# Model for Adaptive Mail Filtration and Classification with Customization


**By**

**SHWETA LODHA**
**(06MCE008)**




**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**
**INSTITUTE OF TECHNOLOGY**
**NIRMA UNIVERSITY OF SCIENCE & TECHNOLOGY**
**AHMEDABAD 382481**

**MAY 2008**

**Major Project**

**On**

# Model for Adaptive Mail Filtration and Classification with Customization

Submitted in partial fulfillment of the requirements

For the degree of

**Master of Technology in Computer Science & Engineering**

By

**Shweta Lodha**
**(06MCE008)**

Under Guidance of

**Dr. Dilip P. Ahalpara**


**Department Of Computer Science & Engineering**
**Institute Of Technology**
**Nirma University Of Science & Technology**
**Ahmedabad 382481**
**May 2008**


This is to certify that Dissertation entitled

# Model for Adaptive Mail Filtration and Classification with Customization

Submitted by

Shweta Lodha

has been accepted toward fulfillment of the requirement

for the degree of

Master of Technology in Computer Science & Engineering

| | |
|---|---|
| Dr. S. N. Pradhan | Prof. D. J. Patel |
| P.G. Co-ordinator | Head of the Department |

Prof. A. B. Patel

Director, Institute of Technology

# *CERTIFICATE*

This is to certify that the work presented here by **Ms. Shweta Lodha (06MCE008)** entitled "**Model for Adaptive Mail Filtration and Classification with Customization"** has been carried out at **Institute Of Technology, Nirma University** during the period **September 2007 – May 2008** is the bonafide record of the research carried out by her under my guidance and supervision and is up to the standard in respect of the content and presentation for being referred to the examiner. I further certify that the work done by her is her original work and has not been submitted for award of any other diploma or degree.

**Dr. Dilip P. Ahalpara**
Scientist SF,
Institute of Plasma Research,
Ahmedabad

**Date: / /**

# ACKNOWLEDGEMENTS

It gives me immense pleasure in expressing my thanks and profound gratitude to **Dr**. **S.N. Pradhan**, Professor, Computer Science Department, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance and continuous encouragement throughout my Major project. I am heartily thankful to him for his precious time, suggestions and sorting out the difficulties of my topic that helped me a lot during this study.

I would like to give my special thanks to **Prof**. **D.J. Patel**, Head, Department of Computer Engineering, Institute of Technology, Nirma University, Ahmedabad for his encouragement and motivation throughout the Major Project, I am also thankful to **Prof. A. B. Patel**, Director, and Institute of Technology for his kind support in all respect during my study.

I am thankful to all faculty members of Computer Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

The blessings of God and my family members make the way for completion of Major Project. I am very much grateful to them.

I am thankful to my dear friends Mr. Pradyumna Lenka and Mr. Pavu Jain who lent their ear when needed and always support and encourages me.

**Shweta Lodha**
Roll No. 06MCE008

# ABSTRACT

The goal of the thesis is to construct an email filter/classifier utilizing several methods. Therefore, it is important to understand the past and present of both incoming emails, spamming and spam filtering. Based on the best-practice solutions of present day, filter is designed and constructed. The thesis has three main sections:

The goal of the first section is to introduce the fact and phenomenon of junk e-mail also called *spam*. The first section covers what spam is and also gives a brief overview about the history of spam and spam filtering. This section is followed by the explanation why spam exists and what it causes. Furthermore, the different kinds of spam are introduced and finally the spam filtering methods are reviewed.

The second section contains the categorization of emails in various categories, based on classification technique named Vector Space Model. Then a description of a vector based algorithm is given. In this algorithm, features are created from individual sentences in the subject and body of a message by forming all possible word pairings from a sentence. Weights are assigned to the features based on the strength of their predictive capabilities for several categories. The predictive capabilities are estimated by the frequency of occurrence of features in collection (emails) as well as application of heuristic rules. The same model can be used for both filtration and classification purpose with varying thresholds. The standard threshold considered here is 0.03.

The third section introduces implementation details for modeling and analyzing our email filtration and classification using JAVA. It is followed by self-constructed libraries and methods to fetch emails from server and filter them, which shows the relevance of the filter. The result of the analysis gives the answer regarding classification technique, considering both the achieved accuracy rate in filtering and the computational resources.

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# 1 INTRODUCTION

## 1.1 HISTORICAL BACKGROUND

In the past decade text categorization has been a highly popular machine learning application. In addition to the standard problem of categorizing documents into semantic topics, a variety of other problem domains have been explored, including categorization by genre, by authorship and even by author's gender. In the domain of personal email messages, text categorization methods have been widely applied to the problem of spam filtering. Other email related problems have also been tackled, such as automatically creating new folders. However, there has been little study of categorizing email into folders also termed as *"email foldering".* Email foldering is a rich and multi-faceted problem, with many difficulties that make it different from traditional topic-based categorization. Email users create new folders, and let other folders fall out of use. It is also interesting to note that email content and foldering habits differ drastically from one email user to another, so while automated methods may perform well for one user, they may fail considerably for another [1].

Furthermore, email arrives in a stream over time, and this causes other significant difficulties. Some email messages only make sense in the context of previous messages. Occasionally all messages in a thread should go to the same folder, but for other times the topic in a thread drifts [1]. The topic associated with a certain email folder can also shift over time. For example, a folder about funding opportunities may at first contain only messages about the National Science Foundation, but later only get new messages about Industrial Partnerships, each of which may have very different word distributions.

A likely reason that the problem of automatic email foldering has not drawn significant attention in the research community is the fact that there has been no standard, publicly available real-world email dataset on which foldering methods could be evaluated, and on which the work of multiple researchers could be compared.

## 1.2 E-MAIL CLASSIFICATION AND FILTRATION

Classification of text documents is an example of supervised learning that seeks to build a probabilistic model of a function that maps documents to classifications. In supervised learning of text, where an entire document represents one example to be classified, a learning algorithm is presented with a set of already classified, or labeled, examples. This set is called the training set. Information Classification deals mainly with large amounts of incoming data. It is based on description of an individual's or a group's information preference, called profiles, which typically represent long-term user's interests [2]. In our context, classification/filtering process aims to select and/or eliminate information from a dynamic data-stream. The description of user's interests (profiles) is the most crucial and difficult operation in the building of an information filtering system. Indeed, this operation addresses the following questions: What are the user's information interests? How are they identified? How are they represented? Can they be updated easily? etc. The effectiveness of filtering is closely dependent on the quality of the profiles representation. In general, the terms contained in a document are different from those the user would use to specify his interests. Thus, it is not easy for users to specify what those interests are because they differ from one user to another, and they are constantly changing [3]. To know how to configure a user's model for an information filtering system, an experimental study can be undertaken to observe how the users process, evaluate and classify their relevant documents or we can say mails. Once a model has been constructed, it can be used to predict the classification of future documents. The accuracy of such models is largely dependent on the "representativeness" of the training data with respect to newly acquired data to be classified. In general, the more the representative training data, the better the performance. A larger number of training examples is often better, because a larger sample is likely to be more reflective of the actual distribution of the data as a whole.

## 1.3 NECESSITY OF CLASSIFICATION AND SPAM FILTERING

Nowadays the implementation of a reliable email classifier becomes more and more important for e-mail users since they have to face with the growing amount of uninvited e-mails. The spam is mainly a cheap and illegal form of advertisement exploiting that thousands of users are easily reachable on the Internet. Although it is illegal, most legislation enforcements fail due to the inability to identify the spammer. The deficit due to the workers time-loss can be measured in thousands of dollars [4]. A new solution in spam filtering always indicates the spammers to try to find the way through the filters. That causes spam filtering to be a never-ending fight.

## 1.4 APPLICATION OF EMAIL FILTER AND CLASSIFIER

In order to try out solving some of the problems mentioned above, an application for email filtration and classification has been designed. This application will have features mentioned below:

- This application will allow users to take backup of their emails on other storage devices like CD, floppy, etc., which will help him to view as per their desire.
- This application will provide facility of pre-defined folders as well as user-defined classified folders.
- In near future, it might be the case that mail servers will charge some amount for reading emails, so this application will be useful for old emails, which we had backed up.
- Downloaded emails will be stored in two ways, in which one is according to category and the other is according to year and month of their reception.

## 1.5 MOTIVATION

It is a general observation that people are relying more and more on emails as a communication medium for their professional benefits. The manual segregation of large number of emails is time consuming and sometimes error prone method.

Therefore we have thought of generating an *Automatic Email Filtration and Classification Application*. This segregation can be very flexible and general such that the user should be able to customize this segregation mechanism based on a set *of* keywords and their frequencies as per his personal choice.

## 1.6 PROBLEM STATEMENT

Emails in our *inbox* are in a mesh form, which is not a proper arrangement. If a user wants to search for an email of his interest of particular category, then he needs to go through his whole list until he locates the email after an involved search. This might be very tedious and time consuming work because it is not possible to filter such a long list of emails manually. So, to provide relief to user, there must be a technique for Automatic Email Classification, in which emails will fall into some predefined email categories like Job, Education, Subscription, etc. and further if user wants to create his own category on behalf of his interest then he can do that also. This adds the useful feature of customization based on user's interest. This application can be further extended by filtering the useless spam emails from inbox. Diagrammatically, the two processes are shown in Figure 1:

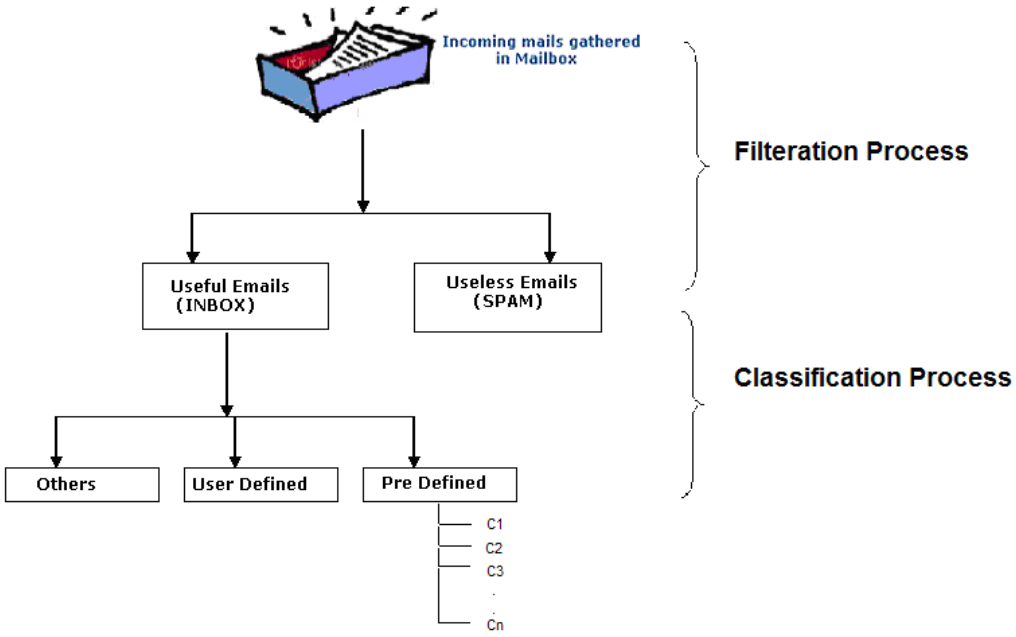


Figure 1: Diagrammatic representation of segregating emails into two stages

## 1.7 SCOPE OF THE WORK

The created desktop application will use to filter and classify emails on client side. The scope of this filtration and classification process is limited only for those mail servers which are supporting POP or IMAP protocol. Nowadays, most of the mail servers are likely to support these two protocols, including Yahoo, Gmail, and Hotmail etc. In essence, we can say that it covers most of the frequently used and popular mail servers.

