

Using Semantic Analysis to Classify Search Engine Spam

Andrew Westbrook
Stanford University
afw@stanford.edu

Russell Greene
Stanford University
rdg12@stanford.edu

1. INTRODUCTION

Anyone who uses a search engine frequently has most likely encountered a high ranking page that consists of nothing more than a bunch of query keywords. These pages detract both from the user experience and from the quality of the search engine. Search engine spam is a webpage that has been designed to artificially inflating its search engine ranking.

Search engines have tried many techniques to filter out these spam pages before they can appear on the query results page. In Section 2 we present a collection of current methods that are being used to combat spam. We introduce a new approach to spam detection in Section 3 that uses semantic analysis of textual content as a means of detecting spam. This new approach uses a series of content analyzers combined with a decision tree classifier to determine if a given webpage is spam. Section 4 discusses the implementation of our approach. Our architecture is augments the search engine Lucene by adding a Java-based spam classifier. The spam classifier makes use of the Wordnet word database and the machine learning library Weka to classify web documents as either spam or not spam. We describe the results of our work in Section 5 and finally present our conclusions and future work in Section 6.

2. RELATED WORK

2.1 Search Engine Spam Detection

2.1.1 *Link Analysis*

Any software that searches an unedited and/or unmoderated set of documents must contend with attempts made by document authors to manipulate the order in which the result set is returned. A ranking method which returns documents based on how many times the search term appears in each document could be easily manipulated by creating documents that include common search terms repeated over and over again. More complex ranking methods are also vulnerable to spam. For example, the cosine similarity method used in latent semantic indexing will always rank a document that is an exact match to the query higher than any other document. Thus, a spammer could create documents which contain only common queries; these documents would then always be first result for these frequent queries. (Presumably, the documents

would be “doorway” pages that redirect users to a different page containing other information.) Because of these concerns, successful search engines have taken steps to prevent web authors from manipulating their results.

The popular search engine Google¹ uses a system called PageRank to determine the order in which it returns results [4]. This ranking method orders pages based on the inbound links to each page. Essentially, when one page links to another it is casting a vote that the target of the link is valuable. This assumes that when a web author links to a page, he considers that page to be valuable. Furthermore, if the web author is not involved in a manipulation attempt, he will not link to a page whose sole purpose is to manipulate the search engine. Thus, spam pages have low PageRank scores and will not be returned at the top of Google’s results.

Although users have found Google resistant to spam, PageRank can be manipulated by artificially altering the link structure of the web. While Page et al. note that this could be done by web authors paying others for inbound links, thought they thought it would be financially infeasible [4]. It seems that their predictions were incorrect as at least one company is in the business of brokering these link sales [9]. Another problem with PageRank is that it only works on interlinked collections of documents. There are many valuable document repositories that do not have links such as newsgroup postings and archived emails. In addition, running PageRank on a small subset of the Web (e.g., the IBM.com website), will not produce as useful results since links from outside documents can not be considered. Although PageRank has been successful at keeping spammers from manipulating results, it is not impenetrable and link analysis is not applicable in certain instances.

2.1.2 *HTML Analysis*

All search engines do some analysis of the HTML elements on a web page in order to determine its ranking. In order to keep authors from simply filling the title and description elements with keywords, search engines will usually only look at a finite number of characters in these fields [12]. Additionally, search engines also look for attempts to hide keywords by putting them at the bottom of a page, in a small font, or in a font whose color closely matches the

¹ www.google.com

background color [8]. While these methods can detect many spam pages, others remain unnoticed by having spam text appear in the web page, masking itself as normal text.

Search engines will also look for “doorway” pages that are setup to rank highly on common searches and then send the user to a different page which would not have ranked as highly [12]. However, many doorway pages are difficult to detect since they use complicated JavaScript code rather than a simple redirect tag.

2.1.3 Human Experts

About.com², Yahoo!³, and the Open Directory Project⁴ all provide directories of pages on frequently requested topics. These directories have been edited and thus manually screened for spam. However, these listings will reflect the biases of their editors. While this may not bother some users, those searching for information on controversial topics may be more comfortable with search results that have not been filtered by a human. Finally, these directories may not be as up-to-date as other search engines since it is difficult for a human editor to keep up with the fast-changing web.

