Islamic University of Gaza

Deanery of Higher Studies

Faculty of Engineering

Computer Engineering Program

Improved Spam Detection using DBSCAN and Advanced digest algorithm

كشف البريد العشوائي باستخدام خوارزميتي تقسيم المجموعات بناء على الكثافة والموجز المطور

Prepared By:
Alaa H. Ahmed

Supervised By:
Prof. Mohammad Mikki

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master in Computer Engineering

1434 هـ - 2013م

بسم الله الرحمن الرحيم

الجامعة الإسلامية – غزة
The Islamic University - Gaza

الجامعة الإسلامية غزة
The Islamic University - Gaza

عمادة الدراسات العليا　　　　هاتف داخلي: 1150

الرقم ج س غ/35/ Ref
التاريخ 2013/06/19 Date

نتيجة الحكم على أطروحة ماجستير

بناءً على موافقة عمادة الدراسات العليا بالجامعة الإسلامية بغزة على تشكيل لجنة الحكم على أطروحة الباحث/ علاء حسن حسني أحمد لنيل درجة الماجستير في كلية *الهندسة* قسم هندسة الحاسوب وموضوعها:

كشف البريد العشوائي باستخدام خوارزميتي تقسيم المجموعات بناء على الكثافة والموجز المطور

Improved Spam Detection using DBSCAN and Advanced digest algorithm

وبعد المناقشة التي تمت اليوم الأربعاء 10 شعبان 1434هـ، الموافق 2013/06/19م الساعة الثانية عشرة ظهراً، اجتمعت لجنة الحكم على الأطروحة والمكونة من:

أ.د. محمد أمين مكي	مشرفاً ورئيساً	....
أ.د. حاتم محمود حماد	مناقشاً داخلياً	....
أ.د. سامي سليم أبو ناصر	مناقشاً خارجيًا	....

وبعد المداولة أوصت اللجنة بمنح الباحث درجة الماجستير في كلية *الهندسة*/ قسم هندسة الحاسوب.

واللجنة إذ تمنحه هذه الدرجة فإنها توصيه بتقوى الله ولزوم طاعته وأن يسخر علمه في خدمة دينه ووطنه.

والله ولي التوفيق ،،،

الجامعة الإسلامية غزة
The Islamic University
عمادة الدراسات العليا

عميد الدراسات العليا

أ.د. فؤاد علي العاجز

Abstract

Mail is one of the most popular and frequently used ways of communication due to its worldwide accessibility, relatively fast message transfer, and low sending cost. Nowadays, detecting and filtering are still the most feasible ways of fighting spam mails. There are many reasonably successful spam mail filters in operation. The identification of spam plays an important role in current anti-spam mechanism.

For improving the accuracy of spam detection, we present an improved Filtering technique which is based on the Improved Digest algorithm and DBSCAN clustering algorithm.

Using this technique, mails are represented using improved digest algorithm and then clustered using DBSCAN clustering algorithm. All similar mails which always categorized as spam are identified and clustered together where good mails that don’t look similar like other mails are not clustered. This method greatly improves the filtering accuracy against latest proposed algorithms by 30 % and improves the resistance of spam detection against increased obfuscation effort by spammers, while keeping miss detection of good mails at a similar level of older filtering methods.

ملخص

يعتبر البريد الإلكتروني أحد أهم وسائل الاتصالات في العالم وذلك لسهولة الإستخدام والمرونة وسرعة نقل البيانات اضافة الى انخفاض التكلفة ، الا ان هناك بعض المعيقات والمشاكل التي قد تواجه مستخدمي البريد ولعل أهملها هو البريد العشوائي المزعج الذي يهدف الي نشر دعاية أو اختراق حسابات المستخدمين .

وقد تعددت الطرق لتصفية هذا النوع من البريد ومنع وصوله للمستخدمين و تعتبر مرحلة تحديد البريد على انه عشوائي المرحلة الأهم وللتعرف على البريد العشوائي قمنا بتقديم طريقة جديدة لتحسين دقة التصفية وكشف البريد العشوائي من خلال استخدام خوارزميتي تقسيم المجموعات بناء على الكثافة والموجز المطور ، وتعتمد هذه الطريقة على تمثيل محتوى البريد من خلال خورازمية الموجز المطو ر والعمل على تصفية وتحديد ماهو عشوائي من خلال خوارزمية تقسيم المجموعات اعتماداً على الكثافة ، وبالتالي فإن البريد العشوائي المتشابه سيتم تجميعه في مجموعات يتم التعرف عليها بينما البريد الصحيح لايتم تصنيف ه داخل المجموعات وبالتالي لاتتم تصفيته ويصل للمستخدم .

وقد أثبت الطريقة المقدمة قدرتها على تحسين دقة الكشف و تصفية البريد العشوائي بنسبة 30% على الطرق المقدمة حديثاً في نفس المجال فيما ابقت على نسبة الخطأ في تصفية أي بريد صحيح مقارنة مع الطرق الحديثة .

Acknowledgement

I would like to express my deepest gratitude for my parents who have always been there to support me. I also thank my wife who has been strongly supportive to me to the end of this thesis.

I also greatly thankful to my supervisor, Prof. Mohammad Mikki, whose encouragement, guidance and support from the initial to the final level enabled me to develop a deep and thorough understanding of the subject. Finally, I offer my regards and blessings to all of those who supported me in any respect during the completion of the thesis.

Alaa H. Ahmed

Table of Content

List of Figures

List of Tables

This page left intentionally blank

Chapter 1 - Introduction

1.1 Spam Mail introduction

The use of internet has been extensively increasing over the past decade and it continues to be on the ascent. Hence the Internet is gradually becoming an important part of everyday life. Internet usage is expected to continue growing and mail has become a powerful tool intended for idea and information exchange. Negligible time delay during transmission, security of the data being transferred, low costs are few of the multifarious advantages that mail enjoys over other physical methods. However there are few issues that spoil the efficient usage of mails. Spam mail is one among them [1].

The term spam is used to describe any “unwanted” thing. Mail spam is a set of unwanted electronic spam mail that contains nearly identical messages sent to huge number of recipients. Spam mail can be not only annoying but also dangerous to recipients. Clicking on links contained in spam mails may send users to phishing and malware .It also may include malware as scripts or other risky executable file attachments. The problem of spam or Unsolicited Bulk Mail (UBE) is becoming a pressing issue [2].

Spammers collect mail addresses from chartrooms, websites, customer lists, newsgroups, and viruses which harvest users address books, and are sold to other spammers. They also use a practice known as "mail appending" or "e-pending" in which they use known information about their target (such as a postal address) to search for the target's mail address. Much of spam is sent to invalid mail addresses. The percentage of spam in mail traffic average is 77.2%.[3]

With the development of internet, the problem of the spam is getting more and more serious. A large number of spam mails spread over the internet, which not only increases the burden of the network, but also brings a lot of inconvenience to users. Therefore, the researching of anti spam technology has become one of the key internet researches.

Spam mail characterized by three main features:

- **Anonymity:** The address and identity of the sender are concealed
- **Mass Mailing:** The mail is sent to large groups of people
- **Unsolicited:** The mail is not requested by the recipients

Despite the Varity of anti-spam applications, spam remains a constant disturbance source. At a minimum, spam can interrupt your busy days, forcing you to spend time opening and deleting mails. In a more serious scenario, spam could unleash a risky virus on your organization's network, halting your servers and desktop machines.

Experts and anti-spam services tend to peg the rate of spam at anywhere from 50 to 90 percent of all mails on the Internet. Although preventing spammers from sending junk mail may never be possible, installing an anti-spam application on your organization's mail server or individual computers can vastly reduce the amount of spam that your staff has to deal with. Anti-spam applications typically use one or more filtering methods to identify spam and stop it from reaching a user's inbox.

1.2 Spam Filter methods

For instance, some spam-filtering methods run a series of checks on each message to determine the likelihood that it is spam. Other spam-filtering techniques simply block all mail transmissions from known spammers or only allow mail from certain senders. And while some spam-filtering methods are completely transparent to both the sender and recipient, others require some degree of user interaction.

While no effective and complete solution to the spam problem is currently available, several moderately successful anti-spam techniques have been proposed, each operating along a different line. Here we present a shortlist of desired filtering techniques:

1.2.1 List-Based Filters

List-based filters attempt to stop spam by categorizing senders as spammers or trusted users, and blocking or allowing their messages accordingly.

Blacklist: This popular spam filtering method attempts to stop unwanted mail by blocking messages from a preset list of senders that you or your organization’s system administrator creates. Blacklists are records of mail addresses or Internet Protocol (IP) addresses that have been previously used to send spam. When an incoming message arrives, the spam filter checks to see if its IP or mail address is on the blacklist; if so, the message is considered spam and rejected.

Though blacklists ensure that known spammers cannot reach users' inboxes, they can also misidentify legitimate senders as spammers. These so-called false positives can result if a spammer happens to be sending junk mail from an IP address that is also used by legitimate mail users. Also, since many clever spammers routinely switch IP addresses and mail addresses to cover their tracks, a blacklist may not immediately catch the newest outbreaks.

Real-Time Blackhole List: This spam filtering method works almost identically to a traditional blacklist but requires less hands on maintenance. That’s because most real time blackhole lists are maintained by third parties, who take the time to build comprehensive blacklists on the behalf of their subscribers. Your filter simply has to connect to the third party system each time a mail comes in, to compare the sender’s IP address against the list.