## 1.8 OUTLINE OF THESIS

The thesis is organized as follows. Chapter 1 will begin by introducing the dissertation problem. Chapter 2 explains, what is spam, their history, types of spam, methods to find against spam and finally chapter is concluded by splashing the light on the necessity of spam filtering. In chapter 3, email classification, type of classification and its need is introduced.

Further, Chapter 4 starts by examining the algorithm named Vector Space Model with its preparatory steps and detail methodology. Chapter 5 focuses on implementation part and its result via various graphs and tables. It also gives brief overview of the libraries used in development work. Appropriate conclusions and future work are discussed in Chapter 6.

# 2 SPAM MAIL FILTRATION

## INTRODUCTION

This chapter introduces the fact and phenomenon of *Unsolicited Commercial Email (UCE),* often called *spam*. An effort is made here to cover various definitions of spam and it also brings out the history of spam. Further, it is followed by a short introduction to the necessity of spam filtering and users' opinion about the topic. Then the different kinds of spam are introduced, with their trends and characteristics and finally the chapter gives a glance at the common filter techniques and the basic solutions for handling spam emails.

## 2.1 THE DEFINITION OF SPAM

There are several definitions for spam. Most of the spam can be termed as *unwanted e-mail* but not all of the unwanted e-mails are spam. Another term would be *unsolicited commercial e-mail* [4], but unfortunately spam is not only advertising material. (Different kinds of unwanted e-mails are discussed later on.) Spam can be also defined as *junk mail* but it implicates the question: what is a junk mail? Although most of the e-mail users know what spam is, but it is not obvious how to define spam and spamming. Wikipedia, the biggest encyclopedia on the Internet gives the following definition for Spam: "E-mail spam involves sending nearly identical messages to thousands (or millions) of recipients." [50] Spamming: "Spamming is the abuse of any electronic communications medium to send unsolicited messages in bulk." [60] As a summary one could agree that spam is something unsolicited, unwanted email that is mostly also an advertisement material. However not all unwanted e-mail letters are spam and not all spam is an advertisement. In other words, there is not a clear cut and unique definition for spam and what is discussed above are only some of its properties that show relationship among spam and other emails and pictorially it can be seen in Figure 2.

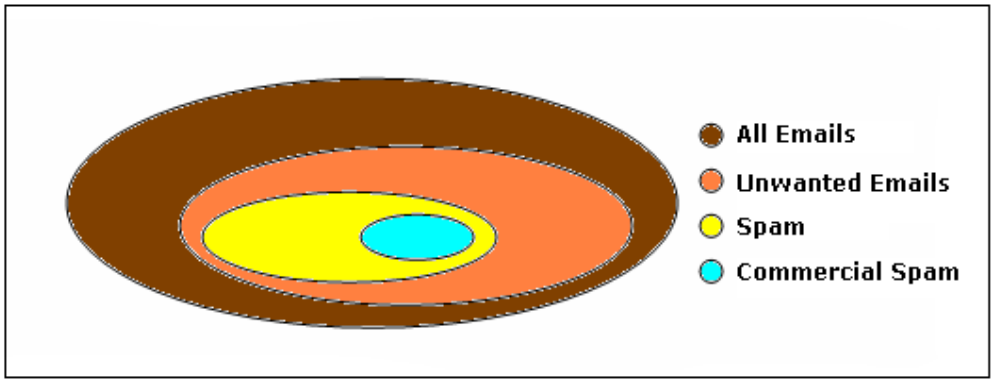


Figure 2: The set of spam, compared to other e-mail sets

## 2.2 THE HISTORY OF SPAM

### The different periods in spam history

The history of spam can be separated into three different stages:

- The first period: In early years spam letters were addressed and sent manually
- The second period: Subsequently spammers started using machines, that led to the dramatical rise in the volume of spam
- The third period: The final part is when machine learning appeared at spam filtering and made the filtering substantially effective.

### The first period – the early years

The era of spamming began in 1978. At the initial years, spam was sent manually. Spamming needed huge amount of human resource and so it was not able to reach millions of users. Spammers used to send the messages one by one and used inner address lists of small communities. These were the tolerable years.

### The second period – letters sent by machines

The first mentionable change happened in 1994, when a programmer was hired in order to write a spammer program. Since that time the amount of received spam letters has been growing uncontrolled. The first very famous spammer was Jeff Slaton, the “Spam King”.

Spamming slowly became a business. The first known e-mail list was offered for sale in 1995 with more than 2 million addresses. After the first

weak attempts in 1997 the first real spam-filter software was made. Nevertheless spam turned to be totally out of control, by the end of 1997 statistics showed an exponential growth. The early spam filters were rule-based software. All the new tricks of the spammers indicated to settle up new rules into the filters. Unfortunately, the spammers were able to try the filters with their messages, so they had the ability to change the letter to get through the filters.

**The third period – machines against machines**

The big breakthrough was in 2002, when Paul Graham published "A Plan For Spam". This article introduced the usage of machine learning and statistical classification techniques on the field of spam filtering. Paul Graham wrote [5]: "I think it's possible to stop spam, and that content-based filters are the way to do it. If we can write software that recognizes their messages, there is no way they can get around that."

With the use of content based technique, filtering not only became significantly more effective, but first time the filter learned the e-mail characteristics of the users. It was the first solution, which made it impossible for the spammers to test the filters before sending their e-mails, hence every users' filter worked with different knowledge base.

From this time spammers could only hope that they found the right method to pass the filters, but they could not be sure whether they could reach a defined number of users. Even if the spammers figured out a lot of new tricks, like good-word attacks, the Content Based filters still have a good chance to filter them out.

## 2.3 THE NECESSITY OF SPAM FILTERING

**Overview**

Nowadays the implementation of a reliable spam filter becomes more and more important for e-mail users since they have to face with the growing amount of uninvited e-mails. The spam is mainly a cheap and illegal form of advertisement exploiting that thousands of users are easily reachable on the Internet. Although it is illegal, most legislation enforcements fail

due to the inability to identify the spammer. In other words, spammers can easily hide themselves and they try to achieve the goal of advertising using minimum investment. The deficit due to the workers time-loss can be measured in thousands of dollars. A new solution in spam filtering always indicates the spammers to try to find the way through the filters. That causes spam filtering to be a never-ending fight. Currently some of the most efficient filters are based on Content Based classifiers, although there are several less complex methods as well to amend the filtering.

**The users' side**

The irritating-rate of spam letters depends on the e-mail habits of the users. For example if somebody receives 10 emails every week, and there is only one spam than it could be easily imaginable that to press the delete button is quicker and more comfortable than implementing any kind of filtering. Spam causes problem usually for users who receive hundreds of e-mails per week. In such a situation, user finds himself in a very frustrating situation to press the delete button hundred of times manually every week. In the initial years, the main problem was the connection speed. Especially users connecting through modem to the Internet were suffered by waiting significantly more time, spent with downloading e-mail. To wait ten minutes to download a couple of emails can be painful, out of which may be 70% of the emails were actually spam letters. Another major problem is recognization of each and every email, which is itself a very time consuming and a tedious job.

Nowadays, when more and more users use broadband Internet connection, the downloading time has reduced significantly. Spam is usually a short message and/or smaller picture. Today the problem is that the users have to face the necessity of deleting the unwanted letters, or the difficulty to pick out the important ones from the big amount of junks. Moreover there is the annoying question: "why I have to get all these junk emails?"

**Facts and figures**

The statistics in April 2005 showed that 67% of the Internet users dis

like spam and 33% of them not only dislike spam, but also feel themselves frustrating enough, while only 33% of the users have no problem with it. Although the Figure 3 shows almost a balanced 33-33-33 percent, it is important to highlight, that two-thirds of the users have problem with spam [8].

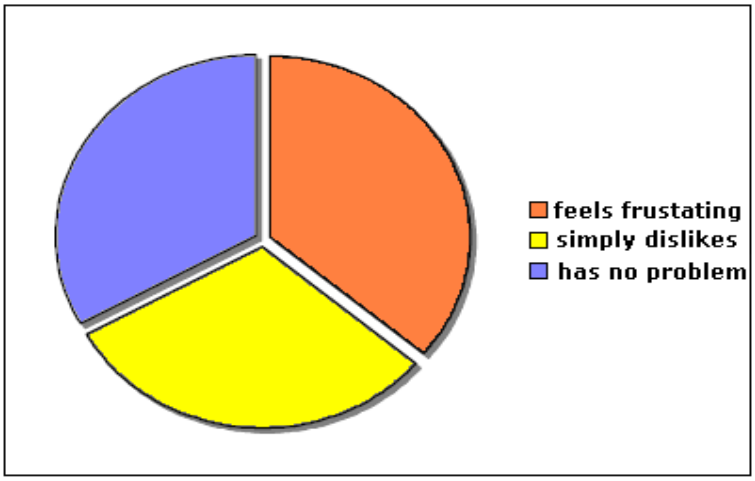


Figure 3: Opinions of the users about spam

The main thing that is important to understand is that advertisement through spam works, only if people take interest in spam emails. If people buy from the offered product or if people can be fooled by phishing, spamming can survive. If nobody would respond to spammers, or buy from them, surely spamming would eventually reach its end.

## 2.4 DIFFERENT KINDS OF SPAM

There are many ways of classifying spam. First, there is the mode of delivery. At the present moment, spam can be sent through emails, newsgroups, mobile phone's Short Message Service (SMS), and instant messenger services. Spam can also be classified according to the sender: strangers, friends and acquaintances, and subscription services [10]. Most importantly, spam can be differentiated by content, the common types of which are commercial spam, political, industrial, chain mails, and others such as self-spreading virus emails.

