integers. To compute the m independent orderings $\prec_i$, we precompute m permutations $\sigma_i : [2^{64}] \rightarrow [2^{64}]$ and compare the permuted values:

$$x \prec_i y \Leftrightarrow \sigma_i(x) < \sigma_i(y)$$

To compute these permutations, we use a subfamily of permutations of the form $\sigma_i = \sigma_i^1 \circ \sigma_i^2$ where σ_i^1 is a bits shuffle and $\sigma_i^2(x)$ is an exclusive OR mask. After having initialized every dimension of the fingerprint vector $V \in \mathbb{N}^m$ to ∞, we evaluate each n-gram of the HTML noise with Procedure 1.

Procedure 1 Insert a string s by minsampling into a fingerprint $V \in \mathbb{N}^m$.

Require: $m > 0$ and V initialized
  $h := preHash(s)$
  $i := h \mod m$
  $h' := \sigma_i(h)$
  if $h' < V[i]$ **then**
    $V[i] := h'$
  end if

3.4 Clustering

We used a variant from the algorithm described in [2]. It does not build the entire *similarity graph* and uses a probing heuristic to find potentially similar pairs.

3.4.1 Using Quasi-Transitivity on Similarity Matrix

By thresholding the similarity matrix (cf. 2.3.1), we obtain a symetric relation (let us call it *similarity graph*):

$$S = \{(x, y) \in D \times D \mid sim(x, y) < threshold\}$$

Similarity graphs are characterized by their *quasi-transitivity* property: if xSy and ySz then there is a high probability that xSz. In other words, the connected components of these graphs are almost equivalence classes. This *quasi-transitivity* is helpful to accelerate the clustering process. If the relation is transitive enough, any element may be used as reference to decide if other elements are in the same class. In practice, though building the full similarity graph is too expensive, the noise induced by sampling raises some interest in using a bit of redundancy to improve the robustness of the process (cf. figure 3).

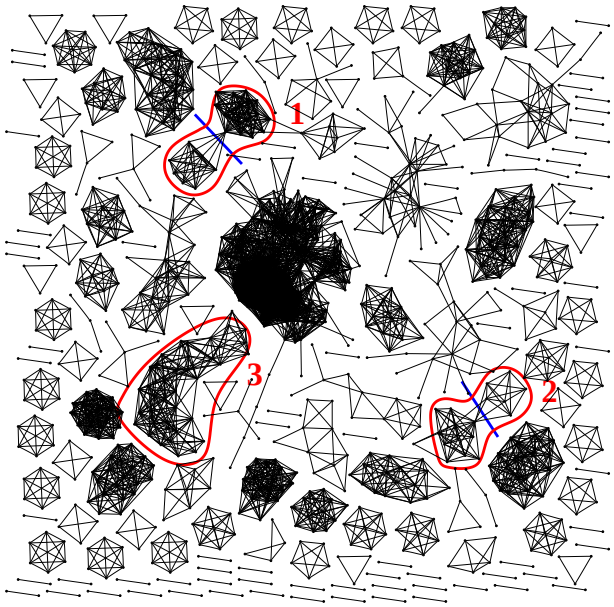


Figure 3: The *quasi-transitivity* of estimated HS-similarity relation is well illustrated by this full similarity graph (realized from 3000 HTML files). With a transitive relation, each connected component would be a clique. For "bunch of grapes" components like (1) and (2), a clear cut may be done but for "worms" components like (3) the similarity diagnostic is less clear.

To approximate the clustering map $Cluster$, we use an algorithm controlled by two parameters: a probe threshold p and a check threshold t. Depending of the aggressiveness of hashing, it may be useful to group fingerprint keys. This introduces a new parameter k to define the size of these groups. We first generate p random subsets of size k in $[m]$. By projecting and indexing fingerprint vectors p times according to these k-subsets, we probe for potential similar pairs which are then checked according to t (See Procedure 2).

Procedure 2 Search HS-similarity classes

Require: $0 < k, p, t \leq m$
  init similarity graph;
  for $i := 0$ to p **do**
    pick a k-subset $s \subseteq [m]$;
    for all pairs (x, y) of fingerprints matching according to s **do**
      if $Sim_b(x, y) > t$ **then**
        add edge (x, y) to similarity graph;
      end if
    end for
  end for
  compute $Clusters$ from connected components of the graph;

3.4.2 Setting Up of Parameters

A good way to choose parameters p, t and k is to make sure that the probability to miss a pair of similar documents during the probing step is under a certain value. With $k = 1$, for a check threshold t and a probe threshold p, the probability of missing a pair of similar documents is dominated by:

$$P_{miss} = \frac{\binom{m-p}{t}}{\binom{m}{t}} = \frac{(m-t-1)\dots(m-t-p+1)}{m\dots(m-p+1)}$$

The actual error rate is lower due to the quasi-transitivity of the similarity graph. With $m = 128$ and $k = 1$ the relation $p \cdot t \geq 512$ ensures an edge missing probability lower than 1%.

4. EXPERIMENTAL RESULTS

We used for experimentation a corpus of five million HTML pages crawled from the web. This corpus was built by combining tree crawl strategies:

- a deep internal crawl of 3 million documents from 1300 hosts of DMOZ directory;
- a flat crawl of 1 million documents from a french search engine blacklist (with many adult content);
- a deep breadth first crawl from 10 non adult spam urls (chosen in the blacklist) and 10 trustable urls (mostly from french universities).

At the end of the process, we estimated roughly that 2/5 of the collected documents where spam.

After fingerprinting, the initial volume of 130 GB data to analyse was reduced down to 390 Mo, so the next step could easily be made in memory.

4.1 One-to-All Similarity

The comparison between one HTML page and all other pages of our test base is a way to estimate the quality of HS-similarity. If the reference page comes from a known web site, we are able to judge the quality of HS-similarity as web site detector. In the example considered here: `franao.com` (a web directory with many links and many internal pages), a threshold of 20/128 gives a `franao` web site detector which is not based on URLS. On our corpus, this detector is 100% correct according to URLS prefixes.

By sorting all HTML documents by decreasing HS-similarity to one randomly chosen page of `franao` web site, we get a decreasing curve with different gaps (Figure 4). These gaps are interesting to consider in detail:

- The first 20,000 HTML pages are long lists of external links, all built from template 1 (Figure 5);
- Around 20,000 there is a smooth gap (1) between long list and short list pages, but up to 95,000, the template is the same;
- Around 95,000, there is a strong gap (2) which marks a new template (Figure 6): up to 180,000, all pages are internal `franao` links built according to template 2;
- Around 180,000 there is a strong gap (3) between `franao` pages and other web sites pages.

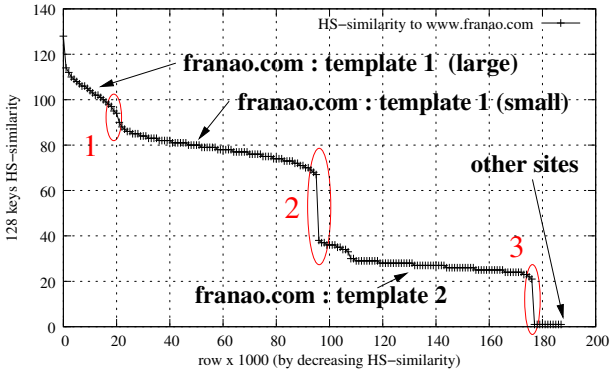


Figure 4: By sorting all HTML documents by decreasing HS-similarity with one reference page (here from `franao.com`) we get a curve with different gaps. The third gap (around 180,000) marks the end of `franao` web site.