A system designed by Bharat and Mihalia [3] uses existing directories of sites on a particular topic to rank search results for that topic. It scours the web for pages that link to a wide variety of sites that are on the same topic but from different sources and stores these pages as “expert pages.” It then ranks pages based on the number of “experts” that link to them. While this system appeared very promising in tests, if it were deployed in an actual search engine, it would be subject to phony expert pages. Since the criteria for an “expert page” is based entirely on the content of the page itself, a spammer could create a number of pages that are designed to be recognized as “expert pages” and use them to manipulate the search engine.

2.2 Text Classification

2.2.1 Identification of email Spam

Androutopoulos et al. tested a Bayesian classifier (developed by Sahami et al.) that determines whether an incoming email is spam [1]. They note that “it seems that the language of current spam messages constitutes a distinctive genre” and using natural classification to distinguish between spam and legitimate email thus makes sense. This classification scheme treats each word as a token and analyzes messages based on the frequency of the words they contain.

Another spam email classifier is Spam Assassin [7]. This classifier tests for the presence of key phrases (e.g.,

pornographic text) and other properties (e.g., an invalid date in the headers) of an email message. It assigns each of these phrases and properties a numeric value and adds the values for a particular email together. If the sum of these values is above a certain threshold, it marks the message as spam.

Although both email and search engine spam are attempts to gain the attention of Internet users, they do not have much in common. Search engine spam is a largely technical task in that spammers are trying to get their results placed as highly as possible. Thus, filters that foil an email spammer may not be sufficient to stop a more technical search engine spammer. Also, since most personal email is not an attempt to sell a product or service, it is easier to classify email spam based on content. Classifying search engine spam in this manner is more difficult since many non-spam webpages exist for commercial purposes and contain many of the same keywords as the spam pages.

2.2.2 Classification of webpages

Quek developed a system to classify webpages into categories using a Bayesian classifier [6]. In addition to using only the textual components of pages, he tried using a couple web-specific classification schemes. One of these was to use only the text contained within header and title tags, as this text is assumed to be representative of the page’s contents. The other was to use the hyperlink structure and the text in the hyperlinks to derive relationships between webpages.

3. Webpage Spam

In order to properly classify spam, we first have to define precisely what constitutes a spam document. This definition is complex because spam in one context may not be spam in another. For example, meta tag information in a webpage is used specifically to list keywords as a means for assisting search engines. However, a listing of keywords in the title of the webpage itself does not add value to the webpage, existing solely as a means to attract search engines. In this case we have similar structures differing in classification solely based on their purpose. Because no specific element of a web page makes it spam, we have to look to the intent of the author.

A webpage is spam if it or a portion of it was created with the purpose of increasing its ranking through use of content that does not add to the user experience.

Unfortunately, intent is known only to the author of the document. We are attempting to derive intent based on the content of the document. We have done this by creating a series of semantic analyzers which report parameters for a given document. It is our belief, that there is a measurable difference in the parameters reported by our semantic analyzers when run on spam and non spam documents.