The United States' Federal Trade Commission studied 250,000 spam messages forwarded by Internet users and identified twelve most likely spam scams [11]:

- business opportunity scams
- making money by sending bulk emailing
- chain letters
- work-at-home schemes
- health and diet scams
- easy money
- get something for free
- investment opportunities
- cable descrambler kits
- guaranteed loans or credits on easy terms
- credit repair scams and
- vacation prize promotions

Spam can also be classified according to the sender's motive. Generally, spam is sent for informational purposes, namely to solicit positive responses. On the other hand, sending of spam solely for the purpose of disrupting the working of a recipient or email service provider's (ESP) network is known as mail-bombing. Spam can also be sent maliciously in order to confuse or misinform the recipients.

There are at least three main different kinds of spam. More than 99% of spam falls into one or more of the categories listed below [5].

**Advertisement spam**

Most spam is commercial advertisement, often a direct product offer. Spam costs the sender very little to send, compared to other advertisement methods. The most common subcategories of the advertisement spam are:

- Online Pharmacy spam: Spam promoting different versions of Viagra, Cialis, Anti-depressant pills that can be purchased online.
- Penny Stock spam: Stock-encouraging spam, encouraging people to buy cheap stocks.

- Porn dating spam: Porn-sites and (sex-) dating sites were often marketed via spam.
- Pirate Software spam: spam offering pirate software, usually much cheaper than the official prices.
- Online Casino spam: Spam promoting gambling in online casinos.
- Fake Degrees spam: Spammers often try to sell fake Degrees and Diplomas.
- Mule job spam: Promoting jobs 'working from home' (which are typically scams, or mule jobs, like laundering money).

**Financial spam**

While advertisement spam have at least a little probability, that the responder could get something for the sent money, the financial spam only tries to fool people and get their money somehow, without the chance to buy anything. The most common financial spam types are the following:

- 419 scams: Usually a plea for help to recover millions of dollars from a bank account in a foreign country (typically Nigeria).
- Lottery spam: Similar to the 419 scam, these spam are telling, 'You have already won X Million' in order to try to extract transfer fees etc.

**Phishing**

Phishing spam is fake alert from banks, Pay Pal, eBay etc, and it asks for confirmation, validation or monitoring of details in order to defraud people of their personal information. Phishing spam are usually linked to fake login sites, which can be used to capture user details (e.g. passwords) in order to use this information to steal money or goods. Phishing e-mails use mostly the listed methods:

- Using the company's Image: the fraudulent e-mails often contain the company's logo and use similar fonts and color schemes.
- Links to the real company's site: The main link in a fraudulent e-mail sends the recipient to the fraudulent phishing web site, but many fraudulent e-mails include other links that send the recipient to sections of the real company's web site.

- E-mail appears to be from the spoofed company: To further convince the recipient that the e-mail originated from the company, the spammers use an e-mail address that appears to be from the company (e.g., @ebay.com, @paypal.com).

**The trends in spam messages**

Although the content of spam letters is in continuous change, it is difficult to pin point any abrupt change taking place in a short duration. Analyze the data of 5 months in Figure 4, that the most common spam is the medicine product offer spam, with a significant rate one third of all spam. The second biggest category is the financial spam, where there is a slight decrement from around 40% to 30%. The direct product offering follows a waving tendency and the adult contents get fortunately less accent. The next figure shows the trends in spam.

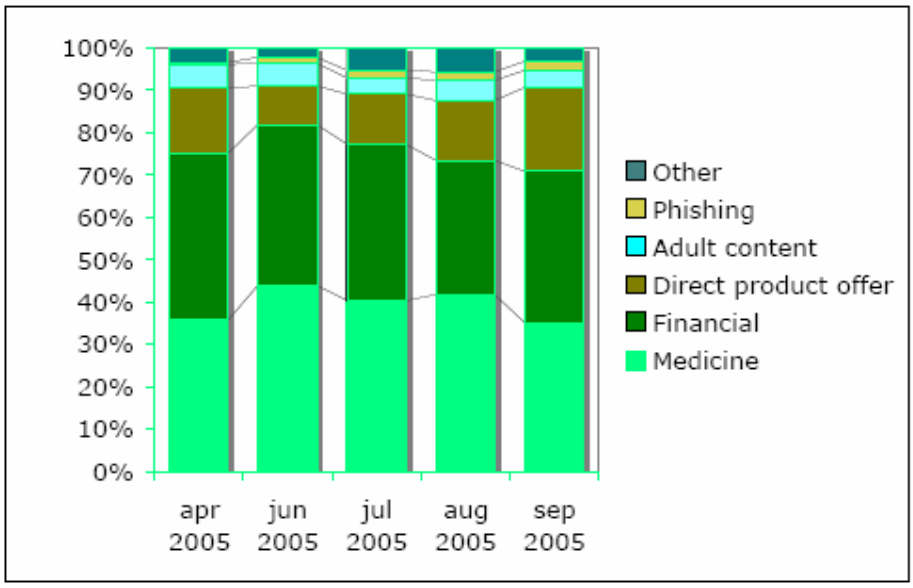


Figure 4: The trends in spam

## 2.5 THE SIMPLE METHODS TO FIGHT AGAINST SPAM

**Overview**

Several different very simple methods have been discovered during the evolution of spam filtering. These are ideal to be combined with more complex parts of spam filtering; therefore the area of spam filtering

becomes very complex. The biggest effectiveness can be reached by using together small parts of filtering methods.

### 2.5.1 Keywords

One of the first solutions is to search for keywords in the e-mail's subject. It means to scan the subject for words, related to spam letters. This is a simple language analysis, works only by match specific phrases. This method has unfortunately several problems, since the spam letters topic changes time by time. This can be handled by a keyword list that is regularly updated, but the smallest change in the words of the subject leads to mismatch (e.g. write "softw@re" in spite of "software"). Filtering based on a single word had a potential success rate of around 80 percent.

### 2.5.2 Blacklist

There is a need to make difference between two levels of blacklisting: the network-level and the address-level blacklisting. The network-level blacklisting is based on creating intentional network outages. The method has the ability the detect spam letters based on its origin rather than its content. Moreover if a legitimate user was accidentally blacklisted, there is no way, to get off the blacklist, hence all mails ware rejected from the blacklisted part of the network.

The address-level blacklist is an updated list of known spam sender addresses. There are on-line accessible blacklists and the user can administrate personal blacklist as well.

### 2.5.3 Whitelist

White listing is the opposite of blacklisting. Content filtering identifies spam, while white listing requires identifying users. A white list is a collection of reliable contacts. If e-mail comes from the members of this list, it should be marked automatically as legitimate letter what is also called *ham*. Just as the blacklisting, the white list also needs a continuous upgrade and refreshment. After all of its drawbacks, white listing is used only for classify letters as legitimate mails, and has nothing to do when the sender is unknown. If the blacklist and white list methods are used

together, further filtering is only required for letters that do not match any of the entries in the two lists.

### 2.5.4 Throttling

The throttling simply slows down the rate at which a single network or host can send traffic. Probably this is the most sensible way to fight spam. For example a legitimate mailing list may send out huge quantities of mail, but each message is addressed to different users on different networks. A spammer on the other hand may use dictionary attack, and tries to find valid e-mail addresses on one network. In this case throttling can lead to a drawback for the spammers, but it also uses more resources from the legitimate senders.

### 2.5.5 Collaborative filtering

To filter in collaborative way, relies on “two hands are better than one” philosophy. It will allow individuals in trusted groups to share message inoculations with the other group members. For example if ten users agree to share their training errors and to see a new spam type or a new mutation has the probability 0.9 for each person. The person, who received the letter first trains not only his, but the filter of all other users. After all if one received 1 spam out of 100 legitimate e-mails, after sharing the training one will receive 1 spam out of 1000. The weakness of collaborative filtering is the participating community itself. In large communities, there is both a high maintenance loop and the risk of false positives.

### 2.5.6 Netiquette and Enforcement of Agreement

Internet service providers and email service providers, such as MSN Hotmail and Yahoo!, are also involved in fighting spam from the sender’s end. Email Service Providers (ESPs) have to act on their own subscribers because spamming brings bad reputation to the ESPs and unnecessarily burden on network resources. When a user violates these terms of agreement by sending spam, the ESP will be able to suspend or terminate his account. Nevertheless, this is not a very effective measure. Although it hinders a spammer from continue using an ESP’s service to conduct spamming related activities.

The distaste for spam by the early adopters of the Internet can best be reflected in the norms known as netiquettes. These netiquettes came about during a time when the Internet was primarily an academic and research facility, of which commercial activities are forbidden.

## 2.6 SUMMARY

A basic explanation of spamming has been outlined. Various arguments in support of and against spam are laid out, and the parties and non-law strategies against spam are identified. On the whole, there seems to be more reasons for Internet users to be against spam than supporting it.

# 3 MAIL CLASSIFICATION

## 3.1 INTRODUCTION

Classification of text documents is an example of supervised learning that seeks to build a probabilistic model of a function that maps documents to classifications. In supervised learning of text, where an entire document represents one example to be classified, a learning algorithm is presented with a set of already classified, or labeled, examples. This set is called the training set. Information Classification deals mainly with large amounts of incoming data. It is based on description of an individual's or a group's information preference, called profiles, which typically represent long-term user's interests. In our context, classification/filtering process aims to select and/or eliminate information from a dynamic data-stream. The description of user's interests (profiles) is the most crucial and difficult operation in the building of an information filtering system. Indeed, this operation addresses the following questions: What are the user's information interests? How are they identified? How are they represented? Can they be updated easily? etc. The effectiveness of filtering is closely dependent on the quality of the profiles representation. In general, the terms contained in a document are different from those the user would use to specify his interests. Thus, it is not easy for users to specify what those interests are because they differ from a user to another, and they are constantly changing [8]. To know how to configure the user's model for an information filtering system, an experimental study can be undertaken to observe how the users process, evaluate and classify their relevant documents or we can say mails. Once a model has been constructed, it can be used to predict the classification of future documents. The accuracy of such models is largely dependent on the "representative ness" of the training data with respect to newly acquired data to be classified. In general, the more representative the training data, the better the performance. A larger number of training examples is often better, because a larger sample is likely to be more reflective of the actual distribution of the data as a whole.

## 3.2 NEED OF CLASSIFICATION

In the last decade, the importance of email as a means of communication has increased dramatically, and, due to this development, the need to organize these emails has grown as well. For companies receiving huge numbers of emails to a single email address, e.g. the address of the company's customer service, it is particularly important to make sure that all emails are responded to within a reasonable amount of time by someone with the knowledge required to reply to the email. In order to meet this challenge, automatic categorization of all incoming emails can be extremely useful because it can help route an email to the right person.

Automatic classification is useful for a wide range of applications such as organizing web, intranet or portal pages into topic directories, filtering news feeds or mails and many more. In the classical scenario, it is often assumed that all topic categories are known and that the training corpus provides example documents for all these categories [14]. The major need of classification is to get rid the user from manual filing of documents and sort the mails automatically into corresponding folders. Its main aim is to save user's time wasted in going through spam mails and reading them.

