International Journal of Engineering Research and General Science Volume 2, Issue 5, August-September, 2014
ISSN 2091-2730

Blog-Spam Detection Using intelligent Bayesian Approach

- Krushna Pandit, Savyasaachi Pandit

Assistant Professor, GCET, VVNagar, E-mail- pandit.krushna11@gmail.com , M - 9426759947

Abstract

Blog-spam is one of the major problems of the Internet nowadays. Since the history of the internet the spam are considered a huge threat to the security and reliability of web content. The spam is the unsolicited messages sent for the fulfillment of the sender's purpose and to harm the privacy of user, site owner and/or to steal available resource over the internet (may be or may not be allocated to).For dealing with spam there are so many methodologies available. Nowadays the blog spamming is a rising threat to safety, reliability, & purity of the published internet content. Since the search engines are using certain specific algorithms for creating the searching page-index/rank for the websites (i.e. google-analytics), it has attracted so many attention to spam the SMS(Social Media Sites) for gaining rank in order to increase the company's popularity. The available solutions to malicious content detection are quite a more to be used very frequently in order to fulfill the requirement of analyzing all the web content in certain time with least possible "false positives". For this purpose a site level algorithm is needed so that it can be easy, cheap & understandable (for site modifiers) to filter and monitor the content being published. Now for that we use a "Bayes Theorem" of the "Statistical approach".

Keywords— Spam-detection, Bayesian approach, blog-spam, autonomous spam detection, text mining

INTRODUCTION

Spam is the use of electronic-messaging systems to send unsolicited bulk messages, especially advertising, indiscriminately. While the most widely recognized form of spam is e-mail spam, the term is applied to similar abuses in other-media: IM-instant messaging spam, Usenet newsgroup spam, Web search engine spam, spam in blogs, wiki spam, online classified ads spam, mobile phone messaging spam, Internet forum spam, junk fax transmissions, social networking spam, Social spam etc.[3]

Spamtraps are often email addresses that were never valid or have been invalid for a long time that are used to collect spam. An effective spamtrap is not announced and is only found bydictionary attacks or by pulling addresses off hidden webpages.[4] For a spamtrap to remain effective the address must never be given to anyone. Some black lists, such as spamcop, use spamtraps to catch spammers and blacklist them.

Enforcing technical requirements of the Simple Mail Transfer Protocol (SMTP) can be used to block mail coming from systems that are not compliant with the RFC standards.[2] A lot of spammers use poorly written software or are unable to comply with the standards because they do not have legitimate control of the computer sending spam (zombie computer). So by setting restrictions on the mail transfer agent (MTA) a mail administrator can reduce spam significantly, such as by enforcing the correct fall back of Mail eXchange (MX) records in the Domain Name System, or the correct handling of delays (Teergrube).

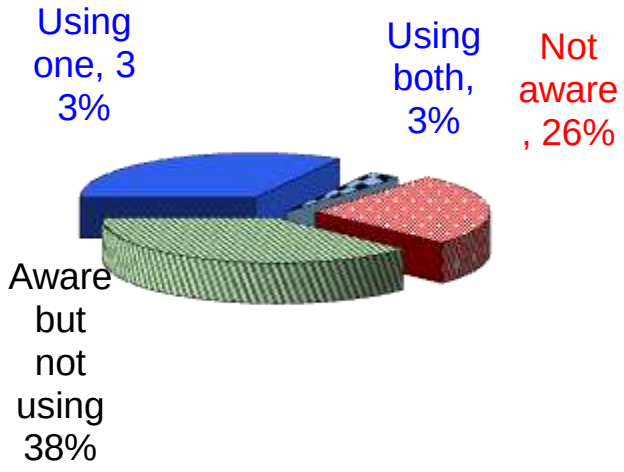

Fig- Current Scenario for Spam-Awareness [By IDA Singapore Inc.][12]

