A Dissertation

on

Bayesian Spam Classification: Time Efficient Radix Encoded Fragmented Database Approach

Submitted in the partial fulfilment of the requirement
for the award of Degree of

MASTER OF TECHNOLOGY
Computer Technology and Application (CTA)

By

NISHTHA JATANA
2K10/CTA/21
M.Tech (COMPUTER TECHNOLOGY AND APPLICATIONS)
DEPARTMENT OF COMPUTER ENGINEERING

Under the supervision of

Dr. Kapil Sharma
Associate Professor
Department of Computer Engineering
DELHI TECHNOLOGICAL UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
BAWANA ROAD DELHI-110042
INDIA
May, 2013

CERTIFICATE

This is to certify that Ms. Nishtha Jatana has carried out the work presented in this Dissertation entitled, **"Bayesian Spam Classification : Time Efficient Radix Encoded Fragmented Database Approach",** under my supervision. The report embodies result of work and studies carried out by herself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else.

Dr. Kapil Sharma
Associate Professor
Department of Computer Engineering
Delhi Technological University

ACKNOWLEDGMENT

I take the opportunity to express my sincere gratitude to the Almighty God for enabling me to be capable enough to do this work. I want to express my sincere thanks to our Head Of Department, Dr. Daya Gupta, Department of Computer Engineering, Delhi Technological University, Delhi for providing well equipped infrastructure support.

I would like to extend my earnest thankfulness to my project mentor Dr. Kapil Sharma, Associate Professor, Delhi Technological University, Delhi for providing valuable guidance and constant encouragement throughout the project.

I feel lucky to be blessed with such family and friends who were always there to support me and my endeavours. I am indebted to my elder sister to be there as a source of inspiration and for supporting me in my endeavours. I am thankful to Mr. Yogesh Bhalla, Ms. Nupur Kinger and Ms.Tanu Sharma for being there as true friends to support and help me whenever I required any kind of help. I would also like to thank two of my students Mr. Milind Rishi and Mr. Nikhil Gupta for helping me with coding in Java.

NISHTHA JATANA
2K10/CTA/21
M.Tech (COMPUTER TECHNOLOGY AND APPLICATIONS)
DEPARTMENT OF COMPUTER ENGINEERING

ABSTRACT

Spam or unsolicited email has become a major problem for companies and private users. The problems associated with spam and various approaches that attempt to deal with it, have been presented here. Statistical classifiers are one such group of methods that show adequate performance in filtering spam, based upon the previous knowledge gathered through collected and classified emails. Learning algorithms that uses the Naive Bayesian classifier have shown promising results in separating spam from legitimate mail. An encoded and fragmented database approach that resembles radix sort technique has been proposed and applied for first time to improve Paul Graham's Naive Bayes machine learning algorithm for spam filtering. The main objective of this work is to reduce overall time in the process of spam detection. Quantitative and qualitative analysis of the proposed technique, performed on two public spam databases (SpamAssasin and Ling Spam) has shown improved time performance. The proposed method has performed up to six times faster than the existing Paul Graham's Bayesian approach.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

$\boldsymbol{M}$	:	Email corpus (spam/ham)
$\boldsymbol{wb}$	:	weighing factor for bad(spam) mail
$\boldsymbol{wg}$	:	weighing factor for legitimate/good(ham) mail
$\boldsymbol{nbad}$	:	number of Emails in spam corpus
$\boldsymbol{ngood}$	:	number of Emails in ham corpus
$\boldsymbol{hb}(\boldsymbol{t})$	:	frequency shown in spam corpus by a token t
$\boldsymbol{hg}(\boldsymbol{t})$	:	frequency shown in ham corpus by a token t
$\boldsymbol{pb}(\boldsymbol{t})$	:	probability that a token t is present in M when M is ham.
$\boldsymbol{pg}(\boldsymbol{t})$	:	probability that a token t is present in M when M is spam.
$\boldsymbol{f}(\boldsymbol{t})$	:	token probability
$\boldsymbol{v}(\boldsymbol{t})$	:	array containing probability of token t.
$\boldsymbol{u}$	:	calculated decisive value if mail is spam or ham.

CHAPTER 1

Introduction

1. Introduction

In modern era, humans are heavily dependent on cyber world. Mammoth number of people use their digital identity to maintain their business and social relations. Today, 3.146 billion email accounts exist worldwide This chapter describes the significance of emails and the major threats associated with spam. Categories of spam filtering techniques are also described here.

1.1 Usage of Email

The use of email is essential when communicating in today's business culture. All kinds of Businesses can effectively use email for multiple tasks and purposes effectively. It is the largely proficient way to communicate with management, clients, colleagues, and vendors. Email is used when management, head of departments and human resources send memorandums and notifications to the company as a whole. For illustration, certain software programs might be shut down for maintenance between specific hours of the evening or night. Emails can be sent informing all employees to report about processed data, fill out time sheets or any other useful information. Email is used when colleagues of the same or different departments need to send and receive information about reports, projects, spreadsheets and research. Emailing is an enormously useful tool because it only takes a very less time to receive the answers to an inquiry.

Canned documentation, such as signed contracts and time sheets, can be sent in a PDF form. This facilitates recipients the ability to review information without altering the document. Thereafter, these documented can be printed, saved and forwarded to others. Employees can converse their needs to the administrative assistant. Businesses send out company newsletters through email. Thus enabling the employees to rapidly open and read information on company stocks, advancement, charitable donations, featured employees food drives. Employees can receive schedule of meetings, conferences, and compulsory training sessions via email.

Emails have become a pre-dominant medium of communication by masses of people to stay in touch with their friends, relatives, and family members. Emails are used for publicity of events, campaigning and even for web promotion. Notifications and bills are nowadays sent through emails to save papers and in turn save trees.

Email id are nowadays used by social networking sites like facebook and twitter for verifying the identity of people. Email has successfully reinstated postal mail and courier services as the favoured form of sending and sharing significant documents and for other communication purposes. Students make use of emails for easy submission of assignments and tutorial sheets. Emails can nowadays be accessed through smart phones anywhere and everywhere .

Important users of spam include:

- Business personnel for maintaining business relations, communication with their employees and keeping track of deals and for promotion of their products.
- Students for sharing notes, submission of assignments etc.
- Teenagers and elders use emails to stay in touch with their friends and family.
- Emails are used for sending bills and notifications.
- Email ids are used by social networking sites for verifying the identity of a person while they create their accounts.

1.2 Threats associated with spam

With the escalating popularity and heavy dependence of electronic mail on social and business life, people and companies have found it an easy way to swiftly disseminate unsolicited messages to a huge number of users at very low costs for the senders. Subsequently, unsolicited or spam emails have dramatically become a major threat that can degrade the fame associated with the electronic mail as a reliable mode of communication. Spam not only consumes considerable time and money for business users and network administrators, it also consumes network bandwidth and server storage space, slows down email servers, and provides a medium to distribute harmful and/or offensive content.

(Davis & Craney, 2001) explains that how the term spam originated from old Monty Python sketch that took place in a restaurant where everything on the menu came with spam (the food product). The word 'spam' was repeated over and over in the sketch and someone used it to coin the meaning of an unsolicited commercial post on Usenet (electronic bulletin boards, or better known as newsgroups) in the early

1990's and it stuck, much to Hormel's apprehension. One of the first spammers was Dave Rhodes, a college student who wanted to make a little extra cash with little extra effort. He sent the artery clogging fat cells in motion when he cross-posted a pyramid scheme on Usenet. It was later turned into email and file uploads to Bulletin Boards with the Header "MAKE MONEY FAST". How well Dave did with his spamming adventure is unknown.

Major threats of spam for network resources and users:

- Annoyance – receiving unsolicited message is extremely nuisance to many users as they waste time, effort and money; in addition to the possibility of carrying offensive content.
- Flooding mailboxes – waste storage space and overload the server; thus it may lead to losing legitimate emails, delaying the server response, or even make it totally unavailable.
- Wasting network bandwidth and processing time.
- Wasting time and money – to install, configure and upgrade anti-spam software
- Carry malicious codes including viruses, root kits, worms, etc.
- Spreading rumours and other fraudulent ads.
- Network attack such as phishing.
- Undermine the usability of the email system.
- Severely impact the quality of service for other legitimate traffic.

People who benefit from spam are those who sell spamming tips and tricks disguised as Internet marketing providers, the software companies that create spamming products, and Internet service providers who support spammers. Marketing to spammers is far more profitable than being a spammer. Hence, it has become an important and requisite aspect of any recent email system to incorporate a spam filtering subsystem.

1.3 Spam filtering methods

Spam filtering methods fall into two broad categories: non-machine learning based and machine learning based. Most of the early implemented anti-spam tools belonged to the first category where the users or the system administrators create rule sets based on specific attributes that characterize spam messages. This human-crafted rule set may depend on a blacklist of known spammers, a white list of trusted senders, or a heuristic set of keywords such as "Amazing offer" either in the subject line or the message content (Wang, 2004), (Jung & Sit, 2004). However, such static rules that depend on the sender address or a fixed set of keywords may not be very helpful as they can be defeated easily. A spammer can change or spoof the sender's address or domain each time. Also a spammer can deliberately avoid/misspell words or forge the content to get around such spam filters. For these methods to be effective, periodic update is required. Manually maintaining and frequently changing a large sophisticated set of rules requires a considerable amount of time and effort to analyze and devise such rules which makes it a boring task.

The success of machine learning (ML) based techniques in text categorization problems (Joachims, 1998), (Yang Y. , 1999), (Sebastiani, 1999), (Sebastiani, Sperduti, & Valdambrini, 2001), (Sebastiani, 2002) and the similarity of spam filtering to these problems have encouraged several researchers to investigate their applicability in spam filtering. Although spam filtering seems to be a simple application of the text categorization task, it has some distinct features that make it a different and challenging problem.

For this work, we have designed a statistical spam filter using Naive Bayesian approach using Paul Graham's probability function and then proposed and implemented a new and time efficient approach for spam filtering.

The spam filters were designed in Java, which was chosen because it is apt at manipulating text and has a rather extensive standard library. The token frequency counts are stored in an MSACCESS database. For testing and training two corpuses have been used on both the systems: SpamAssassin public spam corpus and LingSpam public spam corpus, which are freely available. The spam messages consist of unwanted messages, the ham emails consist of messages that are legitimate mails. Like most problems that involve machine learning, a statistical spam filter can be broken into two components: training and testing. For the project, a random 90

percent of the data was used for training and ten percent was used for testing both the spam filtering systems.

Chapter 2 reveals the existing literature for different types of problems associated with spam, various types of spam and various spam filtering techniques. Chapter 3 shows the implementation of Bayesian spam filtering approach. Chapter 4 proposes a time efficient spam filtering technique. Chapter 5 describes comparative results of the existing technique with the proposed work. Conclusion and suggestions for future work has been included at the end.

CHAPTER 2

Literature Survey

2. Literature survey

This chapter reviews the problems associated with the spam, types of spam, various spam corpuses that are available online, searching techniques used in spam filtering mechanism and various spam filtering approaches .

2.1 The problem of spam

Internet has opened new channels of communication; enabling an email to be sent to a relative thousands of kilometres away. This medium of communication has opened doors for virtually free mass emailing, reaching out to hundreds of thousands users within seconds. However, this freedom of communication has been misused. In the last couple of years spam has become a phenomenon that threatens the viability of communication via email. It was difficult to develop an accurate and useful definition of spam, although every email user will quickly recognize spam messages. Merriam-Webster Online Dictionary[1] has defined spam as "unsolicited usually commercial email sent to a large number of addresses". Some other than commercial purposes of spam are to express political or religious opinions, deceive the target audience with promises of fortune, spread meaningless chain letters and infect the receivers' computer with viruses. Even though one can argue that what is spam for one person can be an interesting mail message for another, most people have agreed that spam is a public frustration. Spam has become a serious problem (Fallows, 2003) because in the short term it is usually economically beneficial to the sender. The low cost of email as a communication medium virtually guaranties profits. Even if a very small percentage of people respond to the spam advertising message by buying the product, this can be worth the money and the time spent for sending bulk emails. Commercial spammers are often represented by people or companies that have no reputation to lose. Because of technological obstacles with email infrastructure, it is difficult and time-consuming to trace the individual or the group responsible for sending spam. Spammers make it even more difficult by hiding or forging the origin of their messages. Even if they are traced, the decentralized architecture of the Internet with no central authority made it hard to take legal actions against spammers. Spam has increased steadily over the last few years. The major problem concerning spam was that it is the receiver who is paying for the spam in terms of their time, bandwidth and disk space.

[1] Merriam-Webster Online Dictionary, http://www.m-w.com/, 2004-03-12

This can be costly even for a small company with only 20 employees who each receive 20 spam emails a day. If it takes 5 seconds to classify and remove a spam, then the company will spend about half an hour every day to separate spam from legitimate email. The statistics shows that 20 spam messages per day is a very low number for a company that is susceptible to spam. There were other problems associated with spam. Messages have content that is offensive to people and might cause general psychological annoyance, a large amount of spam messages can crash unprotected mail servers, legitimate personal emails can be easily lost and more.

There was an immediate need to control the steadily growing spam flood. A great deal of on-going research is trying to resolve the problem. However, email users were impatient and therefore there was a growing need for rapidly available anti-spam solutions to protect them.

2.2 Spam filtering

Lately, Goodman et al. (Goodman, Cormack, & Heckerman, 2007) presented an overview of the field of anti-spam protection, giving a brief history of spam and anti-spam and describing major directions of development. The authors were quite optimistic in their conclusions, indicating learning-based spam recognition, together with anti-spoofing technologies and economic approaches, as one of the measures which together will probably lead to the final victory over email spammers in the near future.

According to the study by Siponen and Stucke (Siponen & Stucke, 2006) about the use of different kinds of anti-spam tools and techniques in companies, filtering is the most popular way of protection from spam. This showed that spam filtering is, and is likely to remain, an important practical application of machine learning. The flow of control of a typical spam filter is shown in Figure 1 on the next page.

As each email arrives, the filter makes its best judgment whether or not it is spam. If a message is classified as spam, it is routed to a junk folder. All ham is moved directly to the user's inbox.

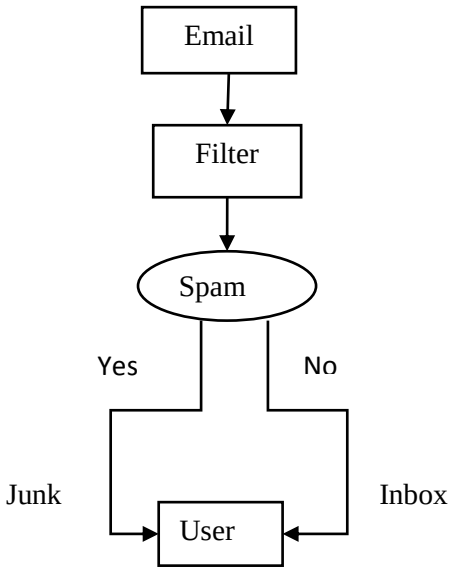

Figure 1 : Flowchart of a typical spam filter

Filters make two types of errors. False negatives are spam messages that are incorrectly passed to the inbox. False positives are ham messages that have been incorrectly classified as spam and sent to the junk folder. If a spam filter is noticeably effective, users can tolerate a few remaining spam in their inbox. However, all ham has a certain value to each user. If a single ham is misplaced or even just delayed, users are negatively affected. Spam filters strive to keep the false positive rate as low as possible. No filter is perfect though, so periodic checks of the junk folder for mistakes are recommended. Study (Sakkis G. , et al., 2001) showed that as no one filter is sufficient. So, stacking multiple filters prove to show better results.