## 3.3 TASK

The task of classifying emails is strongly related to the more general task of automatic text categorization (TC), which aims at categorizing text documents according to their topic by assigning tags from a pre-defined set of categories. The categorization decision is based on the document's contents and usually involves some analysis of the words in the documents. In principle, there are several ways to do mail classification and hence several decisions to be made before starting: One important decision is whether to assign exactly one category to each document (single-label ) or to allow any number of categories to be assigned to the same document (multi-label).

Another decision to make is whether to enforce a classification for every document or to classify only those that can be classified at a certain

confidence level, i.e. with a probability above a certain threshold that needs to be determined in advance. A third alternative is to build the system in such a way that for each document d, it ranks the categories according to their appropriateness to d. Such a ranked list would then help a human expert to take the categorization decision. When building a text classification system, two basic approaches are possible. The most popular approach was to use knowledge engineering, i.e. to manually define a set of logical rules. However, this approach is costly because it involves enormous labor power of not only the knowledge engineer but also a domain expert knowing which documents belong to which of the possible categories. Therefore, even though some real-world applications still use human encoded linguistic rules for mail classification, machine learning has become the dominant approach for this task in the last 15 years. Here, a so-called learner automatically builds a classifier for each category by observing the characteristics of the documents that have been manually assigned this category by a domain expert. As manually tagged data in the form of pre-classified documents is provided by the domain expert, this kind of text classification is an example of supervised learning. Each new document is then classified under the category whose characteristics best match the characteristics of the document. Even though you still need a knowledge engineer and a domain expert, their tasks have become easier. Instead of describing the concept of each category in words, the domain expert now only has to select instances of it. The knowledge engineer, who had to build a classifier by defining rules manually in the knowledge engineering approach, now has to build a system that learns rules, i.e. builds classifiers, automatically. Since these systems work domain-independently, they are also highly portable to different domains. Using machine learning, the dominant approach for mail classification tasks is to use the presence or absence of different words as so-called features, i.e. classify each email based on the words that it does and does not contain. In the most common document representation in mail categorization, each document is seen as a so-called bag-of-words, i.e. the set of all words appearing in the document; context and word order are ignored. Each document is then converted into a vector mapping every feature to a specific value. Feature values can be

either binary (e.g. 1 if the feature is present, 0 otherwise) or continuous (e.g. a weight assigned to each feature based on the importance of the feature).

## 3.4 PLACEMENT OF FILTER

The possible good placement of the filter is between mail store and the mail store. Filter will fetch the new incoming emails from mail store and does the processing task. Then filter deliver that filtered emails on client machine, which is shown in Figure 5.

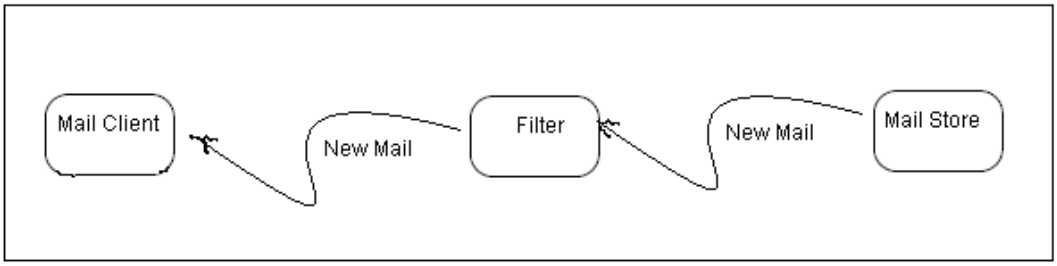


Figure 5: Possible placement of filter

## 3.5 TYPES OF CLASSIFICATION SCHEMES

This thesis will contribute to two types of filter creation schemes, from that one is based on predefined categories and other one is on-behalf of users. The two schemes are:

- Pre-defined Categorization
- Customized Categorization

In case of *Pre-defined Categorization,* some categories are already defined by this application like job, subscription, education, spam and inbox. Consequently, mails regarding these categories will automatically fall into corresponding ones, without user intervention.

Whereas in case of *Customized Categorization,* user will create his/her own category/filter as per his/her interest on basis of the keywords supplied.

## 3.6 SUMMARY

Recent growth in the use of email for communication and the corresponding growth in the volume of email received have made *Automatic Classification* of email desirable. The algorithm, which is implemented and experimented for this purpose is elaborated in upcoming chapters.

# 4 VECTOR SPACE MODEL ALGORITHM

## 4.1 OVERVIEW

Vector Space Model (or term vector model) uses algebraic concept for representing text documents as vectors of identifiers, such as, for example, index terms. It is used in information filtering, information retrieval, indexing and relevance rankings. Many information retrieval systems are based on Vector Space Model that represents document as a vector of index terms. We represents document as a vector of terms in VSM. The basis of the vector space corresponds to distinct terms in the document collection. Components of the document vector are the weights of the corresponding terms that represent their relative importance in the documents.

A document is represented as a vector. Each element in the vector corresponds to a separate term. If a term occurs in the document, its value in the vector is non-zero. Several different ways of computing these values, also known as (term) weights, have been developed [13].

## 4.2 PREPARATORY STEPS

Steps which are to be performed before implementing the Vector Space Algorithm are as follows:

### Data Preprocessing

In the original corpus, the actual messages were embedded in all kinds of header information (i.e. name and address of the sender) and content-types (i.e. text/plain, multipart/mixed etc.). Although information about the email sender may be useful for other email classification tasks, it is likely to harm performance in this case because most senders will only occur once in the corpus. It was treated differently before processing rest of the data. In addition, the file containing the message usually also contained reply message sent by some customer service representative. This reply was deleted as well because it might make the results less representative. After removing these additional data, the email bodies

were saved in an EML file to facilitate further processing. When classifying emails, this pre-processing will not be necessary because the email bodies can be extracted automatically using the JavaMail APIs.

**Identification of categories**

Since this application development work is an independent study and not directly connected to a current project of any organization, the categories used in the application are designed from a general consideration. In other words, the categories do not pertain to any specific organization. Having examined a preliminary set of test emails (comprising of 78 emails), following categories and assignment criteria is decided:

- Job
- Education
- Subscription
- Friends
- Spam
- Inbox

These categories were chosen for several reasons as they allow a very straightforward manual annotation because most of the emails can be classified as belonging to precisely one of these possible categories. Moreover, there seems to be a reasonable number of categories for the restricted amount of data available. Given the criteria specified above and using a set of 900 emails, 828 of these were classified as belonging to exactly one of the categories shown in Table 1:

**Table 1: Categories with corresponding number of emails**

| Category | # of emails |
|---|---|
| Job | 242 |
| Education | 73 |
| Subscription | 119 |
| Spam | 394 |

For the remaining 72 emails, it is impossible to assign exactly one category, even for a human annotator, either because the email content would fit into more than one category or because none of the categories

fit. Initially it was visualized to add a fourth category that would collect all those emails that do not fit in anywhere else; however, this idea was discarded because the contents of the 72 affected emails are much too diverse to be merged into a single category. Such a category was not found to be reasonable. Consequently it was decided to take the 72 emails out of the training data and allow user to define his/her own category. This will presumably make the results of experiments slightly better as compared to situation in which all emails are used for training.

**Attribute Selection Problem**

The term *attribute* describes a statistic that represents a measurement of some aspect of a given user's email activity or behavior. In this thesis work, focus is on those features which can be believed from observation and could be useful in categorizing emails. Similar techniques could also be applied to design attributes for spam detection. There are dozens of features which can be implemented for obtaining the set of statistics which will accurately distinguish the email behavior. One of the attributes which have been selected for this work is MIME types.

**MIME Types**

MIME is an Internet Standard that extends the format of email to support:

- Text in character set
- Non-text attachments
- Multipart message body

MIME defines a collection of e-mail headers for specifying additional attributes of a message including *content type*, and defines a set of *transfer encodings* which can be used to represent data. There are several *content types* as:

- multipart/alternative
- multipart/mixed
- text/plain
- text/html

For mail classification, here focus is only on text/plain type, which includes simple text messages only.

## 4.3 ALGORITHM - VECTOR SPACE MODEL

The Vector Space Model is applied for both the processes named

(a) Filtration Process

(b) Classification Process

The major steps, which involve in applying the Vector Space Model, can be pointed out as below:

### 4.3.1 Loading

The first step in implementing VSM is loading the email to client side. The most appropriate protocols are combined with email programs and the downloaded message is filtered in real-time.

### 4.3.2 Pre-Filtering

Every single method based on text string searching needs quite a reliable pre-filtering to make it easier match the words in the previously stored list of the token dictionary. Without pre-filtering, the tokens kept in the token-dictionary can reach an unlimited size, since the spammers realized in a very short time, that writing the words in their dictionary-form leads to an easy filtering, and so they began to change some letters or put extra characters in (e.g.: softw@re, r_i_c_h). This has an effect not only on the keyword searching methods but also makes statistical filtering more difficult. The pre-filter's role is to remove all unused words (i.e. tokens separated by blank space) and basic punctuation marks (i.e. [,.!?;:"()]) beforehand.

The other possible role of pre-filters is to try changing the words back to their original form and to make the search engines able to find them. An interesting newer method does not try to change the words back, rather tries to find similar words. VSM algorithm, works on this approach. Even so many of the filters are not using pre-filtering and so those deal with larger sets of tokens [14].

### 4.3.3 Tokenization

Tokens are individual small parts of the text. Many filters work with simple word-by-word tokenizing, where the text is separated into words and the words have their own values in the token-dictionary. In this case, one

word will be one token. However, some filters offer word-pair tokenization, which requires more maintenance and more resources for calculation. For example: the word "software" and the world "cheap" can also be found in legitimate e-mails, but the word-pair "cheap software" can be a strong indicator of being spam. So, the decision of spam and useful mail be done on the basis of occurrence of frequency of particular token, as respect with other tokens. After the tokenization, the tokens are handled separately [16] and the training data on which the classification model will be built is provided in the *config* files format. For example, consider that a one new incoming mail named **mail1** has following content:

*"Dear Subscriber, lots of job. Please register here for detail information of new job openings".*

For this email the listed tokens will be, *dear*, *subscriber*, *lots*, *of*, *please*, *register*, *here*, *for*, *detail*, *information*, *new*, *job*, *openings.*

### 4.3.4 Vector Creation and Comparison

After tokenization, one needs to create the vector that gives the information about how many times each word appears in a document. For each and every incoming mail two vectors will be created, for all predefined categories. From this, one vector will be according to the tokens of new incoming mail and other will be according to that of specific category.