Figure 5: `franao.com` template 1 (external links)

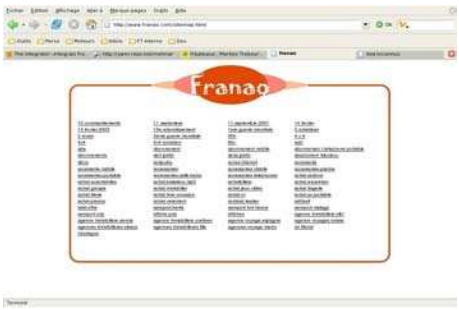


Figure 6: `franao.com` template 2 (internal links)

Table 1: Clusters with highest mean similarity and domain count

URLS	Domains	Mean similarity	Prototypical member (centroïd)	
268	231	1	www.9eleven.com/index.html	Copy/Paste
93148	313	0.58	www.les7laux.com/hiver/forum/phpBB2/membe...	Template (Forums)
3495	255	0.33	www.orpha.net/static/index.html	Template (Apache)
966	174	0.40	www.asliguruney.com/result.php?Keywords=m...	Link farm
122	91	0.74	anus.fistingfisting.com/index.htm	Copy/Paste
1148	173	0.38	www.basketmag.com/result.php?Keywords=gif...	Link farm
19834	164	0.40	www.series-tele.fr/index.html?mo=serie_t...	Template
122	55	0.91	www.ie.gnu.org/philosophy/index.html	Mirror
139	101	0.44	www.reha-care.net/home_buying.htm?r=p	Link farm
218	195	0.21	chat.porno-star.it/index.html	Copy/Paste
177	60	0.67	www.ie.gnu.org/home.html	Mirror
2288	44	0.90	www.cash4you.com/insuranceproviders/index...	Link farm
626900	70	0.52	animalworld.petparty.com/automotivecenter...	Link farm
168	96	0.32	www.google.ca/intl/en/index.html	Mirror
214	61	0.50	shortcuts.00go.com/shorcuts.html	Link farm
42314	112	0.26	forums.cosplay.com/index.html	Template
121	63	0.41	collection.galerie-yemaya.com/index.html	Copy/Paste
555	34	0.68	allmacintosh.digsys.bg/audiomac_rating.h...	Template
114	77	0.29	www.gfx-revolution.com/search/webarchiv.p...	Link farm
286	60	0.35	gnu.typhon.net/home.sv.html	Mirror

4.2 Global Clustering

To clusterize the whole corpus we need to raise the threshold to ensure a low level of false positive. In our experiments we chose $n = 32$, $m = 128$, $k = 1$, $t = 35$ and $p = 20$. With a similarity score of at least 35/128, the number of misclassified URLS seems negligible but some clusters are split into smaller ones.

We obtained 43,000 clusters with at least 2 elements. The table 1 shows the 20 clusters sorted by highest *mean similarity* × *domain count*. (For the sake of readability some mirror clusters have been removed from the list.)

In order to evaluate the quality of the clustering, the first 50 clusters, as well as 50 other randomly chosen ones, were manually checked, showing no misclassified URLS.

Most of the resulting clusters belong to one of these classes:

1. *Template* clusters groups HTML pages from dynamic web sites using the same skeleton. The cluster #2 of figure 1 is a perfect example, it groups all forum pages generated using the PhpBB open source project. The cluster #3 is also interesting: it is populated by Apache default directory listings;

2. *Link farm* clusters are also a special case of *Template*. They contain numerous computer generated pages, based on the same template and containing a lot of hyperlinks between each other;

3. *Mirrors* clusters contain sets of near duplicate pages hosted under different servers. Generally only minor changes were applied to copies like adding a link back to the server hosting the mirror;

4. *Copy/Paste* clusters contain pages that are not part of mirrors, but do share the same content: either a text (e.g. license, porn site legal warning...), a frameset scheme, or the same javascript code (often with few actual content).

Table 2: A sample template cluster

	URL
1	www.les7laux.com/hiver/forum/phpBB2/me...
0.68	ksosclan.free.fr/phpBB2/login.php
0.67	www.quartertothree.com/phpBB2/profile.ph...
0.65	www.lirone.com/forum/login.php?redirect=...
0.64	www.francemule.com/forum/login.php?redir...
0.63	ksosclan.free.fr/phpBB2/login.php?redire...
0.62	90plan.ovh.net/ lillofor/login.php?redir...
0.60	www.artisanatweb.com/phpBB/viewtopic.php...
0.57	www.artisanatweb.com/phpBB/viewforum.ph...
0.54	www.dualforum.com/profile.php?mode=viewp...
0.53	www.dualforum.com/viewforum.php?f=52
0.56	bande2floydiens.free.fr/forum/posting.ph...
0.33	forum.p2pfr.com/posting.php?mode=quote&a...
0.28	a.chavasse.free.fr/phpBB2/viewtopic.php?...
0.25	www.e-hotellerie.com/forum/index.html?c=...

The first two cluster classes are the most interesting benefit of the use of HSS clustering. They allow an easy classification of web pages by categorizing a few of them.

Table 2 shows sample URLS of a cluster with the associated similarity against the center of the cluster. This cluster gathers URLS from forum build with phpBB. Some of these could have been classified with simple methods like a search for the typical string *"phpBB2"* in the URL, but this would overlook some web sites that integrate phpBB with some changes in the display style. Using HSS algorithm allows to gather those forums in the same cluster.

Classical algorithms for similarity could be used to extract the last two cluster classes. Using HSS algorithm enables to quickly build a first clustering of the pages, and next use more expensive methods to refine this clustering, reducing the total computing time.

5. CONCLUSION

We presented in this paper a method to compute a distance based on HTML extra-textual features (*Hidden Style Similarity*). A computationally efficient method to cluster documents based on this similarity has been detailed, and some results on a test corpus have been commented.

This method finds several uses in a search engine back-office process: it makes it possible to cluster pages based on the template and writing style. It enables to find particular instances of well-known pages genre such as forum pages or Apache directory listings, so as to tag or skip them in a search engine response list. Given a set of already known undesirable pages, other pages sharing the same template can be sought after and tagged for deletion or editorial review.

Apart from the editorial detection of web spam, a complementary useful process is to point out large clusters of similar pages spanning several domains: this is often a good hint of either sites mirroring or automatic spam pages generation, both things being valuable information to be processed by a search engine back office.

6. REFERENCES

[1] A. Z. Broder. On the resemblance and containment of documents. In *Proceedings of Compression and Complexity of Sequences*, page 21, 1998.

[2] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. In *Selected papers from the sixth international conference on World Wide Web*, pages 1157–1166, Essex, UK, 1997. Elsevier Science Publishers Ltd.