2.3 Types of spam Email

Types of email have been categorised by (varnsen) as follows:

1) Unsolicited Advertisements

Unsolicited bulk Emails are pretty annoying as they stack up in your spam folder, but for the most part they are pretty low on the spam email ladder. Each day hundreds of billions of email advertisements are sent, most selling miracle weight loss cures, male enhancement products, knock-off merchandise, online degree programs and prescription drugs.

2) Phishing Scams

One of the hardest types of email spam to spot is phishing scam mails. These buggers are designed to look like official emails from financial institutions or big companies like eBay and PayPal, but actually direct victims to equally official looking scam sites. These tricks people into volunteering their usernames and passwords, which are then used by the site owners, the scammers, to compromise the real accounts.

3) Nigerian 419 Scams

If you have an email account, more than likely you've received a seemingly amazing offer out of the blue by some stranger from a faraway land. Someone claiming to be the agent for a long lost relative, a lottery service, an employer or even someone looking for love will offer a large sum of money, only asking for a small percentage in return for their time, insurance, shipping or other seemingly legit reason. The scammer then sends a fake check and asks for money to be wired back. Victims never receive the large sum from the check and are out the small fee, usually a few hundred or thousand dollars.

4) Email Spoofing

More of a technique used to make other email spam tactics seem more believable, many spammers will send messages which appear to originate from a different email address than they actually do. This spoofing technique makes it appear as though a fraudulent email actually came from a trusted source, company or organization. This builds the trust of the victim, making them more likely to take part in whichever scam is included in the message.

5) Trojan Horse Email

Considered ancient in the email spam history books, email worms are nasty little buggers which not only infect the victim computer but also sends itself to everyone in the victim's contact list. The most famous email worm was the ILOVEYOU bug which debuted in the year 2000. It was highly successful as who won't open an email from a loved one titled I love you? Once opened and downloaded, the script attached would damage the local computer and send itself out to everyone the victim knows.

6) Commercial Advertisements

Technically, any unsolicited bulk messages sent indiscriminately are considered spam. This includes when legit websites and companies that you use send out advertisements, newsletters and other junk messages. Most websites these days ask you if you'd like to be included in their communications however some will automatically add you to their mailing list simply for signing up for their site.

7) Anti-Virus Spam

Nobody wants a virus so when victims receive emails saying that their computer is infected, some will believe the claim out of fear. Victims think they're downloading security software but they are actually infecting computers with nasty viruses. To get rid of the virus, the software demands cash to clean up the virus it just installed.

8) Chain Letters

We all have that annoying relative who constantly sends us recycled jokes, funny photos or those sensational claims about President Obama. Despite coming from a friend or family member, this too qualifies as spam. So if Crazy Uncle Ray sends you a message with a hundred lines of forwards at the top, be warned that it's likely junk you've seen before or plain old nonsense.

9) Political or Terrorist Spam

Part scare tactic and part attempt to steal personal information, this type of email spam appears to be from a politician or well-known government office, such as the FBI, claiming that you're in danger. To clear up the threat, the Email asks the victims to fork over personal information and sometimes cash. While rarely does an actual threat ever exist, the trick does get people to volunteer their personal information to unreliable sources.

10) Porn Spam

Pornography is a huge business around the globe, used by a large percentage of the population and a leading source of malicious content. Porn spammers harvest or purchase email addresses of people, send out raunchy advertisements, then direct victims to adult sites.

2.4 Techniques to eliminate spam

There are several approaches which deal with spam. These approaches are discussed in (Kågström, 2008). This section briefly summarizes some common methods to avoid spam and describes the spam filtering techniques used at present.

1) Hiding the Email address

The simplest approach to avoid spam is to keep the email address hidden from spammers. The email address can be revealed only to trusted parties. For communication with less trusted parties a temporary email account can be used. If the email address is published on a web page it can be disguised for Email spiders[1] by inserting a tag that is requested to be removed before replying. Robots will collect the email address with the tag, while humans will understand that the tag has to be removed in order to retrieve the correct email address. For most users this method is insufficient. Firstly, it is time consuming to implement techniques that will keep the email address safe, and secondly, the disguised address could not only mislead robots, but also the inattentive human. Once the email address is exposed, there is no further protection against spam.

2) Pattern matching, whitelists and blacklists

This is a content-based pattern matching approach where the incoming Email is matched against some patterns and classified as either spam or legitimate. Many Email programs have this feature which is often referred to as "message rules" or "message filters". This technique mostly consists of a plain string matching. Whitelists and blacklists, which basically are lists of friends and foes, fall into this category. Whenever an incoming email is matched against an entry in the whitelist, the rule is to allow that email through. However whenever an email has a match against the blacklist, it is classified as a spam. This method can reduce spam up to a certain level and requires constant updating as spam evolves. It is time consuming to determine what rules to use and it is hard to obtain good results with this technique. In (Mertz, 2002) some simple rules are presented. The author claims that he was capable of catching about 80% of all spam he received. However, he also stated that the rules used had, unfortunately, relatively high false positive rates.

[1] Email spiders, or Email robots, are computer programs that scans and collects Email address from Internet.

3) Rule based filters

This is a popular content-based method deployed by spam filtering software such as SpamAssassin[1]. Rule-based filters apply a set of rules to every incoming email. If there is a match, the email is assigned a score that indicates spaminess or non-spaminess. If the total score exceeds a threshold the email is classified as spam. The rules are generally built up by regular expressions and they come with the software. The rule set must be updated regularly as spam changes, in order for the filtering of spam to be successful. Updates are retrieved via the Internet. The advantage of rule-based filters is that they require no training to perform reasonably well. Rules are implemented by humans and they can be very complex. Before a newly written rule is ready for use, it requires extensive testing to make sure it only classifies spam as spam and not legitimate messages as spam. Another disadvantage of this technique is the need for frequent updates of the rules. Once the spammer finds the way to deceive the filter, the spam messages will get through all filters with the same set of rules.

4) Statistical filters

In (Sahami, Dumais, Heckerman, & Horvitz, 1998), it is shown that it is possible to achieve remarkable results by using a statistical spam classifier. Since then many statistical filters (Zhang, Zhu, & Yao, 2004) have appeared. The reason for this is simple; they are easy to implement, have a very good performance and require a little maintenance. Statistical filters require training on both spam and non-spam messages and will gradually become more efficient. They are trained personally on the legitimate and spam emails of the user.

5) Email verification

Email verification is a challenge–response system that automatically sends out a one-time verification email to the sender. The only way for an email to pass through the filter is if the sender successfully responds to the challenge. The challenge in the verification email is often a hyperlink for the sender to click. When this link is clicked, all emails from that sender are allowed through. Bluebottle[2] and ChoicEmail[3] are two such systems.

[1] SpamAssasin, http://www.spamassassin.org/index.html
[2] Bluebottle, http://www.bluebottle.com/
[3] ChoicEmail, http://www.digiportal.com/index.html

The advantage of this method is the ability to filter almost 100% of the spam. However, there were two drawbacks associated with this method. The sender is required to respond to the challenge which necessitates extra care. If this challenge is not recognized the email will be lost. Verifications can also be lost due to technical obstacles such as firewalls and other email response systems. It can also cause problems for automated email responses such as online orders and newsletters. The verification email also generates more traffic.

6) Distributed blacklists of spam sources

These filters use a distributed blacklist to determine whether or not an incoming email is spam. The distributed blacklist resides on the Internet and is frequently being updated by the users of the filter. If a spam passes through a filter, the user reports the email to the blacklist. The blacklist is updated and will now protect other users from the sender of that specific email. This class of blacklists keeps a record of known spam sources, such as IP numbers that allow SMTP relaying. The problem involved in using a filter entirely relying on these blacklists is that it will generally classify many legitimate emails as spam (false positive). Another downside is the time taken for the networked based lookup. These solutions may be useful for companies assuming that all their email communications are with other serious non-listed businesses. Companies offering this service include MAPS[1], ORDB[2] and Spamcop[3].

7) Distributed blacklist of spam signatures

These blacklists work in a same manner to that described in 2.4.6. The difference is that these blacklists consist of spam message signatures instead of spam sources. When a user receives a spam, that user can report the message signature (typically a hash code of the email) to the blacklist. In this way, one user will be able to warn all other users that a certain message is spam. To avoid non-spam being added to a distributed blacklist, many different users must have reported the same signature. Spammers have found an easy way to fool these filters; they simply add a random string to every spam. This will prevent the email from being detected in the blacklist.

[1] Mail Abuse Prevention System LLC (MAPSSM), http://mailabuse.com/
[2] Open Relay DataBase (ORDB), http://ordb.org/
[3] Spamcop, http://www.spamcop.net/

However spam fighters attempt to overcome this problem by adapting their signature algorithms to allow some random noise. The advantage being that these kinds of filters rarely classify legitimate messages as spam. The greatest disadvantage is they are not able to recall much of the spam. Vipul's Razor[1] uses such a blacklist and states that it catches 60%-90% of all incoming spam. Another disadvantage is the time taken for the network lookup.

8) Money Email stamps

The idea of email stamps is not new, having been discussed since 1992, but it is not until recently that major companies have considered using it to combat spam. The sender would have to pay a small fee for the stamp. This fee could be minor for legitimate email senders, while it could destroy business for spammers that send millions of emails daily. There are two stamp types; money stamps and proof-of-work stamps (discussed later). GoodmailSystems[2] is developing a system for money stamps. The basic idea is to insert a unique encrypted id to the header of each sent email. If the recipient ISP is also participating in the system, the id is sent to Goodmail where it is decrypted. Good mail will now be able to identify and charge the sender of the email. Today there are many issues requiring solutions before such a system can be deployed.

9) Proof-of-work Email stamps

At the beginning of 2004, Bill Gates, Microsoft's chairman, suggested that the spam problem could be solved within two years by adding a proof-of-work stamp to each email. Camram[3] is a system that uses proof-of-work stamps. Instead of taking a micro fee from the sender, a cheat-proof mathematical puzzle is sent. The puzzle requires a certain amount of computational power to be solved (matter of seconds). When a solution is found, it is sent back to the receiver and the email is allowed to pass to the receiver. The puzzle Camram[3] is using is called Hashcash. Whether it is money or proof-of-work email stamps, many oppose the idea, not only because emailing should be free, but also because it will not solve the spam problem. To make this approach effective, most ISP's would have to join the stamp program.

1 Vipul's Razor, http://razor.sourceforge.net/
2 Goodmail, http://www.goodmailsystems.com/
3 Camram, http://www.camram.org/

As long as there are ISP's that are not integrated into the stamp system, spammers could use their servers for mass emailing. It could then still be possible for the legitimate email users to pay to send emails, while spam is still flooding into the inboxes of users. Many non-profit legitimate mass email users will probably have to abandon their newsletters due to the sending cost.

10) Legal measures

In recent years many nations have introduced anti-spam laws, in December 2003, president George W. Bush signed the CAN-SPAM15 act, the Controlling the Assault of Non-Solicited Pornography and Marketing Act. The law prohibits the use of forged header information in bulk commercial email. It also requires spam to include opt-out instructions. Violations can result in fines of $250 per email, capped at $6 million. In April 2004 the first four spammers were charged under the CANSPAM law. The trial is still on, but if the court manages to send out a strong message, this could deter some spammers. The European Union introduced an anti-spam law on the 31st of October 2003 called "The Directive on Privacy and Electronic Communications". This new law requires that companies gain consent before they send out commercial Emails. Many argue that this law is toothless since most of the spam comes from the outside of European Union. In the long-run legislation can be used to slowdown the spam flood to some extent, but it will require an international movement. Legislation will not be able to solve the spam problem by itself, at least not in the near future.

The most commonly used methods for eliminating spam were described above. Perhaps legislation is the best option in the long run. However, it requires a world wide effort and this process could be slow. Presently users need to protect themselves and for the moment statistical filters are the most promising method for this purpose. They have superior performance, can adapt automatically as spam changes and in many cases are computationally efficient.

2.5 Public benchmark spam corpora (DATASETS)

Regardless of the similarity of spam filtering and text categorization, creating a spam corpus was not as easy as in text categorization task. While it is easy to collect spam messages (*e.g.* from sites such as http://spamarchive.org), it is not easy to collect legitimate email messages for privacy reasons. The common practice of mixing spam from one site and legitimate mails from several other sources lead to biased training of the classifier since the corpus distribution may not reflect the true distribution. It is better to have the collection from the same source where the filter is to be deployed. Apart from that, a number of spam corpora have been made publicly available by their creators and have been used in evaluating various spam filtering techniques. Some of them are available in raw format such as SpamAssassin; others are available in a pre-processed format either with limited number of pre-selected attributes (such as spambase) or using encoded terms to protect privacy (such as PU1). Pre-processed corpus may lose information that is necessary for certain filtering methods. In the following, are briefly described some of these spam corpora.

• **Spambase** – This corpus is available only in a pre-processed form through UCI Machine Learning Repository (http://mlearn.ics.uci.edu/databases/spambase/). The database has been created in June-July 1999 by Mark Hopkins, Erik Reeber, George Forman, and Jaap Suermondt at Hewlett-Packard Labs. It consists of 4601 instances of legitimate and spam Email messages with 39.4% being spam. Each instance is represented by a vector of 58 dimensions. The first 57 are pre-selected attributes and the last dimension is a label describing the category of the message as spam or legitimate. The attributes include the frequency of various keywords extracted from the original messages (*e.g.* "money"), the frequency of special characters (*e.g.* semicolon, exclamation mark, dollar sign), and the length of sequences of consecutive capital letters in the message. Attributes 49 to 57 are heuristic attributes of messages. Attributes 1-48 give the percentage of words in the Email message for the respective keyword indicated in the attribute name. Attributes 49-54 give the percentage of characters in the Email message for the respective character indicated in the attribute name. Attributes 55 and 56 give the average and maximum lengths, respectively, of uninterrupted sequences of capital letters in the message. Attributes 57 gives the total number of capital letters in the message. Attribute number 58 in the dataset is the true

class (legitimate = 0, spam = 1). The dataset has no missing attribute values. Since the original contents of messages are not available, Spambase is much more restrictive than other datasets. This dataset is used in (Hidalgo, López, & Sanz, 2000), (Huai-Bin, Ying, & Zhen, 2005), (Yang & Elfayoumy, 2007), (El-Alfy & Al-Qunaieer, 2008).

• **LingSpam** – This dataset is available at (csmining.org/index.php/ling-spam-datasets.html). It is a mixture of spam and legitimate messages collected via a Linguist mailing list (a moderated mailing list about the science and profession of linguistics). It is available from (http://www.aueb.gr/users/ion/publications.html). This corpus includes 2893 messages out of which 2412 are labelled as legitimate and 481 are labelled as spam with a spam rate of 16.63%. This dataset has been used in (Sakkis G. , et al., 2003) to empirically evaluate the memory-based approach for anti-spam filtering or mailing lists. Since the number of messages in this corpus is relatively small when compared to established benchmarks for text categorization, they used 10-fold stratified cross-validation to increase the confidence in their experimental findings when using small datasets. Other authors have used this corpus as well (Androutsopoulos I. , Koutsias, Chandrinos, Paliouras, & Spyropoulos, 2000d), (Luo & Zincir-Heywood, 2005), (Sakkis G. , et al., 2001), (Zhang, Zhu, & Yao, 2004), (Zhou, Mulekar, & Nerellapalli, 2005), (Zorkadis, Panayotou, & Karras, 2005b) (Schneider, 2003), (Yang, Nie, Xu, & Guo, 2006).