Since blackhole lists are large and frequently maintained, organization's IT staff won't have to spend time manually adding new IP addresses to the list, increasing the chances that the filter will catch the newest junk mail outbreaks. But like blacklists, real time blackhole lists can also generate false positives if spammers happen to use a legitimate IP address as a conduit for junk mail. Also, since the list is likely to be maintained by a third party, you have less control over what addresses are on or not on the list.

White list: A white list blocks spam using a system almost exactly opposite to that of a blacklist. Rather than letting you specify which senders to block mail from, a white list lets you specify which senders to allow mail from; these addresses are placed on a trusted users list. Most spam filters let you use a white list in addition to another spam fighting feature as a way to cut down on the number of legitimate messages that accidentally get flagged as spam. However, using a very strict filter that only uses a white list would mean that anyone who was not approved would automatically be blocked.

Some anti-spam applications use a variation of this system known as an automatic white list. In this system, an unknown sender's mail address is checked against a database; if they have no history of spamming, their message is sent to the recipient's inbox and they are added to the white list.

Greylist: A relatively new spam filtering technique, greylists take advantage of the fact that many spammers only attempt to send a batch of junk mail once. Under the greylist system, the receiving mail server initially rejects messages from unknown users and sends a failure message to the originating server. If the mail server attempts to send the message a second time a step most legitimate servers will take , the greylist assumes the message is not spam and lets it proceed to the recipient's inbox. At this point, the greylist filter will add the recipient's mail or IP address to a list of allowed senders.

Though greylist filters require fewer system resources than some other types of spam filters, they also may delay mail delivery, which could be inconvenient when you are expecting time sensitive messages.

1.2.2 Content-Based Filters

Rather than enforcing across the board policies for all messages from a particular mail or IP address, content-based filters evaluate words or phrases found in each individual message to determine whether a mail is spam or legitimate.

Word-Based Filters: A word-based spam filter is the simplest type of content-based filter. Generally speaking, word-based filters simply block any mail that contains certain terms.

Since many spam messages contain terms not often found in personal or business communications, word filters can be a simple yet capable technique for fighting junk mail. However, if configured to block messages containing more common words, these types of filters may generate false positives. For instance, if the filter has been set to stop all messages containing the word "discount," mails from legitimate senders offering your nonprofit hardware or software at a reduced price may not reach their destination. Also note that since spammers often purposefully misspell keywords in order to evade word-based filters, your IT staff will need to make time to routinely update the filter's list of blocked words.

Heuristic Filters: Heuristic (or rule-based) filters take things a step beyond simple word-based filters. Rather than blocking messages that contain a suspicious word, heuristic filters take multiple terms found in a mail into consideration.

Heuristic filters scan the contents of incoming mails and assigning points to words or phrases. Suspicious words that are commonly found in spam messages, such as "Rolex" or "Viagra," receive higher points, while terms frequently found in normal mails receive lower scores. The filter then adds up all the points and calculates a total score. If the message receives a certain score or higher (determined by the anti-spam application's administrator), the filter identifies it as spam and blocks it. Messages that score lower than the target number are delivered to the user.

Heuristic filters work fast minimizing mail delay and are quite effective as soon as they have been installed and configured. However, heuristic filters configured to be aggressive may generate false positives if a legitimate contact happens to send an mail containing a certain combination of words. Similarly, some savvy spammers might learn which words to avoid including, thereby fooling the heuristic filter into believing they are benign senders.

Rule-based filters, like Spam Assassin [5], assign a spam filter a score to each message based on whether the message contains features typical of spam, such as keywords, URLs or, in the case of HTML messages, background colors

Bayesian Filters: Bayesian filters employ the laws of mathematical probability to determine which messages are legitimate and which are spam. In order for a Bayesian filter to effectively block spam, the end user must initially "train" it by manually flagging each message as either junk or legitimate. Over time, the filter takes words and phrases found in legitimate mails and adds them to a list; it does the same with terms found in spam.

To determine which incoming messages are classified as spam, the Bayesian filter scans the contents of the mail and then compares the text against its two-word lists to calculate the probability that the message is spam. For instance, if the word "valium" has appeared 62 times in spam messages list but only three times in legitimate mails, there is a 95 percent chance that an incoming mail containing the word "valium" is junk.

Because a Bayesian filter is constantly building its word list based on the messages that an individual user receives, it theoretically becomes more effective the longer it's used. However, since this method does require a training period before it starts working well, you will need to exercise patience and will probably have to manually delete a few junk messages, at least at first.

Bayesian tools, like SpamProbe [6], assign a frequency based probability to tokenized words (or pairs/triples) as spam indicators based on previous experiences

1.2.3 Collaborative Content Filtering

An important feature of spam, which can be exploited for detecting it easier, is its bulkiness. A spam bulk mailing consists of many copies of the same original spam message, each sent to a different recipient or group of recipients. The different copies from the same bulk are usually obfuscated, i.e. modified a bit in order to look different from each other.

Spammers apply obfuscation in order to make collaborative spam detection more difficult. Indeed, in collaborative spam detection it is important to have a good technique for determining which mails belong to the same bulk. This allows, after observing an initial portion of a bulk, for the bulkiness scores to be assigned to the remaining mails from the same bulk. If the collaborative spam detection is based purely on the evaluation of bulkiness, each recipient must be equipped with white lists of all the bulky sources from which she or he wants to receive mails.

Having a good technique for determining which mails belong to the same bulk also allows for the individual evidence of spamminess to be joined, if such evidence is generated by collaborating filters or users for some of the mails from an initial portion of the bulk. The observed bulkiness and the estimated spamminess of a bulk can then be used to better filter the remaining mails from the same bulk. Collecting and using the evidence of spamminess is especially useful if the reputation of spam reporters is evaluated and used, in which case the collaborative detection may be relatively safe to use even if the recipients are not equipped with white lists of the bulky sources from which they want to receive mails.

Chapter 2 – Background

2.1 Introduction

The most important step for collaborative spam techniques is the operation of comparing mails to measure the similarity between them. Spam mail bulk contains similar mails. Spammers always try to make some modification to make new spam mail looks different. This modification methods includes add extra text, replace some words with its synonyms, add html code and make multiple copy of the same mail string. To compare two mails we need to represent each mail in a way so we can compare it easily. Digest is one of the used methods to represent a specific text in a specific number of bits. The clustering of mails is done by two steps: getting the digests of mails and clustering the digests. In this chapter, we will introduce some digest algorithms used in anti-spam field and the DBSCAN clustering algorithm of similar data.

2.2 Digest Algorithms

There are many digest algorithms in the anti-spam field. In the distributed anti-spam mechanism DCC [7] (Distributed Checksum Clearinghouse), there are two digests Dig l and Dig 2 for each mail. Dig l is the MD5 value of the mail body after removing the simple characters such as comma and semicolon, etc. Dig 2 is the MD5 value of the words set which is composed of special words in the mail. Using the MD5 algorithm can ensure different mails to have different digests, but it can't do well with the usual spam attack strategy. For Dig l, if the spam attacker adds some additional information in the mail, the Dig l will be entirely different. For Dig2, if the spam attacker exchanges the positions of some sentences in the mail, the Dig 2 will be entirely different. So the digest algorithms in the DCC mechanism aren't strong enough to be used in anti-spam field.

The CTPH – Context Triggered Piecewise Hashing [8] is a text digest algorithm which is based on fragments hash. This algorithm divides the text into fragments first, and then calculates the hash values of all the fragments, finally gets a character string composed of the hash value as the digest. CTPH determines the similarity of the two texts by computing the edit distance of the digests. The CTPH algorithm can identify the similar texts accurately with editing differences, so it has been widely used in computer forensic and anti-spam field. However, this algorithm doesn't do well with the usual spam attack strategy neither. Adding special characters after some sentences can make CTPH digests of similar mails completely different.

The Nilsimsa Algorithm proposed by E. Damiani[9] and DHTnil [10]. Nilsimsa is a simple local sensitive hash function; A local sensitive hash function is a function producing similar digests for similar documents. Nilsimsa takes a document or a text string as input and provides as output a 32-byte code representing the distribution of the trigrams in the text.

Nilsimsa operates by using a window of 5 characters that slides along the text of the message one character at a time as shown in figure 2.1. When a new character enters the window, the algorithm generates the trigrams associated with the window and passes each of them to a hash function $h\,(\,)$. The hash function $h\,(\,)$ computes a value $i = h(trigram)$ between 0 and 255 that corresponds to the $i - th$ counter in an array of integers of size 255, called accumulator, and whose value is increased by 1. After the text analysis, the accumulator will present in the $i - th$ cell the number of trigrams that have been found in the text producing i by the application of the hash function.

The relative frequency of each bucket is compared with the average bucket frequency observed for a large collection of messages (typically, all the messages received by the user) and the value representing this ratio is associated with the bucket. Then, the ratio of each bucket is considered and if the $i - th$ ratio is greater than the median, $the\ i - th$ bit of the Nilsimsa code is set to 1; it is set to 0, otherwise. In this way a 32-byte code is produced.

The original version of Nilsimsa used a simple method to compute the bits in the digest, comparing the cardinality of the trigrams of each bucket with the average for all the buckets. This technique however, is not robust enough against aimed addition therefore, to increase robustness; they introduced the usage of the median as the term of comparison.