[3] D. Fetterly, M. Manasse, and M. Najork. Spam, damn spam, and statistics: using statistical analysis to locate spam web pages. In *WebDB '04: Proceedings of the 7th International Workshop on the Web and Databases*, pages 1–6, New York, NY, USA, 2004. ACM Press.

[4] D. Fetterly, M. Manasse, and M. Najork. Detecting phrase-level duplication on the world wide web. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 170–177, New York, NY, USA, 2005. ACM Press.

[5] A. Gray, P. Sallis, and S. MacDonell. Software forensics: Extending authorship analysis techniques to computer programs. In *3rd Biannual Conference of International Association of Forensic Linguists (IAFL '97)*, pages 1–8, 1997.

[6] Z. Gyöngyi and H. Garcia-Molina. Web spam taxonomy. In *First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2005.

[7] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. In *VLDB*, pages 576–587, 2004.

[8] N. Heintze. Scalable document fingerprinting. In *1996 USENIX Workshop on Electronic Commerce*, November 1996.

[9] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. of 30th STOC*, pages 604–613, 1998.

[10] B. Kessler, G. Nunberg, and H. Schütze. Automatic detection of text genre. In *Proceedings of the 35th Annual Meeting of the ACL and 8th Conference of the EACL*, pages 32–38, 1997.

[11] T. Lavergne. Unnatural language detection. In *Proceedings of RJCRI'06 : Young Scientists' conference on Information Retrieval*, 2006.

[12] T. McEnery and M. Oakes. Authorship identification and computational stylometry. In *Handbook of Natural Language Processing*. Marcel Dekker Inc., 2000.

[13] S. Meyer Zu Eissen and B. Stein. Genre classification of web pages. In S. Biundo, T. Frühwirth, and G. Palm, editors, *Proceedings of KI-04, 27th German Conference on Artificial Intelligence*, Ulm, DE, 2004. Published in LNCS 3238.

[14] C. J. Van Rijsbergen. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow, 1979.

[15] A. Westbrook and R. Greene. Using semantic analysis to classify search engine spam. Technical report, Stanford University, 2002.

[16] J. Zobel and A. Moffat. Exploring the similarity space. *SIGIR Forum*, 32(1):18–34, 1998.

Adversarial Information Retrieval Aspects of Sponsored Search

Bernard J. Jansen
College of Information Sciences and Technology
The Pennsylvania State University
University Park, PA, 16801, USA
jjansen@acm.org

ABSTRACT

Search engines are commercial entities that require revenue to survive. The most prevalent revenue stream for search engines is sponsored search, where content providers have search engines service their links to users in response to queries or in a contextual manner on relevant Web sites. In exchange for providing this service, content providers pay search engines based on the number of clicks (i.e., a click being a visit by a user to the content providers Web page). This business model has proven to be very effective for the search engines, content providers, and searchers. However, click fraud, a unique form of adversarial information retrieval, threatens this business model and, therefore, the "free search" that has rapidly become indispensable to the daily lives of many people. In this paper, we outline how sponsored search is a unique form of information retrieval – not just a mode of advertising, what is click fraud, how click fraud happens, and what are some possible countermeasures.

1. INTRODUCTION

Web search engines provide information access to millions of users per day. For many people, Web search engines are now the primary method for finding information, news, and products, according to a recent report on Internet usage [11]. Given this importance, there is increasing attention being paid to search engine spam and other adversarial information retrieval (IR) techniques by content providers to secure undeserved highly ranked positions in the search engine results listings. However, most of the attention in adversarial IR is focused on the algorithmic listings.

Major search engines offer at least two types of results on a search engine results page (SERP), non-sponsored and sponsored results. Sponsored search is an increasingly important, popular, and uniquely contextual form of information interaction on the Web, and is subject to spamming (i.e., click fraud). However, sponsored search and adversarial techniques to subvert it have received little attention in the research community. This lack of consideration is surprising given that the negative effect of spam on the sponsored search process may have greater implications than on the algorithmic procedure.

Sponsored search is the process by which content providers pay Web search engines to display specific links in response to user queries alongside the algorithmic (a.k.a., organic or non-sponsored) links. The sponsored search mechanism plays a critical role in financing the "free" search provided by search engines that have rapidly become essential to many Web users. A distinctive type of interaction involving information push-and-pull, sponsored search also is increasingly important in locating information on the Web. Because of the uniquely dynamic contextual relationship among participants, sponsored search is a distinctive form of IR, and there are significant social and political repercussions if the process is significantly compromised.

In the paper, we provide an overview of sponsored search to demonstrate that it is a unique form of IR and much more than "just online advertising", which may be a common misperception [c.f., 13, 14]. We then discuss click fraud, highlighting how the sponsored search process is susceptible to spamming. We demonstrate how click fraud occurs. We conclude with a discussion of the implications of click fraud and possible mechanisms to combat it.

AIRWeb'06, August 10, 2006, Seattle, Washington, USA.

2. LITERATURE REVIEW

The primary business model for these search engines is sponsored search, where commercial corporations, small businesses, and other entities or individuals pay the search engines to service links that appear on SERP when searchers enter certain key phrases as queries. The content providers may also pay to have their listings presented on Web sites that the search engine's or the content provider deem relevant to the sponsored search links.

The economic impact of paid search is immense. Sponsored search was an $8 billion industry in 2004 and vital to the success of most major search engines. For example, Google received 99% of its $3.1 billion revenue from paid search in 2004; Yahoo! received 84% of its $3 billion, and AOL received 12% of its $1 billion, according to Tim McCarty of Time magazine [12]. In 2005, Web search engines displayed approximately 13 billion sponsored links in a given week, according to Nielsen/NetRatings (http://www.tekrati.com/firmnews/?id=5756). The investment firm Piper Jaffray estimates that online advertising will exceed $55 billion globally by 2010 (http://www.clickz.com/news/article.php/3569361). Without a doubt, sponsored search is now and the foreseeable future the primary business model for Web search engines [5].

For a review of how sponsored search works see [7]. Accounting is one reason that sponsored search is so popular for businesses and organizations. In most models of

advertising, there is little accountable with the cost being impressions (i.e., how many times and when a particular advertisement is shown). However, this is also the key area for an adversarial IR technique known as click fraud.

3. ADVERSARIAL IR ASPECT OF SPONSORED SEARCH

Sponsored search significantly reduces spam content that many times occurs with algorithmic listings. In fact, search engine spam was the primary motivation for the development of the sponsored search paradigm [1]. The reason that sponsored search helped reduce spam is that there is a cost motive for the provider and search engine to present relevant content, and the search engines have review processes consisting of both automated and manual aspects to help ensure this. These monetary factors significantly reduce spam content.

3.1 Click Fraud

However, there is the issue of click fraud with sponsored search. Click fraud is the intentional clicking on a sponsored link where the perpetrator does not intend to buy (or use) the products or services advertised. We use the term "buy", since most sponsored links are ecommerce related. However, more and more non-commercial entities are entering the sponsored search market. So, "buy" may soon be too restrictive. Regardless, click fraud is one of the fastest growing problems on the Web, according to iProspect (http://www.iprospect.com/media/newsletter_october_meech.htm?ipsrc=media&reftype=pi&sptype=osmx). Click fraud has not been widely perceived as search engine spamming [6], but its negative effect is severe.