Spam-detection

Given that the objective of web spam is to improve the ranking of select search results, web spamming techniques are tightly coupled to the ranking algorithms employed (or believed to be employed) by the major search engines. As ranking algorithms evolve, so will spamming techniques.[1] For example, if web spammers were under the impression that a search engine would use click-through information of its search result pages as a feature in their ranking algorithms, then they would have an incentive to issue queries that bring up their target pages, and generate large numbers of clicks on these target pages. Furthermore, web spamming techniques evolve in response to countermeasures deployed by the search engines.[6] For example, in the above scenario, a search engine might respond to facetious clicks by mining their query logs for many instances of identical queries from the same IP address and discounting these queries and their result click through/s in their ranking computation.[9] The spammer in turn might respond by varying the query (while still recalling the desired target result), and by using a "bot-net" (a network of third-party computers under the spammer's control) to issue the-queries and the click -throughs on the target results.

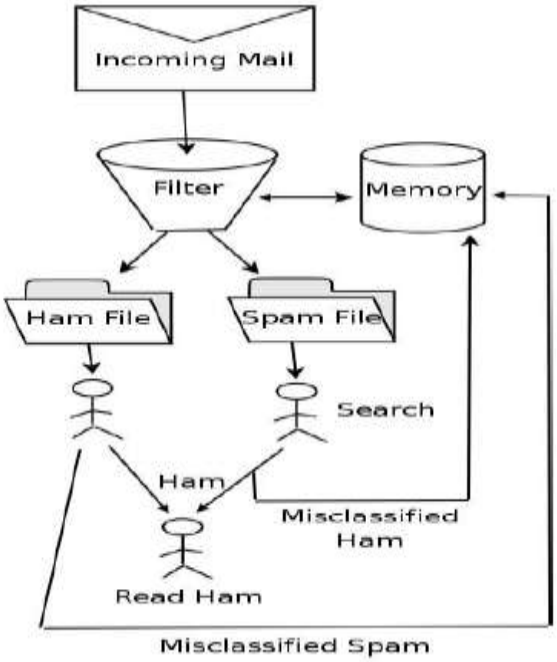

Fig- Spam Detection Model(Message content)

Given that web spamming techniques are constantly evolving, any taxonomy of these techniques must necessarily be ephemeral, as will be any enumeration of spam detection heuristics.[5]
However, there are a few constants:

- Any successful web spamming technique targets one or more of the features used by the search engine's ranking algorithms.
- Web spam detection is a classification problem, and search engines use machine learning algorithms to decide whether or not a page is spam.

In general, spam detection heuristics look for statistical anomalies in some of the features visible to the search engines.

BLOG SPAMMING

Spam in blogs (also called simply **blog spam, comment spam,** or **social spam**) is a form of spam-dexing- done by automatically posting random comments or promoting commercial services to blogs, wikis, guest-books, or other publicly accessible online discussion boards.[11] Any web application that accepts and displays hyperlinks submitted by visitors may be a target.

Adding links that point to the spammer's web site artificially increases the site's search engine ranking.[9] An increased ranking often results in the spammer's commercial site being listed ahead of other sites for certain searches, increasing the number of potential visitors and paying customers. There are various solutions available for spam-detection, some of them are described as below[2]:

1. **General spam-avoidance approaches**

Disallowing multiple consecutive submissions

It is rare on a site that a user would reply to their own comment, yet spammers typically do. Checking that the user's IP address is not replying to a user of the same IP address will significantly reduce flooding. This, however, proves problematic when multiple users, behind the same proxy, wish to comment on the same entry. Blog Spam software may get around this by faking IP addresses, posting similar blog spam using many IP addresses.

Blocking by keyword

Blocking specific words from posts is one of the simplest and most effective ways to reduce spam. Much spam can be blocked simply by banning names of popular pharmaceuticals and casino games.{{Citation needed|date=September 2012}

This is a good long-term solution, because it's not beneficial for spammers to change keywords to "vi@gra" or such, because keywords must be readable and indexed by search engine bots to be effective.[4]

nofollow

Google announced in early 2005 that hyperlinks with rel="nofollow" attribute[4] would not be crawled or influence the link target's ranking in the search engine's index. The Yahoo and MSN search engines also respect this tag.