To determine if two messages present the same textual content, their Nilsimsa digests are compared, checking the number of bits in the same position that have the same value. The Nilsimsa Compare Value is the number of bits that are equal in two digests minus 128. For two randomly chosen 256-bit sequences, they expect an average of 128 equal bits, that is, a Nilsimsa Compare Value equal to zero. The maximum value of the Nilsimsa Compare Value is 128, for two identical digests. They have proposed the best threshold to compare digests of two mails is 74. Where if the Nilsimsa compare value is larger or equals to 74 then the two mails are similar and belong to the same bulk but if Nilsimsa compare value is less than 74 then the two mails are not similar and don’t belong to the same bulk.

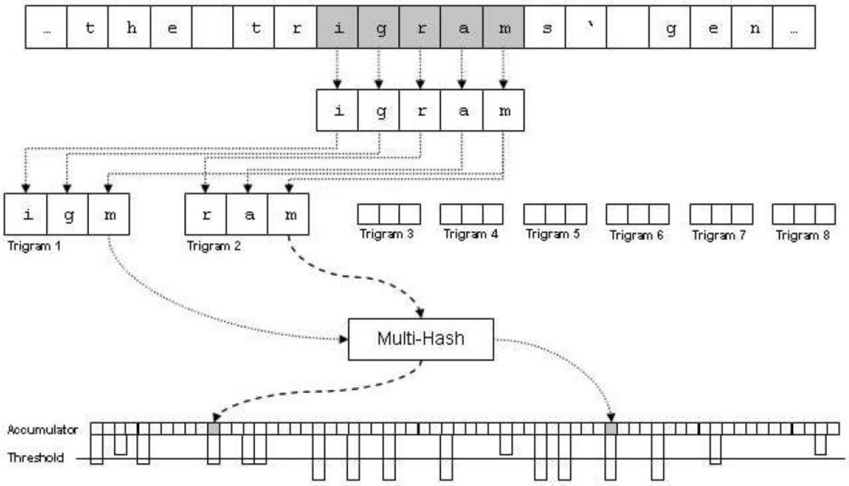

Figure 2.1 Nilsimsa Algorithm

2.3 Clustering Algorithms

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called cluster) are more similar to each other than to those in other groups (clusters)[11].

Cluster analysis has been widely used in numerous applications, including market research, pattern recognition, data analysis, and image processing. In business, clustering can help marketers discover interests of their customers based on purchasing patterns and characterize groups of the customers. In biology, it can be used to derive plant and animal taxonomies, categorize genes with similar functionality, and gain insight into structures inherent in populations. In geology, specialist can employ clustering to identify areas of similar lands; similar houses in a city and etc. data clustering can also be helpful in classifying documents on the Web for information discovery.

In the literature, many clustering algorithms have been proposed. These algorithms differ from each other by the criteria considered which lead to different categories of clustering algorithms. Although it is difficult to find strict categorization of the clustering algorithms because the categories may overlap, the following categorization is helpful to discriminate the clustering algorithms [12]:

Partitioning methods: A partitioning method creates K partitions (or clusters) such that K ≤ n where n is the total number of objects. It creates an initial portioning and then iteratively moves the objects from one cluster to another to improve the partitioning. Good clustering is that the similarity between objects in the same cluster is high whereas the dissimilarity between objects in the different clusters is high. The K-means algorithm is a commonly used partitioning method [13].

Hierarchical methods: A hierarchical method creates a hierarchal structure of the data objects. Then a given number K of clusters determines how to cut the hierarchy. It can be either agglomerative or divisive.

Density-based methods: The idea behind these methods is to group dense objects into clusters. An object is dense if its neighborhood in a given clusters contains at least minimum number of objects. DBSCAN [14] and OPTICS are typical examples of density based clustering.

Grid-based methods: These methods divide the object space into a finite number of cells that form a grid structure. Therewith connected cells are grouped in a cluster.

Model-based methods: This approach creates a mathematical model for each of the clusters and finds the best fit of the data to the given model. A main advantage is that these methods automatically determine the number of clusters based on standard statistics.

Threshold clustering: This method mainly depends on comparison between every two objects in the dataset. A threshold value is used to check if the two objects are similar or not, similar objects are clustered into as single cluster. A cluster to be considered it needs to consist of a threshold number of objects.

To cluster the mail digests, first we need to know the three main features of the digests in the space:

1. Digest of the spam is gathering over digest space
2. Shape of the digest subspace is unknown
3. Digests of regular mails are distributed over digest space.

According to the three main features of the digests mentioned above, we can see that the digest density plays an important role in distinguishing between regular mails and the spam, which is also important in the spam classification. So the density-based clustering algorithm DBSCAN [14] is a good choice.

Chapter 3 - Related Work

3.1 Introduction

In this chapter, we review some of the significant and recent research papers in the field of spam filtering using collaborative content filtering method. We present these activities and discuss their advantages and the disadvantages.

3.2 State of the art

A well known technique for detecting whether mails belong to the same spam bulk is presented and evaluated in the "Open Digest paper" by Damiani et al. [9] (OD stands for Open Digest). OD-paper is often cited in the literature related to digest-based collaborative spam filtering, as it gives very positive results and conclusions on the resistance of the technique to the increased obfuscation effort by spammers. It also shows that the technique is expected to have very low false positives.

The technique produces similar digests out of similar mails, and uses them to find out which mails belong to the same bulk. The digests are produced from the complete mail or from the complete predefined parts of the mail. The digest queries are submitted to a global database, and the replies indicate the number of similar messages (queries) observed by the database.

The open digest technique from the OD paper represents an mail by a 256-bits digest. The transformation is performed using Nilsimsa hashing [15]. This is a locally sensitive hash function, in sense that small changes in the original document may impact only few bits of the digest. That means that similar documents will have similar digests, in sense of a small Hamming distance between them. With the standard hash functions small changes in the original document usually result in a digest that is completely different from the digest of the original document. OD paper gives a detailed description of the Nilsimsa hashing. In summary, a short sliding window is applied through the mail. For each position of the window, the trigrams from the window are identified that consist of the letters from the predefined window positions.

The collected trigrams are transformed, using a standard hash, to the positions between 1 and 256, and the accumulators at the corresponding positions are incremented. Finally, the accumulators are compared to the mean or to the median of all the accumulators, and the bits of the digest are set to 0 or 1, depending on whether the corresponding accumulators are bellow or above the threshold.

The digests are called "open" because: a) the digests computation method is assumed to be publicly known; b) the used similarity hashing hides original mail text, so the privacy of the content is preserved even if the digests are openly exchanged for collaborative filtering.

Open digest algorithm generate a single digest for the whole mail and depend on threshold clustering for check similarity and cluster similar mails so increased spammer mail modification will decrease the accuracy of the algorithm significantly.

The peer to peer system for collaborative spam filtering by Zhou et al. [16] is another well known and often cited digest based anti spam technique. It uses multiple digests per mail, created from the strings of fixed length, and sampled at random mail positions. They apply however the exact matching instead of a similarity matching between the digests, as required by the rest of their system to work. Even modest spam obfuscation is able to alter some of the bits of such generated digests, which prevents their system from detecting spam bulks. Their analysis results in a different conclusion, because they use rather unrealistic obfuscation (which alters the created digests with a very small probability) to test their solution.

M. S. Pera etal, [17] presents a new tool, JunEX, which relies on the content similarity of mails to eradicate junk mails. JunEX compares each incoming mail to a core of mails marked as junk by each individual user to identify unwanted mails while reducing the number of legitimate mails treated as junk, which is critical.

JunEX analyzes the content of an incoming mail and compare it with a previously marked junk mail by the user using word similarity in a word correlation matrix. This content similarity detection approach relies on pre-computed degrees of similarity among words in different documents.

The system proposed by Sarafijanovic and Le Boudec [18] produces multiple digests per mail, from the strings of fixed length, sampled at random mail positions, and it uses similarity matching. Additionally, they extend experiments of open digest paper it uses artificial immune system algorithms to process the digests before and after exchanging them with other collaborating systems, in order to control which digests will be activated and used for filtering of the incoming mails. The system shows good performances in detecting spam bulk under a specific spammer model, but an additional evaluation is needed for more general conclusions about its abilities. As the factorial analysis is missing, it is not clear whether the observed good performances are due to the way the digests are produced (e.g. as compared to the standard digest from the OD paper), or due to the advanced algorithms used by the system.

The direct comparison of the above explained different ways of producing the digests from mails, according to our best knowledge, has not yet been scientifically evaluated.

An improved Nilsimsa digest has been proposed by Sarafijanovic, Perez and Le Boudec [19]. They proposed and evaluated a modified version of the original digest technique. The modified technique uses the same Nilsimsa hashing function, but instead of producing one digest from the complete mail, it produces multiple digests per mail, from the strings of fixed length, sampled at random mail positions. Basically, they only have changed the way of producing the digests from mails. For fairness of the comparison, they have evaluated both techniques while having in mind the same simple detection algorithm that was the basis from the original OD paper experiments.

They performed the experiments for the NCV threshold value 90, as it looked as a more reasonable choice between the two values we used in the single digest experiments.

For mail comparison and clustering they keep the same experiments as in the case with single digest in open digest paper, with the only difference in mail to mail comparison.