• **PU1** – This corpus was created and used by (Androutsopoulos, et al., 2000a) of personal and spam messages (http://www.aueb.gr/users/ion/publications.html). It includes 1099 messages out of which 481 messages were marked as spam and 618 messages were marked as legitimate. The spam ratio is 43.77%. The corpus is pre-processed. All header fields and html tags were removed leaving only the subject line and the body of each message. Then each message was converted to lowercase and strings of non-alphabetic characters were replaced with a single white space. Each token was mapped to a unique integer to protect privacy. There are four versions of the corpus: with or without stemming and with or without stop word removal. It has been later used in (Androutsopoulos, et al., 2000a); (Carreras & Marquez, 2001), (Clark, Koprinska, & Poon, 2003), (Androutsopoulos, Paliouras, & Michelakis, 2004), (Zhang, Zhu, & Yao, 2004), (Schneider K. , 2003), (Cormack & Bratko, 2006).

• **PU2, PU3, and PUA** – These three corpora were introduced in (Androutsopoulos, Paliouras, & Michelakis, 2004). They were collected and processed in a similar fashion as PU1. The total number of messages in PU2 is 721 of which 579 are legitimate and 142 are spam. PU3 contains 4139 messages out of which 2313 are legitimate and 1826 are spam. Finally, PUA has 1142 messages of which 571 are legitimate and 571 are spam.

• **ZH1** – A Chinese corpus collected by Le Zhang at the Natural Language Processing Lab at Northeastern University and is made publicly available at (http://www.nlplab. cn/zhangle/spam/zh1.tar.bz2). It consists of 1205 spam messages and 428 legitimate Learning Methods for Spam Filtering 189 messages with a spam rate of 73.79%. The messages in the corpus are all simplified Chinese text encoded with GB2312/GBK. Unlike English language where clear explicit boundaries exist between words, Chinese text is written continuously without word delimitation. (Zhang, Zhu, & Yao, 2004) have used this corpus with three other corpora (PU1, Ling-Spam, and SpamAssassin) in evaluating four machine learning techniques. The text in the corpus was first segmented into words using a Chinese word segmenter developed by the Natural Language Processing Lab at Northeastern University. Then all messages are pre-processed to tokenize all Chinese text in the header fields, message body, and sender and recipient names.

• **SpamAssassin** – A large collection of raw spam and legitimate messages is made publicly available by SpamAssassin (http://spamassassin.org/publiccorpus). Thus, it is possible to evaluate the contribution of the header alone, the body alone, and/or both the header and the body. There are several versions of the corpus. In the version labeled 20030228, there are 1897 spam messages and 4150 legitimate messages with a spam ratio of 31.37%. It has been used in a number of studies (Chuan, Xianliang, Mengshu, & Xu, 2005), (Zhang, Zhu, & Yao, 2004), (El-Alfy & Al-Qunaieer, 2008), (Yang, Nie, Xu, & Guo, 2006)

• **Enron**-Available at http://www-2.cs.cmu.edu/~enron/ , this corpus has a large collection of legitimate email messages that were collected during the legal legislation of Enron Corporation. A brief introduction and analysis of this dataset is presented in

(Klimt & Yang, 2004). The original raw dataset contains 619,445 messages belonging to 158 senior level management users. The dataset was cleaned up and attachments were removed by a research group at the Stanford Research Institute (SRI). It has been used in (Bekkerman, McCallum, & Huang, 2004), (Webb, Chitti, & Pu, 2005).

• **TREC (2005 – 2007) Public Spam Corpora** – The 2005 TREC Public Spam Corpus (trec05p-1) contains 92,189 Email messages in raw form, with a chronological index labelling each message as spam or ham (*i.e.* legitimate Email). 52,790 messages are labelled spam while 39,399 are labelled ham. The corpus was created for the TREC Spam Evaluation Track based on Enron corpus and spam messages collected in 2005. Besides its availability as full public corpus, the messages are divided into four subsets. The 2006 TREC used two spam corpora one English (trec06p) and one Chinese (trec06c). The trec06p corpus has a total of 37822 messages of which 12910 are labelled as ham and 24912 are labelled as spam whereas the trec06c contains 64620 messages of which 21766 are ham and 42854 are spam. These corpora are made available through (http://trec.nist.gov/). They are also used in a number of publications at TREC Spam Track (http://trec.nist.gov/pubs.html).

S.no	Name of Dataset	Year	Total mails	% Spam mails	% Ham Mails
1.	Spambase	1999	4601	39.4 %	60.6 %
2.	LingSpam	2000	2893	16.63 %	83.37 %
3.	PU1	2000	1099	43.7 %	56.3 %
4.	PU2	2004	721	80.3 %	19.7 %
5.	PU3	2004	4139	55.8 %	44.2 %
6.	PUA	2004	1142	50 %	50 %
7.	ZH1	2004	1633	73.79 %	26.21 %
8.	SpamAssassin	2005	6047	31.37 %	68.6 %
9.	Enron	2006	30041	45 %	55 %
10.	TREC 2007	2007	75419	66.5 %	33.5 %

Table 1 : List of datasets

2.6 Search techniques

A paper has reviewed the two most known techniques(Linear search and Binary Search) (Asagba, Osaghae, & Ogheneovo, December 2010).

2.6.1 Linear search

The most obvious algorithm is to start at the beginning and walk to the end, testing for a match at each item. This algorithm has the benefit of simplicity; it is difficult to get wrong, unlike other more sophisticated solutions. The above code follows the convention of this article, they are as follows:

The algorithm itself is simple. A familiar $0\ to\ (n-1)$ loop to walk over every item in the array, with a test to see if the current item in the list matches the search key. The loop can terminate in one of two ways. If it reaches the end of the list, the loop condition fails. If the current item in the list matches the key, the loop is terminated early with a break statement. Then the algorithm tests the index variable to see if it is less than size (thus the loop was terminated early and the item was found), or not (and the item was not found).

2.6.2 Self organising search

As explained in (Kohonen, 2001); the lists that do not have a set order requirement, a self organizing algorithm may be more efficient if some items in the list are searched for more frequently than others. By bubbling a found item toward the front of the list, future searches for that item will be executed more quickly. This speed improvement takes advantage of the fact that 80% of all operations are performed on 20% of the items in a data set. If those items are nearer to the front of the list then search will be sped up considerably. The first solution that comes to mind is to move the found item to the front. With an array this would result in rather expensive memory shifting. Filling the hole left by removing the found item and then shifting the entire contents of the array to make room at the front is dreadfully expensive and probably would make this algorithm impractical for arrays. However, with a linked list the splicing operation required to restructure the list and send the item to the front is quick and trivial.

For a linked data structure, moving an item to a new position over large distances has a constant time complexity of $O(1)$, whereas for contiguous memory such as an array, the time complexity is $O(N)$ where N is the range of items being

shifted. A solution that is just as effective, but takes longer to reach the optimal limit is to swap the found item with the previous item in the list. This algorithm is where arrays excel over linked lists for our data set of integers. The cost of swapping two integers is less than that of surgery with pointers. The code is simple as well.

2.6.3 Binary search

All of the sequential search algorithms have the same problem; they walk over the entire list. Some of our improvements work to minimize the cost of traversing the whole data set, but those improvements only cover up what is really a problem with the algorithm. By thinking of the data in a different way, we can make speed improvements that are much better than anything sequential search can guarantee. Consider a list in ascending sorted order. It would work to search from the beginning until an item is found or the end is reached, but it makes more sense to remove as much of the working data set as possible so that the item is found more quickly. If we started at the middle of the list we could determine which half the item is in (because the list is sorted). This effectively divides the working range in half with a single test. By repeating the procedure, the result is a highly efficient search algorithm called binary search.

2.7 Statistical classifiers

A classifier's task is to assign a pattern to its class. The pattern can be a speech signal, an image or simply a text document. For example in spam classification, the classifier would assign a message as either spam or legitimate class. Historically, rule-based classifiers were mainly used until the end of the 1980s. Rule-based classifiers are simple but require classification rules to be written. Writing rules for high accuracy is difficult and time consuming. By the end of 1980s, when computers were becoming more efficient, statistical classifiers started to emerge. Statistical classifiers use machine learning to build its classifier from previously labelled (the class is known) training data. Machine learning techniques have been shown in (Mitchell, 1997). For example, a statistical spam classifier (Rennie, 2000) is trained on labelled legitimate and spam messages and a speech recognition classifier is trained on different labelled voices. The classifier uses characteristics of the pattern to classify it into one of several predefined classes. Any characteristic can be referred as a feature.

2.7.1 Features and classes

A feature is any characteristic, aspect, quality or attribute of an object. For example, the eye colour of a person or the words in a text documents are features. A good feature is one that is distinctive for the class of the object. For example, the word 'Viagra' is found in many spam messages but not in many legitimate, hence it is a good feature. In most cases many features makes the classification more accurate. The combination of n features can be represented as an n -dimensional vector, called a feature vector. The feature vector is defined as $F = \{f_1, f_2, \dots f_n\} : 1 \leq i \leq n \; where \; f_i \; is \; a \; feature$. The n-dimensionality of the feature vector is called the feature space. By examining a feature vector the classifier's task is to determine its class. If m is the number of classes, then the class vector is defined as $C = \{c_1, c_2, \dots c_n\} \; where \; 1 \leq k \leq m \; is \; a \; unique \; class$.

2.7.2 Text categorization

Text categorization (Yang & Liu, 1998) is the problem involved in classifying text documents to a category or class. Text categorization is becoming more popular as the amount of digital textual information grows. feature selection in text categorizations is explained in (Yang & Pedersen, 1997). The problem of classifying an Email message as spam or legitimate message has be considered as a text categorization problem (Sebastiani, 1999) (Sebastiani, 2002). Another popular area of use is Web page categorization to hierarchical catalogues. Statistical text classifiers can be divided into two categories, generative and discriminative. The generative approach used an intermediate step to estimate parameters while the discriminative models the probability of a document belonging to a class directly. There are arguments for using discriminative methods instead of involving the intermediate step of generative approaches. Recent studies (Ng, Jordan, Dietterich, Becker, & Ghahramani, 2002) have shown that the performances of generative and discriminative approaches are highly dependent on the corpus training data size. There are many statistical filters in the literature. An extensive study (Yang & Liu, 1998) compared several filters including Supported Vector Machines (SVM), k-Nearest-Neighbor (kNN), Neural Networks (NNet) (Jimenez, 1998), (Gavrilis & Dermatas, 2006) and Naive Bayesian (NB) (M. Sahami, 1998). Naive Bayesian is the only generative algorithm from these four. A brief introduction to these classifiers follows.

SVM (Vapnik, 1995), (Blanzieri & Bryl, 2007), (Cristianini & Shawe-Taylor, 2000), (H Ducker, 1999), (Eryigit & Tantug, 2005) separates two classes with vectors that pass through training data points. The separation was measured as the distance between the support vectors and is called the margin. The time involved in finding support vectors that maximize the margin is, in the worst-case scenario, a quadratic. SVM have shown promising results concerning text categorization problems in several studies (Vapnik, 1995). A recent study (Androutsopoulos, Paliouras, & Michelakis, 2004) demonstrated that its performance was good with reference to the spam domain.

Another classifier, k-Nearest Neighbour (kNN), maps a document to features and measures the similarity to the k-nearest training documents. Scores are created for each of the classes of the k-nearest documents based on the similarity. The document was then classified as the class with the greatest similarity. This approach has been available for over four decades and has proved to be the top-performer on Reuters corpus (topic classification of text documents). Neural Networks (NNet) is commonly used in pattern analysis and has been applied to text categorization by (Yang & Liu, 1998). NNets are expensive to train and memory consuming as the number of features grow. Among the described classifiers the NB classifier is the simplest in terms of its ease of implementation. When compared to the others, it is also shown to be computationally efficient. Tests carried out by (Yang & Liu, 1998) showed that NB underperformed the others. Another study (Androutsopoulos, et al., 2000a) shows that NB outperforms kNN. For this work NB is used as classifier not only for its simplicity and computational efficiency, but also because of a belief that with a good probability estimator and careful feature selection it does not necessarily under-perform discriminative methods. Several techniques have been surveyed and evaluated in (Androutsopoulos, Paliouras, & Michelakis, 2004), (Tretyakov, 2004), (Lai & Tsai, 2004), (Zhang, Zhu, & Yao, 2004), (Gansterer, Ilger, Lechner, Neumayer, & StrauB, 2005), (Carpinter & Hunt, 2006), (Blanzieri & Bryl, 2007), (Khorsi, 2007), (Lai C.-C. , 2007).

Androutsopoulos *et al*. (Androutsopoulos I. , Koutsias, Chandrinos, & Dpyropoulos, 2000b) applied the *k*-NN classifier to spam filtering and obtained comparable results to the Naïve Bayesian classifier. (Sakkis G. , et al., 2003)

presented a thorough empirical investigation of a memory-based learning for anti-spam filtering for mailing lists using the Ling-Spam dataset.

In the rest of this section, the basic elements of the theory relevant to NB will be clarified by using simple examples from everyday life. (GFIMailEssentials, 2007a) explains why Naive Bayesian is most effective among other statistical approaches. The detailed description of Naive Bayesian and its elements will follow in the next chapter.

2.7.3 Basics about probability theory

The probability that an event X occurs is a number that can be obtained by dividing the number of times X occurs by the total number of events. The probability is always between 0 and 1, or it can be expressed as percentage. For example, the probability of a six sided die showing 6 is $P(6) = 1/6$ or it is approximately equal to 16.67 %. Two events are independent if they do not affect each other's probabilities. For example, the events "tossing a coin" and "rolling a die" are independent because the probability of the coin landing on its head is not affected by the probability of rolling a six on a die.

For independent events, the probability of both occurring is called a joint probability and it is calculated as a product of the individual probabilities. The probability for event X is $P(X)$ and for event Y is $P(Y)$. If X and Y are independent, then their joint probability is expressed as in equation1 given below.

$$P(X \wedge Y) = P(X) \times P(Y) \qquad (1)$$

Using the example with the coin and die, the probability that the coin lands on head and the six is rolled is

$$P(Head) \times P(6) = P(Head \text{ ^}6) = \frac{1}{2} * \frac{1}{6} = 1/12$$

Events are dependent when their probabilities do affect each other. In this context conditional probabilities are defined and calculated. For example, the probability of drawing a heart from a complete card deck is 25%. If the second card is drawn without reinserting the first card, the probability of that card being a heart is

lower since one card has already been removed. Therefore, the probability of drawing the second card is dependent on the first one. Conditional probability is denoted as $P(Y|X)$, which is read as "the probability that Y occurs given that X has occurred".

Conditional probability is formally defined as

$$P(Y|X) = P(X \wedge Y)/P(X) \quad (2)$$

If the events X and Y are independent, then the conditional probability for Y given that X has occurred is equal to the probability of Y. This result can be easily obtained by substituting (1) into (2).

$$P(Y \mid X) = P(X \wedge Y)/P(X) \ = P(X).P(Y)/P(X) \ = P(Y) \quad (3)$$

The unconditional (prior) probability of an event X, $P(X)$, is the probability of the event before any evidence is presented. The evidence is the perception that affects the degree of belief in an event. The conditional probability of an event is the probability of the event after the evidence is presented. The spam classification in Naive Bayesian is based upon Bayes theorem that defines the relationship between the conditional probabilities of two events.