Using rel="nofollow" is a much easier solution that makes the improvised techniques above irrelevant. Most weblog software now marks reader-submitted links this way by default (with no option to disable it without code modification). A more sophisticated server software could spare the nofollow for links submitted by trusted users like those registered for a long time, on awhitelist, or with a high karma. Some server software adds rel="nofollow" to pages that have been recently edited but omits it from stable pages, under the theory that stable pages will have had offending links removed by human editors.

Other websites like Slashdot, with high user participation, use improvised nofollow implementations like adding rel="nofollow" only for potentially misbehaving users. Potential spammers posting as users can be determined through various heuristics like age of registered account and other factors. Slashdot also uses the poster's karma as a determinant in attaching a nofollow tag to user submitted links.

rel="nofollow" has come to be regarded as a microformat.

Validation (reverse Turing test)

A method to block automated spam comments is requiring a validation prior to publishing the contents of the reply form. The goal is to verify that the form is being submitted by a real human being and not by a spam tool and has therefore been described as a reverse Turing test. The test should be of such a nature that a human being can easily pass and an automated tool would most likely fail.[5]

Many forms on websites take advantage of the CAPTCHA technique, displaying a combination of numbers and letters embedded in an image which must be entered literally into the reply form to pass the test. In order to keep out spam tools with built-in text recognition the characters in the images are customarily misaligned, distorted, and noisy. A drawback of many older CAPTCHAs is that passwords are usually case-sensitive while the corresponding images often don't allow a distinction of capital and small letters. This should be taken into account when devising a list of CAPTCHAs. Such systems can also prove problematic to blind people who rely on screen readers. Some more recent systems allow for this by providing an audio version of the characters. A simple alternative to CAPTCHAs is the validation in the form of a password question, providing a hint to human visitors that the password is the answer to a simple question like "The Earth revolves around the... [Sun]". One drawback to be taken into consideration is that any validation required in the form of an additional form field may become a nuisance especially to regular posters. One self published original research noted a decrease in the number of comments once such a validation is in place.

Disallowing links in posts

There is negligible gain from spam that does not contain links, so currently all spam posts contain (an excessive number of) links. It is safe to require passing Turing tests only if post contains links and letting all other posts through. While this is highly effective,

spammers do frequently send gibberish posts (such as "ajliabisadf ljibia aeriqoj") to test the spam filter. These gibberish posts will not be labeled as spam. They do the spammer no good, but they still clog up comments sections.

2. Distributed approaches

This approach is very new to addressing link spam. One of the shortcomings of link spam filters is that most sites receive only one link from each domain which is running a spam campaign. If the spammer varies IP addresses, there is little to no distinguishable pattern left on the vandalized site. The pattern, however, is left across the thousands of sites that were hit quickly with the same links.

A distributed approach, like the free LinkSleeve uses XML-RPC to communicate between the various server applications (such as blogs, guestbooks, forums, and wikis) and the filter server, in this case LinkSleeve. The posted data is stripped of urls and each url is checked against recently submitted urls across the web. If a threshold is exceeded, a "reject" response is returned, thus deleting the comment, message, or posting. Otherwise, an "accept" message is sent.[7]

A more robust distributed approach is Akismet, which uses a similar approach to LinkSleeve but uses API keys to assign trust to nodes and also has wider distribution as a result of being bundled with the 2.0 release of WordPress. They claim over 140,000 blogs contributing to their system. Akismet libraries have been implemented for Java, Python, Ruby, and PHP, but its adoption may be hindered by its commercial use restrictions. In 2008, Six Apart therefore released a beta version of their TypePad AntiSpam software, which is compatible with Akismet but free of the latter's commercial use restrictions.

Project Honey Pot has also begun tracking comment spammers. The Project uses its vast network of thousands of traps installed in over one hundred countries around the world in order to watch what comment spamming web robots are posting to blogs and forums. Data is then published on the top countries for comment spamming, as well as the top keywords and URLs being promoted.

