CS 8803 – AIAD
Prof Ling Liu

Project Proposal for Automated Classification of Spam Based on Textual Features

Gopal Pai

Under the supervision of
Steve Webb

Motivations and Objectives

Spam, which was until recently a characteristic which was associated only with emails has spilled over to the web. Web Spam is a new problem which is prevalent due to people trying to find ways around search engine algorithms in the aim of getting their website ranked higher on search engines. In many cases, the content on these sites may be inferior and it may either not be of any benefit to the user to visit these sites or it may even be harmful if the site is malicious. This definition is a hazy one and classification of pages as spam may vary from person to person.

In this project, we attempt to design a system which classifies a page based on its content into distinct categories of spam and ham. This system if successfully built would also provide a method to generate a heuristic for the quality of a page without having to store huge crawls and process these using multiple servers in real time. Although such a methodology might not be as reliable as a rank got from a regular web crawl, depending on its robustness, it might possibly be put to various other uses which follow.

- Email Spam Detection - This can be used in a application which can feed back into a email spam classifier. For every email which has a link in it, a system which uses a heuristic such as the one specified by us can assign a quality metric to the link in the email. Depending on the quality of links in the mail, the mail can be classified as spam / non spam.

- Personalized ratings for web pages - This can be used in an application where some knowledge can be gained from the pages that a user likes / dislikes pages once the system has enough data, it can associate ratings automatically for pages that the user browses. The data for the application depends on the user manually classifying a set of pages as spam or ham. Also a personalized search can be created using this where we can have an intermediate analyzing layer between the normal search results and the user so that the results can be reordered so that the ones which are most likely to be appealing to the user are on top.

- This technique can be extended into classification into classes other than spam and ham. For example, detecting whether a page is safe for kids is another problem which can be tackled using this.

Related Work

This project follows the efforts of Fetterly et al [1] and Drost et al [2]. In [1], the authors investigated a set of metrics pertaining to web pages and showed that for those metrics, spam pages follow a definite pattern which is distinct from pages which can be termed as ham. They proved that it is possible to detect distinct outliers in the graph plots of these metrics which can be termed as spam. As metrics in this study, they used features like length of hostnames, host to machine ratios , out links from a page and inlinks. They concluded that the graph structure of the web is not the only criteria to classify pages, but that simpler features like these could also be used to build a heuristic ranking system. In [2], the authors tried to build a simple system to classify link spam. We attempt to build a more robust system using a greater amount of features from the web pages and build a similar system which can classify web pages into spam and ham.

Machine Learning has also been used to improve search results by filtering Nepotistic links in [4]

and work of classifying web sites into different categories has been performed in [5]. Similar work has also been done in the field of classifying emails into these categories by the process of reading tokens from the mail, using the headers from the mail as various features using which a support vector machine based classifier can be built to segregate the messages into distinct categories. In email spam classification, using a black list and white list of servers and checking the frequency of mails from a particular ip are also used as other categories using which spam is flagged.

The only problem associated with such a learning system is that a reasonably large database of previously classified mails is required to generate the first model for the system. We define a simple technique for obtaining such a database and also define a set of metrics which might be useful and propose the creation of such a system for web pages instead of emails.

Proposed Work

To start with the system we need a set of pages which we can use as a training set for our classifier. We now discuss the source for this training set. In order to get preclassified spam pages, we use an email spam archive being maintained at Georgia Tech. We extract links from these spam emails and we assume that they are low quality pages in terms of web rank as well. For the preclassified ham pages, we use a list of URLs which have been classified with a high value by James Caverlee in his Source Rank Research.

Once we have this list of URLs we intend to crawl them and save the HTML page corresponding to each of them.

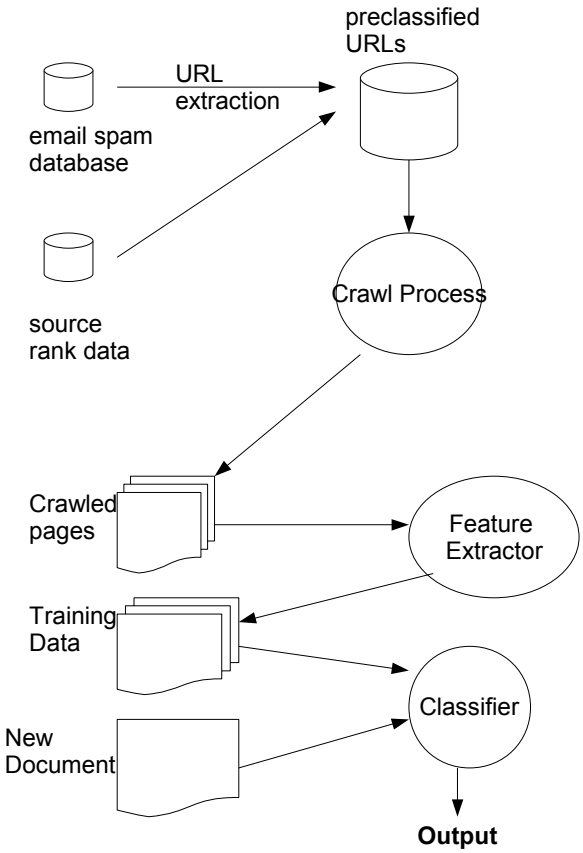
We have started with a broad list of metrics which would feed as features from these pages into the classifier.
These features can be classified into the following categories.
- Meta Data on the Page
- Tags on the web page
- Content in the web page
- Whois information of the domain
- Domain Information
- Inlinks to the page