2.7.4 Bayes theorem

Bayes theorem has provided a way to calculate the probability of a hypothesis, here the event Y, given the observed training data, here represented as X :

$$P(Y \mid X) = (P(X \mid Y)P(Y))/P(X) \quad (4)$$

This simple formula has enormous practical importance in many applications. It is often easier to Calculate the probabilities, $P(X|Y)$, $P(Y)$, $P(X)$ when it is the probability $P(Y|X)$ that is required. This theorem is central to Bayesian statistics, which calculates the probability of a new event on the basis of earlier probability estimates derived from empirical data. The following section explains the different ways of performing statistical analyses using the classical and the Bayesian statistics.

2.7.5 Classical vs. Bayesian statistics

Here, we spot the difference between the two statistics.

Using statistics

Statistics is used to draw conclusions from data and to predict the future in order to answer research questions such as "Is there a relationship between a student's IQ and height?" To answer such questions students' IQ and height data must be collected. This can be achieved by performing experiments. For example, an IQ-test is given and the height of each student is recorded. A plot is made of IQ versus Height and it is then possible to detect whether or not a correlation exists. Statistical tests can be applied to answer research questions, to confirm or reject certain hypothesis. There are two essential statistical methods, classical (Hinton, 2004) and Bayesian (Bullard & Pazzani, 2001).

Consider the scores from one hundred students that have taken a test. Each test is marked with a score between zero and fifty. The collected data is somewhat uninformative as it is merely a list of numbers. To improve the presentation it is possible to add up the number of people who achieved the same mark. This is called the frequency for each mark. For example, 3 people scored 13 points, 10 scored 21 points etc. This information can now be represented as a histogram, where the mark is assigned to the x-axis and the number of students with that mark along the y-axis.

This presentation is called the frequency distribution. Frequency distributions are important in statistical analysis as they provide an informative representation of the data. Statistical tests can be applied to frequency distributions to answer research questions, to confirm or reject certain hypothesis.

Objective and subjective probabilities

In classical statistics all attention is devoted to the observed data, the frequencies which are generally collected from repeated trials. For example, in the case where the research question consists of deciding whether a particular die is biased or not, multiple rolls are required to obtain sufficient data. The data is then used as evidence to determine whether the observed results are significantly different to the expected for a non-biased die. This can be used as evidence that a die is biased. Now consider the scenario where a Casino employee, who is an expert on biased dice, is present and claims that there is a 98% certainty that a particular die is biased.

However, this additional information does not benefit the test as such subjective degrees of belief are ignored in classical statistics.

As opposed to classical, Bayesian statistics takes a subjective degree of belief into account, the prior data. It allows us to use the information offered by the expert in our prediction as to whether or not the die is biased. In fact many experts could be consulted and asked for their opinions. With Bayesian statistics it is possible to take subjective probabilities together with the collected data to obtain the probability of the die being biased.

Inference differences

Another difference between classical and Bayesian statistics is how their inference is performed. In classical statistics an initial assumption or the hypothesis about the research question is first made. It is usually called a null hypothesis. A single or several alternative hypotheses can also be defined. Then the relevant evidence or data are collected. This evidence measures how different the observed results are from the expected if the null hypothesis was true. The measurement is given in terms of a calculated probability called the p-value. It is the probability of obtaining the observation found in the collected data, or other observations which are even more extreme.

The significance level is the degree of certainty that is required in order to reject the null hypothesis in favour of the alternative. A typical significance level of 5% is usually used. The notation is $a = 0.05$. For this significance level, the probability of incorrectly rejecting the null hypothesis when it is actually true is 0.05. If higher protection is needed, a lower a can be selected. Once the significance level is determined and the p-value is calculated the following conclusion is drawn. If the probability of observing the actual data under the null hypothesis is small ($p < a$) the null hypothesis is not true and it can be rejected. This means that the alternative hypothesis is accepted. The converse is not true. If the p-value is big ($p > a$), then there is insufficient evidence to reject the null hypotheses. For example, let the null hypothesis, "the die is unbiased" be assumed to be true. If the die is rolled many times and 70% of all outcomes are sixes, the statistical test will calculate the probability of obtaining a six in 70% of outcomes or higher (p-value). The probability distribution for the outcomes observed using an unbiased die is used for this calculation.

If the p-value is lower than 0.05 then according to classical statistics the null hypothesis can be rejected and the die will be considered to be biased.

In Bayesian statistics (Bruyninckx, 2002), however, a probability is really an estimate of a belief in a particular hypothesis. The belief that a six occurs once in every six rolls of the die comes from both, prior considerations about fair die and the empirical results that have been observed in the past. Bayesian statistics evaluates the probability of a six by taking the previous data collected into consideration. For many researchers this approach is more intuitive than the inference of classical statistics.

2.8 Naive Bayesian Spam Filtering

Here we discuss the generalized Naive Bayesian spam filtering model and its variants.

2.8.1 Naive Bayesian spam filtering model

A general Naive Bayesian spam filtering can be conceptualized into the model presented in Figure 1. It has been evaluated by (Androutsopoulos I. , Koutsias, Chandrinos, Paliouras, & Spyropoulos, 2000). It consists of four major modules, each responsible for four different processes: message tokenization, probability estimation, feature selection and Naive Bayesian classification.

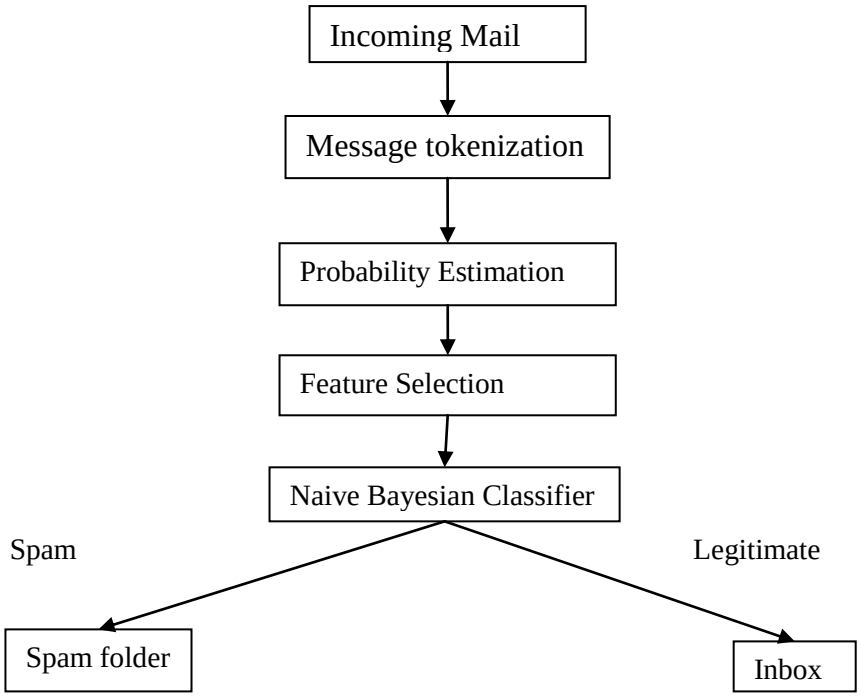

Figure 2 : A model of Naive Bayesian spam filtering

2.8.2 Naive Bayesian Classifier

In terms of a spam classifier Bayes theorem (4) can be expressed as

$$P(C \mid F) = (P(F \mid C)P(C))/(P(F)) \quad (5)$$

where F = { f_1 ,..., f_n } is a set of features and $c = \{good, spam\}$ are the two classes. This assumption is however not true as words in emails are not independent. For example an email with the word 'Viagra' is likely to co-occur with 'purchase'. However even though the independence assumption is not true the classifier works well, at least on the spam domain. One argument (Domingos & Pazzani, 1996) is that with the independence assumption the classifier would produce poor probabilities, but the ratio between them would be approximately the same as would occur using conditional probabilities. Using the somewhat 'Naive' independence assumption gave birth to its name Naive Bayesian classifier. Using the assumption for independence, according to (1), the joint probability for all n features can be obtained as a product of the total individual probabilities.

$$P(C \mid F) = P(C)\ \pi\ (P\ (f_i \mid C))/P(F) \quad (6)$$

Inserting (6) into (5) yields

$$P(C \mid F) = P(C)\ \pi\ (P\ (f_i \mid C))/P(F) \quad (7)$$

The denominator $P(F)$ the probability of observing the features in any message and can be expressed as :

$$P(F) = P\ (C_k)\ \pi\ P\ (f_i \mid C) \quad (8)$$

Inserting (8) into (7) the formula used by the Naive Bayesian Classifier is obtained

$$P(C \mid F) = (P(C)\ \pi\ P\ (f_i \mid C)) \div (P\ (C_k))\ \pi\ (P\ (f_i \mid C)) \quad (9)$$

The formal representation (9) may appear to be complicated.

However, if $C = spam$ then (9) can basically be read as: "The probability of a message being spam given its features equals the probability of any message being spam multiplied by the probability of the features co-occurring in a spam divided by the probability of observing the features in any message". To determine whether or not a message is spam the probability given by (9) is compared to a threshold value.

$$t = \lambda/(1 + \lambda) \tag{10}$$

If $P(C = spam|F) > t$ then the message is classified as spam. For example, when $\lambda = 0.9$ then $t = 0.9$ meaning that blocking one legitimate message is of the same order as allowing 9 spam messages to pass.

2.8.3 Paul Graham's plan for spam filtering

He was not the first, but Paul Graham is widely considered to have written the seminal work on statistical spam detection. In August 2002 he posted an essay to his website titled 'A Plan for Spam' (Graham P. , 2002) He clearly laid out an algorithm for filtering ham and spam. The user starts with two corpora (collections of messages): one of ham, the other of spam. The initial training stage takes place first.

1. Tokenize every message.
2. Count the number of times each token appears in each corpus. Two tables are created, one for each corpus. The tables map tokens to their counts.
3. Create a third table mapping each token to its spaminess probability.

In most current spam filters, just one token database is built. It contains three columns: the token, the count of each token in the ham corpus, and the count of each token in the spam corpus. The individual token probabilities can be calculated as needed, which eliminates the need for the third table. The first step, tokenization, is a key area of research. In his first essay, Graham used a simple definition of a token. He included alphanumeric characters, dashes, apostrophes, and dollar signs in tokens. Everything else was considered a token separator. All-digit tokens and HTML comments were ignored. Case is also ignored.

Graham doubled the 'good' count of a token to favour fewer false positives (ham incorrectly classified as spam). Graham's Token Probability Function - Simplified are given as under :

$$g(w) = 2 * numTimesSeenInHam/numHam \quad (11)$$

$$b(w) = numTimesSeenInSpam/numSpam \quad (12)$$

$$p(w) = b(w)/((b(w) + g(w))) \quad (13)$$

Tokens are only considered if seen more than five times in total. Graham handled tokens that occur in one corpus but not the other by assigning them 0.01 or 0.99 for only ham or spam, respectively. These two values are also hard limits for token probabilities. Tokens should never be 0.0 or 1.0.

Once initial training is complete, new messages can be processed.

1. Tokenize the new message.
2. Choose the 15 unique most interesting tokens.
3. Calculate the combined probability.

Interesting tokens are those tokens farthest from a probability of 0.5 in either direction. These interesting tokens form the decision matrix of the filter. Graham did not say how he broke ties when filling the decision matrix.

He dealt with hapaxes (words never seen before) by assigning them a value of 0.4, which is slightly hammy. Note, however, that tokens are still only considered if seen more than five times in total. If the probability is more than 0.9, the message is classified as spam. A simplified version is as shown below :

$$P = \frac{x_1 x_{2\ldots\ldots\ldots} x_{15}}{x_1 x_{2\ldots\ldots\ldots} x_{15} + (1 - x_1)(1 - x_2) \ldots (1 - x_{15})} \quad (14)$$

Notice a potential problem if hard limits were not used. If two tokens had probabilities of 0.0 and 1.0, a divide-by-zero error would occur. Graham refers to his method as Bayesian filtering (Graham P. , 2002).

However, the term Bayesian filtering is now used as a catch-all phrase for statistical spam filters loosely based on Graham's work.

Bayes' rule is:

$$P(C\ |F) = \frac{P(F|C)P(C)}{P(F|C)P(C) + \ P(F\ |C')P(C')} \tag{15}$$

In the context of spam filtering, C is the condition that 'the message is spam', C' means 'the message is not spam', and F is the feature being considered (the token). P(C|F) is the probability a message containing the feature is spam. This the desired overall probability, P, we are after. P(F|C) is the probability a spam message contains the feature. This is represented by the individual token probability p(w). P(C) is the probability a random message is spam. Graham's combined probability equation shown below simplifies bayes rule.

$$P(C|F) = \frac{P(F|C)}{P(F|C) + \ P(F|C')} \tag{16}$$

Substituting x for $P(F|C)$ and $(1 - x)$ for $P(F|C')$, and accounting for many features, gives Graham's combined probability function. This corresponds to assuming $P(C) = P(C') = 0.5$, equal a priori probabilities that a message is spam or ham. Graham's method results in probabilities with little uncertainty. Most message classification scores end up close to either 0.0 or 1.0. A year after his first plan, Paul Graham wrote an update to 'A Plan for Spam', titled 'Better Bayesian Filtering' (Graham P. , 2003). He presented a more elaborate definition of a token. Now he suggested preserving case. Previously, periods and commas were treated as delimiters, but they are now included in tokens if they are between two digits. This approach allows IP addresses and prices to remain intact. Graham's better plan also included the idea of marking header data. Tokens within specific header fields were marked as such. For example, if the token brownba@cs.okstate.edu is found in the To field of a header, that token would become To*brownba@cs.okstate.edu (where * is some character not allowed in tokens). At the time, Graham marked tokens inside the To, From, Subject, and Return-Path lines, and within URLs. Graham also discussed what to do about HTML. He settled on noticing some tokens and ignoring the rest. He focused on the a, img, and font tags in HTML, as these are likely to contain URLs.

In 'Better Bayesian Filtering', Paul Graham also presented a more theoretical topic of degeneration by marking header tokens and including more types of tokens

will increase the filter's vocabulary. This can make a filter more discriminating, but with a growing vocabulary, the probability that a token has never been seen before also rises. Degeneration allows a new token to be treated as a less specific version of itself. The premise is that a new token's probability of 0.4 is probably not as accurate and useful as the probability of some similar token seen already. For example, if the token Subject*longer!!! is not found in the database, the following degenerate case would be tried: Subject*longer, Subject*Longer!!!, Subject*longer, longer!!!, Longer!!!, longer, etc. The probability of the degenerate case farthest from 0.5 would be used. This token's probability would most likely be more indicative than 0.4.

Paul Graham's personal filter is effective. He trained his filter with ham and spam corpora each of about 4000 messages. Over the next year, he received about 1750 spam. He claims to have caught 99.5% of spam with 0.03% false positives over that period.