Click fraud can take various forms, but the final result is usually the same. Content providers pay for unproductive traffic generated by perpetrators who repeatedly click on a content provider's sponsored link with no intention of buying anything. Click fraud produces revenue for the major search engines and the Web sites that display the links. This is because the clicks generate sales commissions based on the content provider's bid even if the click does not result in a sale. In sponsored search, content providers are contractually obligated to pay for all valid clicks. However, the search engine has discretion over what is valid. According to [10], based on an analysis of more than 1,000 content providers, Google and Yahoo!'s sponsored search programs suffered a click fraud rate of 12%, translating to more than $1.5 billion of Google's ad revenue in 2005. The 12% click fraud rate correlates well with that reported in [9]. However, some content providers complain that their individual click fraud rate is as high as 35% [10]. See [9] for an overview of the click fraud issue.

3.2 Click Fraud Implementation

Why and how does click fraud occur? As for the why, sometimes one content provider tries to deplete a competitor's sponsored search budget (most content providers have monetary limits for any period). In other instances, the owners of Web sites servicing sponsored links click on these links to generate commissions for themselves. Finally, some even more "ethically challenged" individuals set up fictitious Web sites targeted at high payoff sponsored terms. These Web sites exist solely to generate commissions for clicks on sponsored links. The owners of these Web sites typically use automated tools to both set up and generate clicks.

In the first case, depleting a competitor's budget, the motivation is usually to increase the cost of advertising for the other content provider, exhausting the rival's budget. Once the rival's link has dropped out of the search engine's listing, more traffic is diverted to the remaining sponsored links. This type of click fraud is fairly easy to implement. For example, see Figure 2.

Figure 2 contains a snippet from a SERP, namely the sponsored link section. From Figure 2, we see that the initial query "Jim Jansen" retrieved three sponsored results with the sponsored link **Jim Jansen** in the first position. With repeated submissions of the query "Jim Jansen" and subsequent clicks on the sponsored result, we see that the link **Jim Jansen** soon drops out of the sponsored listing once the daily budget has expired. The effects are (1) the content provider of the link **Jim Jansen** pays the search engine for each of these clicks, (2) once the budget for the link **Jim Jansen** is exhausted that link no longer appears, depriving the content provider of traffic, and (3) with the link J**im Jansen** gone, the other links move up in ranking. Studies show that about 30% of searches involve a click on a sponsored link [8] and that the higher a link is in the results listing, the more visits that Web site will receive [2, 3].

For the other two cases of click fraud, the motivation is money. This version of click fraud consists of Web site owners who service sponsored content on their sites and then click these links to generate commissions. See Figure 3 for an example.

We see in Figure 3 that this particular Web site serves contextual sponsored links from Google. Many times these links appear during a normal visit to the Web site or one can trigger their appearance via searching conducted on the Web site. In Figure 3, the query "Jim Jansen" to the Web site's local content prompted the display of the sponsored link **Jim Jansen**. By clicking on this sponsored link, the content provider will pay the search engine, who will then split the payment with the Web site owner.

Numerous software packages are available that will assist in setting-up or nearly automatically setting-up a Web site targeted at high pay-off key words, automate the clicks, and disguise the Internet Protocol (IP) address. The process is marketed as something easy to do, as shown in Figure 4.

3.3 Click Fraud Prevention

What are some potential click fraud countermeasures?

- **Automated and Human Filters**: Search engines currently employ both automated and human filters in an attempt to identify current and prevent future click fraud. Search engine also appear to be making reasonable attempts to reimburse or not charge clients for click identified as click fraud. However, given the string of class action lawsuits, it is apparent significantly more needs to be done.

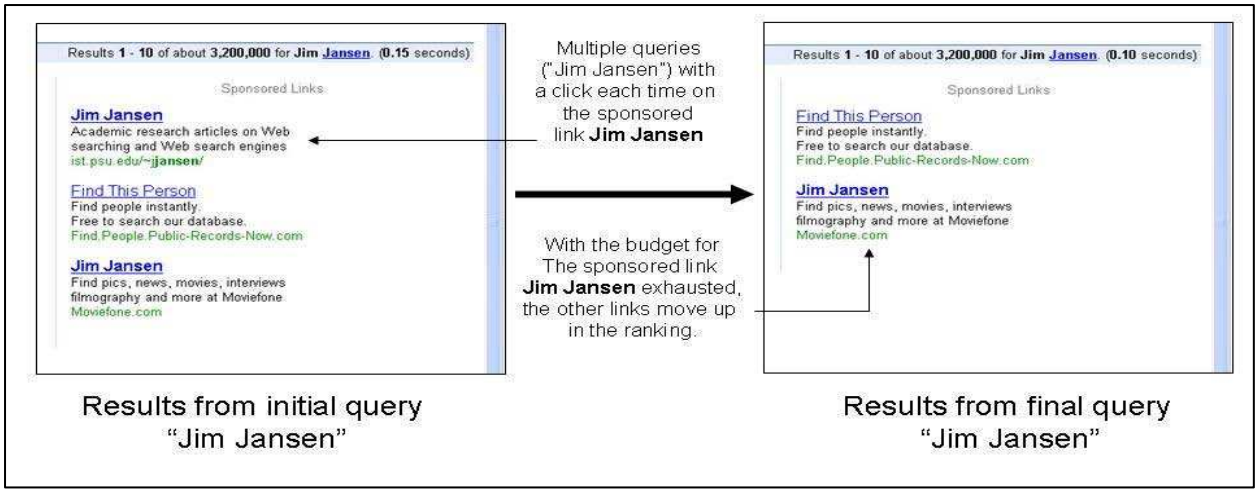


Figure 2. An Example of Click Fraud on the Sponsored Listings of a Web Search Engine.

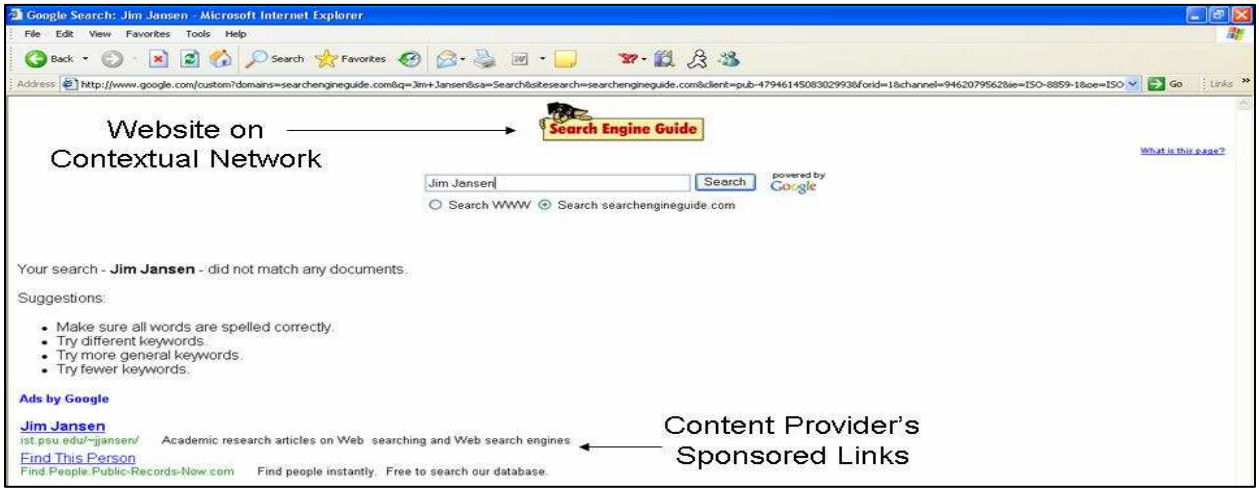


Figure 3. An Example of Contextual Link Where Click Fraud Can Occur.