3. Application-specific anti-spam methods

Particularly popular software products such as Movable Type and MediaWiki have developed their own custom anti-spam measures, as spammers focus more attention on targeting those platforms.[8] Whitelists and blacklists that prevent certain IPs from posting, or that prevent people from posting content that matches certain filters, are common defenses. More advanced access control lists require various forms of validation before users can contribute anything like link-spam. The goal in every case is to allow good users to continue to add links to their comments, as that is considered by some to be a valuable aspect of any comments section.

PROPOSED ALGORITHM/ SOLUTION:

1. Scan entire text(including headers, html & javascripts) for retrieving tokens
2. Map the tokens in 2 hash-tables(one for each corpus)
3. Count number of times each token occurs in each corpus(ignore the case)
4. Create 3rd hash-table for mapping each token to the probability (that the input message is a spam)
   (let ((g (* 2 (or (gethash word good) 0))) (b (or (gethash word bad) 0))) (unless (< (+ g b) 5) (max .01 (min .99 (float (/ (min 1 (/ b nbad)) (+ (min 1 (/ g ngood)) (min 1 (/ b nbad)))))))))

   - **word** is the token whose probability is to be-calculated

   - **good** & **bad** are the hash tables created in 1st step
   - arc formula

5. The most interesting 15 tokens are used to calculate the probability of message to be a spam(use combined probability)
6. Treat the mail as spam if the probability of spam is calculated to be more than 0.9

RESULTS

In Spam-detection it is to evaluate 2 ratios for calculating correctness of an algorithm.

False positive rate

The **false positive rate** is the proportion of absent events that yield positive test outcomes, i.e., the conditional probability of a positive test result given an absent event.

The false positive rate is equal to the significance level. The specificity of the test is equal to **1** minus the false positive rate.[9]

In statistical hypothesis testing, this fraction is given the Greek letter α, and $1 - \alpha$ is defined as the specificity of the test. Increasing the specificity of the test lowers the probability of **type I errors**, but raises the probability of **type II errors** (false negatives that reject the alternative hypothesis when it is true).

False negative rate

The **false negative rate** is the proportion of events that are being tested for which yield negative test outcomes with the test, i.e., the conditional probability of a negative test result given that the event being looked for has taken place.[9]

In statistical hypothesis testing, this fraction is given the letter β. The "power" (or the "sensitivity") of the test is equal to $1 - \beta$.

Checking words: false positives

People tend to be much less bothered by spam slipping through filters into their mail box (false negatives), than having desired email ("ham") blocked (false positives). Trying to balance false negatives (missed spams) vs false positives (rejecting good email) is critical for a successful anti-spam system. Some systems let individual users have some control over this balance by setting "spam score" limits, etc. Most techniques have both kinds of serious errors, to varying degrees. So, for example, anti-spam systems may use techniques that have a high false negative rate (miss a lot of spam), in order to reduce the number of false positives (rejecting good email).

Detecting spam based on the content of the email, either by detecting keywords such as "viagra" or by statistical means (content or non-content based), is very popular. Content based statistical means or detecting keywords can be very accurate when they are correctly tuned to the types of legitimate email that an individual gets, but they can also make mistakes such as detecting the keyword "cialis" in the word "specialist" (see also Internet censorship: Over- and under-blocking). The content also doesn't determine whether the email was either unsolicited or bulk, the two key features of spam. So, if a friend sends you a joke that mentions "viagra", content filters can easily mark it as being spam even though it is neither unsolicited nor sent in bulk. Non-content base statistical means can help lower false positives because it looks at statistical means vs. blocking based on content/keywords. Therefore, you will be able to receive the friend who sends you a joke that mentions "viagra".