² <http://www.about.com/>

³ <http://www.yahoo.com/>

⁴ <http://dmoz.org/>

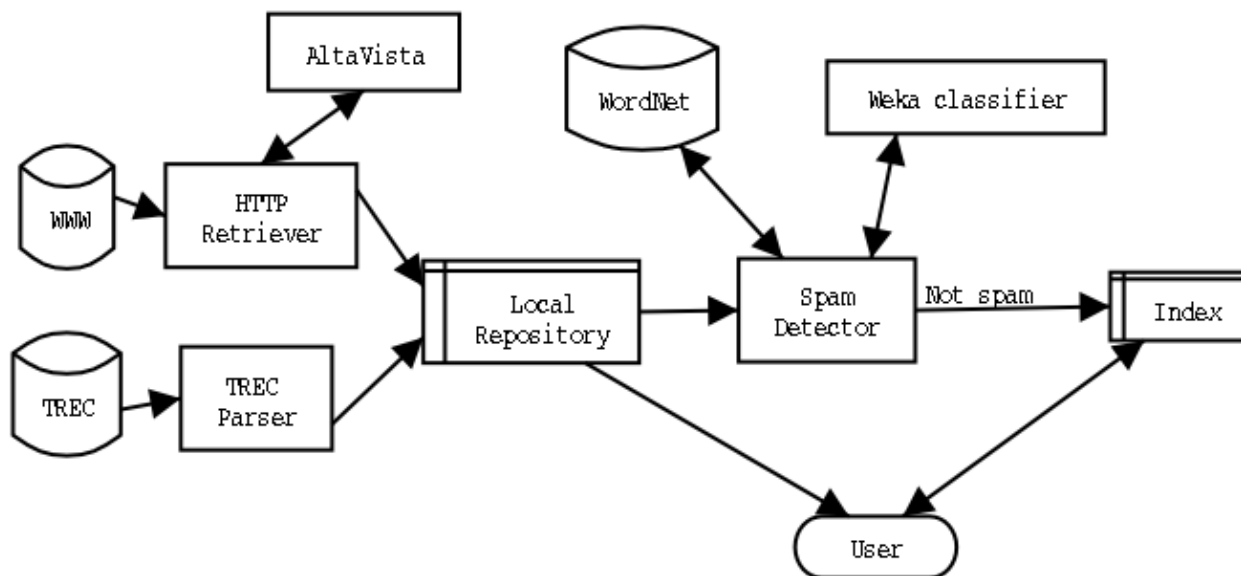


Figure 4-1: Dataflow through our architecture

3.1 Semantic Analyzers

The motivation behind our analyzers lies in the fact that written English has certain consistent statistical properties. These include everything from sentence length, to the frequency of verbs. For example, from simulations over the Information Retrieval Text Research Collection (TREC) data, we found that the average sentence length is 18 words. Intuitively this makes sense as very short or very long sentences can be awkward to read.

We have three semantic analyzers which work together to extract parameters for a given document.

3.1.1 Sentence Length

This analyzer calculates the average sentence length in a document. While a very simple concept, this can catch a variety of different types of spam documents. For example, spam documents often contain many key words in an attempt to increase the breadth of queries that will return the document. Often the words will just be placed in the webpage with no attempt at a proper syntactic structure. The sentence length analyzer will detect this practice and report a large aberration from the norm.

3.1.2 Stop Word Occurrence

The frequency of stop word occurrences in a document is measured by this analyzer. Stop words are the set of the most frequently occurring words in a collection of documents. They are given their name because most search engines ignore these terms when they are presented in a query because including them in a query does not help narrow down the results. Collecting information about stop word frequency in a document can help detect spam pages because an author trying to create spam may not include

stop words in their spam efforts because it is known that these words are filtered out in the query stage.

3.1.3 Part of Speech Analysis

The third analyzer is the most sophisticated of the set. This one uses the WordNet [13] database to ascertain part of speech information for all the words in a document with the intention of collecting frequencies of noun, verb, adjective, and adverb usage. This analyzer can be useful because spam pages often have more nouns than non-spam pages because most query terms involve nouns.

While all three analyzers report parameters on the documents, there is still the problem of using these collected parameters to determine if a given webpage is spam. We solved this problem by using the Weka [10] machine learning libraries to implement a pruned C4.5 decision tree which we fed the parameters from the three analyzers to classify documents as either spam or not spam.

4. System Architecture

We have developed software to load documents into a local file repository, index, and query those documents. Under normal operation, our indexer runs our classifier to determine if a file is spam and only indexes it if it is not. For purposes of training our classifier, we modified the indexer to write statistics about each document to a file. Figure 4.3 gives an overview of this architecture.

4.1 Loading files into the repository

In order to load TREC data files into our local repository, we developed a Java program that separates each TREC data file into separate files each composed of one article. By separating these files into articles, the TREC data more closely matches the nature of data on the World Wide Web.

		average sentence length	stopword freq.	noun freq.	verb freq.	adjective freq.	adverb freq.
Nonspam	mean	162.573	0.1882	0.178	0.075	0.1004	0.0553
	median	38	0.1952	0.1691	0.0757	0.0943	0.0531
	std dev	519.061	0.1101	0.0991	0.0486	0.0599	0.0352
Spam	mean	412.588	0.1078	0.1265	0.0594	0.07	0.0326
	median	227	0.0452	0.097	0.0537	0.0522	0.0159
	std dev	576.261	0.1112	0.0929	0.0411	0.0502	0.0303
TREC	mean	18.6785	0.3529	0.2752	0.1023	0.1846	0.0905
	median	18	0.3605	0.275	0.1016	0.1842	0.0933
	std dev	10.2198	0.0554	0.052	0.0369	0.0409	0.0309