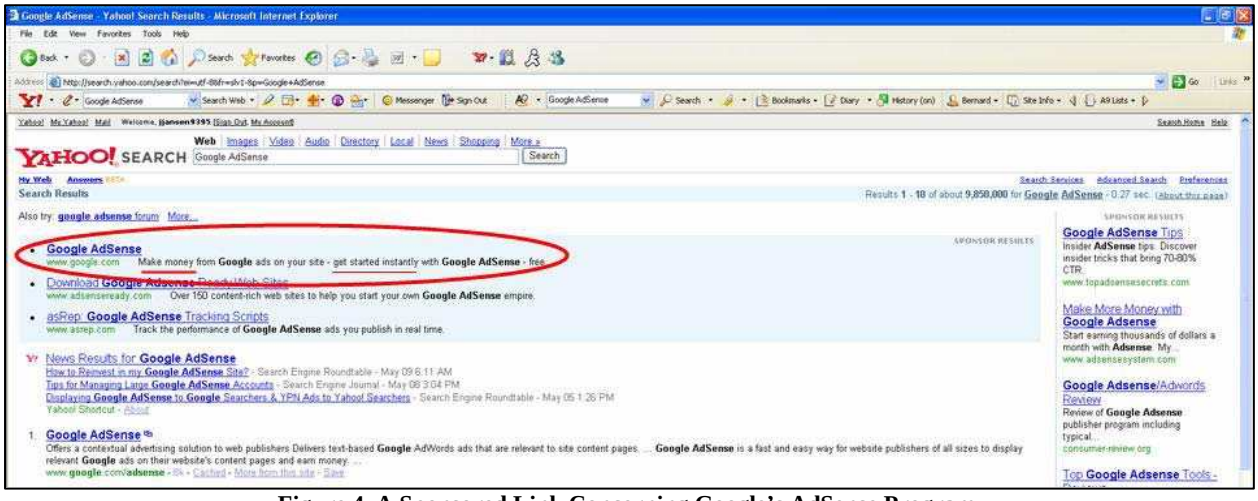


Figure 4. A Sponsored Link Concerning Google's AdSense Program.

Certainly, more sophisticated automated filters and data mining techniques need to be employed, more human effort needs to be engaged, and much better communication about these efforts to the customers and public.

- **Pay-per-action Paradigm**: One partial solution is a shift in paradigm from pay-per-click to pay-per-action. With pay-per-action, the advertiser only pays if the visitor actually executes an action, such as purchasing a product. However, pay-per-action is not the total answer though, as research

reports that many searchers visit a sponsored link multiple times before a purchase [4]. Additionally, some of the traffic generated should be based on the ability of the content provider to construct enticing sponsored links.

- **Block Blacklisted IP addresses**: There are various databases of blacklisted IPs (c.f., http://www.declude.com/Articles.asp?ID=97 and http://www.moensted.dk/spam/), which maintain lists of IPs that are know email spammer sites. These spam lists are also known as Realtime Blackhole List (RBL). The company Mail Abuse Prevention System (MAPS) LLC actively maintains records of RBLs. These are IP addresses whose owners refuse to stop others from using their servers for spam. Email servers routinely block messages from these IPs. However, click fraud perpetrators also use these IP. It would seem reasonable that the search engines could take measures to block clicks from these IPs or reimburse content providers for clicks from these IP addresses.

- **Aggressive monitoring of click fraud perpetrators:** Click fraud is similar to what occurred in the online music industry. The Recording Industry Association of America's (RIAA) campaign against illegal file sharing via peer-to-peer networks is a good example of the effect that an aggressive operation can achieve (c.f., http://www.techweb.com/wire/story/TWB20031105S0006). The RIAA's effort significantly reduced illegal copying of copyrighted audio files. Many content providers have been critical of the major search engines for their lack of aggressive pursuit of click fraud abusers. Aggressive action against click fraud would raise the cost of click fraud, and could reduce the number of folks doing it.

- **Search engines must make efforts to ensure trust**: In sponsored search, content providers sign contracts to pay for all valid clicks, with the search engine determining what is a valid click. Trust is a, if not the, critical element in the sponsored search paradigm. Although the major search engines do make efforts to identify click fraud, sponsored search is not subject to independent auditing. San Antonio-based Click Forensics Inc. recently set up a free service that intends to issue quarterly reports on the frequency of click fraud [10]. Whether through independent auditing or internal efforts, content providers and searchers must have trust in the process if it is to be a long-term business model.

4. CONCLUSION

It appears that the sponsored search model will have increasing impact as new players enter the field. Within this extremely dynamic paradigm, click fraud threatens the entire process. Although media reports rates as high as 50%, studies in the area indicate click fraud rates of between 12% and 16% [9]. This translates into billions of dollars per year, and it jeopardizes the entire model as it decreases trust in the system, which is the basis of any IR process. In this regard, the onus is on the search engines and related researchers to develop methods to combat this threat. These methods run the gamut from technological, to business processes, to regulatory measures.

5. REFERENCES

[1] Battelle, J., *The Search: How Google and Its Rivals Rewrote the Rules of Business and Transformed our Culture*. New York: Penguin Group, 2005.

[2] Brooks, N., *The Atlas Rank Report I: How Search Engine Rank Impacts Traffic*, Accessed on 1 August 2004 on the World Wide Web at http://www.atlasdmt.com/media/pdfs/insights/RankReport.pdf.

[3] Brooks, N., *The Atlas Rank Report II: How Search Engine Rank Impacts Conversions*, Accessed on 15 January 2005 on the World Wide Web at http://www.atlasonepoint.com/pdf/AtlasRankReportPart2.pdf.

[4] Brooks, N., Repeat Search Behavior: Implications for Advertisers, *Bulletin of the American Society for Information Science and Technology*, vol. 32, pp. 16-17, 2006.

[5] Chiang, K.-P., Clicking Instead of Walking: Consumers Searching for Information in the Electronic Marketplace, *Bulletin of the American Society for Information Science and Technology*, vol. 32, pp. 9-10, 2006.

[6] Gyongyi, Z. and Garcia-Molina, H., Web Spam Taxonomy, in *Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb '05), the 14th International World Wide Web Conference (WWW2005)*, 2005. Chiba, Japan. 10-14 May. pp. 39-47.

[7] Jansen, B. J., Paid Search, *IEEE Computer*, Forthcoming.

[8] Jansen, B. J. and Resnick, M., An examination of searchers' perceptions of non-sponsored and sponsored links during ecommerce Web searching, *Journal of the American Society for Information Science and Technology*, forthcoming.