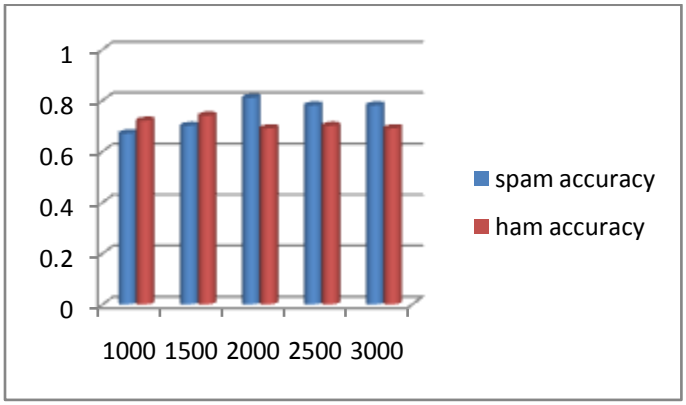

Fig – Algorithm accuracy for different input-data sizes

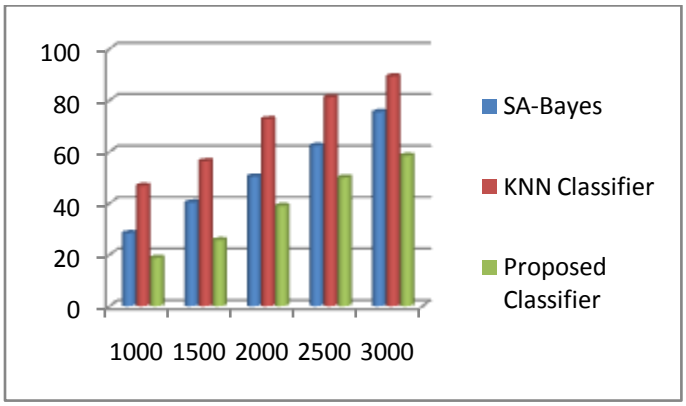

Fig – Execution time for different input-data sizes

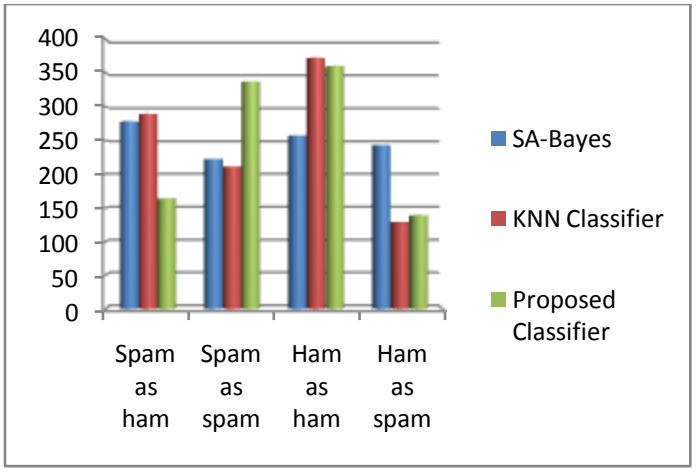

Fig –Spam-ham precision for various algorithms

Table 1 – recall & precision for classifier

Class	**recall**	**Precision**
Spam	78.5%	67.5%
Non-spam	81.5%	72%

RELATED WORK

The common e-mail client has the capability to sort incoming e-mail based on simple strings found in specific header fields, the header in general, and/or in the body. Its capability is very simple and does not even include regular expression matching. Almost all e-mail clients have this much filtering capability. These few simple Text-filters can correctly catch about 80% of the spam. Unfortunately, they also have a relatively high false positive rate.[2]

White list/verification filters[7]

A fairly aggressive technique for spam filtering is what I would call the "whitelist plus automated verification" approach. There are several tools that implement a whitelist with verification: TDMA is a popular multi-platform open source tool; ChoiceMail is a commercial tool for Windows; most others seem more preliminary.

A whitelist filter connects to an MTA and passes mail only from explicitly approved recipients on to the inbox. Other messages generate a special challenge response to the sender. The whitelist filter's response contains some kind of unique code that identifies the

original message, such as a hash or sequential ID. This challenge message contains instructions for the sender to reply in order to be added to the whitelist (the response message must contain the code generated by the whitelist filter). Almost all spam messages contain forged return address information, so the challenge usually does not even arrive anywhere; but even those spammers who provide usable return addresses are unlikely to respond to a challenge. When a legitimate sender answers a challenge, her/his address is added to the whitelist so that any future messages from the same address are passed through automatically.