Figure 5-1 Summary Statistics for Three Corpora

		average sentence length	stopword freq.	noun freq.	verb freq.	adjective freq.	adverb freq.
Nonspam vs Spam	mean difference	250.01	0.0804	0.0515	0.0157	0.0304	0.0266
	95% CI min	110.6	0.0531	0.0284	0.0052	0.0175	0.015
	95% CI max	389.43	0.1078	0.0747	0.0261	0.0432	0.0304

Figure 5-2 Differences in Expected Parameter Values

To load web pages into the repository, we developed a Perl program that uses AltaVista to get a list of documents matching a certain query and then download all those documents to the local file system. We used Perl because both an AltaVista scraper [2], and a HTTP fetcher [5] are freely available in this language.

We decided to store all files in our repository locally because we suspect that many of the spam documents we will be analyzing are ephemeral. (Many probably violate their ISPs terms of service; others are likely from short-lived businesses.) Using local storage, we can avoid worrying about whether a document that was available for one test will still be available for another

4.2 Indexing

Our indexing engine is based on a flexible architecture that allows us to crawl a directory tree on the local file system and process each file encountered. There are three stages of processing a file

1. Pre-process. This includes reading the file from disk and extracting the natural language portions from any markup language.

2. Spam detection. Determine the value of each classification factor for the document. Use our Weka based decision tree model to classify the document as either spam or non-spam.
3. Indexing. Add the document to a Lucene index file if it is classified as non-spam.

We have designed our system so that the pre-processing and spam detection phases are each performed by separate objects. Objects which are used to detect spam implement the SpamDetector interface. Similarly objects which are used for pre-processing implement the DocumentPreProcessor interface. This way, we can change our method of pre-processing or spam detection by using a different class of object for these roles.

4.3 Learning

In order to use a decision tree classifier, it is necessary to first train the model on sample data. We accomplished this by running the indexing process in a mode where it wrote the parameters from the semantic analyzers for each document to a file. We then manually classified each document in the collection indicating whether or not it was spam. Finally, we fed this data file into the Weka libraries to create and train the decision tree classifier.