[9] Kitts, B., LeBlanc, B., Meech, R., and Laxminarayan, P., Click Fraud, *Bulletin of the American Society for Information Science and Technology*, vol. 32, pp. 14-16, 2005.

[10] Liedtke, M., *Click Fraud Concerns Hound Google*, in *ABC News Money*, 2006 http://abcnews.go.com/Technology/wireStory?id=1934655&CMP=OTC-RSSFeeds0312.

[11] Madden, M., *Internet Penetration and Impact*, Accessed on 9 May 2006 on the World Wide Web at http://www.pewinternet.org/PPF/r/182/report_display.asp.

[12] McCarthy, T., Yahoo! Goes to Hollywood, *Time*, vol. 165, pp. 50-53, 2005.

[13] Metaxas, P. T. and DeStefano, J., Web Spam, Propaganda and Trust, in *Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb '05), the 14th International World Wide Web Conference (WWW2005)*, 2005. Chiba, Japan. 10-14 May. pp. 70-78.

[14] Nicholson, S., Sierra, T., Eseryel, U. Y., Park, J.-H., Barkow, P., Pozo, E. J., and Ward, J., How much of it is real? Analysis of paid placement in Web search engine results, *Journal of the American Society of Information Science and Technology*, vol. 57, pp. 448-461, 2006.

Web Spam Detection with Anti-Trust Rank

Vijay Krishnan
Computer Science Department
Stanford University
Stanford, CA 4305
vijayk@cs.stanford.edu

Rashmi Raj
Computer Science Department
Stanford University
Stanford, CA 4305
rashmi@cs.stanford.edu

ABSTRACT

Spam pages on the web use various techniques to *artificially* achieve high rankings in search engine results. Human experts can do a good job of identifying spam pages and pages whose information is of dubious quality, but it is practically infeasible to use human effort for a large number of pages. Similar to the Trust Rank algorithm [1], we propose a method of selecting a seed set of pages to be evaluated by a human. We then use the link structure of the web and the manually labeled seed set, to detect other spam pages. Our experiments on the WebGraph dataset [3] show that our approach is very effective at detecting spam pages from a small seed set and achieves higher precision of spam page detection than the Trust Rank algorithm, apart from detecting pages with higher pageranks [10, 11], on an average.

1. INTRODUCTION

The term Web Spam refers to the pages that are created with the intention of misleading a search engine [1]. In order to put the tremendous amount of information on the web to use, search engines need to take into account the twin aspects of relevance and quality. The high commercial value associated with a web page appearing high on the search results of popular search engines, has led to several pages attempting spamdexing i.e. using various techniques to achieve higher-than-deserved rankings. Though it is not difficult for a human expert to recognize a spam web page, it is a challenging task to automate the same, since spammers are constantly coming up with more and more sophisticated techniques to beat search engines. Recent work [1], addressed the problem of Web Spam detection by exploiting the intuition that good pages i.e. those of high quality are very unlikely to point to spam pages or pages of low quality. They propagate *Trust* from the seed set of good pages recursively to the outgoing links. However, sometimes spam page creators manage to put a link to a spam page on a good page, for example by leaving their link on the comments section of a good page, or buy an expired domain. Thus, the trust propagation is *soft* and is designed to attenuate with distance. The Trust Rank approach thus starts with a seed set of trusted pages as the *teleport set* [2] and then runs a biased page-rank algorithm. The pages above a certain threshold are deemed trustworthy pages. If a page has a trust value below a chosen threshold value then it is marked as spam.

AIRWEB'06, August 10, 2006, Seattle, Washington, USA.

In our work, we exploit the same intuition, in a slightly different way. It follows from the intuition of [1] that it is also very unlikely for spam pages to be pointed to by good pages. Thus we start with a seed set of spam pages and propagate *Anti Trust* in the reverse direction with the objective of detecting the spam pages which can then be filtered by a search engine.

We find that on the task of finding spam pages with high precision, our approach outperforms Trust Rank. We also empirically found that the average page-rank of spam pages reported by Anti-Trust rank was typically much higher than those by Trust Rank. This is very advantageous because filtering of spam pages with high page-rank is a much bigger concern for search engines, as these pages are much more likely to be returned in response to user queries.

1.1 Our Contributions

- We introduce the Anti-Trust algorithm with an intuition similar to [1], for detecting untrustworthy pages.
- We show that it is possible to use a small seed set of manually labeled spam pages, and automatically detect several spam pages with high precision.
- We propose a method for selecting seed sets of pages to be manually labeled.
- We experimentally show that our method is very effective both at detecting spam pages as well as detecting spam pages with relatively high PageRanks.

2. ANTI-TRUST RANK

Our approach is broadly based on the same *approximate isolation* principle [1], i.e it is rare for a good page to point to a bad page. This principle also implies that the pages pointing to spam pages are very likely to be spam pages themselves. The Trust Rank algorithm started with a seed set of trustworthy pages and propagated *Trust* along the outgoing links. Likewise, in our Anti-Trust Rank algorithm, *Anti-Trust* is propagated in the reverse direction along incoming links, starting from a seed set of spam pages. We could classify a page as a spam page if it has Anti-Trust Rank value more than a chosen threshold value. Alternatively, we could choose to merely return the top n pages based on Anti-Trust Rank which would be the n pages that are most likely to be spam, as per our algorithm.

Interestingly, both Trust and Anti-Trust Rank approaches need not be used for something very specific like detecting

link spam alone. The *approximate isolation* principle can in general enable us to distinguish *good* pages from the not-so-good pages such as pages containing pornography and those selling cheap medication. Thus, for the purpose of our work we consider pages in the latter category as spam as well.

2.1 Selecting the Seed Set of Spam pages

We have similar concerns to [1], with regard to choosing a seed set of spam pages. We would like a seed set of pages from which *Anti-Trust* can be propagated to many pages with a small number of hops. We would also prefer if a seed set can enable us to detect spam pages having relatively high PageRanks. In our approach, choosing our seed set of spam pages from among those with high PageRank satisfies both these objectives.

Pages with high PageRank are those from which several pages can be reached in a few hops if we go backward along the incoming links. Thus this helps in our first objective. Also, having high PageRank pages in our seed set makes it somewhat more probable that the spam pages we detect would also have high PageRanks, since high PageRanks pages often get pointed to by other pages with high PageRank. We therefore select our seed set of spam pages from among the pages with high PageRank. This helps us nail our twin goals of fast reachability and detection of spam pages with high PageRank.

2.2 The Anti-Trust Algorithm

- Obtain a seed set of spam pages labeled by hand. Assign pages with high PageRanks for labeling by a human in order to get a seed set containing high PageRank pages.
- Compute T to be the Transpose of the binary web-graph matrix.
- Run the biased PageRank algorithm on the matrix T, with the seed set as the teleport set.
- Rank the pages in descending order of PageRank scores. This represents an ordering of pages based on estimated Spam content. Alternatively, set a threshold value and declare all pages with scores greater than the threshold as spam.

3. EXPERIMENTS

3.1 Dataset

We ran our experiments on the WebGraph dataset, [3]. We chose data corresponding to a 2002 crawl of the "uk" domain containing about 18.5 millions nodes and 300 million links.