They defined, for the considered multiple digest approach, the NCV between two mails to be the maximum NCV over all the pairs of the digests between the two compared mails

They first repeated and extended some of the open digest paper experiments, using the simplest spammer model from that paper. They found that the conclusions of the open digest paper are rather misleading. Then they proposed and evaluated, under the same spammer model, a modified version of the original digest technique. The modified version greatly improves the resistance of spam detection against increased obfuscation effort by spammers, while keeping miss detection of good mails at a similar level.

The Previous clustering techniques of the mails mainly uses the threshold clustering method. This method clusters by scanning every digest and comparing each two digests in the digests set. It can ensure that in the final result, each two digests in the same group are similar. In this method, the threshold is determined by experiment. A larger threshold will reduce the similarity in the group and a smaller threshold will increase the number of groups in the result. By analyzing the result, it is easy to find some similar mails are clustered into several groups. Using this method, the results will be different when the input order is different. Clearly, the reason for the problems above is that the shape of the spam digest subspace is irregular. The DBSCAN algorithm can solve these problems.

Aiming at the problem exists in the spam clustering, Ying, Kai and Jianzhong [20] proposed a new clustering method based on DBSCAN clustering algorithm and Nilsimsa open digest algorithm. Their proposed algorithm first generate a single digest for the whole mails using open digest algorithm then it clustered the generated digests using DBSCAN clustering algorithm. DBSCAN clustering algorithm has increased the accuracy of clustering operation instead of threshold clustering. There proposed algorithm suffers from the shorting of open digest algorithm which mainly depends on single digest for the whole mail.

Moniza and Asha [21] designed an assorted spam detection system. They proposed a system, which possesses hash value to implement an efficient near duplicate matching scheme. Their Spam Detection system can operates in 4 stages , first it generate the mail abstraction using HTML content in mail, and this newly devised abstraction can more effectively capture the near duplicate phenomenon of spams. Then it generates a tree structure named SpTrees, to store large amounts of the mail abstractions of reported spams. SpTrees contribute to the accomplishment of the efficient near duplicate matching with a more sophisticated mail abstraction, after that they calculate a hash value for every subsequence in Sp tree and finally it filter the mails based on near-duplicate matching algorithm.

Chapter 4 – Improved Digest with DBSCAN Filtering Method

4.1 Introduction

In this chapter we will first introduce the current research situation and then explain our proposed algorithm in details.

4.2 Research Situation

At present the latest proposed algorithm for filtering spam using collaborative spam filtering technique by Ying [20] used open digest algorithm with DBSCAN clustering algorithm to achieve the highest accuracy of filtering different received mails.

Older digest clustering papers used only a digest algorithm which represents mails into digest and then starting comparing every pair of mails to cluster all received mails which known as threshold clustering method. This clustering method clusters digests by scanning every digest and comparing each two digests of digests set. It can ensure that in the final result, each two digests in the same group are similar. In this method, the threshold is determined by experiment. A larger threshold will reduce the similarity in the group and a smaller threshold will increase the number of groups in the result. By analyzing the result, it is easy to find some similar mails are clustered into several groups. Using this method, the results will be different when the input order is different. Clearly, the reason for the problems above is that the shape of the spam digest subspace is irregular.

Using DBSCAN has solved problem of threshold clustering algorithm but the use of open digest algorithm cause high spam misdetection against increased obfuscation efforts of spammers.

Our proposed algorithm uses the improved digest algorithm proposed by Sarafijanovic [19] with DBSCAN Clustering algorithm. Modified version of open digest greatly improves the resistance of spam detection against increased obfuscation effort by spammers, while keeping miss detection of good mails at a similar level.

Improved digest algorithm generates multiple digests for a single mail. The generation process include dividing mail to multiple fixed length text, the division operation is done by random. The Nilsimsa compare value has been increased from 74 to 90 because of the high probability of similarity between parts of two mails. This higher value allows higher resistance against increased spammer obfuscation and prevents false positives since the similarity measurement will be between digests of mail divisions not the whole mail.

4.3 Improved digest with DBSCAN Algorithm

Based on the problem exists in the threshold clustering with open digest algorithm, we propose a new clustering method based on DBSCAN clustering algorithm with improved digest algorithm. The new proposed algorithm works in two phases: the first phase is digest generation process which includes the division and representations of mails divisions using improved digest algorithms. The second phase is the clustering of the generated mail divisions digests using DBSCAN clustering algorithm. Figure 4.1 illustrates the main two phases of the proposed algorithm, when a mail received it is first enter the digest generation process where its output are group of digests for each mails, these digests are the input for the second phase which is clustering phase. In clustering phase similar mails is grouped in single spam mails cluster based on similarity between their digests where mails that there digests don't look like any other digest are considered as noise and not clustered. These not clustered mails are regular mail (not spam).

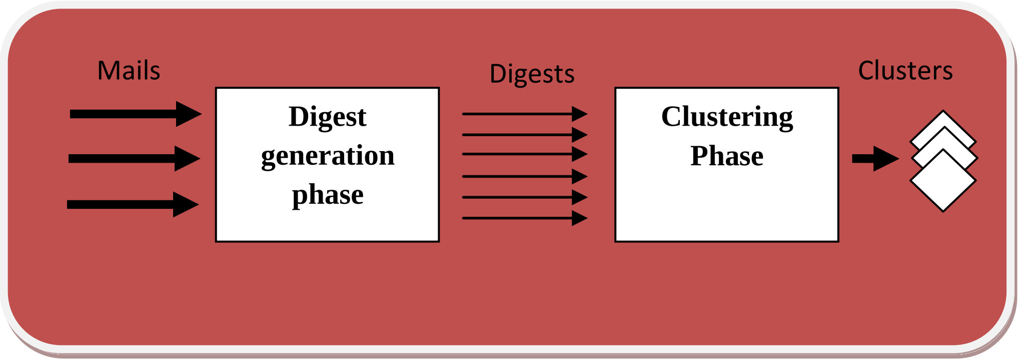

Figure 4.1 Improved digest with DBSCAN Algorithm phases

In this section, we introduce the digest generation process first, and then briefly describe Nilsimsa Digest Space; finally we describe the clustering process using the DBSCAN algorithm and the improved open digest algorithm.

4.3.1 Digest Generation Process

The first phase of the proposed algorithm is the digest generation phase. Digests will represent mails to measure the similarity between them. Similar digests generated from similar mails and a bulk of similar mails will be considered as spam mails. Good or regular mails will always look different that other mails so good mails represented by digests that not similar to other mails digests. When a group of mails is received, multiple digests per mail are generated. The digest generation phase includes the following steps:

1. For each mail; it is first trimmed by removing all spaces including single space and tabs.
2. Mails are divided into random fixed length strings. The length of each division is 60 characters, this value is not optimized because of the variation of the length of mail body. The randomization selection and division of mails make the comparison operation more realistic and make it difficult to the spammers to add specific modification to bypass the spam filter.
3. For each random fixed length string we generate 256 bit digest using Nilsimsa algorithm. The generation process using Nilsimsa digest algorithm has been illustrated in section 2.2 the more similar mail divisions the more similar digest bits.

Digest generation process is shown in figure 4.2 the mails first fetched from the mails pool then it is divided into multiple fixed length strings and finally these fixed length strings are passed to Nilsimsa digest algorithm to get their individual digests. Figure 4.3 shows the pseudo code for digest generation process.

When all mail is fetched, divided and get digest generated for each division; it is delivered to DBSCAN clustering algorithm to be clustered based on similarity.

All similar mails (Spam Bulk) are clustered to a single cluster where good mails are considered as outliers or noise and aren’t included in any spam cluster.

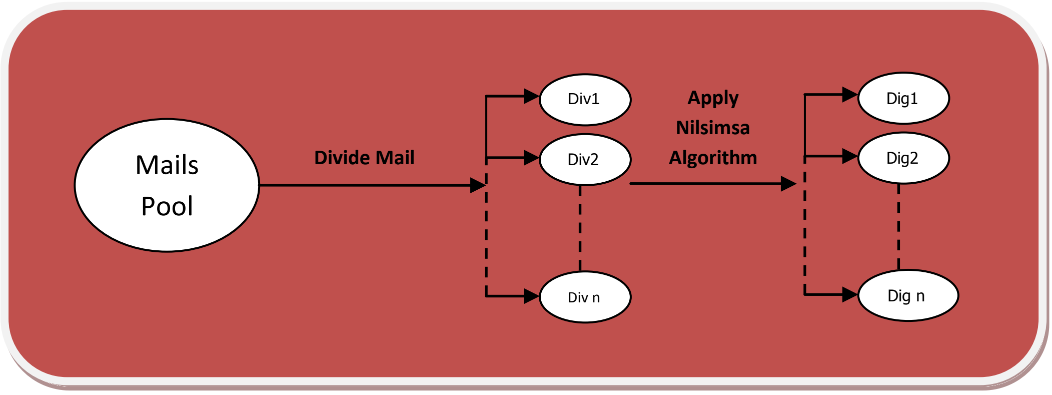

Figure 4.2 Digest Generation Process

```
GetDigest(Mail Pool)
While (more mails found) {
Mail.Fetch()
Mail.Trim()
Divide mail into 60 character fixed length String
      While (more mail division) {
      Return Get Nilsimsa Digest (Mail Division)
      }
                    }
```