For example, consider two predefined categories as Job and Subscription. Both these categories are as follows:

Job= {job, openings, experience, freshers}
Subscription= {subscription, register, unsubscribe, member, subscriber}

Here corresponding keywords for Job and Subscription are shown in form of Job and Subscription sets respectively. After generating this information, next thing is to decide the length of vector. Here, the length of vector will be calculated by the sum of total number of tokens of all pre

defined categories. So, according to the example taken above, this length will be 9, as 4 is due to Job category's keywords and 5 is due to Subscription category's keywords.

Further, these 9 words (i.e. experience, freshers, job, member, openings, register, subscriber, subscription, unsubscribe) will be stored in some sort of temporary storage like hash table for calculation purpose. After getting the length of vector, next step is vector creation, which is nothing, but simply an array of frequency of particular word. As it is discussed previously that there is need of two vectors for calculation. So, considering the message *"Dear Subscriber, lots of job. Please register here for detail information of new job openings":* Here one vector named $V1_{incoming}$ will be created on behalf of incoming mail as,

For incoming mail, $V1_{incoming}$ = {0, 0, 2, 0, 1, 1, 1, 0, 0}

and other vector V2 will be created as per number of predefined categories.

In given example, there are two predefined categories (i.e.; job and subscription). So, two values of V2 will be calculated over here which can be named as $V2_{job}$ and $V2_{subscription}$.

For job category, $V2_{job}$= {1, 1, 1, 0, 1, 0, 0, 0, 0}

For subscription category, $V2_{subscription}$= {0, 0, 0, 1, 0, 1, 1, 1, 1}

After getting the vectors, next step is to compare the values between two vectors for similarity match of documents, i.e. match
V1 with $V2_{job,}$ and
V1 with $V2_{subscription.}$

### 4.3.5 Similarity Match

The goal is to efficiently match documents (emails here), distributed throughout the network, that share strong similarities in their content.

The goal here is to focus on highly similar documents. The similarity of two documents can be expressed through a mathematical function θ, which is known as Cosine Similarity,

$$\theta = \frac{V1 * V2}{|V1| * |V2|} \quad \text{...eq (1)}$$

where θ is value of similarity match parameter and V1, V2 are two vectors. Value of V1 * V2 is just the simple multiplication of corresponding elements and |V1|, |V2| are the norms of the two vectors respectively which can be given by:

$$|V| = \sum_{i=1}^{n} X_i^2 \quad \text{...eq (2)}$$

here vector V is made up of *n* elements $x_i$ where i=1 to n.
Now again, moving back to previous example:

For Job category, value of θ is calculated as,
$\mathbf{V1*V2_{job}}$ = 0*1 + 0*1 + 2*1 + 0*0 + 1*1 + 1*0 + 1*0 + 0*0 + 0*0= 3

and value of norms is,
|V1| = 0+0+4+0+1+1+1+0+0 = 7
$| V2_{job} |$ = 1+1+1+0+1+0+0+0+0 = 4

Hence $\theta_{v1, v2job}$ = 3/ (7*4) = 0.107

So, the same calculation for Subscription category, we will get $\theta_{v1,}$ $v2_{subscription}$ = 0.057.

For the purpose of matching similarity among documents, vectors had been prepared. Now next step is to assign them ranking as in ascending or descending order, to get the maximum value of θ. The range of value of θ lies between -1 and 1 as {-1, 1}. Here values 1 and -1 correspond to perfect match and mismatch respectively. The discussion of value of θ is brought up in upcoming chapters.

### 4.3.6 Ranking Documents

To establish a procedure for vector space model, it is assumed that index terms are single words extracted from documents. Once the term weights are determined and θ is calculated, we need a ranking function to measure similarity between the query and document vectors [10]. Here the higher the rank, lower the rate of matching of the document to that corresponding category, means the category which has the lowest rank, will contain the incoming mail and all mails will be categorize in this same manner.

## 4.4 SUMMARY

As far as Vector Space Model is concerned, it can be applied to both *filtration* and *classification* process and give very good results. The accuracy of results depends on the threshold of similarity match value θ, to a great extent. The various values of θ are experimented and 0.0003 was finalized as a threshold.

# 5 IMPLEMENTATION AND RESULTS

## IMPLEMENTATION

In creation of email filtration and classification system, some libraries are used for implementation of various functions. These libraries are easily compatible with JAVA. The description of some of the major libraries is as follows:

### 5.1 Useful libraries

#### 5.1.1 Lucene API

Apache Lucene is a high-performance, full-featured text search engine library written entirely in Java. It is a technology suitable for nearly any application that requires full-text search, especially cross-platform. It is based on very powerful indexing and can be used for large collections also. Lucene consists of core APIs that allow indexing and searching of text. Given a set of text files, Lucene can create indexes and allows searching those indexes with complex queries. Explanation of Lucene can be further summarized under the following points:

**Analyzing Text to Be Indexed**

Lucene processes the text to be indexed with *analyzers*. Analyzers are used to tokenize text, extract relevant words, discard common words, stem the words (reduce them to the root form, meaning that *bowling*, *bowler* and *bowls* are reduced to *bowl*), and perform any other desired processing before storing it into the index. Lucene provides an Analyzer abstract class, and three implementations of Analyzer. Glossing over the details:

- SimpleAnalyzer: SimpleAnalyzer seems to just use a tokenizer that converts all of the input to lower case.
- StopAnalyzer: StopAnalyzer includes the lower-case filter, and also has a filter that drops out any "stop words", words like articles (a, an, the, etc) that occur so commonly in English that they might as well be noise for searching purposes. StopAnalyzer comes with a set of stop words, but you can instantiate it with your own array of stop words.

- StandardAnalyzer: StandardAnalyzer does both lower-case and stop-word filtering, and in addition tries to do some basic clean-up of words, for example taking out apostrophes (') and removing periods from acronyms (i.e. "T.L.A." becomes "TLA").

**Searching Indexes**

Once created, indexes can be searched by providing complex queries specifying the field and the term that needs to be searched. For example, the user query abstract:system AND email:abc@mit.edu will result in all documents that contain system in the abstract and have the email address abc@mit.edu. Documents that match the query are ranked based on the number of times the terms occurs in the document and the number of documents that have the terms. Lucene implements a ranking mechanism and gives us the flexibility to modify it if required.

**Class Summary**

The classes of Lucene API are useful for searching and indexing purpose. The brief summary of classes is given in Table 2:

**Table 2: JAVA Lucene Class Summary**

| JAVA Class | Usage |
|---|---|
| Analyzer | An Analyzer builds token streams, which analyze text. |
| LowerCaseFilter | Normalizes token text to lower case. |
| LowerCaseTokenizer | LowerCaseTokenizer performs the function of LetterTokenizer and LowerCaseFilter together. |
| TokenFilter | A TokenFilter is a token stream whose input is another token stream. |
| FilterIndexReader | A FilterIndexReader contains another IndexReader, which it uses as its basic source of data, possibly transforming the data along the way or providing additional functionality. |
| IndexReader | IndexReader is an abstract class, providing an interface for accessing an index. |

| | |
|---|---|
| IndexWriter | An IndexWriter creates and maintains an index. |
| InputStream | Abstract base class for input from a file in a Directory. |
| ParseException | This exception is thrown when parse errors are encountered. |

### 5.1.2 Mail API

The Java Mail API offers a simple, consistent way to write mail applications regardless of the mail service provider (mail server) used and the operating environment (i.e.: Windows, Linux or Web application server). It even accommodates many of the differences on how mail servers are set up with security constraints. Whether someone use an in-house mail server or an outside mail server provided by ISP, one should be able to use the Java Mail API to send and receive mail easily. The Java Mail API implements a lot of the mail functions defined in the World Wide Web mail standards (known as RFCs).

#### Key Java Mail classes

The Java Mail API has a few key classes which are use to send and receive mail. Summary of classes can be shown in the below Table 3:

**Table 3: JAVA Mail Class Summary**

| **Java Mail Class** | **Usage** |
|---|---|
| Session | This is the first class that is used. It creates an instance of a session with support for mail services: send, receive and so on. It has many properties and default values used in a mail session. Transports and Stores are created for a specific session. |
| MimeMessage | This is used to create a simple mail message to send or receive. You will set and get different message properties such as from, recipient, subject and text. |
| Transport | This is a class that represents a mail transport. It provides application support for naming, connection and listening to connection events. |

| | |
|---|---|
| Authenticator | Some mail servers, such as Yahoo, require a mail user to authenticate himself to the server. If authentication is needed, you create a subclass of Authenticator to implement a getPasswordAuthentication method that returns a PasswordAuthentication object with a user name and password. In our example, I created MailAuthenticator as the subclass. When the server checks authentication, the getPasswordAuthentication method is called by the Java Mail framework. |
| Store | A Store object represents a mail store you connect to, such as a POP3 mailbox. Once connected to a store, you can access folders such as "Inbox" to receive your mail. |
| Folder | A folder holds mail in a mail store. If you are using POP3 protocol, the only folder you'll access is "Inbox". If your server supports another protocol, such as IMAP4, you can access other named folders for mail as well. |
| MimeMultipart | A MimeMultipart object is returned when you access the MimeMessage with the getContent method. This object can contain multiple MimeBodyPart objects. |
| MimeBodyPart | A MimeBodyPart object lets you access content objects and headers from a mail message using an input stream. There's also a setText method to set the content easily. Each body part has a MIME type that determines what the content access methods should be. The default MIME type is "text/plain". |
| POP3Message | This is a subclass of MimeMessage specific to a POP3 service.The package for this class is com.sun.mail.pop3 |

**Java Mail Environment properties**

The Java Mail API services use different environment properties to configure their behavior, primarily for the different mail protocols supported by the service. The major properties are shown in Table 4:

**Table 4: JAVA Mail Environment Properties**

| Property name | Usage | Default value |
|---|---|---|
| Mail.store.protocol | Specifies default message access protocol. Session.getStore() returns a store supporting this protocol. You can also explicitly specify the protocol you want to use for a session. | First protocol found in the config files (javamail.properties) |
| Mail.host | Store and Transport connect methods use this value to find the default mail host. | The local machine |
| Mail.user | Specifies the mail user when connecting to a mail server if not explicitly set on the connect method. Generally, you'll want to explicitly set this on the connect. | user.name |
| Mail.protocol.host | Sets the protocol-specific mail server overriding the mail.host setting. We set this as a Java System value for SMTP mail to connect to the Yahoo SMTP mail server as follows: System.setProperty("mail.smtp.host", "smtp.mail.yahoo.com"); | Mail.host |
| Mail.protocol.auth | This defines whether the server requires an authentication object to connect. In our example, we set this with a system value as follows: System.setProperty("mail.smtp.auth", "true"); | False |

### 5.2 Protocols Implemented

The very first step in creation of this desktop client side application requires the mails to be on our client side. So, to fetch emails from mail server to client machine, appropriate protocol must be implemented. The

two protocols which are most popular among mail servers are IMAP and POP. These protocols allow downloading mails on client machine which can be used for further processing. The brief description of both the protocols is as follows:

### 5.2.1 IMAP

IMAP stands for **I**nternet **M**essage **A**ccess **P**rotocol and was defined in 1986 at Stanford University. It permits a client email program to access remote messages as if they were local and download messages to their machines, which means they can still access their emails when they are not connected to the server. It support for concurrent access to shared mailboxes and client software need now to be aware about server's file format. IMAP supports both *online* and *offline* mode of operation, but its special strength is in online mode in which mail is not copied on client side and can be accessed directly on server.