2.8.4 Pantel and Lin's plan for spam filtering

The AAAI-98 Workshop on Learning for Text Classification took place four years before Graham's first essay on spam detection. Two papers presented at this conference, one by Pantel and Lin (Pantel & Lin, 1998) and the other by Sahami, Dumais, Heckerman, and Horvitz of Microsoft Research (M. Sahami, 1998), formed the foundation for our current state-of-the-art spam filters. Catching 92% of spam with 1.16% false positives, Pantel and Lin's filter performed better than the filter from Microsoft Research. However, this is noticeably worse than Paul Graham's 99.5%/0.03% accuracy achieved four years later. A few differences in the way Pantel and Lin operated compared to Graham, outlined below, could have attributed to the decreased accuracy.

The first difference is the data Pantel and Lin used. They used what is considered a very small set of training messages: 160 spam and 466 ham.

In contrast, Graham trained with about 4000 messages each of spam and ham. With such a small training set as that used by Pantel and Lin, many tokens in the testing phase would be new and thus considered slightly hammy. Also, not only did they train with few messages, their messages were not complete. They removed the headers from all messages. With the classification based solely on the body of the

message, a lot of potentially incriminating data has been lost. It is highly recommended not to remove any information from your messages.

The data fed into Pantel and Lin's filter was substantially different from Graham's data, and so was the way they tokenized. They defined a token in two ways. A token may be a consecutive sequence of letters or digits, or it can be a consecutive sequence of non-space, non-letter, and non-digit characters. Tokens of the second type are limited to a maximum length of three characters.

Additionally, Pantel and Lin used an algorithm to remove suffixes from tokens. For example, the token waited would be reduced to wait, and meetings would be treated as meet. This 'stemming' could have been an optimization or a step to combat the small set of training data.

Pantel and Lin used another interesting technique to derive information from their data. Instead of stripping suffixes, they pulled trigrams from words. They defined a trigram as each three letter sequence of consecutive letters in a word. A large amount of information is lost when words are reduced to trigrams. However, this reduction did not significantly hurt their performance. Pantel and Lin, and Sahami et al. deserve the credit for originating the idea of a statistical spam filter, although similar techniques had been used for decision processes in other contexts. Paul Graham made the process more efficient and more widely known.

2.8.5 SpamProbe spam filter

SpamProbe is an open-source spam filter developed by Brian Burton (Burton B. , 2003) (Meyer & Whateley, 2004). Burton credits Paul Graham for the initial ideas, but Burton has implemented some alternative approaches designed to improve performance. SpamProbe's tokenizer boasts more rules than those originally proposed by Graham. The tokenizer allows certain non-text characters ('.', ',', '+', '-', ' ', and '$') within tokens. All other non-alphanumeric characters are delimiters. Purely numeric tokens are ignored.

The token 127.0.0.1 is valid, but 127 is not. All tokens are converted to lower case, which will lead to a smaller database. Tokens containing punctuation are broken down by repeatedly removing the head of the token. For example, cs.okstate.edu will result in tokens cs.okstate.edu, cs, okstate.edu, okstate, and edu. This is designed to capture domain names from URLs. Graham's individual token probability function is

retained, but the hard limits are now 0.000001 and 0.999999, and the hapax value is 0.300000.

SpamProbe has many user-configurable options. For example, it can recognize HTML tags, but by default ignores them. In either case, whether all or no HTML tags are used, URLs inside HTML are always retained. By default, header data is marked for tokens inside the Received, Subject, To, From, and Cc lines. This is referred to as the 'normal' set of header fields. The marked set can be changed to all header fields, no header fields, or all header fields excluding X- fields. The X- header fields in any Email consist of optional lines added by user Email clients. Spammers have been known to insert seemingly hammy material in X- header fields, since these fields are not usually visible to users. For example, X-mailer is a common X- header line. Spammers can insert the name of a common Email client to give the illusion that messages were sent from that client. Header tokens are marked by prefixes consisting of an H, the field name, and an ' '. For example, if the term tok was in the To field, the token Hto tok would be produced. Since SpamProbe converts all terms to lower case, marked header tokens will never be confused with body tokens.

In his first plan, Paul Graham mentioned the idea of tokenizing word pairs instead of just single words. Burton has implemented this idea in SpamProbe. By default, all single and two-word phrases are counted. For example, when the string 'one two three' is tokenized, the tokens 'one', 'one two', 'two', 'two three', and 'three' are generated. Optionally, the user can choose any phrase length. This idea of word pairs gives the tokenizer a sense of context. An important difference between SpamProbe and Graham's filter is the decision matrix. Graham used the fifteen most interesting, unique tokens in every case.

Burton implemented a more dynamic approach in SpamProbe. By default, a decision matrix of 27 tokens is used. Furthermore, tokens may be repeated up to two times if they appear in the message twice. Both the window size and the number of repeats may be adjusted by the user. A potentially important note should be made regarding tokens that have never been seen before. SpamProbe scores these tokens with a constant value like Graham, but they are allowed to appear in the decision matrix if slots remain empty.

In other words, SpamProbe will fill all slots of a decision matrix if the message size is greater than or equal to the size of the decision matrix. Optionally, a variable-

sized array of tokens can be used in SpamProbe. This array starts at size five and allows tokens to repeat up to five times each.

To prevent a single token from dominating the window, the array size is variable. All significant tokens of probability 0.1 or 0.9 in the message are added to the array.

Burton claims slightly lower spam detection accuracy but fewer false positives with this approach. Brian Burton also addressed the lack of uncertainty in Graham's combined probability function. SpamProbe uses the modified function shown below. This small change of using the nth root of products produces smoother probabilities.

Spam probe's combined probability function are given as under :

$$S = (x_1 x_2 \dots \dots . . x_n)^{1/n} \tag{17}$$

$$\mathrm{G} = ((1 - x_1)(1 - x_2) \dots . (1 - x_n))^{1/n} \tag{18}$$

$$P = S/(S + G) \tag{19}$$

Burton also differs from Graham in using a 0.7 spam threshold. Burton claims over 99% accuracy using SpamProbe with his own email. However, accuracy claimed by authors and researchers should not be expected by all users. Everybody's email is different, and often corpora show a plateau that is rarely surpassed with any filter optimization.

2.8.6 Gary Robinson's spam filter

The development of two additional combination functions is credited to Gary Robinson (Robinson G. , 2003). These functions have been employed with great success in many spam filters.

Robinson's geometric mean function is show on the next page. This function is quite similar to Burton's combination function in SpamProbe. They both use the nth root of products and return values other than 0.0 or 1.0.

Robinson's geometric mean function are given as under :

$$P = 1 - \left((1 - p_1) * (1 - p_2) * \dots \dots (1 - p_n)\right)^{1/n} \quad (20)$$

$$\text{Q} = 1 - \left((p_1) * (p_2) * \dots \dots (p_n)\right)^{1/n} \quad (21)$$

$$S = \frac{1 + \frac{(P - Q)}{(P + Q)}}{2} \quad (22)$$

Robinson has also proposed an altered token probability function. He has named this function $f(w)$ as shown below , a degree of belief. In this function, $p(w)$ can be calculated as before in Graham's essay, s is a tuneable constant, x is an assumed probability given to words never seen before (hapaxes), and n is the number of messages containing this token.

Initial values of 1 and 0.5 for s and x, respectively, are recommended. Robinson suggests using this function in situations where the token has been seen just a few times. An extreme case is where a token has never been seen before. In this case, the value of x will be returned. As the number of occurrences increases, so does the degree of belief.

Robinson's Degree of belief function is :

$$f(w) = \frac{(s * x) + \left(x * p(w)\right)}{s + n} \quad (23)$$

In Robinson's degree of belief function, $p(w)$ can be calculated as Graham did, but he suggests another slight modification. The formula below shows how instead of using the total number of occurrences of a token in a ham or spam corpus, Robinson used the number of messages containing that token. Robinson believes Graham's method performs slightly better than his since Graham's counting method does not ignore any of the token occurrences data.

Robinson's used the following equations:

$$g(w) = 2 * \frac{numTimesSeenInHam}{numHam} \tag{24}$$

$$b(w) = \frac{numTimesSeenInSpam}{numSpam} \tag{25}$$

$$p(w) = \frac{b(w)}{(\, b(w) + g(w))} \tag{26}$$

CHAPTER 3

Spam Filtering Using Paul Graham's Bayesian Approach

3. Spam filtering using Paul Graham's Bayesian Approach

In this chapter, the actual working of Paul Graham's Bayesian spam filter is depicted along with its pseudo code and flowchart.

3.1 How the Bayesian spam filter works

Bayesian filtering is based on the principle of supervised machine learning that most events are dependent and that the probability of an event taking place in the future can be inferred from the prior occurrences of that event.

This technique has been applied to classify spam. If some part of text occurs frequently in spam and not in legitimate mail, thereafter it can be reasonably assumed that this email is most likely spam.

3.1.1 Creation of a custom-made Bayesian word database

Before actual email filtering using Bayesian approach, the system needs to be trained by generating a database with words and tokens (such as the recurrent words like 'get rich' 'buy', $ sign, IP addresses and domains, etc), collected from samples of spam mails and legitimate mails (referred to as ‘ham’).

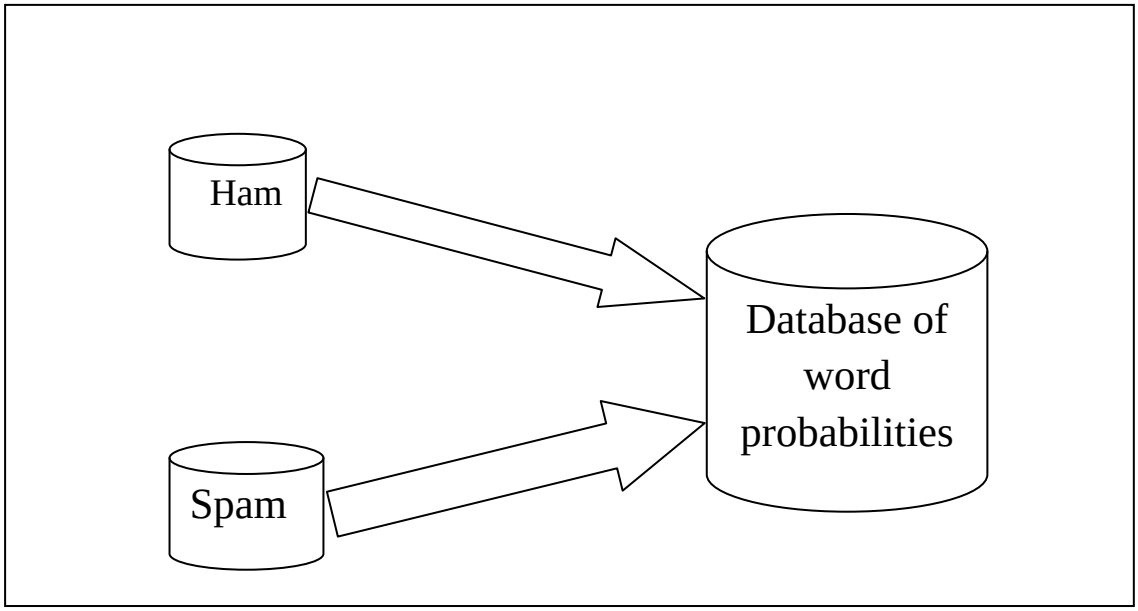

Figure 3 : Creating a word database for the filter

The token (word) is then assigned a probability value; the probability is based on calculations that consider how often that word occurs in spam and not in legitimate mail. This can be carried out by analysis of the users' outbound mail and identified spam: All the tokens in both databases of mail are analyzed to produce the probability that a particular token is most likely spam.

This word probability is found as illustrated here: If the word "buy" occurs in 500 of 6,000 spam mails and in 8 out of 600 legitimate emails, for example, then probability that is it spam would be 0.949 (that is, [500/2000] divided by [8/600 + 500/2000]).

3.1.2 Creating the legitimate(ham) database

It is significant to remember that the analysis of ham mail is performed on the organization's mail, and is thus customized to that particular organization. For instance, a financial institution might use the word "credit" many times over and would get many false positives if using a universal anti-spam rule set.

While the Bayesian filter, if customized to a company initially through training period, marks of the company's legitimate outbound mail (and recognizes "credit" as often being used in legitimate messages), and therefore has a more efficient spam detection rate and very less false positive rate.

3.1.3 Creating the spam database

The Bayesian filter also needs on a spam data file along with the ham data. This spam data file includes collection of known spam and must be regularly updated with the latest spam by the spam filter. This makes the Bayesian filter aware of the latest spam tricks, consequentially giving a high spam detection rate.

3.1.4 How is actual filtering done

When the ham and spam databases have been created, the word probabilities can be calculated and the filter can now be used. On arrival of new mail, the significant (frequent tokens) are found. From these tokens, the probability of the new message(Spam score) is calculated by the Bayesian filter. If this calculated probability is greater than a threshold, say 0.9, then the message is declared as spam. This approach to Bayesian spam filtering is highly effective.

3.2 Pseudo code : Graham's Bayesian filtering technique

The algorithm of Graham's Bayesian filtering technique as explained by authors in (Yeh & Chiang, 2008) is divided into two modules:

Module 1: Create token probability database

Module 2: Detection

MODULE 1 : CREATE TOKEN PROBABILITY DATABASE

- Input: *Email Corpus M ; Weighing parameters* wb, wg *; Token parsing function T*
- Output *: Value of spam indicator* $f(t)$ *for each token* t *in M*
- Steps

1) Prepare 2 Email corpus : One for spam (total Emails : nbad) and another for non-spam mails (total Emails: ngood)

2) For each mail M_x in corpus, parse mail body to get a set of tokens T_x.

3) For each token t, count the frequency shown in spam corpus denoted as $hb(t)$ and in non-spam corpus denoted as $hg(t)$

4) For each token t, get the probability of the token shown in spam and non-spam mails respectively

$$prob(M\ includes\ t|M\ is\ spam) = hb(t)/nbad$$
$$prob(M\ includes\ t\ |M\ is\ non-spam) = hg(t)/ngood$$

5) For each token t , calculate token probability

$$f(t) = \max(0.01, pb(t)/(pb(t) + pg(t))$$

$$where \quad pb(t) = \min(1, wb * \frac{hb(t)}{nbad})$$
$$pg(t) = \min(1, wg * \frac{hg(t)}{ngood})$$

MODULE 2 : DETECTION

- Input: *Incoming Email M_x ; Token probability database* $F = \{ f(t) | t \text{ is a token in } M\}$ *; system parameters* k, h, f_{null}
- Output *: Whether or not if the Email M_x is classified as spam*
- *V and B are temporary arrays for holding token*
- Steps

1) For a new incoming mail M_x , parse Email body to get a set of tokens T_x. Set $V = Empty$.

2) For each token t in T_x , set token probability value V_t to $f(t)$ if t ϵ M; else set $V_t = f_{null}$. Add record (t, V_t) to V.

3) Let array B=empty . For each token record (t, V_t) ϵ V, count $b_{t=}|V_t - 0.5)|$. Add (t, b_t) to B.

4) Sort B according to the b_t values in ascending order. Find out top K tokens with highest b_t values. Assume they are $\{a_1, a_{2.....} a_k\}$.

5) Compute $a = \Pi_{1 \le i \le k} a_i$, $b = \Pi_{1 \le i \le k}(1 - a_i)$. *Set* $u = a/(a + b)$.