Figure 4.3 Digest Generation Process Pseudo code

4.3.2 Nilsimsa Digest Space

To be able to apply a clustering algorithm to any type of data we need to specify data dimension and how to measure similarity or distance. Nilsimsa digest space is a 256-dimensional space, each dimension values 0 or 1. We define the Nilsimsa digest space as ∂ , define the digests of mail as m={s_1, s_2,…., s_n} (***n***= number of divisions per mail , $\boldsymbol{m} \in \partial$, $s_i = \{d_1 \ , d_2 \ , \ldots\ldots , d_{256}\}$, $d \in \{0,1\}$), define the distance between two digests ***m1*** and ***m2*** as follows:

$$\mathbf{distance}\ (\mathbf{s_1}, \mathbf{s_2}) = \sqrt[2]{\sum_{\mathbf{1}}^{\mathbf{256}} (\mathbf{d_{1i}} - \ \mathbf{d_{2i}})^{\mathbf{2}}} \qquad \textbf{(Equation } 1\ )$$

$$\mathbf{distance}\ (\mathbf{m_1}, \mathbf{m_2}) = \left. \sum_1^3 \mathbf{distance}\ (\mathbf{s_x}, \mathbf{s_y}) \middle/ \mathbf{3} \right. \qquad \text{(Equation 2)}$$

Where x , y are the indices of the most three similar divisions between m1 , m2

As shown above the dimension space is specified based on the base that each mail is divided into ***n*** divisions and each division has a 256 bit digest. As shown in equation 1 and equation 2, to compute the distance between two mail divisions digests we need to count the number of different bits at the same location at the two 256 bit digests the digests of divisions of both mails are compared and the average of the distance between most three similar division digests is considered.

Open digest algorithm proposed Nilsimsa Compare Value (NCV). Where NCV between two digests is equal to the number of the equal bits at the same positions in the two digests, minus 128 (for the digests of 256 bits). The higher NCV indicates the higher similarity of the texts from which the digest are computed. The threshold of NCV values proposed to be 74. If NCV is bigger than or equal to 74 then the two mails are similar, else they are different.

In improved open digest space, d_i values only 0 or 1, so the distance is equal to the number of different bits between two division digest. Improved open digest defines that when the distance between two digests is smaller than or equal to 38 (128-90) where 90 is the new NCV Threshold defined by improved open digest, the two digests are similar. Based on threshold clustering method; if any two digests in the group are similar and the number of digests exceeds the threshold, the group can be called a cluster of spam. The ideal distribution of the spam digests in digest space is as shown in Figure 4.4 (a).

However, if the spam digests distribute in irregular shapes as shown in Figure 4.4 (b), using the threshold clustering method may lead to the result that large cluster is divided into several small clusters, and the number of the clusters increases. Using the DBSCAN can cluster such an irregular shape cluster together into a large cluster.

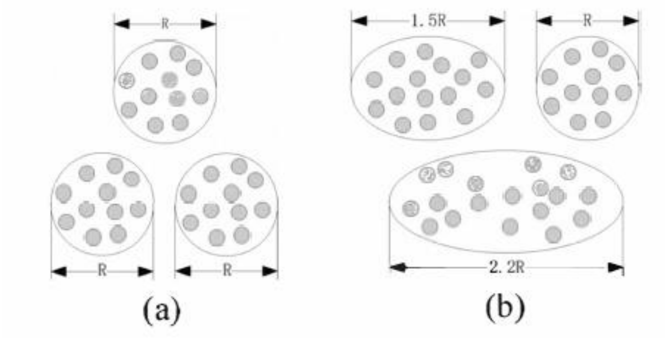

Figure 4.4 distribution of the spam digest in digest space

Similar divisions digests indicates similar mails so the comparison between two mails is held between all both mails digests. We focus on the maximum similar digests of both mails so we have taken the average of the maximum similar three division digests and get the average distance between them. The value of three is not optimized but it provides the best results.

4.3.3 Clustering mails using DBSCAN

DBSCAN [14] is significantly more effective in discovering clusters of arbitrary shape than other well known algorithm.

DBSCAN requires two parameters: ε (eps) and the minimum number of points required to form a cluster (MinPts). It starts with an arbitrary starting point p that has not been visited from the group of points D. This point's ε-neighborhood is retrieved, and if it contains sufficiently many points less than or equal to MinPts it is called a core point and a cluster is started. Otherwise, the point is labeled as noise. Note that p might later be found in a sufficiently sized ε -environment of a different point and hence be made part of a cluster.

If in the range of p's ε radius the number of the elements is less than MiniPts, we can call p as the boundary, p is marked as noise node temporarily. Then, DBSCAN will dispose the next element in set D. As the first and the last step is the same as the threshold clustering method, so the two steps are ignored here. The main workflow of DBSCAN clustering for the proposed algorithm is shown as follows:

Step1: Scan the mail ***p*** in the set ***D*** one by one. Judge whether it has been clustered in a cluster. If so, skip this mail, otherwise turn to Step2. If the scan of all the digests in the set ***D*** is completed, then turn to Step3.

Step2: Get the number of neighbors of ***p*** within the range of ε. This step is done by calculating the distance between ***p*** and all other mails. The calculation of distance between two mails as shown in previous section include the average of the smallest three distances between all of the two mail digests distances. If the number isn't less than MinPts, set the digest ***p*** as the core mail, then scan each of the neighbors of ***p*** and turn to Step l for recursive queries. Finally, all elements from recursive clustering are marked as a new cluster, and then turn to Step l to dispose the next mail of set ***D***.

Step3: Scan all the mails in set ***D***, if a mail isn't in a cluster, it should be marked as a noise mail, and the corresponding mail should be regular.

As shown in Figure 4.4, we set MinPts to 3.There are three mails within A's radius of ε. So it meets the demand, ***A*** can serve as a core. Do the recursion from the three digests. Take digest ***B*** for example, there are single mail within ***B***'s radius of ε. So ***B*** is a boundary point. The recursion stops when the boundary digests doesn't meet the density demand.

As there is no other mail within the radius of mail ***N***, the mail ***N*** is a noise one, this mail is regular. We should note that all of the mail must be queried. In order to show the process clearly, Figure 4.5 only shows the query processes of several mails.

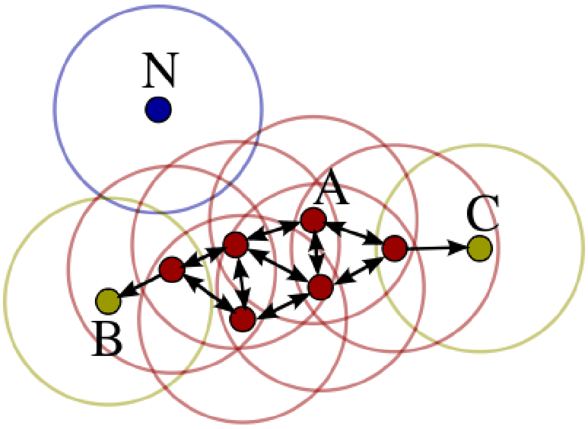

Figure 4.5 Large spam mails clustered correctly

```
DBSCAN(D, eps, MinPts)
   C = 0
   for each unvisited mail P in dataset D
      mark P as visited
      NeighborMails = GetNeighbors(P, eps)
      if sizeof(NeighborMails) < MinPts
         mark P as NOISE
      else
         C = next cluster
         expandCluster(P, NeighborMails, C, eps, MinPts)

expandCluster(P, NeighborMails, C, eps, MinPts)
   add P to cluster C
   for each mail P' in Neighbormails
      if P' is not visited
         mark P' as visited
         NeighborPts' = GetNeighbors (P', eps)
         if sizeof(Neighbormails') >= MinPts
            NeighborMails = NeighborMails joined with NeighborMails'
      if P' is not yet member of any cluster
         add P' to cluster C

GetNeighbors(P, eps)
   for each mail X in dataset D
        GetDistance(P , X)
return all Mails within P's neighborhood where eps> GetDistance(P , X)
(including P)

GetDistance (Mail P , Mail X )
Vector distances
for each digest d1 in Mail P digests

          for each digest g1 in Mail X digests

              distances.add (NilsimsaDistance (d1 , g1 ))

return the Average of the Smallest three distances of distances Vector
```

Figure 4.6 DBSCAN Pseudo code

Using DBSCAN clustering algorithm to cluster the digests of multiple fixed length mail divisions can improve the accuracy of clustering operation better than the threshold clustering since the number of digests for a large number of mails can be considered a high density data which DBSCAN clustering algorithm has designed to deal with.

DBSCAN clustering algorithm categorizes any good mails as noise and they will not be clustered in any of the generated clustered. Figure 4.7 summarizes the clustering method using DBSCAN clustering Algorithm.

The pseudo code shown in figure 4.6 illustrates the steps of DBSCAN algorithm.