### 5.2.2 POP

POP stands for **P**ost **O**ffice **P**rotocol. It is oldest one and consequently the best known. It was originally defined in October 1984. But it has gone through several variations since its fist incarnation. The current version is POP3. POP was designed to support *offline* mail processing, in which user periodically invokes a mail client program that connects to the server and downloads all the pending mails to the user's own machine.

## 5.3 Required Tools

Some of the software environmental tools are required to bring the application in working condition. The major tools required are:

- Platform: Any platform which supports JVM
- Environment: Net Beans 6.0
- Language: JDK 1.6.0

## 5.4 Experimental Setup

### 5.4.1 Corpora used for classification

As a testcase, here *sltestset* email corpus is used which was collected and generated by Miss Shweta Lodha. This corpus contains messages from 4

different users and some newsgroups which are around 900. Email corpora emails characteristics are given in Table 5:

**Table 5: Statistics of General Email Corpora**

| Corpus | # of Emails |
|---|---|
| User 1 | 100 |
| User 2 | 400 |
| User 3 | 50 |
| User 4 | 150 |
| News Group Messages | 200 |

### 5.4.2 Data Flow Diagram showing Filtration and Classification Processes

The diagram drawn in Figure 6 is the Data Flow Diagram for both the filtration and classification processes.

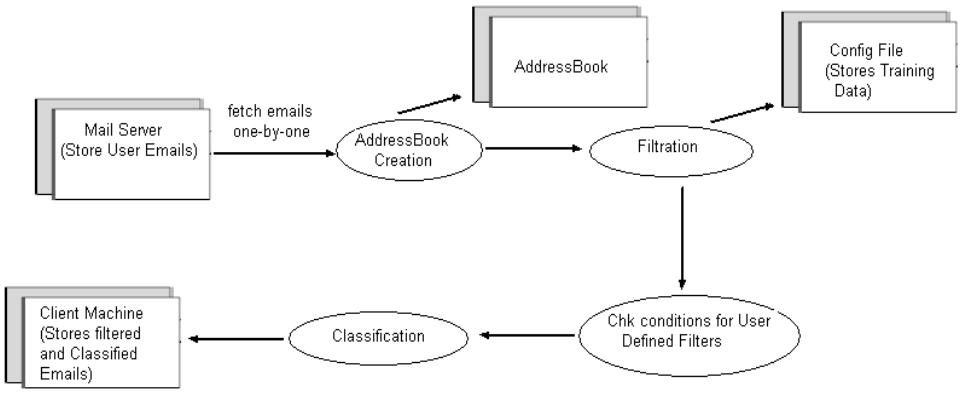


Figure 6: DFD showing Classification and Filtration Process

### 5.4.3 Sequence Diagram showing Filtration and Classification Processes

The Figure 7 reflects the view of sequence of tasks to be performed by mail server, user and the application.

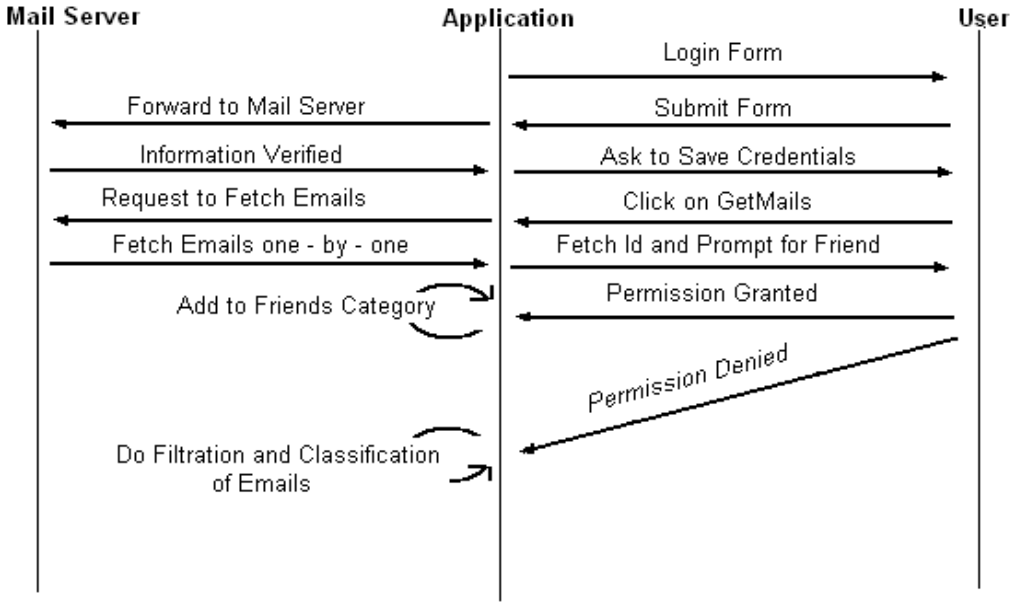


Figure 7: Sequence Diagram showing Classification and Filtration Process

### 5.4.4 Flow of Filtration and Classification Processes

The flow graph drawn in Figure 8, reflect both the filtration and classification process with the step of user-defined filter.

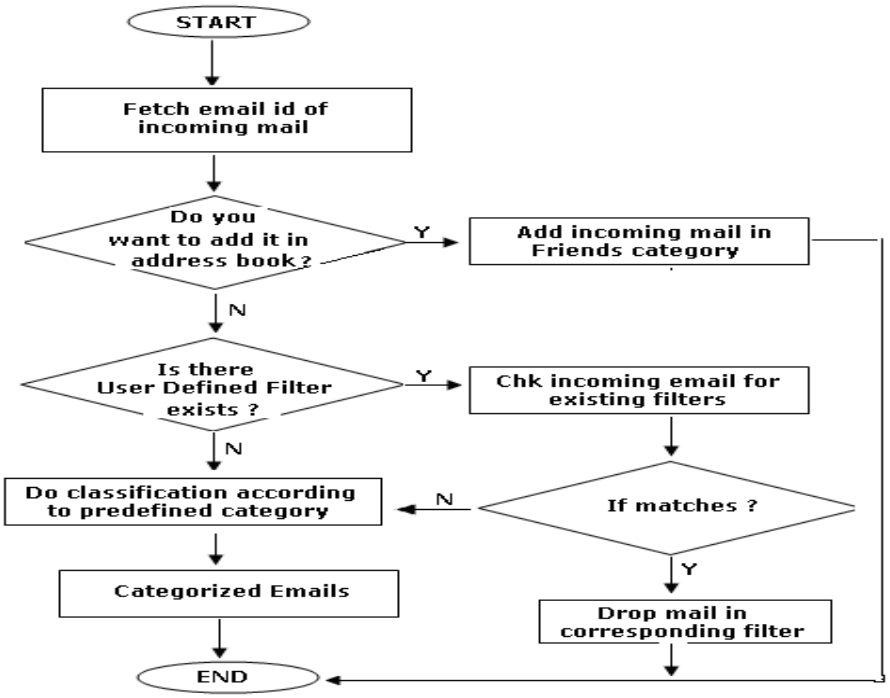


Figure 8: Flow Chart showing Classification and Filtration Process

### 5.4.5 Processing Steps

- **Check for availability of email client**

Before getting start with this desktop application, user must be sure that there must be a mail client installed on that corresponding desktop system, which will allow user to read emails. An example of such email client includes Microsoft's Outlook Express, Mozilla's Thunderbird, Mail Washer etc.

- **Set the properties of account**

This is the very first module of application which will prompt for the user account information like User Id, password, Server name, Port number and protocol to be used. After filling all the entries, *Connect* button is to be pressed, to bring the mails on client side from mail server, as in Figure 9.

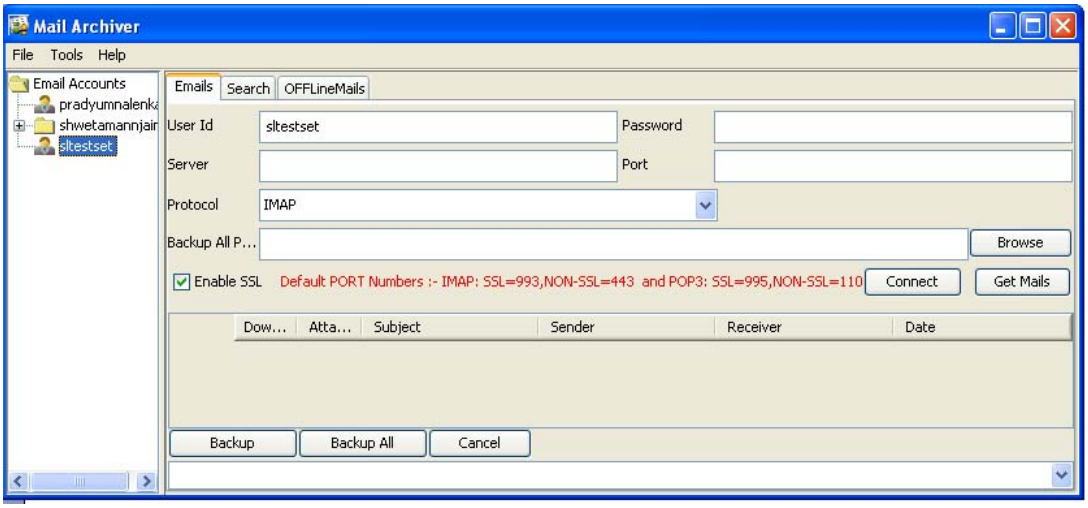


Figure 9: Screenshot showing the settings of an user account

- After getting connect with mail server, Get Mails button is to be pressed in order to see the list of emails as listed in Figure 10:.