3.2 Evaluation Metric

Clearly, the only perfect way of evaluating our results is to manually check if the pages with high Anti-Trust score are indeed spam pages and vice-versa. It was observed in [1] that this process is very time consuming and often hard to do in practice.

We however circumvented this problem by coming up with a heuristic which in practice selects spam pages with nearly

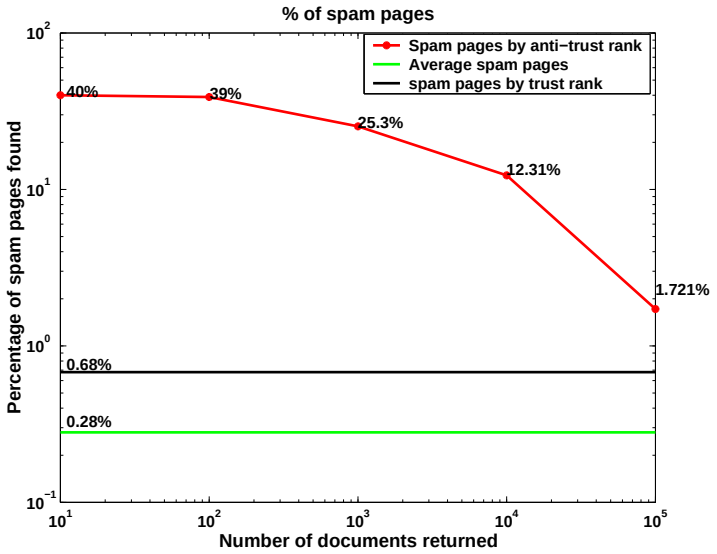


Figure 1: Comparison of the precisions of Anti-Trust Rank and Trust Rank at various levels of recall, against the naive baseline of total percentage of spam documents in the corpus. It can be seen what Anti-Trust Rank does significantly better than Trust Rank which is in turn clearly better than the naive baseline.

100% precision and also a recall which is a reasonable fraction of the set of true spam pages, on our dataset.

The Heuristic: We compiled a list of substrings whose presence in a URL almost certainly indicated that it was a spam page, on our dataset. As one would expect, our list contained strings like viagra, casino and hardporn. Thus, this heuristic enables us to measure the performance of our Anti-Trust Rank algorithm and compare it against the Trust Rank algorithm with a good degree of reliability. It seems reasonable to expect that the relative scores obtained by the spam detection algorithms with the evaluation being heuristic based would be representative of their actual performance in spam detection, since our heuristic has a pretty reasonable recall and is independent of both the Trust Rank and Anti-Trust Rank algorithms and would not give the algorithms we are looking at, an unfair advantage.

As per this heuristic, out of the 18,520,486 pages, 0.28 % i.e. 52,285 were spam pages.

3.3 Choosing the Seed Set

We chose the top 40 pages based on page rank from among the URLs that got flagged as spam by our heuristic. For comparing with Trust-Rank we picked the top 40 pages based on inverse page rank, among the pages marked non-spam by our heuristic. We also manually confirmed that the seed sets were indeed spam in the former cases and trustworthy pages in the latter case. We also studied the effect of increasing the seed set size in Anti-Trust rank. We found that we could benefit substantially from a larger seed set. Also we used the common α value of 0.85 i.e. the probability of teleporting to a seed node was 0.15.

3.4 Results and Analysis

From figure 1, we can see that both Anti-Trust Rank and

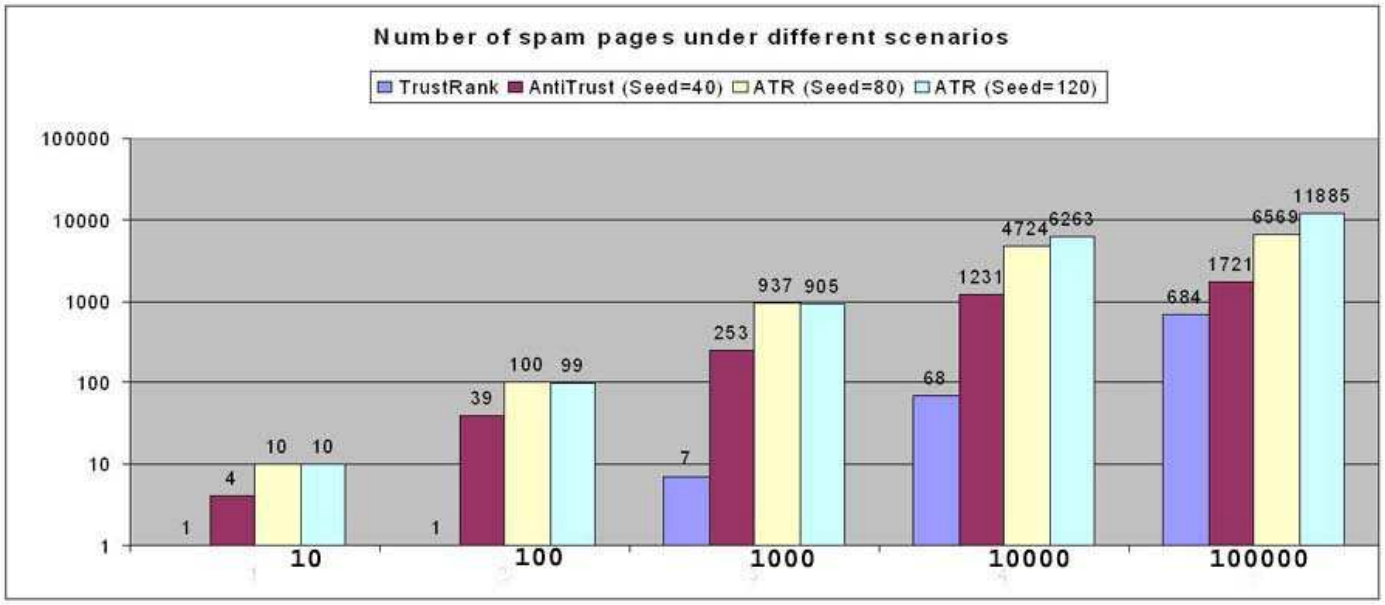


Figure 3: Comparison of the performance of Trust Rank with a seed set of 40 pages against Anti-Trust rank with 40, 80 and 120 pages respectively. The X-axis represents the number of documents selected having the highest Anti-Trust and lowest Trust scores. The Y-axis depicts, how many of those documents were actually spam(as measured by our heuristic). We observe that Anti-Trust rank typically has a much higher precision of reporting spam pages than Trust rank. Also, Anti-Trust rank benefits immensely with increasing seed-set size.