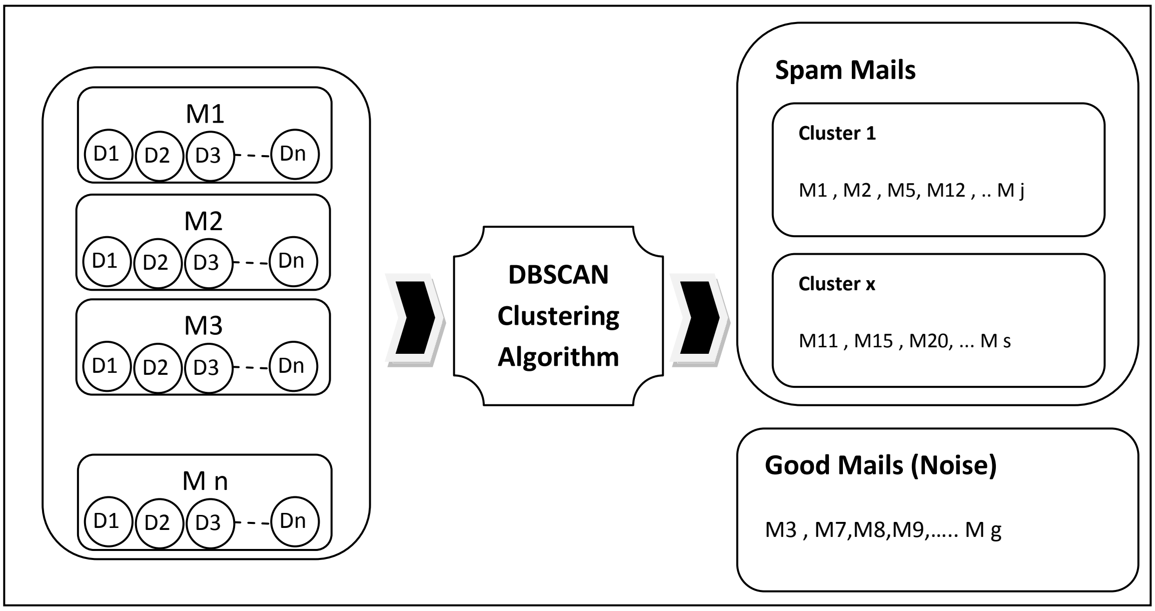

Figure 4.7 Clustering phase using DBSCAN Clustering

Chapter 5 - Experiments and results

5.1 Introduction

In this section, experiment environments and an evaluation method are introduced. First we will discuss the DBSCAN Parameters then we will describe different evaluation methods and finally evaluation process includes implementation and evaluation of proposed algorithm along with latest three algorithms including open digest, improved open digest and DBSCAN with open digest.

5.2 DBSCAN Parameters

DBSCAN requires two parameters: ε (eps) and the minimum number of points required to form a cluster (MinPts), for the first parameter ε (eps) value based on improved open digest algorithm it is equal to 38 since the proposed NCV equals to 90, so the threshold of distance between two mails will be 128 minus 90 which equals to 38.

For the second parameter MinPts based on the authors of DBSCAN [14], this parameter mainly depends on the density of the data being clustered. So for our evaluation to compare the performance of the proposed technique against DBSCAN with open digest technique we have evaluated both algorithm using multiple value of MinPts against accuracy where the accuracy of a measurement system is the degree of closeness of measurements of a quantity to that quantity's actual (true) value. So we can define accuracy as following:

$$\boldsymbol{Accuracy = \frac{Number\ of\ spam\ mails\ Clustered}{number\ of\ ALL\ Spam\ Mails}} \qquad \text{(Equation 3)}$$

We have used a collection of spam groups from Spam assassin public corpus [22]. The experiments of calculating the best MinPts that provides highest accuracy for the first collection of data that’s contains 90 mails from 20030228_spam.tar.bz2 spam repository from the Spam assassin public corpus [22] includes the following steps:

- 90 mails are selected from 20030228_spam.tar.bz2 spam repository from the Spam assassin public corpus[22]
- Then we start our proposed algorithm and DBSCAN with open Digest algorithms with different values of MinPts parameter including 2 – 15 range. The results as shown in figure 5.1

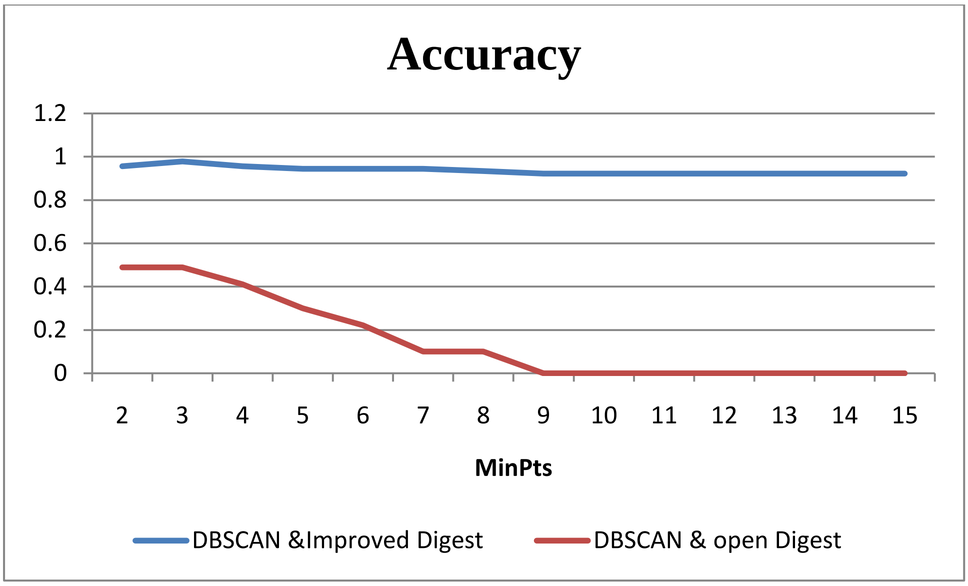

Figure 5.1 Best MinPts Value for 90 mails of 20030228_spam.tar.bz2

Based on the results shows in figure 5.1 for the range of MinPts from 2 to 15 MinPts=3 provides the highest accuracy for both algorithms so we have used MinPts=3.

For the second experiments which includes 200 mails from 20021010_spam group, we experiments MinPts values of range 2 -15 and the results shown in figure 5.2

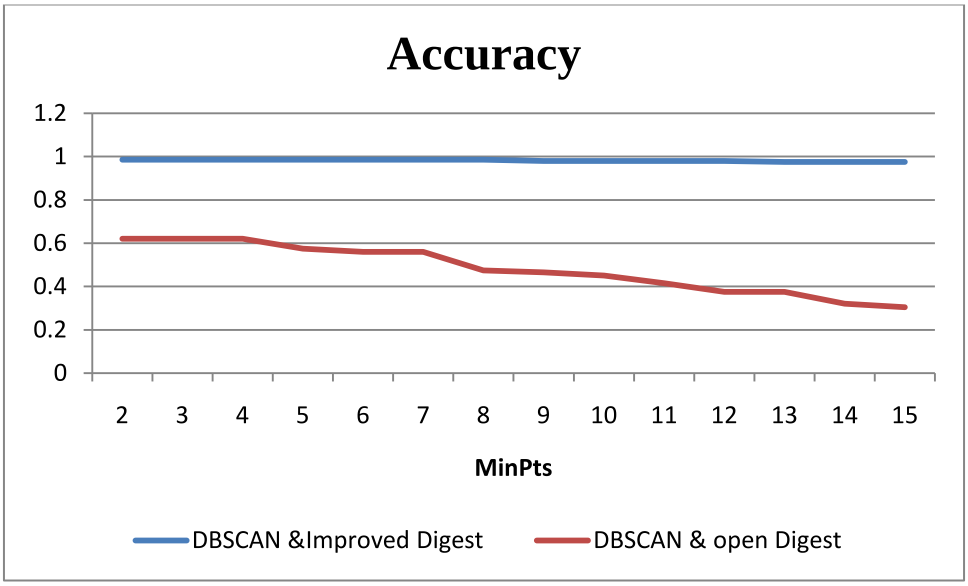

Figure5.2 Best MinPts Value for 200 Mails from 20021010_spam group

Based on Figure 5.2 we can use MinPts values from 2 – 4 since the best results for both algorithms resulted with these values, we will use MinPts =3.

5.3 Clustering Evaluation method

There are different methods to measure and evaluate the efficiency of clustering algorithms. These methods applied to the results of any clustering algorithm. The evaluation steps always apply pre-clustered and define data based known simulation data and then compare the resulted clusters with predefined data cluster.

We have searched for simulation data to measure and evaluate our proposed algorithm with latest algorithms. Latest papers like Ying[20] uses their own predefine data and doen’t mention any reference to it where Sarafijanovic[19] has used simulation data from Spam assassin public corpus [22] , this organization is specialized in searching and applying spam filter techniques and provides a real collection of data for simulation purposes. These collections of spam and ham mails are shown in table 5.1:

No.	Collection Name	Last Modified
1.	20021010_easy_ham.tar.bz2	2004-06-29 03:28
2.	20021010_hard_ham.tar.bz2	2004-12-16 19:56
3.	20021010_spam.tar.bz2	2004-06-29 03:27
4.	20030228_easy_ham.tar.bz2	2004-06-29 03:27

5.	20030228_easy_ham_2.tar.bz2	2004-06-29 03:27
6.	20030228_hard_ham.tar.bz2	2004-12-16 19:56
7.	20030228_spam.tar.bz2	2004-06-29 03:27
8.	20030228_spam_2.tar.bz2	2004-06-29 03:27

Table 5.1Simulation Data Collection Form SpamAssasin

To measure and evaluate our proposed algorithm against latest collaborative spam filter algorithms we have measure accuracy, precision and recall values.

The accuracy of clustering is determined by comparing the clusters obtained by the experiments with the real spam clusters specified by Spam assassin public corpus [22] using accuracy as mentioned in equation 3.