6) If $u > h$ then output *spam* else output *non-spam.*

3.3 Flow Chart for Paul Graham's Bayesian filtering technique

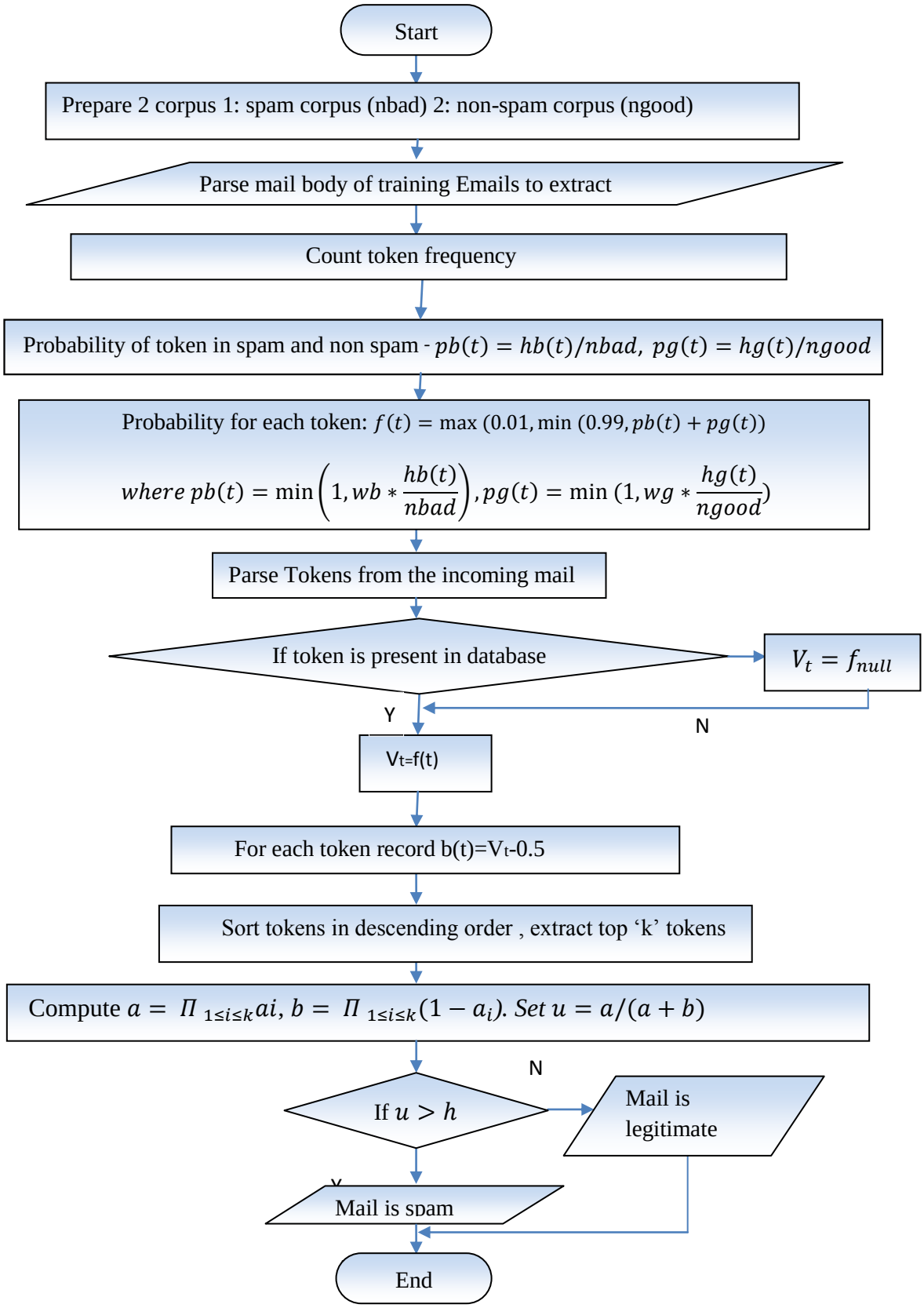

Figure 4 : Flow Chart for Paul Graham's Bayesian filtering technique

3.4 Merits of Bayesian Spam Filtering

The merits of Bayesian approach to spam filtering are discussed below :

1. The Bayesian approach considers the whole message -It identifies keywords that denote spam, but it also identifies words that denote legitimate mail. For instance: every email that has the word “buy” and “sell” is not spam. The benefit of the Bayesian method is that it takes the most fascinating words and indicates that a message is spam by showing the probability. In other words, Bayesian filtering is a very clever approach as it examines all aspects of a message, and not only keyword checking that declares a mail as spam on the finding a single word that indicates spam.

2. A Bayesian filter is persistently self-updating - It learns from new incoming spam and new legitimate outbound mails, the Bayesian filter grows and adapts to new spam techniques. For instance, when spammers started using “s-e-l-l“ instead of “sell“, it thrives in escaping keyword checking until “s-e-l-l“ is also incorporated in the keyword database. The Bayesian filter automatically notices these falsifying tactics; in fact if the word “s-e-l-l“ exists, this indicates high spam probability, since this word is highly unlikely in a ham mail.

3. The Bayesian approach can be tailored for the user - It discovers the mailing habits of the company and recognizes them, for instance, the word ‘credit’ might indicate spam if the company running the filter is, say, a banking firm, while it would not indicated as spam if the company is a car dealing company.

4. The Bayesian approach supports multiple languages and is international – Bayesian anti-spam filter is adaptive and thus can be used for any language worldwide. The Bayesian filter also considers certain languages divergence or varied usage of certain words in various areas, even if the language spoken is same. This enables makes the filter more efficient.

5. A Bayesian filter is more intelligent than a keyword filter – A tricky spammer who wishes to fool a Bayesian filter may use either fewer words that typically indicate spam (like Viagra, buy etc), or more words that generally indicate valid mail (such as a valid phone number etc). Doing this is unfeasible as the spammer will have to know

the contact details of each recipient - but a spammer cannot easily gain access to this for every intended recipient.

Bayesian filtering, if implemented the rightly and customized to particular company is very effective technology to filter spam. The learning period is a must here where the system learns to identify ham and spam. Over time, the Bayesian filter becomes even more efficient and effective as it learns more about the organization's mailing habits.

It is significant, however, to remember while evaluating anti-spam software that if the product has advanced and tailored Bayesian analysis, then it can only be judged after a few weeks(learning period). It might be the case that basic anti-spam software performs better but after the learning period, the Bayesian filter shows more efficient results, thus outperforming the conventional filters.

CHAPTER 4

Time Efficient Radix Encoded Fragmented Database Approach

4. Time Efficient Radix Encoded Fragmented Database Approach

This chapter presents the proposed method. The pseudo code shows the tokenization method with an example. The flowchart below explains the working of the spam filter using the proposed method of tokenization.

4.1 Pseudo code of the proposed method

The previously defined techniques define simple tokenization method that is extraction of various words in the email and storing them in a single data-base. A new scheme is proposed and implemented here wherein simple text tokens are encoded and stored in distributed buckets for faster retrieval.

4.1.1 Encoding Scheme used in the method:

The words are encoded by considering ASCII values of the alphabets and then finding the absolute of the difference of consecutive words. The encoding scheme used can be explained using the following example:

Token= into

If not already, convert the token into lower case. Now, we subtract ASCII value of each alphabet occurring in the token from the next alphabet and retrieve an absolute value of the difference. All such differences of a particular token in sequence are appended to form a new string, hereby called as code.

Hence, for Token=into

$$i - n = abs\,(73 - 78) = 5$$

similarly,

$$n - t = 6 \quad \text{and} \quad t - o = 5 \;\Rightarrow code = 565.$$

4.1.2 Distribution of data-base

The encoded word is then stored in the distributed database(buckets) as explained below :

The data base is divided into 26 different buckets named from 0 to 25.

The name of the bucket signifies the difference “d” of the first two alphabets of the tokens to be stored in it. Storage of words in the bucket can be explained using the following example :

For token=into, $d = 5$.

Hence, the token coded as 565 is stored in bucket 5. The bucket stores all such entries in sorted order.

4.2 Flow Chart for the proposed method

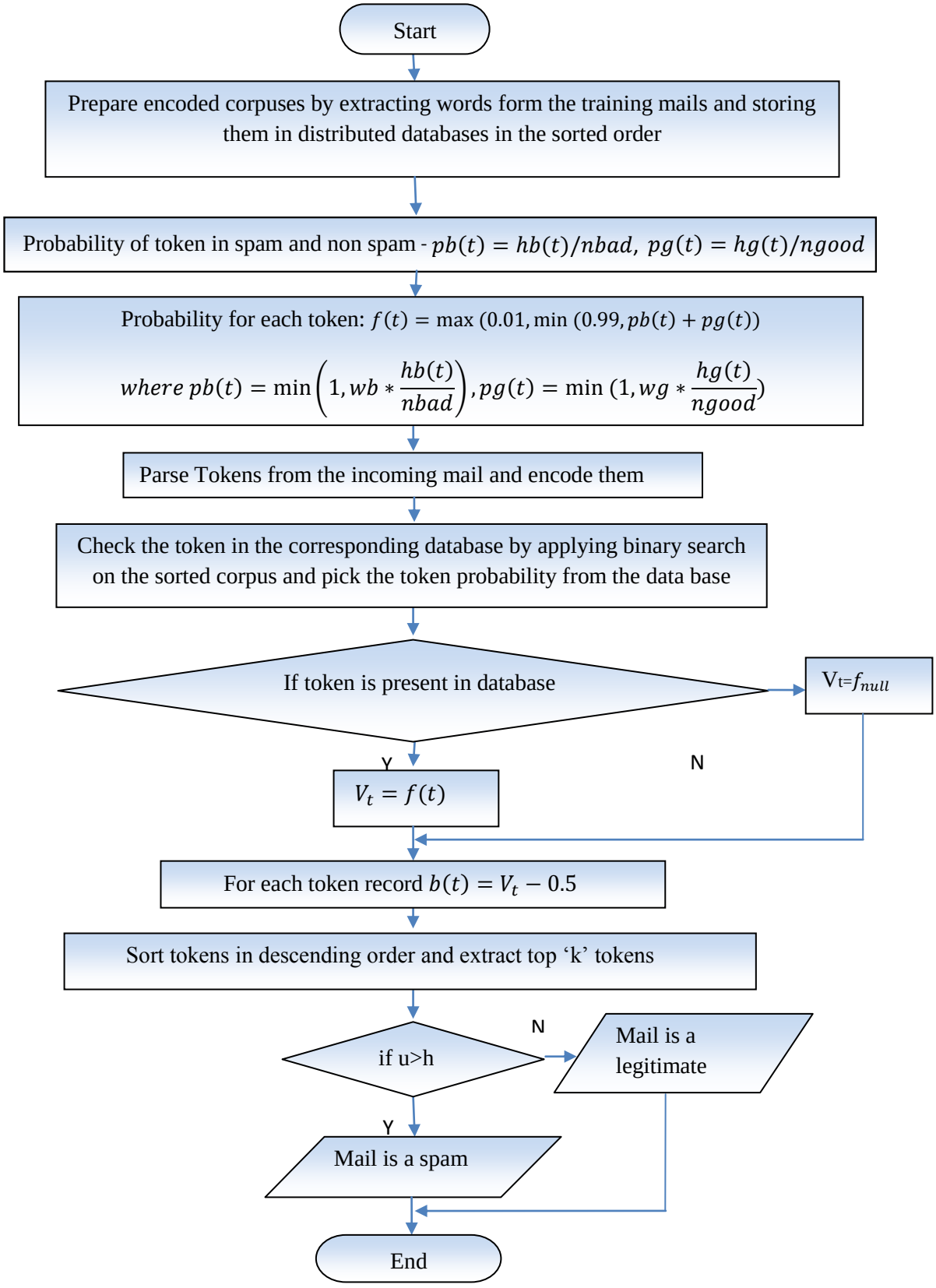

Figure 5: Flow Chart for the proposed method

4.3 Merits and Demerits of the proposed method

- Due to encoding scheme used for creation of codes of the tokens in the corpus, the database is more secure as it is difficult to decode them.
- Distribution of spam and ham corpus into distributed databases helps in faster retrieval as the difference ‘d’ causes a combination of two alphabets to be searched at a time rather than moving sequentially with single alphabets.
- Time of retrieval of a token from the buckets can now be further improved by using a faster binary search method rather than the usual linear search.
- These distributed databases can be accessed even more faster by using parallel processing or using cloud computing by fragmenting the databases over varied locations.
- These fragmenting is also advantageous in terms of security as the separated databases can be distributed at varied locations. So, in case of a calamity or any other potential damage, this database will not be damaged in its entirety.

The proposed scheme, however, has a demerit that pre-processing is required for database creation for code calculation.

CHAPTER 5

Analytical Comparative Results

5. Analytical comparative results

This chapter shows the results of qualitative and quantitative analysis of the Paul Graham's approach for spam filtering and the proposed method. Both filtering and time efficiency has been compared here.

5.1 Comparing Filtering Efficiency

Naive Bayesian spam filtering using Paul Graham's approach as mentioned and precisely explained in (Yeh & Chiang, 2008) is implemented and tested on two publicly available spam corpuses (SpamAssasin and LingSpam). Figure 6 and Figure 8 shows the ratio of spam mails and ham mails in the two corpuses used here. The proposed method is also tested over these two spam corpuses. 90% of the dataset is used for training purpose and the rest 10% of the dataset is used for testing purpose. The results are depicted in the figures below. The results shown in Figure 7 and Figure 9 clearly show that both the methods show similar results in terms of filtering efficiency. The proposed method makes no change in the filtering technique and the results are comparable for both the methods. Figure 10 shows the combined results on both the datasets.

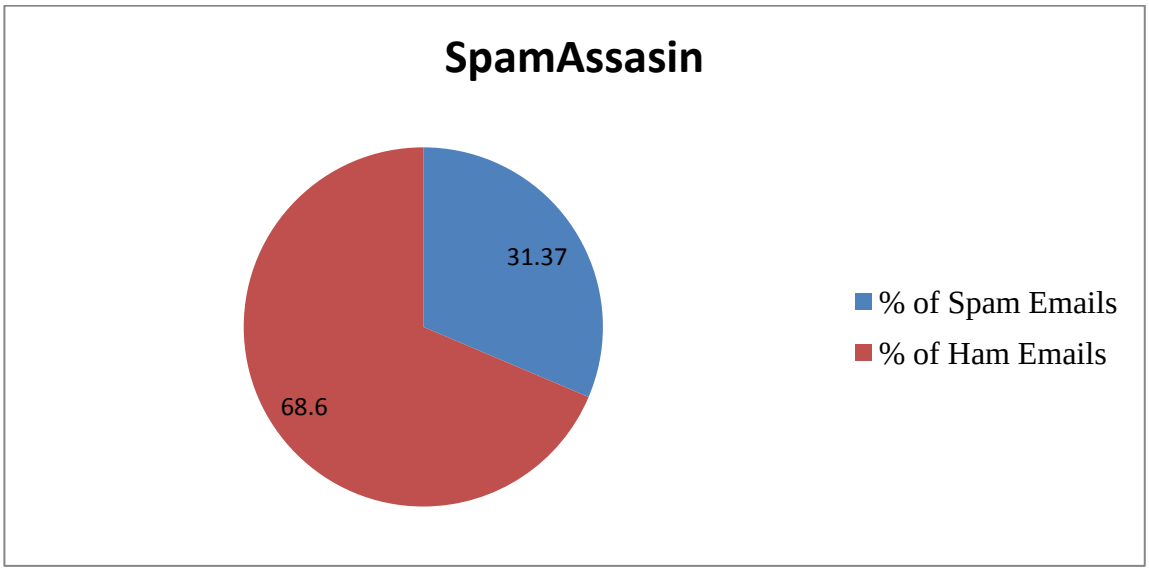

Figure 6 : Ratio of mails (Ham, Spam) in SpamAssasin Dataset

The figure 6 above shows the ratio of ham and spam Emails in SpamAssasin dataset. Here, 31.37 % of the total Emails in this dataset (shown in blue in the figure) represent spam Emails. Rest 68.6 % of the total Emails are ham mails.