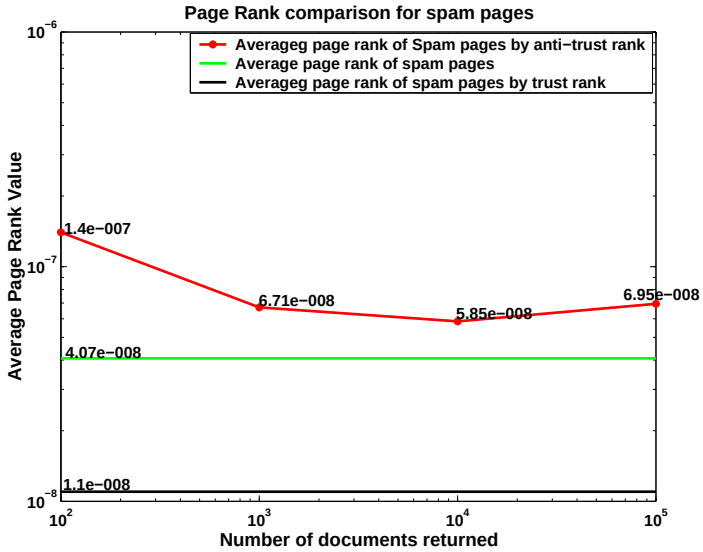


Figure 2: Comparison of the page ranks of spam pages returned by Anti-Trust Rank and Trust Rank at various levels of recall, against the baseline of average page rank of spam pages in the corpus. It can be seen that while Anti-Trust Rank returns spam pages with higher-than-average page ranks, Trust Rank returns spam pages with clearly lower-than-average page ranks.

Trust Rank are significantly better than the naive baseline corresponding to a random ordering of the pages, for which the precision of reporting spam would merely be the percentage of spam pages in the corpus. However we also see that Anti-Trust rank typically does much better than Trust Rank at different levels of recall.

This is intuitive because Trust Rank is capable of reporting with high confidence that the pages reachable in short paths from its seed set are trustworthy, while it cannot be expected to say anything with high confidence about pages that are far away from the seed set. Likewise our Anti-Trust Rank approach is capable of reporting with high confidence that the pages from which its seed set can be reached in short paths are untrustworthy.

Also, from figure 2, we find that the average rank of spam pages returned by Trust Rank is even lower than the average page rank of all spam pages. Anti-Trust rank however manages to return spam pages whose average page rank is substantially above the overall average page rank of all spam pages. The ratio of average page ranks of spam pages reported by Anti-Trust Rank and Trust rank was over 6:1 for different levels of recall. Thus, Anti-Trust rank has the added benefit of returning spam pages with high page rank, despite the fact that it has a significantly higher precision than Trust Rank at all levels of recall that we explored.

This is intuitive because, by starting with seed spam pages of high page rank, we would expect that walking backward would lead to a good number of spam pages of high page rank.

Figure 3 compares the performance of Trust Rank against Anti-Trust rank with an equal seed size of 40 and also show performance of Anti-Trust Rank with larger seed sets of 80 and 120 respectively. It shows the precisions achieved by

Trust Rank and Anti-Trust Rank at various levels of recall such as 10, 100, 1000, 10000 and 100000 web pages. We find that apart from achieving better precision of spam page detection than Trust Rank for the same seed set size, increasing the seed set size in Anti-Trust rank can lead to dramatic improvement in performance.

An analysis of success of these algorithms in picking trustworthy pages would not be very useful. This is because our corpus has over 99% trustworthy pages, and it would be very hard to conclude anything about the performance of these algorithms given that they would all attain a precision of well over 99% and would differ merely by a tiny fraction of a percent.

4. RELATED WORK

The BadRank algorithm [13] relies on intution similar to ours, namely that pages pointing to spam pages are likely to be spam themselves. The SpamRank algorithm [12] attempts to tackle link spam and assumes that spam pages have a biased distribution and attempts to compute the extent of underserved pagerank for a web page. The taxonomy of web spam has been well defined by [4]. There are many pieces of work on combating link spam. The problem of trust has also been studied in other distributed fields such as P2P systems [5]. Other approaches rely on detecting anomalies in statistics gathered through web crawls [7]. Approaches such as [8], focus on higher-level connectivity between sites and between top-level domains for identifying link spams. The data mining and web mining community has also worked on identifying link farms. Various farm structures and alliances that can impact ranking of a page has been studies by [6]. [9] identifies link farm spam pages by looking for certain patterns in the webgraph structure.

5. CONCLUSION AND FUTURE WORK

We have proposed the Anti-Trust Rank algorithm, and shown that it outperforms the Trust Rank algorithm at the task of detecting spam pages with high precision, at various levels of recall. Also, we show that our algorithm tends to detect spam pages with relatively high PageRanks, which is a very desirable objective.

It would be interesting to study the effect of combining these both the Trust Rank and Anti-Trust Rank methods especially on data containing a very high percentage of spam pages. It would also be interesting to attempt combining these link-based spam detection techniques with techniques that take text into account, such as text classifiers trained to detect spam pages.

Acknowledgements

We would like to thank Zoltán Gyöngyi, Dr. Anand Rajaraman and Dr. Jeffrey D. Ullman for helpful discussions. We would also like to express our gratitude to Paolo Boldi and Sebastiano Vigna whose compressed WebGraph dataset, with its useful Java API's made it very convenient for us to run experiments on a significant sized subgraph of the web.

6. REFERENCES

[1] Combating Web Spam with Trust Rank. Z. Gyöngyi, H. Garcia-Molina and J. Pedersen. Proc. of the 30st International Conference on Very Large Data Bases (VLDB), 2004.

[2] Topic-sensitive Page Rank. Taher Haveliwala. Proc. of the 11th International World Wide Web Conference, 2002.

[3] The WebGraph dataset. Online at: *http://webgraph-data.dsi.unimi.it/*

[4] Web Spam Taxonomy. Zoltán Gyöngyi, Hector Garcia-Molina. Proc. of the First International Workshop on Adversarial Information Retrieval on the Web (at the 14th International World Wide Web Conference), 2005.

[5] The EigenTrust algorithm for reputation management in P2P networks. S. Kamvar, M. Schlosser, and H. Garcia-Molina. Proc. of the Twelfth International World Wide Web Conference, 2003.

[6] Link Spam Alliances. Zoltán Gyöngyi, Hector Garcia-Molina. Proc. of the 31st International Conference on Very Large Data Bases (VLDB), 2005.

[7] Spam, Damn Spam, and Statistics. Dennis Fetterly, Mark Manasse and Marc Najork. Proc. of the Seventh International Workshop on the Web and Databases (WebDB 2004), 2004, Paris, France.

[8] Links to Whom: Mining Linkage between Web Sites. K. Bharat, B. Chang, M. Henzinger, and M. Ruhl. Proc. of the IEEE International Conference on Data Mining, 2001.

[9] Identifying Link Farm Spam Pages. Baoning Wu, Brian D. Davison. Proc. of the 14th International World Wide Web Conference, 2005.

[10] The PageRank citation ranking: Bringing order to the web. L. Page, S. Brin, R. Motwani and T. Winograd. Technical Report, Stanford University, 1998.

[11] The Anatomy of a Large-Scale Hypertextual Web Search Engine. Sergey Brin and Lawrence Page. Proc. of the 7th International World Wide Web Conference, 1998.

[12] SpamRank Fully Automatic Link Spam Detection. Andras A. Benczur, Karoly Csalogany, Tamas Sarlos, Mate Uher. Proc. of the First International Workshop on Adversarial Information Retrieval on the Web (at the 14th International World Wide Web Conference), Chiba, Japan, 2005.

[13] BadRank. Online at: *http://pr.efactory.de/e-pr0.shtml*