Precision and recall are the basic measures used in evaluating search strategy. Precision is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved. It is usually expressed as a percentage where recall is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. It is usually expressed as a percentage.

$$\boldsymbol{Precision = \frac{T_p}{T_p+F_p}} \qquad \text{Equation 3}$$

$$\boldsymbol{Recall = \frac{T_p}{T_p+F_n}} \qquad \text{Equation 4}$$

Where T_p is the number true positives (spam mails that has correctly clustered), F_p is the number of false positive (good mails that clustered as with spam mail) and $F_n$is the number of false negatives (Spam mail not clustered and specified as good mail).

Recall and precision are used to measure the ratio of false positives and false negatives of clustered mails

5.4 Implementation of Algorithms:

To evaluate our proposed algorithm against latest algorithms we have implemented it using Java programming language. We have used Java because of its flexibility and ability to implement any algorithm easily and efficiently.

The core of the proposed algorithm and latest algorithms is the open digest algorithm which generates a 256 bit digest from a given text whether this text is a complete mail or a fixed length string of a given mail. So the first class called "*Nilsimsa Class*" this class implements open digest algorithm. The main method of this class is digest method which generate a 256 bit digest, another important method called *compare (Dig1 , Dig2)* which returns the number of similar bit between two digests.

The second class named "*dbscan*" which implements DBSCAN algorithm , All mail digests are provided as a vector of strings to this class. A method named "*applyDbscan*" apply DBSCAN algorithm on provided digests and return clustered mails as a vector of lists, each list contains a group of similar mails that clustered.

To get the distance between two compared mails we use two different methods *getDistance(string1 , string2)* and *getDistanceAll(string1,string2)*. These two methods are used to implement both algorithms DBSCAN with open digest and the proposed algorithm DBSCAN with improved digest.

getDistance(string1,string2) method is used to implement the proposed algorithm DBSCAN with improved digest. Two compared mails are passed to this method , first it divide each mail based on the length of string division which is 60 characters, then it get the digest of each division, after that it compares all digests combinations of two mails and store the number of different bits in a collection. Finally it returns the distance which is the average of the smallest three values of the collections.

getDistanceAll(string1,string2) method is used to implement the DBSCAN with open digest. Two compared mails are passed to this method , first it gets the digest of each division, then it compares digests of the two mails and return the number of different bits.

To implement open digest and improved digest algorithms, a "*Threshold*" class is added to apply threshold clustering. We use the same methods of *getDistance(string1,string2)* and *getDistanceAll(string1,string2).*

5.5 Evaluation Experiments

To check the accuracy of the proposed algorithm against latest algorithms using accuracy equations as illustrated in equation 3. The proposed algorithm is experimented against latest algorithms including threshold clustering with open digest algorithm, threshold clustering with improved digest and DBSCAN with open digest algorithm Steps of the experiment include:

1- 90 of mails are sampled randomly from the used spam repository (we use 20030228_spam.tar.bz2 spam repository from the Spam assassin public corpus [22]);
2- For the measured algorithms parameter the threshold value for open digest is 54 where higher values indicates similar mails. The threshold value for improved digest algorithm is 90, if two mails have similar bits is larger than 90 they are considered to be similar .For both previous algorithm the minimum number of points in a single cluster is 4. For DBSCAN with open digest we assign the following parameters MinPts= 3 and ε (eps) =128-54=74 where 54 is the threshold value for open digest .The proposed algorithm MinPts=3 and ε (eps) =128-90=38 where 90 is the threshold of the improved digest. the MinPts Parameter value is set to 3 where this value provides the best accuracy value based on figure 5.1
3- The results are shown in figure 5.3.

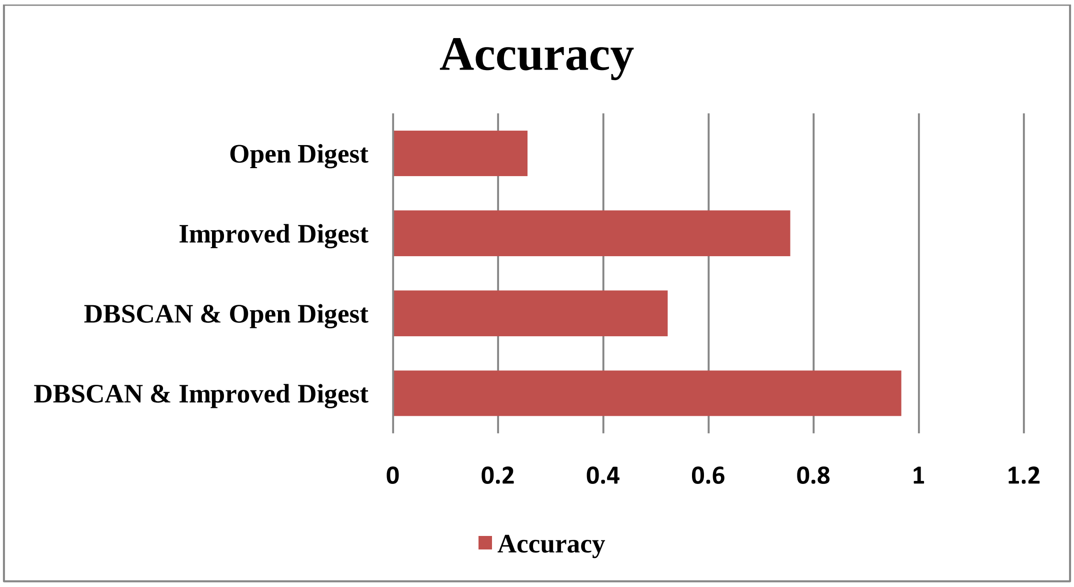

Figure 5.3 proposed algorithm accuracy against latest algorithms for 90 spam mails of 20030228_spam group

The same experiment is held again for a second group of spam mails including 200 spam mails from 20021010_spam group [22] and the results is show in figure 5.4.

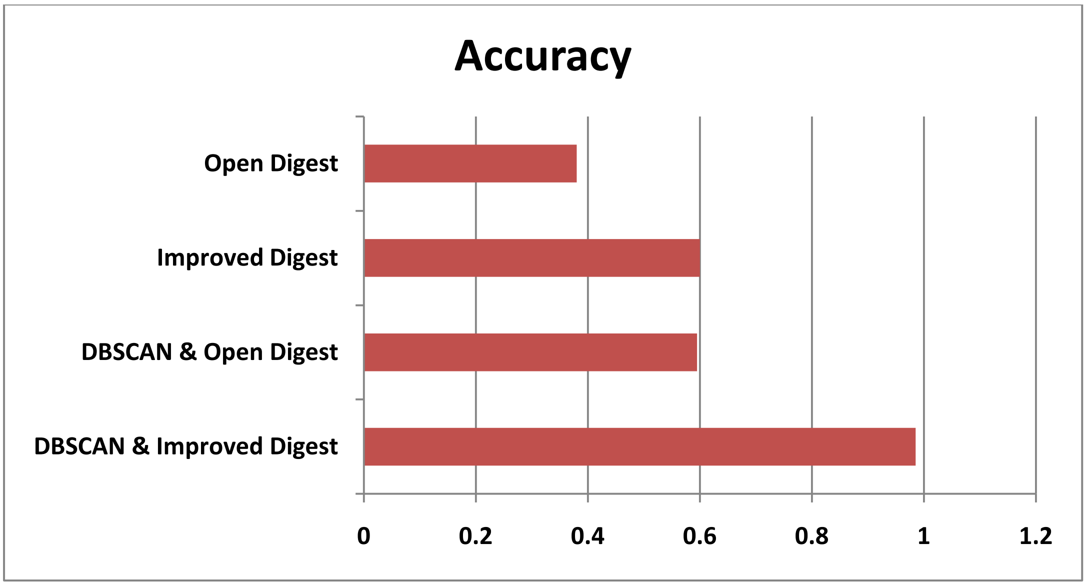

Figure 5.4 Accuracy of proposed algorithm against latest algorithms 200 spam mails of 20021010_spam group

Based on results shown in figure 5.3, 5.4, DBSCAN with improved digest provided the highest accuracy where it can capture 87 spam mails out of 90 spam mails from 20030228_spam group and 197 out of 200 spam mails from 20021010_spam group , as we can see from the result the impact of using DBSCAN with the improved digest algorithms has raised the accuracy of improved digest only to about 30%. The proposed algorithm has exceeded the accuracy of DBSCAN with open digest with about 35%.