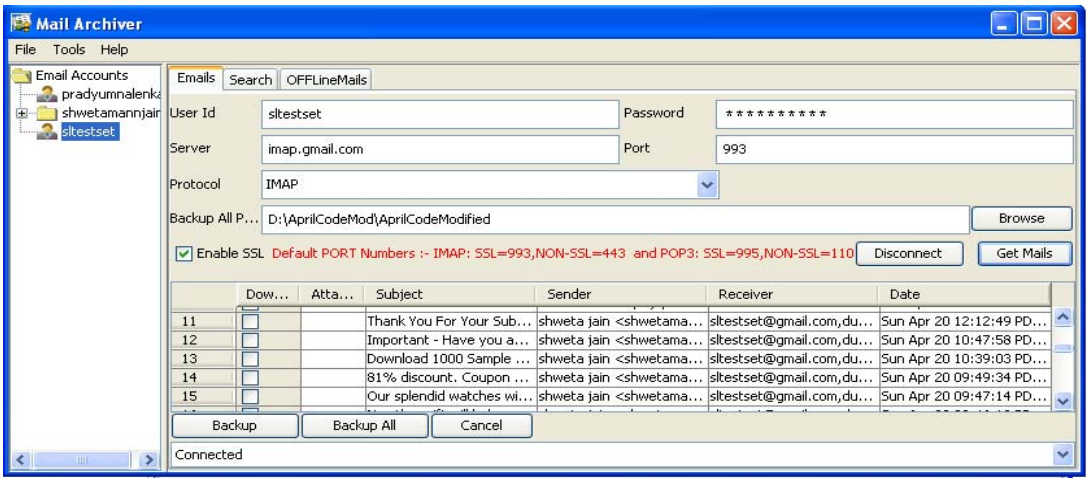


Figure 10: Screenshot showing the list of emails coming from mail server

- Now next step is to create the Address Book. This will be created as and when emails are downloading. When the mail is coming first time on this client machine, then message box will appear , prompting user , whether he/she is willing to add this particular contact in address book. If user accepts it, then that email id will be store in address book, and will be never asked in future. But if user denied, then application will prompt again next time, when that mail will arrive. So, in this way Friends List will be created on behalf of this address book.
- Next good feature of this client application is to take backup of emails stored on mail server. Now, if user wants to download emails at his client machine for backup purpose, he/she need to check the checkbox given in front of all emails and then need to click on Backup button. If user is willing to download all his emails then first he/she need to give the path in the space provided and then click on Backup All button, without checking the checkboxes for each and every emails individually.
- Up till now, user will be able to see the mails which are filtered as SPAM and INBOX. These two folders can be viewed by right clicking on account information in left panel. The option given there for this purpose is *REFRESH.* If users want to see these categories during mails are downloading then he/she need to periodically refresh it.

Graphical view of INBOX and SPAM folders can be seen in Figure 11 in the left panel of the tree given.

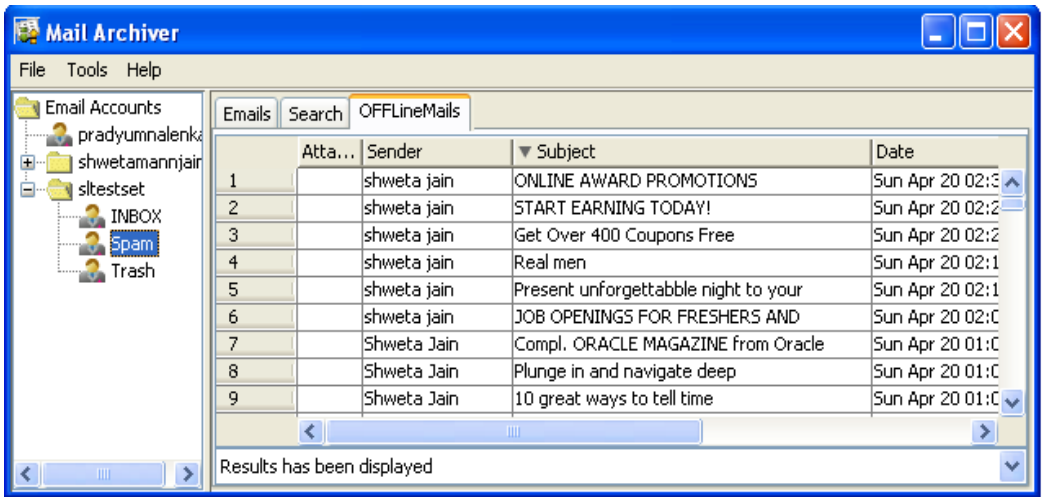


Figure 11: Screenshot showing Filtration Process

- Till step 6, *Filtration* process is complete. Now the thing remain is the *Classification* process, means the *categorization* of the emails. Now, to view the pre-defined categories, user need to click on *SHOW MAILS* option, which is also given in the same pop-up in which refresh option was given. When user will click on show mails, new window will open, which will show the various categories in tree structure. Here, user can read those categorized emails by clicking on *open* option, which is given on right click of each and every email. The new window for categorization is shown in Figure 12:

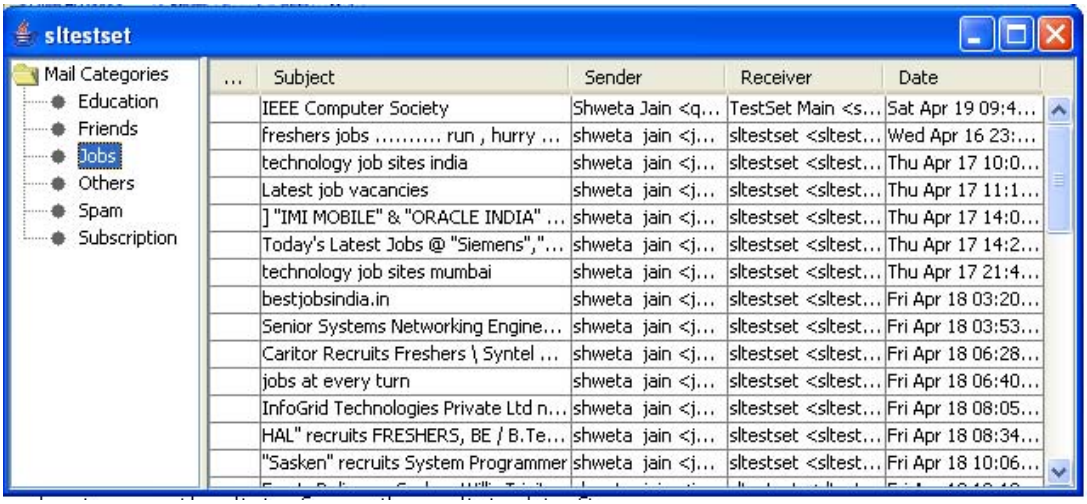


Figure 12: Screenshot showing Classification Process

By going through above 7 steps, user can easily complete both the Filtration and Classification process on emails. But, problem might occur if any user's mails are not fitting into any of the predefined categories. So, to overcome from this problem, facility of *User Defined Classifier* is also provided to user, which will facilitate him/her to define his/her own categories on basis of the keywords and conditions supplied by him. These user defined categories can be created from Tools Menu, Add Filter and the dialog box for this new category can be seen in Figure 13 :

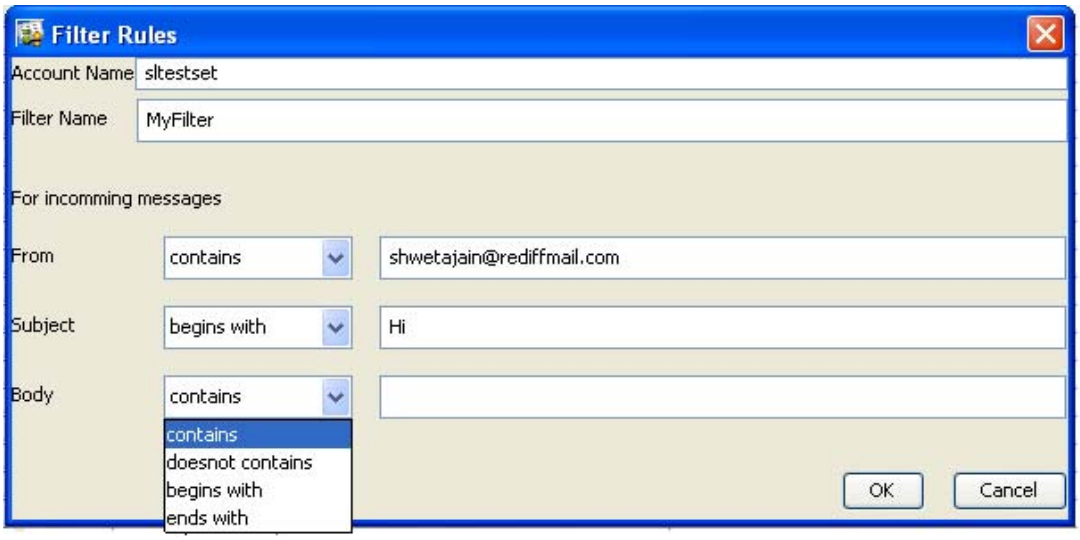


Figure 13: Screenshot showing User Defined Filter

## 5.5 DECISION PARAMETER

### Definition of threshold

Up to now, performance had been investigated in the case where a filtration and classification decision was enforced for each email. An alternative path is to define a confidence threshold *t* for similarity match parameter θ, such that an email *e* is classified under category *c* if and only if the probability of *e* belonging to *c* exceeds t. If the probability is below t, the email may either be classified manually or put into a pre-defined default category.

The threshold can be derived in two different ways: analytically (i.e. based on a theoretical method indicating how to compute the best threshold value) or experimentally. Experimental policy was taken and tested different values for t in order to find out which of them would maximize

effectiveness. Figure 14 shows the results for threshold values of 0.01, 0.03, 0.05, 0.35 and 0.50.

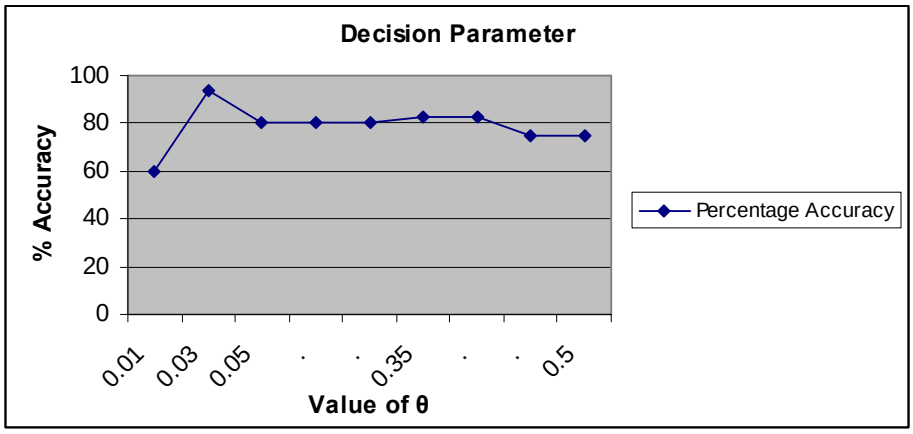


Figure 14: Graph reflecting different values of θ

Evaluating the different threshold values as to accuracy and percentage of classifiable emails, there seems to be no clear winner. 30% may be a good choice because it brings a considerable improvement in accuracy.

### 5.6 RESULTS

As a result it turned out that how many mails are spams and how many are legitimates among such a large collection.

(1) The graph drawn in Figure 15 shows the number of emails and the percentage accuracy of result obtained.