Distributed adaptive blacklists

Spam is almost by definition delivered to a large number of recipients. And as a matter of practice, there is little if any customization of spam messages to individual recipients. Each recipient of a spam, however, in the absence of prior filtering, must press his own "Delete" button to get rid of the message. Distributed blacklist filters let one user's Delete button warn millions of other users as to the spamminess of the message.[4]

Tools such as Razor and Pyzor (see Resources) operate around servers that store digests of known spams. When a message is received by an MTA, a distributed blacklist filter is called to determine whether the message is a known spam. These tools use clever statistical techniques for creating digests, so that spams with minor or automated mutations (or just different headers resulting from transport routes) do not prevent recognition of message identity.[6] In addition, maintainers of distributed blacklist servers frequently create "honey-pot" addresses specifically for the purpose of attracting spam (but never for any legitimate correspondences). In my testing, I found *zero* false positive spam categorizations by Pyzor. I would not expect any to occur using other similar tools, such as Razor.

There is some common sense to this. Even those ill-intentioned enough to taint legitimate messages would not have samples of *my* good messages to report to the servers -- it is generally only the spam messages that are widely distributed.[5] It is*conceivable* that a widely sent, but legitimate message such as the developerWorks newsletter could be misreported, but the maintainers of distributed blacklist servers would almost certainly detect this and quickly correct such problems.

Rule-based rankings

The most popular tool for rule-based spam filtering, by a good margin, is *SpamAssassin.* There are other tools, but they are not as widely used or actively maintained. SpamAssassin (and similar tools) evaluate a large number of patterns -- mostly regular expressions -- against a candidate message. Some matched patterns add to a message score, while others subtract from it. If a message's score exceeds a certain threshold, it is filtered as spam; otherwise it is considered legitimate.[7]

Some ranking rules are fairly constant over time -- forged headers and auto-executing JavaScript, for example, almost timelessly mark spam. Other rules need to be updated as the products and scams advanced by spammers evolve. Herbal Viagra and heirs of African dictators might be the rage today, but tomorrow they might be edged out by some brand new snake-oil drug or pornographic theme. As spam evolves, SpamAssassin must evolve to keep up with it.

Bayesian word distribution filters

It suggested, building Bayesian probability models of spam and non-spam words. The general idea is that some words occur more frequently in known spam, and other words occur more frequently in legitimate messages. Using well-known mathematics, it is possible to generate a "spam-indicative probability" for each word. Another simple mathematical formula can be used to determine the overall "spam probability" of a novel message based on the collection of words it contains.[7]