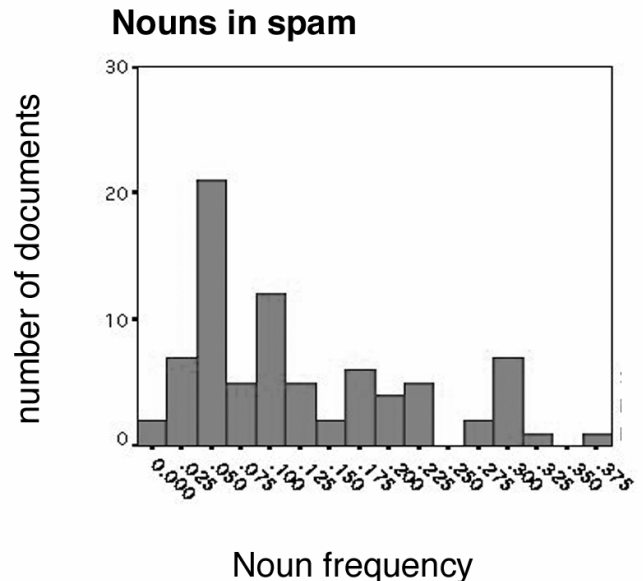
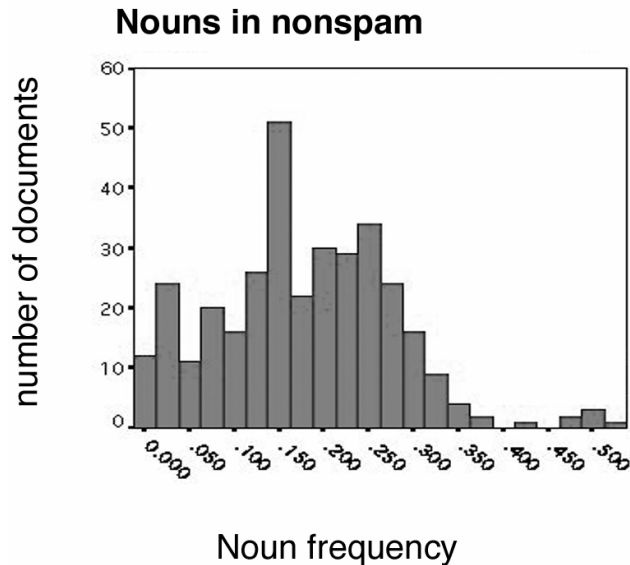
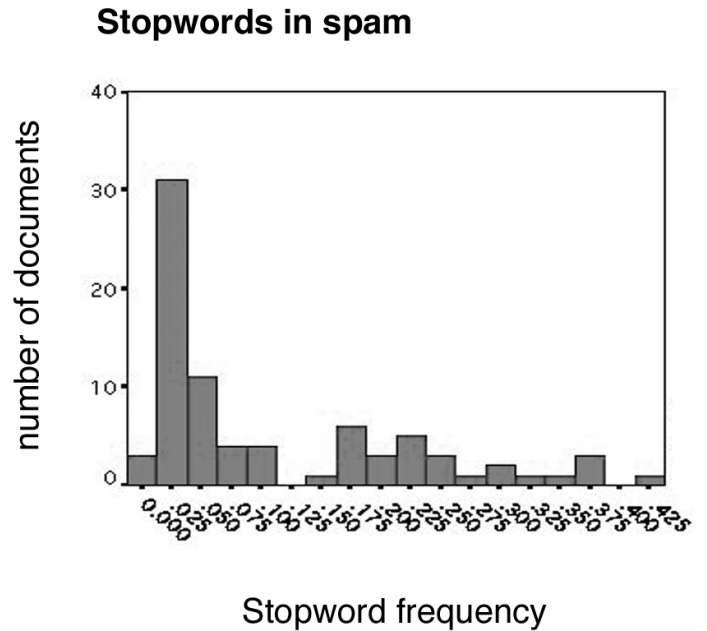
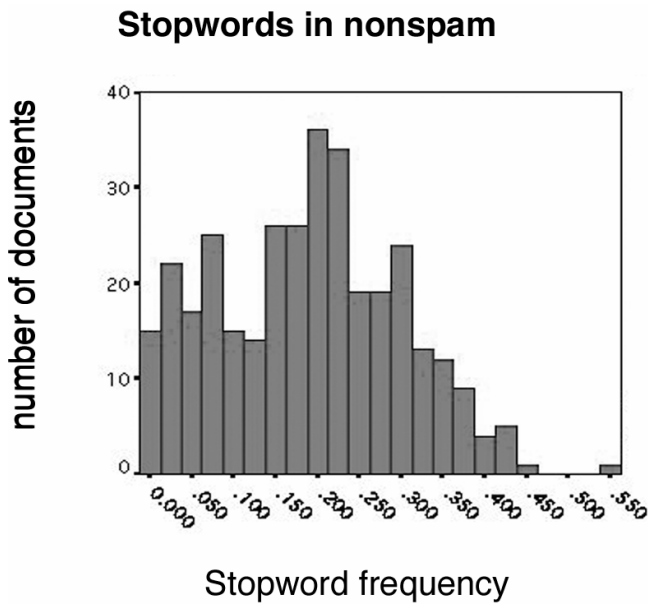


Figure 5-3: Histograms for noun and stopword frequency. Note the overlap between the distribution of spam and non-spam documents.

4.4 Query

We provide a query interface to allow users to test our system. For control purposes, this query system provides a way for users to search a Lucene index of only documents that have passed our spam detection and one of all documents, regardless of their classification.

5. Results

5.1 Initial Results

We computed summary statistics for the parameters described in section 3 on three data sets: spam, non-spam, and TREC [11]. These are shown in Figure 5-1. The spam and non-spam pages were found by performing five queries on AltaVista and manually classifying the top one hundred

results of each. We selected five queries that we thought were likely to result in a large number of spam documents: “MP3”, “breast”, “college girl”, “Apple IBM Dell Gateway”, and “ford chevy nissan toyota honda.”⁵ Of the five hundred AltaVista results, 337 were non-spam, eighty were spam, and the remaining eighty-three either could not be fetched by our HTTP retriever or were not in English. For control purposes, we also looked at 4815 TREC documents (which are known to contain only natural language and headings). These were gathered from the

⁵ The last two queries were chosen because we felt that it would be unlikely for a non-spam page to contain so many different brand names.