Currently we have about 50 features without taking into account the text associated with the page.

Our first attempt will be to grab all of these metrics from the set of pages which we have crawled. We will then experimentally figure out the metrics which are most distinct in terms of spam and ham pages. We will use these set of features as the important ones in our classification phase.

Once the main set of features are got, we feed the data into a custom classifier written using WEKA [3]. Weka is a collection of open source machine learning algorithms for data mining tasks. It contains tools for data pre-processing, classification, regression, clustering, association rules . We intend to use different classification techniques to build a model around the existing data. Naive Bayesian, Support Vector Machines with hard and soft bounds and with varying kernels are different instances of learning techniques whose behavior we intend to study on this data. At this point, new data can be fed into the system and we can monitor the success or failure of this application based on the correctness of the classification which we can extract from our tool for each of the classifiers used

Graphical depiction of the flow of the process



Plan of Action

The resources required for the computing process of this project are in the following phases
1 - Crawling of spam URL's
2 - Crawling of ham URL's
3 - Extraction of features from the crawled documents
4 - Using the Features for building a classification model
5 - Classification of new data using the classification model

Of the tasks above, the crawl process is a one time activity and is not a task which we expect to be able to perform regularly. Consequently, efficiency is not a key concern in this activity. We expect to extract 500,000 and crawl URL's for both spam and ham pages.

Extraction of features from the documents will be a incremental process. In the inital extraction, we

can extract all the features which we have mentioned above. Any extra features will have to added to the set later by a seperate process. The features will be stored in xml files which will make it easy to access and restructure them for further use.

However one feature which we expect problems with is that of the inlinks to the page. The API's provided by search engines like google do not permit such a large number of queries (of the order of 1 million) to be made to it. Due to these constraints, it is possible that we might have to not include this metric in the final set which we evaluate.

The classification building part and the classification of new pages is the next part of the process and as mentioned, this will be conducted using WEKA. During this critical phase, we shall evaluate each of the metrics in the set and determine which of these can be used. We will proceed in this phase in two sub phases. In the first we shall evaluate all the metrics apart of the tokens on the page , which would lead to a explosion of features and in the second we will evaluate the tokens and determine which of the tokens can be termed as useful.

A rough bi-monthly milestone set for the project is as follows,
Feb end - Extract the set of URLs from spam mails and get the ham URL's and Crawl the web pages
March mid - Extract the features of the crawled pages
March End - Start of classifier coding and Evaluation of metrics apart from the text tokens on the page
Apr Mid - Evaluate the important textual tokens on the page.
Apr End - Have basic framework for classification of new pages

Evaluation and Testing Methodology

In order to test the correctness of the various classification methods, we could perform a random samplings of the spam and ham pages and build the models on these alone. Then we could test the models builts against the rest of the data sets and check for the accuracy of classification. The size of the sample used will have to be determined experimentally and the results will feed in directly to the classifier. We intend to start with smaller samples of 1000's of pages and scale upwards until we get an optimal performance.

As a second level of testing, we could build a framework to build a random walk of the web and classify all of these pages. Some manual verification will have to be done on these pages later to determine how effective this technique has been.

Bibliography

[1] D Fetterly, M Manasse, M Najork, “Spam Damn Spam and Statistics” in Proceedings of the 7th International Workshop on the Web and Databases: colocated with ACM SIGMOD/PODS 2004 WebDB '04
[2] I Drost, T Scheffer, “Thwarting the Nigritude Ultramarine: Learning to Identify Link Spam” in Proceedings of the European Conference on Machine Learning. 2005
[3] Ian H. Witten and Eibe Frank (2005) "Data Mining: Practical machine learning tools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[4] B. D. Davison, "Recognizing Nepotistic Links on the Web," in Artificial Intelligence for Web Search, pp. 23--28, 2000.
[5] E. Amitay et al., "The Connectivity Sonar: Detecting Site Functionality by Structural Patterns," in Proceedings of the 14th ACM Conference on Hypertext and Hypermedia, pp. 38--47, 2003.