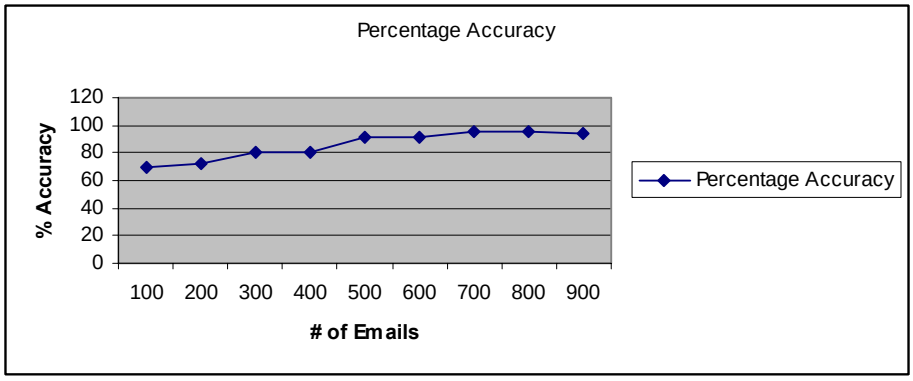


Figure 15: Graph between number of emails and % accuracy earned

**Table 6: Statistics of Spam Filtering Corpora**

| Corpus | # of emails | # of Spam Emails | # of Legitimate emails |
|---|---|---|---|
| **User 1** | 100 | 80 | 20 |
| **User 2** | 400 | 80 | 320 |
| **User 3** | 50 | 4 | 46 |
| **User 4** | 150 | 40 | 110 |
| **News Group** | 200 | 190 | 10 |

**(2)** The graph plot in Figure 16 shows the various categories like Job, Spam, Education, Subscription and Others with respect to all five users named User 1, User 2, User 3, User 4 and News Group.

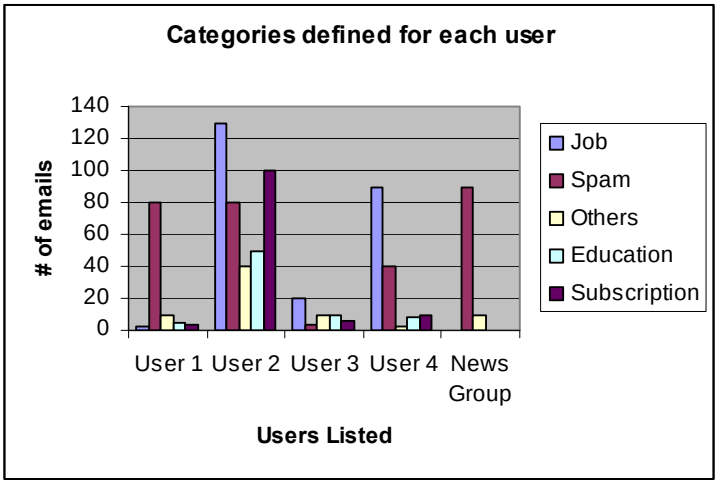


Figure 16: Graph showing users and corresponding categories

**Table 7: Users with their corresponding email categories**

| Users | Job | Spam | Others | Education | Subscription |
|---|---|---|---|---|---|
| **User 1** | 2 | 80 | 10 | 5 | 3 |
| **User 2** | 130 | 80 | 40 | 50 | 100 |
| **User 3** | 20 | 4 | 10 | 10 | 6 |

| | | | | | |
|---|---|---|---|---|---|
| **User 4** | 90 | 40 | 2 | 8 | 10 |
| **News Group** | - | 190 | 10 | - | - |

**(3)** The graph drawn in Figure 17, gives the clear cut view of mail filtration process, which defines Spam and Useful emails.

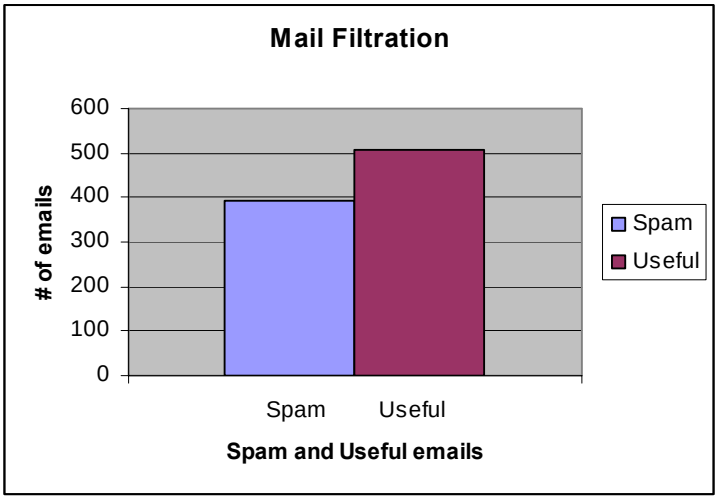


Figure 17: Graph for Spam and Useful emails

**Table 8: Filtration process reflecting Spam and Useful emails**

| **Categories** | **# of emails** |
|---|---|
| **Spam** | 394 |
| **Useful** | 506 |

**(4)** Figure 18, throws the light on various categories derived from whole email corpus named sltestset.

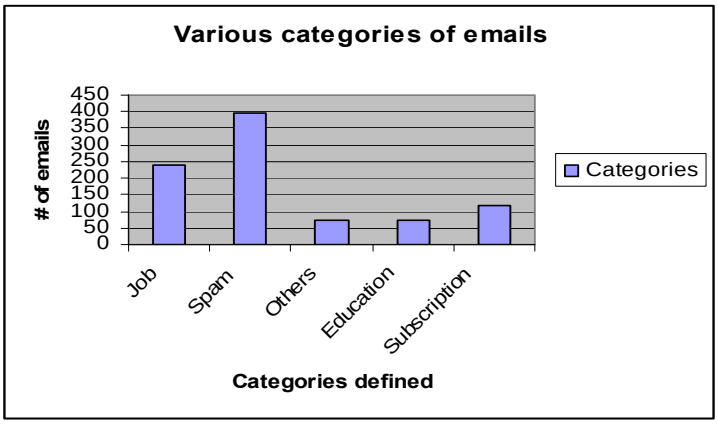


Figure 18: Graph of categories defined from whole corpus

**Table 9: Number of emails in corresponding categories**

| **Categories** | **# of emails** |
|---|---|
| Job | 242 |
| Spam | 394 |
| Others | 72 |
| Education | 73 |
| Subscription | 119 |

## 5.5 SUMMARY

To summarize the result of the thesis it can be said, that criteria of an email classifier makes sense and has its ground. Although thesis deals with concrete test examples of email corpus, the above described methodology can be applied to both the filtration and classification processes. Analysis of Spam and Ham mails provided a good basis for the creation of this classifier.

# 6 CONCLUSION AND FUTURE WORK

## 6.1 CONCLUSION

Dissertation was started by recognizing the three steps. The first one is downloading the emails from corresponding mail servers, next is filtration process and the last one is categorization of emails into pre-defined categories. Vector Space Model is implemented in its bit modified form. The algorithm implements the following steps: A>Whole email is tokenized. B> Frequencies are calculated for vector formation. C> Criteria for similarity match is taken into consideration, based on the strength of capabilities of spam/legitimate emails. This predictive strength is based on the frequency of occurrence in spam/legitimate collections as well as on heuristic rules. D> During filtration, spam and ham (legitimate) evidence is obtained. E> During classification, ham emails are further categorize into some of the predefined categories and further user have ability to create his/her own filter as per his interest.

Config files are defined for listing the rules of email categorization. We empirically determined the best weights (frequencies) to be assigned to features for all categories. We also defined a threshold parameter to trade - off between the spam and ham emails.

The algorithm is implemented on Windows but can work well with other platforms also because JAVA is used here for achieving platform independency.

## 6.2 FUTURE WORK

Lots of work had been done for the desire of achieving best results. On the basis of plain/text results are almost accurate but as we know that emails contains many more things besides this plain/text. The other email parts are HTML links, images, attachments etc. So, for getting more accurate results,

all these factors should also be taken into consideration and all this will lead to further enhancements as a future work.

Further improvements to the algorithm include:

1. Reading and using images during filtration and classification
2. Attachments are also required to be taken into consideration for making decisions
3. Identification of more heuristic rules to classify emails for more higher accuracy

# REFERENCES

Literature from various books and research papers has been carried out to support present work. The name of referred book, papers and site references are summarized as below.

**Papers**


[1] Paul A. Henry, “Spam Defence”, Science Direct

[2] Minoru Sasaki and Hiroyuki Sasaki, “Spam Detection using Text Clustering”, IEEE CNF

[3] Evan P. Greenberg and David R. Cheriton, “Enforcing Bulk Mail Classification”

[4] Ron Bekkerman and Andrew McCallum, “Automatic Categorization into Folders”

[5] A plan for spam by Paul, http://www.paulgraham.com/spam .html

[6] Shouqiang Liu and Deyu Qi, “Study on Spam Filtering System”, IEEE CNF

[7] What is spam, and how do I tell whether a message is spam or not?, http://www.clearswift.com/support/technotes/item.aspx?ID=1561

[8] Phaedra Hise, “Mom Spam: The Cyber-Scourge of Families Everywhere,” Salon, 20 December 1999, available from http://www.salon.com/mwt/feature/1999/12/20/spam/

[9] U.S. Federal Trade Commission, “FTC Names Its Dirty Dozens: 12 Scams Most Likely to Arrive via Bulk Email,” FTC Consumer Alert (July 1998), available from http://www.ftc.gov/bcp/conline/pubs/alerts/doznalrt.htm

[10] Deepak P and Sandeep Parameshwaran, “Spam filtering using spammail communities”, IEEE CNF, 2005 Symposium on 31 Jan.-4 Feb. 2005 Page(s):377 – 383

[11] O. Nouali and P. Blache, “A semantic vector space and features-based approach for automatic information filtering”, Science direct

[12] Art B. Owen, ”Information Retrieval, and the Vector Space Model”, Proceedings of the 2006 Australasian workshops on Grid computing and e-research - Volume 54 ACSW Frontiers '06

[13] DIK L. LEE,”Document Ranking and the Vector-Space Model”, IEEE CNF

[14] Wenlei Mao and Wesley W. Chu, “Free Text Medical Document Retrieval via Phrase based Vectore Space Model”, Computer Scrienc Department, University of California, Los Angeles

[15] Art B. Owen, “Information Retrieval and Vector Space Model”, www- stat.stanford.edu/~owen/courses/399/ir4up.pdf

[16] William W. Cohen, “Learning rules that Classify Emails”, www.cs.cmu.edu/~wcohen/postscript/aaai-ss-96.ps

**Books**

- Ending Spam - Bayesian Content Filtering and the Art of Statistical Language Classification by Jonathan A. Zdziarski

- Java Complete Reference by Herbert Schildt