1989 AP data (1913 articles), the 1987 *Wall Street Journal* data (2173 articles), and DOE data (729 articles).

We used 319 of our webpages as training data to build a classifier using Weka as described in section 4. (ninety-eight documents were set aside for later use as testing data.)

After examining the tree we found that after pruning, our tree consisted of only one node, a “no” classifier. This means that for any document input, the decision tree would classify the document as non-spam. Even using the most powerful open-source decision tree implemented by Weka, we could not determine a classifier that split documents into spam and non-spam.

5.2 Analysis of initial results

In order for a classifier based on our six parameters to work, there must be some statistically significant difference between the expected value of at least one of the parameters in spam and non-spam documents. Based on an independent-sample-t-test, such a difference exists for all of the parameters with extremely high probability. The expected mean differences and their 95% confidence intervals are shown in Figure 5-2.

However, a significant difference alone does not mean a classifier can be created as there may still be substantial overlap between the distribution of a parameter for spam and the distribution of the same parameter for non-spam. That is, there is a significant range of values for each parameter such that documents scoring within that range stand a substantial chance of being both spam and non-spam. Histograms for stop word frequency and noun frequency for both spam and non-spam are shown in figure 5-3. They depict the overlap between the two corpora. Overlap is also present in other parameters, but histograms for these are not shown due to space considerations. One can also observe this overlap by noting that, for each of the parameters, the mean value for non-spam is within one standard deviation of the mean value for spam.

Due to this overlap, a classifier that selected a cutoff point for a particular parameter and classified all documents above it as spam and all those below it as non-spam (or vice-versa) would have a substantial number of false positives and/or false negatives. The scatter plot in Figure 5-4 shows how many spam and non-spam documents would be classified correctly if such a classifier based on the noun frequency parameter were used. Each point on the plot represents a potential cutoff value. Sixteen potential cutoff values were examined, ranging from the median value for spam (0.097) to the median value for non-spam (0.169). Note that none of these classifiers gets more than 80% of either type of document correct. Also, the cutoff point that is the best compromise (0.1258) only classifies 64.5% of the spam correctly and 71.1% of the non-spam correctly. Again, results for other parameters are similar (except for sentence length which does not seem to work at all as a

classifier due to its extremely large variance). This level of classification is not sufficient for filtering documents that a search engine will index.

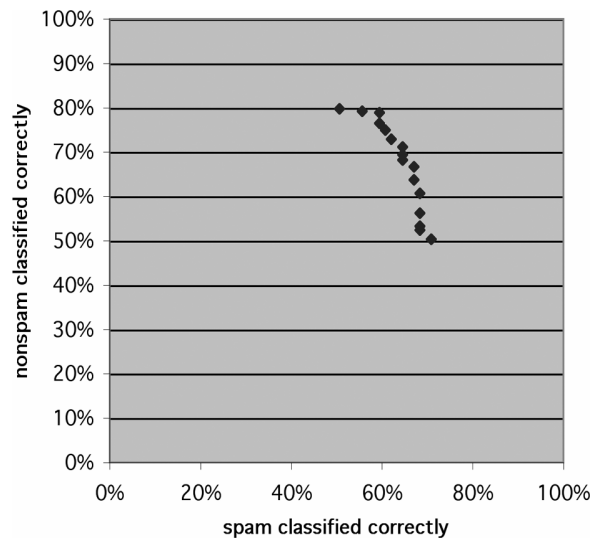


Figure 5-4 Potential cutoff points for a classifier based on noun frequencies.

5.3 Explanation of results

Our original hypothesis was that non-spam webpages would consist of a substantial quantity of natural language, while spam webpages would not. An example spam webpage is presented in figure 5-5. As you can see, the page consists of nothing more than two pictures and a list of comma separated keywords in the guise of sentences. However, as we manually classified web pages, we observed that many of the non-spam webpages we examined contained little natural language as well. Figure 5-6 shows an example of this. Here, the webpage is for car dealerships in Colorado. There are few sentences; instead, it has links for every make of car which point to a separate page for that make. This is not spam because the webpage is allowing users to refine what they are looking for by selecting a type of car.