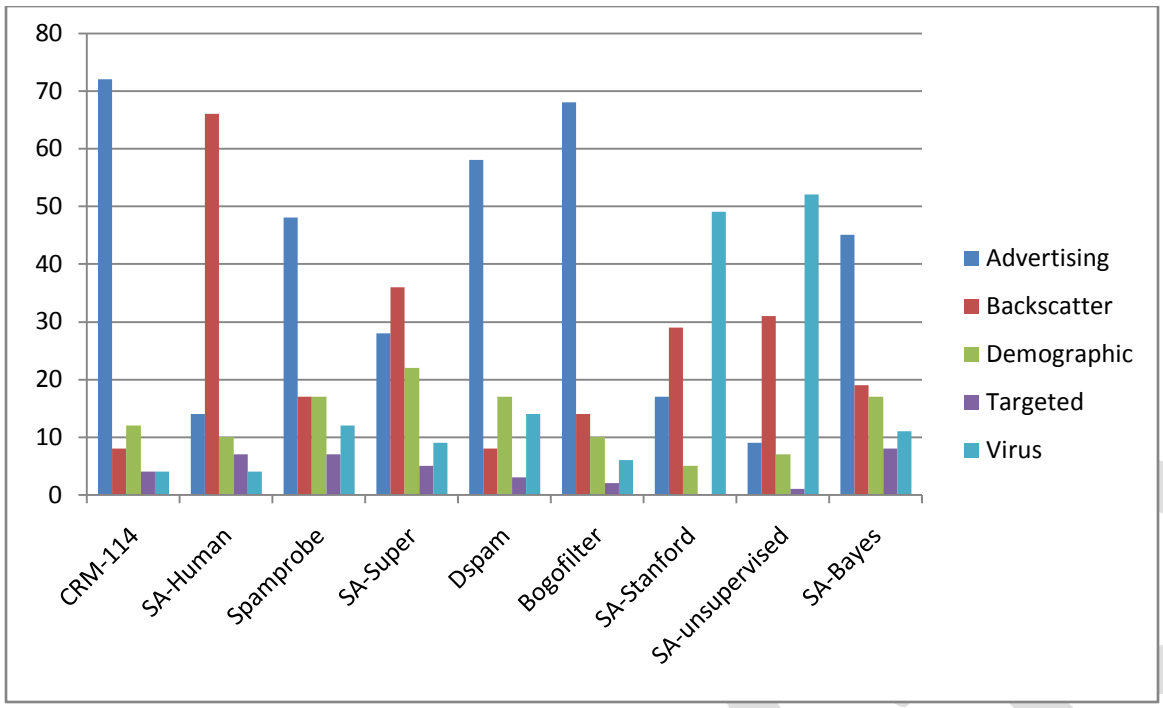

Fig- 6: The analysis of the available open source methods

PROS & CONS OF THE PROPOSED ALGORITHM:

Pros:

- Ease of usage (can be used as a simple web service)
- Cost effective solution(less memory consumption with simple structure)
- Reliable performance(no false positives)
- Better resource utilization(uses previously available content to analyze)
- Ease of tracing(for future aspects)

Cons

Requires hectic-analysis in order to attain approximate 100% accuracy, due requirement for good-word and bad-word data-corpus/s

CONCLUSION:

This approach gives comparative reduction in false-positive(ham as spam) and false negative(spam as ham) ratios w.r.t. simple Bayesian-classifier and KNN classifier. It also gives continuous and linear order of growth for increasing input-data sizes.

Thus, it can be concluded that the statistical approach as this proposed prototype, be a promising & satiable solution for our targeted domain of application or usage

REFERENCES:

[1] Nikita Priyanka Renato "A Survey on blog bot Detection Techniques" - IJARCSSE(International Journal of Advanced Research in Computer Science and Software Engineering) Dec 2013
[2] Sahil Puri, Dishant Gosain, Mehak Ahuja, Ishita Kathuria, Nishtha Jatana"Comparison and analysis of spam detection algorithms"- International Journal of Application or Innovation in Engineering & Management (IJAIEM) 2013
[3] Grabianowski "How Spam Works". HowStuffWorks. Discovery Communications. Archived from the original on September 8, 2012.
[4] "SLV : Spam Link Verification". LinkSleeve. 2012
[5] T.A Meyer and B Whateley "SpamBayes: Effective open-source, Bayesian based, email classification system"

[6] "Augmenting Naive Bayes Classifiers with Statistical Language Models" –Fuchun Peng University of Massachusetts – Amherst
[7] Enrico Blanzieri and Anton Bryl "A survey of learning-based techniques of email spam filtering"-
[8] Banit Agrawal, Nitin Kumar, and Mart Molle. Controlling spam emails at the routers. In Proceedings of the IEEE International Conference on Communications, ICC 2005, volume 3, pages 1588–1592, 2005.
[9] P Boykin and Vwani Roychowdhury. Leveraging social networks to fight spam. Computer, 38(4):61–68, 2005.
[10] Enrico Blanzieri and Anton Bryl. Evaluation of the highest probability svm nearest neighbor classifier with variable relative error cost. In Proceedings of Fourth Conference on Email and Anti-Spam, CEAS'2007, page 5 pp., 2007.
[11] Hrishikesh Aradhye, Gregory Myers, and James Herson. Image analysis for efficient categorization of image-based spam e-mail. In Proceedings of Eighth International Conference on Document Analysis and Recognition, ICDAR 2005, volume 2, pages 914–918. IEEE Computer Society, 2005.
ITU. ITU survey on anti-spam legislation worldwide. - 2005