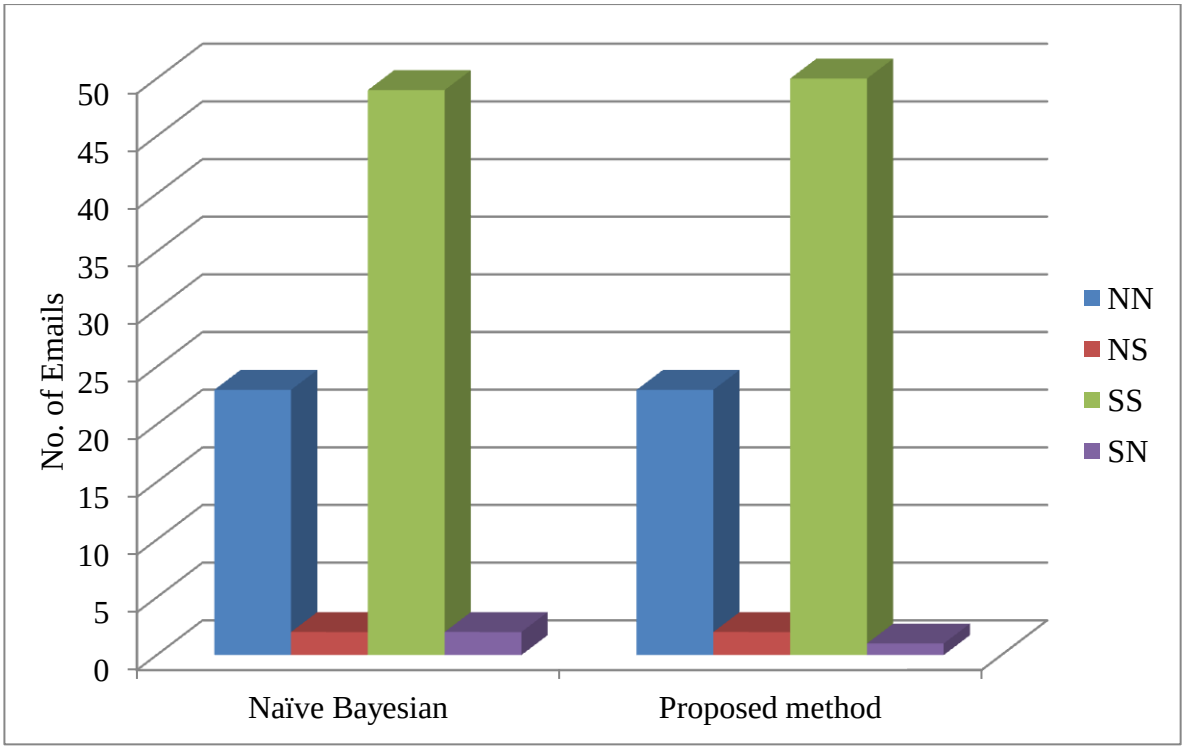

Figure 7 : Comparison of proposed method with existing on SpamAssasin dataset

The Figure 7 shows the results of the proposed scheme for measuring the filtering on SpamAssasin dataset. The metrics used are as follows:

NN denotes non-spam Emails that are correctly classified by the filter as non-spam.
NS denotes non-spam Emails that are correctly classified by the filter as spam.
SS denotes spam Emails that are correctly classified by the filter as spam.
SN denotes spam Emails that are correctly classified by the filter as non-spam.

The colour scheme used here is :
Blue represents NN (non-spam classified as non-spam)
Red represents NS (non-spam classified as spam)
Green represents SS (spam classified as spam)
Purple represents SN (spam classified as non-spam)

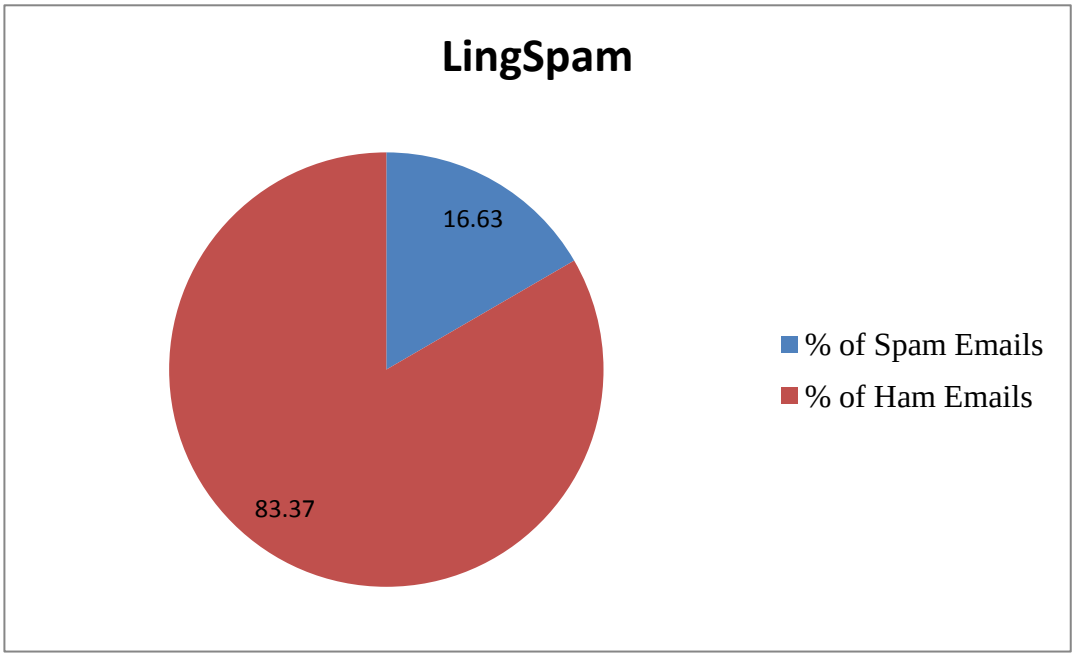

Figure 8 : Ratio of Emails (Ham, Spam) in LingSpam Dataset

The Figure 8 above shows the ratio of ham and spam Emails in LingSpam dataset. Here, 16.63 % of the total Emails in this dataset (shown in blue in the figure) represent spam Emails. Rest 83.37 % (shown in red) of the total Emails are ham Emails.

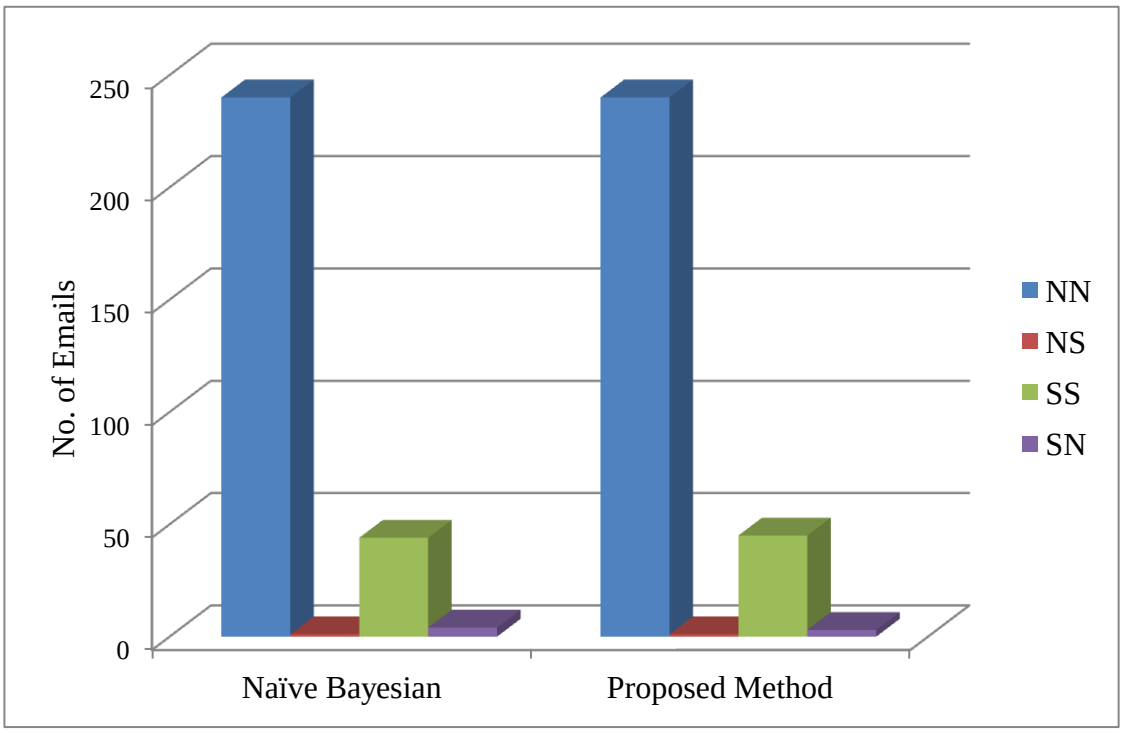

Figure 9 : Comparison of proposed method with existing on LingSpam dataset

The Figure 9 above shows the results of the proposed scheme for measuring the filtering on SpamAssasin dataset. The metrics used are as follows:

NN denotes non-spam Emails that are correctly classified by the filter as non-spam.
NS denotes non-spam Emails that are correctly classified by the filter as spam.
SS denotes spam Emails that are correctly classified by the filter as spam.
SN denotes spam Emails that are correctly classified by the filter as non-spam.

The colour scheme used here is :
Blue represents NN (non-spam classified as non-spam)
Red represents NS (non-spam classified as spam)
Green represents SS (spam classified as spam)
Purple represents SN (spam classified as non-spam)

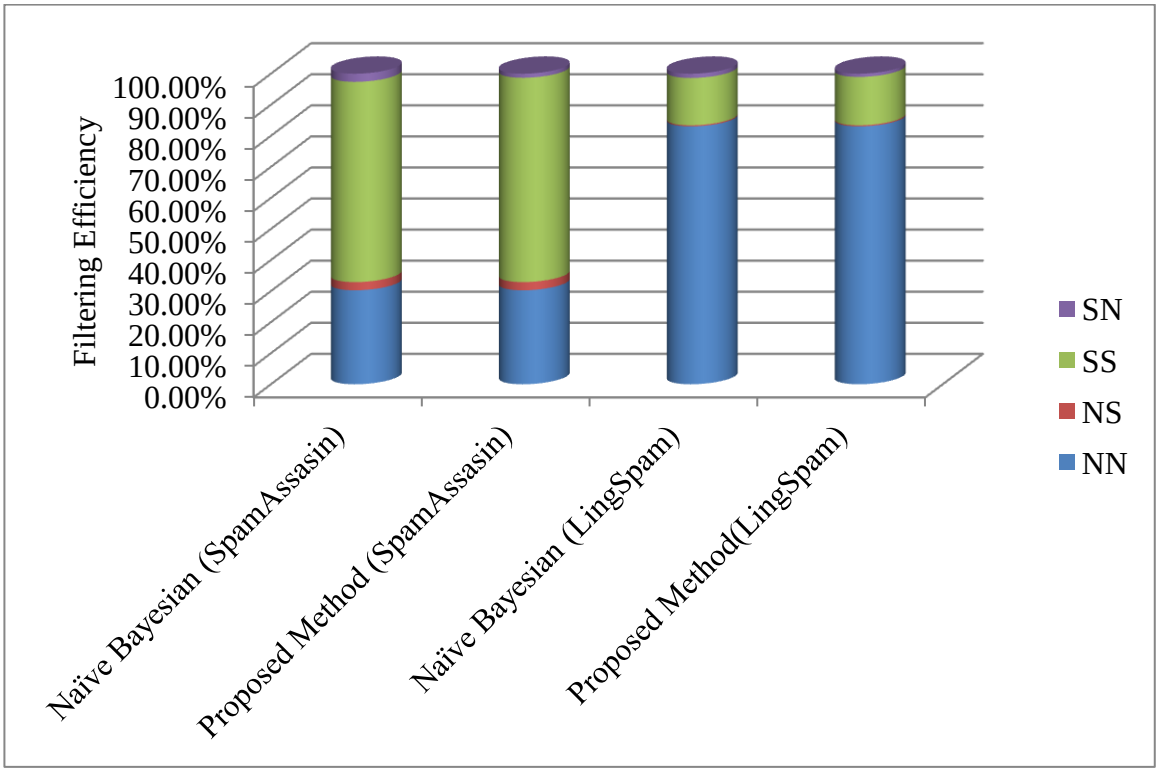

Figure 10 : Overall result on the two datasets for filtering efficiency

Figure 10 shows the combined results on both the datasets. This figure clearly depicts that the proposed scheme is as efficient as the original Naive Bayesian spam filtering in terms of spam filtering efficiency. The colour scheme used is as shown above.

5.2 Comparing Time Efficiency

The proposed method is basically designed to improve the time efficiency of the spam classification. The following figure clearly depicts that the proposed method works 5-6 times faster than the classical method used in content spam filtering. The faster execution results due to the distributions of the encoded corpuses prepared ; on which binary search has been applied for faster retrieval.

The simulation results shown that the proposed method gives the filtering results in very less time as compared to the original method used by Paul Graham. For SpamAssasin dataset; Paul Graham's Naive Bayesian spam filtering technique took 25 minutes to execute and the proposed method took 5 minutes to complete detection of mails in the testing part of the dataset. For LingSpam dataset; Paul Graham's Naive Bayesian spam filtering technique took 20 minutes to execute and the proposed method took 3.5 minutes to generate the results. Figure 11 below depict these results. The blue bar represents the time (in minutes) required for spam filtering using the Naive Bayesian filtering technique. The red bar represents the time (in minutes) required for spam filtering using the proposed method.