As noted earlier, just because a page has semantic similarities to spam pages doesn't mean that its author was trying to manipulate a search engine. Web pages often contain lists of links, navigation bars, and titles/headings that to a natural language processor, would look identical to a list of spam keywords.

This effect was most likely magnified by the way we obtained our sample population. Because search engine results tend to return the main page for a site rather than pages within the site, our sampling from this population of pages resulted in many more pages containing navigational elements than would a random sample of the Web.

5.4 Comparing TREC data to Web Data

Although natural language content does not seem to be a good way to differentiate between spam and non-spam, it may have other uses in ranking search results. If we assume that the user is using a search engine to find information relating to a query, then the user would rather go to a page containing information, as opposed to one containing only links to information. With this in mind, ranking pages by their natural language content might be able to improve the search engine. We hypothesize that the parameters described in section 3 would be a good way to develop a classifier to distinguish natural language documents.

In order to test this hypothesis, we built another C4.5 decision tree classifier using Weka. This time, we classified all TREC documents under the classification trec and all webpages as non-trec. To test this new classifier we ran both the original test set of 98 webpages and a new set of 125 TREC articles from the 1989 AP data set. All of the TREC data was correctly classified as trec (100% correct) and 64 of the web pages were classified as non-trec (65.4% correct).

More promising was the fact that many of the webpages classified as trec contained proportionally large amounts of natural language data. For example the classifier classified the webpage <http://www.svtc.org/cleancc/pubs/2001report.htm> as TREC data. Examining the webpage, one finds that it contains very few links, and 14 pages of pure text.

► Facts about Leasing a car or truck

Click on the Make below and contact Colorado Auto Dealers.

Acura dealers	BMW Dealers	Buick dealers
Cadillac Dealers	Chevy dealers	Chrysler Dealers
Dodge dealers	Ford Dealers	GMC DEALERS
Honda Dealers	Hyundai Dealers	Infiniti Dealers
Isuzu Dealers	Jeep dealers	Lexus Dealers
Lincoln/Mercury	Mazda Dealers	Mercedes Benz dealers
Mitsubishi Dealers	Nissan dealers	Oldsmobile Dealers
Pontiac dealers	Subaru Dealers	Toyota Dealers
VW dealers	Volvo dealers	Used car dealers
Subaru Dealers	Suzuki Dealers	Kia Dealers

AutoNetUSA.com

AutoNetUSA.com

Questions or Comments E-Mail info@coloradocarddealers.com
 Copyright 1999-2001 AutoNetUSA, Inc.
 Colorado car dealers is maintained by COAutoNet.com Service Center (303) 751-9892

Figure 5-6 Non-spam document with high link structure

sham nissan toyota honda ford chevy auto parts accessories Page 1 of 3

sham nissan toyota honda ford chevy auto parts accessories

Browse Parts Replacement Accessories Performance. Chilton's Manuals Brands Partners Tires, Tire Rack. Customize your vehicle with an array of accessories from bulbs to bike-racks, alarm systems and more. PARTS FOR ALL VEHICLES. Choose a Make. Acura, Alfa Romeo, AM General, AMC, Audi, Bentley, BMW, Buick, Cadillac, Chevrolet, Chrysler, Crosley, Dacia, Daihatsu, De Soto, Dodge, Eagle, Edsel, Fiat, Ford, Frazer, General Motors, Geo, GMC, Henry J, Hiltman, Honda, Hudson, Hyundai, Infiniti, International, Isuzu, Jaguar, Jeep, Kaiser, Kia, Land Rover, Lexus, Lotus, Lincoln, Mazda, Mercedes, Mercury, Merkur, MG, Mitsubishi, Nash, Nissan, Oldsmobile, Opel, Packard, Peugeot, Plymouth, Pontiac, Porsche, Rambler, Renault, Rolls Royce, Saab, Saturn, Studebaker, Subaru, Sunbeam, Suzuki, Toyota, Triumph, Volkswagen, Volvo, Yugo.