The third experiment examine the precision and recall values, experiment includes the following steps:

- 60 of mails are sampled randomly from the used spam repository (we use 20030228_spam.tar.bz2 spam repository from the Spam assassin public corpus [22]);
- 20 of mails are sampled randomly from the used ham repository (we use 20030228_easy_ham.tar.bz2 ham repository and 20021010_hard_ham.tar.bz2 from the Spam assassin public corpus [22] the hard ham group contains spam messages which are closer in many respects to typical spam.
- The four algorithms have been experiment against the specified number of spam mails and ham mails, recall and precision has been calculated based on equation 4 and equation 5. The results is shown in table 5.2 and figure 5.5

Measured Value	open Digest	DBSCAN & Open Digest	Improved digest	DBSCAN & Improved Digest
False Negative	38	33	18	2
True Positive	12	17	42	58
False Positive	0	0	0	1

Table 5.2 clustering results for spam and ham mail

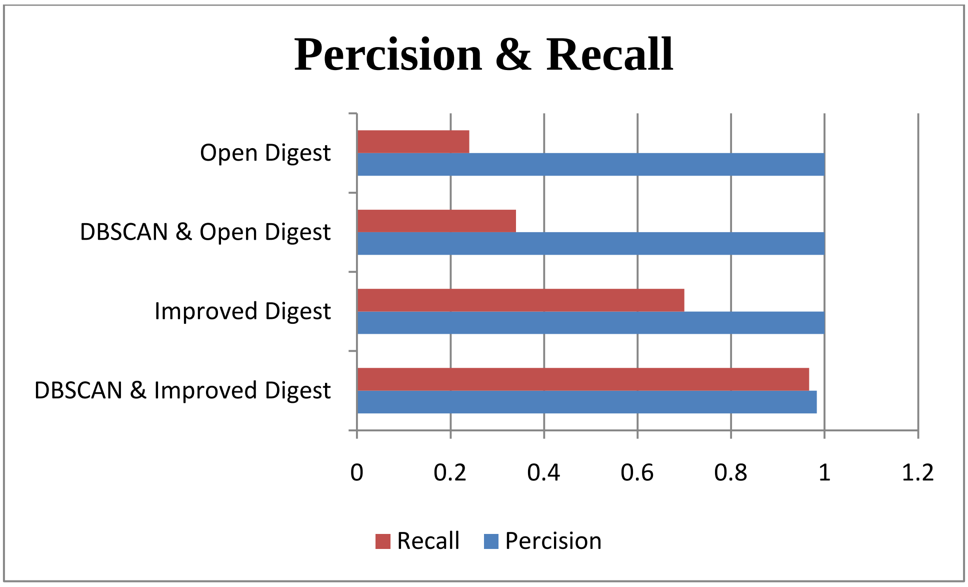

Figure 5.5 Precision and recall values of proposed algorithm against recent Algorithms

Based on results shown in Figure 5.5 the proposed algorithm provides the best recall and precision values. As we can see the recall value for the proposed algorithm is the highest value since most of spam mails have been clustered but other algorithms don’t cluster high number of spam and consider it as ham mails.

5.6 Results and discussion

Based on experiments results explained in previous section, we can see that the proposed algorithm performance has overcome the performance of latest algorithms including DBSCAN with open digest, improved Nilsimsa and Nilsimsa open digest.

The evaluation methods measure accuracy, precisions and recall values which are very critical measures for any searching algorithm. Based on results on figure 5.3 and figure 5.4 the accuracy with proposed algorithm has increased with about 40% than DBSCAN with open digest algorithm accuracy. This accuracy improvement resulted from using DBSCAN with improved Nilsimsa digest algorithm instead of old Nilsimsa open digest. The improved digest results are more accurate than old open digest with about 35 %.

The impact of using improved digest algorithm which include the operation of mail division into multiple fixed length strings and comparing these strings digest instead of comparing the whole mail digests has increased the opportunity to measure any similarity between any parts of any couple of mails. This impact made the critical improvement related to digest algorithm used in the representation phase of any mail.

The proposed algorithm accuracy increased with about 25% than improved digest with traditional threshold clustering. This accuracy improvement resulted from using DBSCAN clustering instead of threshold clustering which provides more accurate results.

The impact of using DBSCAN clustering algorithm instead of using threshold clustering has increased the accuracy of the proposed algorithm. The irregularity of spam mail cluster shape is considered one of the obstacles that face threshold clustering since lots of mails can’t be cluster or big spam mail cluster can be divided into multiple small clusters based on the threshold value specified by the algorithm. DBSCAN can deal with any irregular shape of the spam cluster and cluster it into one single cluster with a very low miss rate.

The precision and recall values are very critical for the returned value of any search algorithm. The number of false positive (number of good mails that classified as spam mail) and false negatives (number of spam mails classified as good mail) indicates another measure of algorithm accuracy.

Based on results in figure 5.5 the proposed algorithm have the smallest number of false negatives where open digest algorithm and DBSCAN with open digest have the highest number of false negatives. Which means proposed algorithm provides higher recall values than other algorithms with about to 40 %. This high value of recall resulted from the highest accuracy of the proposed algorithm.

The number of false positives of proposed algorithm is acceptable regarded to other compared algorithms since the proposed algorithm consider any sign of similarity between any pair of mails as the impact of using improved digest algorithm. The recall value of the proposed algorithm is the same as recall values of open digest and improved digest algorithms where recall value of DBSCAN with open digest has deceased. We expect this decreased value of recall because of using DBSCAN which can cluster any density reachable point. This is a double edged feature which can enhance the accuracy of search algorithm but may cluster good mail that may be similar to a single spam mail.

Chapter 6 Conclusion

6.1 Concluding Remarks

In this thesis, we have proposed and implemented spam filtering method using DBSCAN with improved digest algorithm which was proved to be a very effective solution to the problem of detecting spam bulks which mainly depends on spam mails similarity which while keeping miss detection of good mails at a similar level of older filtering methods.

The proposed algorithm used improved digest algorithm to represent incoming mails for similarity check and DBSCAN clustering algorithm to cluster similar mails with each other. The proposed algorithm has increased the accuracy of searching and clustering spam mails with 30% than other latest algorithms.

As a final conclusion, the DBSCAN with improved digest algorithms have been an accurate spam bulk detecting algorithms which can be merged with other types of other spam filtering techniques to build a full spam filter.

6.2 Future Work

We can do further research to optimize the division phase of the proposed algorithm. We have divided the delivered mails based on fixed length strings. This string length needs to be optimized to make the proposed algorithm to work faster.

Another value needs to be optimized is the number of maximum similar digests between two mail groups of digests. We have assigned this number to three but if it can be optimized it will provide better results.

References

[1] C. Pu and S. Webb, Observed trends in spam construction techniques: A case study of spam evolution. In Proc. of the 3rd Conf. on Mail and Anti-Spam, 2006

[2] L.F. Cranor and B.A. LaMacchia, Spam! Communications of the ACM, Augest 1998

[3] Spam Report .[Online] http://www.securelist.com/en/analysis/204792230/Spam_Report_April_2012

[4] Brian Satterfield , "Learn how different spam-fighting techniques work", techsoup newsletter ,Novermber 2006 .

[5] Wikipedia Spam Assassin. [Online], http://en.wikipedia.org/wiki/SpamAssassin

[6] SorceForge SpamProbe. [Online], http://spamprobe.sourceforge.net

[7] Rhyolite distributed checksum clearinghouse. .[Online] http://www.rhyolite.com/dcc/

[8] Jesse Kornblum, "Identifying almost identical files using context triggered piecewise hashing", Digital Investigation, vol. 3(sl):9 1-97, 2006

[9] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati. An open digest-based technique for spam detection. In Proceedings of The 2004 International Workshop on Security in Parallel and Distributed Systems, San Francisco, CA, USA, September 2004

[10] Zhang Jianzhong, Lu Hongbo, Lan Xiaofeng, Dong Dafan, "DHTnil: An approach to publish and lookup Nilsimsa digests in DHT". Proc. of the 2008 International Conference on High Performance Computing and Communications (HPCC-08), Dalian, China , 2008 .

[11] Ian H. Witten , Eibe Frank " Data Miningm Practical Machine Learning Tools and Techniques",second edition, 2005

[12] J. Han and M. Kamber, "Data Mining: Concepts and Techniques", 2nd edition, Elsevier, 2006

[13] J. MacQueen, "Some methods for classification and analysis of multivariate observation". In: Berkeley Symposium on Mathematical Statistics and Probability. University of California Press, pp. 281–297, 1967.

[14] M. Ester et. al. "A density-based algorithm for discovering clusters in large spatial databases with noise," In Second international conference on knowledge discovery and data mining", pp. 226–231. Portland, OR: AAAI Press, 1996

[15] Nilsimsa.[Online] https://github.com/jwilkins/nilsimsa.

[16] F. Zhou, L. Zhuang, B. Zhao, L. Huang, A. Joseph, and J. Kubiatowicz. Approximate object location and spam filtering on peer-to-peer systems. In Proceedings of ACM/IFIP/Usenix Int'l Middleware Conf., LNCS 2672, pp. 1-20.

[17] M.S. Pera and Y. -K. Ng, "Using Word Similarity to Eradicate Junk Mails," Proc. 1 6th ACM Int'I Conf.lnformation and Knowledge Management (CIKM), pp. 943 - 946, 2007.

[18] S. Sarafijanovic and J.-Y. Le Boudec. Artificial immune system for collaborative spam filtering. In Proceedings Of NICSO 2007, The Second Workshop on Naturen Inspired Cooperative Strategies for Optimization, Acireale, Italy, November 8-10, 2007.

[19] Slavisa Sarafijanovic , Sabrina Perez, JeanYves Le Boudec. “Improving DigestBased Collaborative Spam Detection”, MIT Spam Conference, Cambridge, Massachusetts, USA, March 27-28, 2008

[20] Wu Ying, Yang Kai, Zhang Jianzhong,” Using DBSCAN Clustering Algorithm in Spam Identifying”, 2010 2nd International Conference on Education Technology and Computer (ICETC), Nov,2010

[21] P. Moniza and P. Asha, “An Assortment of Spam Detection System” , International Conference on Computing, Electronics and Electrical Technologies [ICCEET], 20 1 2

[22] SpamAssassin-Public-Corpus. .[Online] http://spamassassin.org/publiccorpus/, March 2013