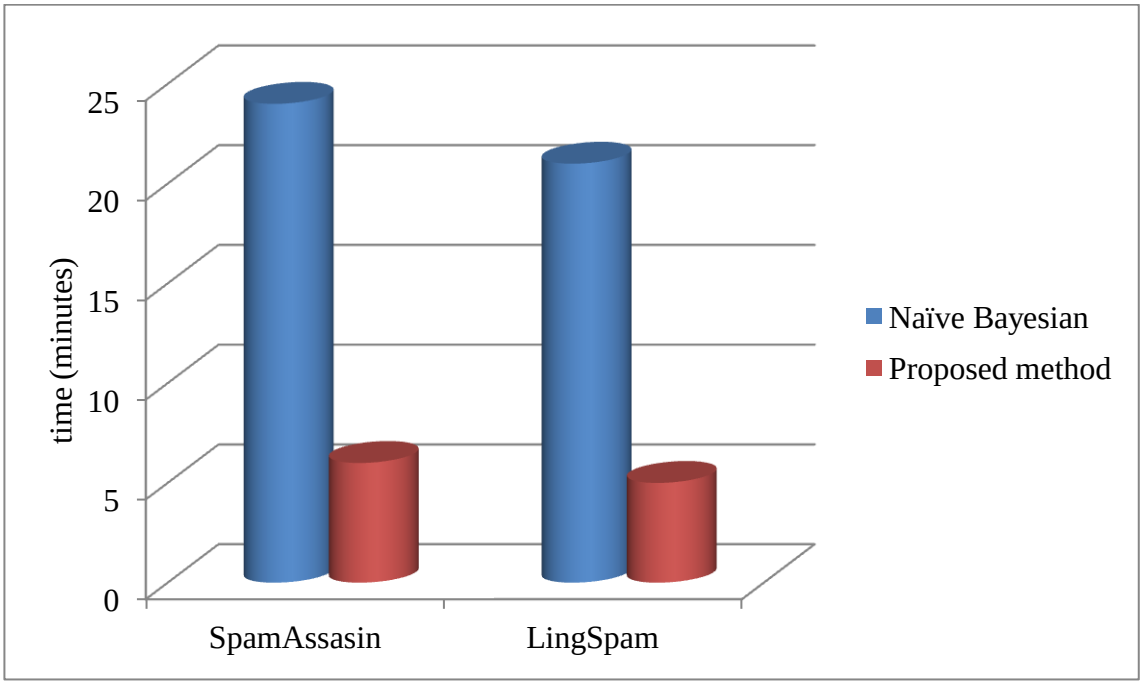

Figure 11 : Time efficiency comparison of proposed method with existing method

Conclusion

This dissertation reviews various existing approaches for Spam detection and includes results of implementation of a statistical approach: Naive Bayesian Spam Filtering using Paul Graham's filtering technique. Thereafter, an encoded and fragmented database approach that resembles radix sort technique has been proposed and applied to improve Paul Graham's Naive Bayes machine learning algorithm for spam filtering technique.

The results clearly show that proposed method works much faster (as much as six times faster) as compared to the original Paul Graham's spam filtering technique as it is faster to search the fragmented database as opposed to searching on the whole database; and also due to binary search applied onto the sorted encoded database. This method of fragmenting the encoded databases can be applied to various systems requiring managing and searching on databases.

Suggestions for Future work

Searching time may further be improved using hash function to create distributed databases. The distributed databases can be implemented over cloud computing and the results can be improved all the more. The distributed databases can be implemented and accessed using parallel computing/processing.

REFERENCES

Androutsopoulos, I., Koutsias, J., Chandrinos, K., Paliouras, G., & Spyropoulos, C. (2000). An Evaluation of Naive Bayesian Anti-Spam Filtering. *Proceedings of the 11th European Conference on Machine Learning*, (pp. 9-17). Barcelona, Spain. Retrieved from http://arxiv.org/pdf/cs/0006013v1.pdf

Androutsopoulos, I., Koutsias, J., Chandrinos, K., Paliouras, G., & Spyropoulos, C. (2000d). An evaluation of naïve Bayesian anti-spam filtering. *Proceedings of the Workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning (ECML 2000)*, (pp. 9–17). Barcelona, Spain.

Androutsopoulos, I., Koutsias, J., Chandrinos, V., & Dpyropoulos, D. (2000b). An experimental comparison of naïve Bayesian and keyword-based anti-spam filtering with personal e-mail messages. *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (pp. 1-13). New York.

Androutsopoulos, I., Paliouras, G., & Michelakis, E. (2004). Learning to Filter Unsolicited Commercial E-Mail. *Athens University of Economics and Business and National Centre for Scientific Research "Demokritos"*.

Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C., & Stamatopoulos, P. (2000a). Learning to filter spam email: A comparison of a naive bayesian and a memory-based approach. *In Workshop on Machine Learning and Textual Information Access.* 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2000).

Asagba, P. O., Osaghae, E. O., & Ogheneovo, E. E. (December 2010). Is Binary Search Technique Faster Than Linear Search. *Scientia Africana, Vol. 9 (No. 2) December 2010*, 83-92.

Bekkerman, R., McCallum, A., & Huang, G. (2004). *Automatic categorization of email into folders: Benchmark experiments on Enron and SRI corpora.*

Blanzieri, E., & Bryl, A. (2007). Instance-based spam filtering using SVM nearest neighbor classifier. *American Association for Artificial Intelligence*.

Bruyninckx, H. (2002, November). Bayesian probability. doi:10.1.1.120.4949

Bullard, F., & Pazzani, M. (2001). A Brief Introduction to Bayesian StatisticsBeyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier. *Domingos* (pp. 105–112). Bari,Italy: Proceedings of the 13th Int. Conference on Machine Learning.

Burton, B. (2003). Spamprobe-bayesian spam filtering tweaks. *In Proceedings of the Spam Conference.* Retrieved from http://spamprobe.sourceforge.net/paper.html.

Carpinter, J., & Hunt, R. (2006). Tightening the net: A review of current and next generation spam filtering tools. *Computers and Security, 25*, 566–78.

Carreras, X., & Marquez, L. (2001). Boosting trees for anti-spam email filtering. *Proceedings of Fourth International Conference on Recent Advances in NaturalLanguage Processing.* Tzigov Chark, Bulgaria. Retrieved from http://www.lsi.upc.es/ ~carreras/pub/boospam.ps.

Chuan, Z., Xianliang, L., Mengshu, H., & Xu, Z. (2005). A LVQ-based neural network antispam email approach. *ACM SIGOPS Operating Systems Review, 39*(1), 34-39.

Clark, J., Koprinska, I., & Poon, J. (2003). A neural network based approach to automated email classification. *Proceedings of the IEEE/WIC International Conference on Web Intelligence (WI'03).*

Cormack, G. V., & Bratko, A. (2006). Batch and online spam filter comparison. *Third Conference on Email and Anti-Spam (CEAS'06).* Mountain View, California.

Cristianini, N., & Shawe-Taylor, J. (2000). An Introduction to support vector machines and other kernel-based learning methods. *Cambridge University Press*.

Davis, S., & Craney, G. (2001). How Do I Stop Spam?

Domingos, P., & Pazzani, M. (1996). Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier. *Proceedings of the 13th Int. Conference on Machine Learning*, (pp. 105–112). Bari, Italy.

El-Alfy, E.-S. M., & Al-Qunaieer, F. (2008). A fuzzy similarity approach for automated spam filtering. *Proceedings of IEEE International Conference on Computer Systems and Applications (AICCSA'08).* Doha, Qatar.

Eryigit, G., & Tantug, C. (2005). A comparison of support vector machines. *memory-based and naïve bayes techniques on spam recognition. Proceedings of the International Conference on Artificial Intelligence and Applications*, (pp. 457–462). , Innsbruck.

Fallows, D. (2003). *Spam: How it is hurting email and degrading life on the internet.*

Gansterer, W., Ilger, M., Lechner, P., Neumayer, R., & StrauB, J. (2005). Anti-spam methods – state-of-the-art. *University of Vienna, Austria*.

Gavrilis, D., & Dermatas, E. (2006). Neural recognition and genetic features selection for robust detection of e-mail spam. *Lecture Notes in Computer Science, , SpringerBerlin/Heidelberg, 3955*.

GFIMailEssentials. (2007a). Why Bayesian filtering is the most effective anti-spam technology. *White Paper*. Retrieved from http://www.gfi.com/whitepapers/GFI

Software, (2007b). How to keep spam off your network. White Paper. Available from:http://www.gfi.com/whitepapers/

Goodman, J., Cormack, G. V., & Heckerman, D. (2007). Spam and the ongoing battle for the inbox. *Communications of the ACM*, (pp. 25-33).

Graham, P. (2002). *A plan for spam.* Retrieved from http://www.paulgraham.com/spam.html.

Graham, P. (2003). Better Bayesian Filtering. *In Proceedings of the 2003 Spam Conference.* Retrieved from http://www.paulgraham.com/better.html

H Ducker, D. W. (1999). Support vector machine for spam categorization. *IEEE Transaction on Neural Networks, 10*(5), 1048-1054. doi:10.1109/72.788645

Hidalgo, J., López, M. M., & Sanz, E. P. (2000). Combining text and heuristics for cost-sensitive spam filtering. *Proceedings of the 2nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning, 7*, pp. 99-102.

Hinton, R. (2004). Statistics Explained. *2nd Edition, Routledge*.

Huai-Bin, W., Ying, Y., & Zhen, L. (2005). SVM classifier incorporating feature selection using GA for spam detection. *Lecture Notes in Computer Science*, 1147-1154.

Jimenez, D. (1998). Dynamically weighted ensemble neural networks for classification. *Proceedings of IEEE International Joint Conference on Neural Networks*, (pp. 753–756). Anchorage,Alaska, USA.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *Proceedings of 10th European Conference on Machine Learning(ECML-98)*, (pp. 137–142). Chemnitz, Germany.

Junejo, K. N., Yousaf, M. M., & Karim, A. (2006). A two-pass statistical approach for automatic personalized spam filtering. *Proceedings of 17th European Conference on Machine Learning (ECML) and 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD).* (ECML/PKDD 2006) Berlin.

Jung, J., & Sit, E. (2004). An empirical study of spam traffic and the use of DNS black lists. *Proceedings of Fourth ACM SIGCOMM Conference on Internet Measurement.* Taormina, Sicily, Italy.

Kågström, J. (2008, July). *Codeode*. Retrieved from Techniques to eliminate spam: http://www.codeode.com/techniques-to-eliminate-spam

Khorsi, A. (2007). An overview of content-based spam filtering techniques. *Informatica, vol 31, 269-277*.

Klimt, B., & Yang, Y. (2004). Introducing the Enron corpus. *Proceedings of the First Conference on Email and Anti-Spam (CEAS).*

Kohonen, T. (2001). Self-organizing maps. *Third Extended Edition, Springer, New York*.

Lai, C.-C. (2007). An empirical study of three machine learning methods for spam filtering. *Knowledge-Based Systems, 20*(3), 249–254.

Lai, C.-C., & Tsai, M.-C. (2004). An empirical performance comparison of machine learning methods for spam e-mail categorization. *Proceedings of the Fourth International Conference on Hybrid Intelligent Systems (HIS'04).*

Luo, X., & Zincir-Heywood, N. (2005). Comparison of a SOM based sequence analysis system and naïve bayesian classifier for spam filtering. *Proceedings of IEEE International Joint Conference on Neural Networks (IJCNN'05), 4,* pp. 2571–2576.

M. Sahami, S. D. (1998). A Bayesian Approach to filtering Junk E-mail. *In proceedings of AAAI workshop On Learning for Text Categorization.* Madison, Visconsin.

Mertz, D. (2002). Spam filtering techniques. Retrieved from http://www.ibm.com/developerworks/linux/library/l-spamf/index.html#author1

Meyer, T. A., & Whateley, B. (2004). SpamBayes: Effective open-source. *Bayesian based.* email classification system. Proceedings of the First Conference on Email and Anti-Spam(CEAS). Retrieved from http://www.ceas.cc/papers-2004/136.pdf.

Mitchell, T. M. (1997). Machine learning. *McGraw Hill, New York, NY*.

Ng, A. Y., Jordan, M. I., Dietterich, T., Becker, S., & Ghahramani, Z. (2002). On discriminative vs generative classifiers: A comparison of logistic regression and naive Bayes. *Advances in Neural Information Processing Systems (NIPS) 14*.

Pantel, P., & Lin, D. (1998). SpamCop: a spam classification and organization program. *Proceedings of AAAI Workshop on Learning for Text Categorization*.

Rennie, J. (2000). Ifile: An application of machine learning to e-mail filtering. *KDD-2000 Text Mining Workshop*.

Robinson, G. (2003). A Statistical Approach to the Spam Problem. *Linux Journal*(107).

Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998). A Bayesian approach to filtering junk e-mail. *Proceedings of AAAI'98 Workshop on Learning for Text Categorization, Madison, WI,* 55-62. Retrieved from http://research.microsoft.com/~horvitz/junkfilter.htm.

Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D., & Stamatopoulos, P. (2001). Stacking classifiers for anti-spam filtering of e-mail. *Proceedings of the 6th Conf. on Empirical Methods in Natural Language Processing*, (pp. 44-50). Pittsburgh.

Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D., & Stamatopoulos, P. (2003). A memory-based approach to anti-spam filtering. *Information Retrieval, 6*(1), 49-73.

Schneider, K. (2003). A comparison of event models for naïve Bayes anti-spam e-mail filtering. *Proceedings of the 10th conference of the European Chapter of the Association for Computational Linguistics (EACL'03).* Budapest, Hungary.

Sebastiani, F. (1999). A tutorial on automated text categorization. *Proceedings of the First Argentinean Symposium on Artificial Intelligence (ASAI-99)*.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys (CSUR), 34*(1), 1-47.

Sebastiani, F., Sperduti, A., & Valdambrini, N. (2001). Boosting algorithms for automated text categorization. *ERCIM News*(44).

Siponen, M., & Stucke, C. (2006). Elective anti-spam strategies in companies: An international study. *In Proceedings of HICSS '06, 6*.

Tretyakov, K. (2004). Machine learning techniques in spam filtering. *Institute of Computer Science, University of Tartu, Data Mining Problem-oriented Seminar, MTAT.03.177*, 60-79.

Vapnik, V. (1995). The nature of statistical learning theory. *Springer, New York*.

Varnsen, K. (n.d.). *Types of Spam E-mails*. Retrieved from Ranker: http://www.ranker.com/list/types-of-spam-e-mails/kel-varnsen?page=1

Wang, C.-C. (2004). Sender and receiver addresses as cues for anti-spam filtering. *Journal of Research and Practice in Information Technology, 36*(1), 3–7.

Webb, S., Chitti, S., & Pu, C. (2005). An experimental evaluation of spam filter performance and robustness against attack. *International Conference on Collaborative Computing:Networking.* Applications and Worksharing.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval, 1*, 69-90.

Yang, Y., & Elfayoumy, S. (2007). Anti-spam filtering using neural networks and Bayesian classifiers. *Proceedings of the 2007 IEEE International Symposium on Computational Intelligence in Robotics and Automation, Jacksonville, FL, USA.*

Yang, Y., & Liu, X. (1998). A re-examination of text categorization methods. *School of Computer Science, Carnegie Mellon University*.

Yang, Y., & Pedersen, J. O. (1997). A comparative study of feature selection in text categorization. *Proceedings of the 14th International Conference on Machine Learning.*

Yang, Z., Nie, X., Xu, W., & Guo, J. (2006). An approach to spam detection by naïve Bayes ensemble based on decision induction. *Sixth International Conference on Intelligent Systems Design and Applications.* (ISDA '06).

Yeh, C.-C., & Chiang, S.-J. (2008). Revisit Bayesian Approaches for Spam Detection. *The 9th International Conference for Young Computer Scientists*, (pp. 659- 664). doi:10.1109/ICYCS.2008.434

Zhang, L., Zhu, J., & Yao, T. (2004, December). An Evaluation of Statistical Spam Filtering Techniques. *ACM Transactions on Asian Language Information Processing*, 243-269. doi:10.1145/1039621.1039625

Zhou, Y., Mulekar, M. S., & Nerellapalli, P. (2005). Adaptive spam filtering using dynamic feature space. *Proceedings of 17th IEEE International Conference on Tools with Artificial Intelligence*, (pp. 302–309). ICTAI'05.

Zorkadis, V., Panayotou, M., & Karras, D. A. (2005b). Improved spam e-mail filtering based on committee machines and information theoretic feature extraction. *Proceedings of IEEE International Joint Conference on Neural Networks*, *1*, pp. 179–184. IJCNN '05.