You have chosen to view accessories for all vehi. Select products below or choose your vehicle to your search. Additives Chemical Cleaners, Coola Cooling Additives Oil, Oil Treatments Lubricants Sealants and Adhesives, Alarms Security alarms Horns, Alarms, Security Products, Car Care, Car Wash car Wax, Chrome finish Other Dress Up Accessories, Battery Chargers Accessories, Balle Accessories, batteries, Battery Trays Boxes Boots Cables Cleargets Bed Protection Liners, bed liner Bedliners Bedmats Tie Downs, Ropes Straps Box Ground Effects Kits Body Kits, Air Dams, Flares Front Spoilers Bug Guards Air Deflectors Cowls Ground Effects Roll Pans Wings Rear Spoilers B Manuals Software Repair Manuals Books Bump Guards Bumpers Roll Bars Grills Grill Guards Li Guards Accessories Covers Tops Car Truck Cove Convertible Tops Soft Tops Misc. Exterior Cove Tops Electronics Audio Accessories Cellular Pbc Cruise Control Power Accessories Exterior Accessories Bike Racks, Ski Racks Roof Carrier Systems Brac License Plate Frames Louvers, Vis Vents Luggage Racks, Roof Racks Van Ladders Mirrors Exterior Mud Flaps Sunroofs Moon Roo Downis Ropes Straps Tool Boxes Gauges Air Fue Radio Monitor Electrical Gauges Face Plates Gau Accessories Mechanical Gauges Mounting Caps Panels Pods Shift Lights Warning Lights Speedometers Tachometers Hardware Fasteners



http://www.your1stopshopping.com/auto_parts.htm 12/3/2002

Figure 5-5 Spam document

6. Conclusions and Future Work

Due to the similarities between spam and non-spam our original semantic analyzers are not an effective method to classify spam content. Since spam and non-spam documents are so similar, it is sometimes very difficult for a human to differentiate between the two. Because of these similarities, it is unlikely that any natural language analysis method will be successful in differentiating between spam and non-spam.

However, using semantic analyzers to determine the usefulness of information on a webpage had much more promising results. Assuming the user is more interested in finding a quick answer to their query, a page with more textual information should have a higher rank. Our analyzers could help to determine this rank.

In order to better classify web documents it is our belief that it is necessary to take advantage of the meta-information that is included in the html as well as the link structure. With this extra information at hand, a spam analyzer will have a better chance of being able to classify spam vs. one that only looks at plain text.

7. REFERENCES

- [1] Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, and Constantine D. Spyropoulos. "An Experimental Comparison of Naive Bayesian and Keyword-Based Anti-Spam Filtering with Personal E-mail Messages." In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval* (July 2000).
- [2] John Heidemann. The AltaVista WWW::Search module. (Perl module.) http://www.isi.edu/~johnh/SOFTWARE/WWW_SEARCH_ALTAVISTA/
- [3] Krishna Bharat and George A. Mihala. "When Experts Agree: Using Non-Affiliated Experts to Rank Popular Topics." *ACM Transactions on Information Systems* (January 2002). 47-58.
- [4] Larry Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. "The PageRank Citation Ranking: Bringing Order to the Web." 1999. Technical Report, Stanford University. <http://dbpubs.stanford.edu/pub/1999-66>.
- [5] libwww-perl. (Perl module.) Gisle Aas. <http://www.linpro.no/lwp/>
- [6] Choon Yang Quek. "Classification of World Wide Web Documents." 1997. Senior Honors Thesis, Carnegie Mellon University. <http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-11/www/wwkb/choon-thesis.html>.
- [7] Spam Assassin. (Software.) <http://www.spamassassin.org/>.
- [8] Danny Sullivan, ed. "Search Engine Placement Tips." Last updated October 14, 2002. <http://searchenginewatch.com/webmasters/tips.html>.
- [9] Danny Sullivan, ed. "Google Sued over PageRank Decrease." November 4, 2002. The Search Engine Report. <http://searchenginewatch.com/sereport/02/11-searchking.html>.
- [10] Ian Witten and Eibe Frank. Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation. Morgan Kaufmann, 1999.
- [11] Text Research Collection Volume I. (Data set.) Revised March 1994. National Institutes of Standards and Technology
- [12] "Web Marketing Now: Search Engine Tips." Webpage, copyright 2001. <http://www.webmarketingnow.com/tips/search-engine-cops.html>.
- [13] WordNet. (Software library and database.) Cognitive Science Laboratory at Princeton University. <http://www.cogsci.princeton.edu/~wn/>