AENSI Journals

Australian Journal of Basic and Applied Sciences

ISSN:1991-8178

Journal home page: www.ajbasweb.com

Firefly and BAYES Classifier for Email Spam Classification in a Distributed Environment

[1]Karthika Renuka Dhanaraj and [2]Visalakshi Palaniswami

[1]*Anna University, Department of Information Technology, PSG College of Technology, Coimbatore, India.*
[2]*Anna University, Department of ECE, PSG College of Technology, Coimbatore, India*

ARTICLE INFO

Article history:
Received 19 August 2014
Received in revised form
19 September 2014
Accepted 29 September 2014
Available online 8 November 2014

Keywords:
Spam, feature selection, firefly, classification, Apache Hadoop, Distributed environment, E-mail, Naïve Bayes classifier, Spam, SVM, CSDMC2010 SPAM.

ABSTRACT

Background: In today's computer world, Internet plays a main role. Internet mail system is a store and forward mechanism used for the purpose of exchanging documents across computer network through Internet. Unsolicited Bulk E-mail(UBE) known as spam mail is considered to be a major threat. Email spam is a serious worldwide problem which causes problems for almost all computer users. There are many spam filters approaches to identify the incoming message as spam- white list / black list, Bayesian analysis, keyword matching, mail header analysis, postage, legislation, and content scanning etc. **Objective:** To filter spam mails many classification algorithms are used. Various machine learning algorithms that are used for classification includes Decision trees, Support Vector Machines, Naive Bayes schemes, Neural networks and Clustering. To deal with more number of clients the system is implemented in a distributed manner to increase its performance. An application when implemented in a distributed environment serves to be more efficient since it parallelizes its execution. **Results:** For Spambase dataset the accuracy of the proposed algorithm performs 26.7% better than Neural Network and 2.4% better than PSO algorithm and for CSDMC2010 SPAM the accuracy of the proposed algorithm performs 82.29% better than Neural Network and 2.56% better than PSO algorithm. **Conclusion:** A considerable reduction in the computation time is achieved when implemented in a distributed manner. The Application processes a huge amount of data and hence reduces its processing time considerably. The result analysis is done by comparing the efficiency of spam classification using Naïve Bayes, Support Vector Machine (SVM) with PSO and Firefly Algorithm for feature selection.

To Cite This Article: Karthika Renuka Dhanaraj and Visalakshi Palaniswami, Firefly and BAYES Classifier for Email Spam Classification in a Distributed Environment. ***Aust. J. Basic & Appl. Sci.,*** *8(17): 118-130, 2014*

INTRODUCTION

Nowadays the percentage of people accessing the Internet has increased rapidly. Most of them are using email for communication. Email evolution makes it possible to communicate in an easy, effective, fast and cheaper way. Managing these emails becomes a significant problem for individuals and organizations. Therefore, spammers prefer to send spam through such kind of communication. Nowadays, almost every second user has an E-mail, and consequently they are faced with spam problem. Spam is a big problem both for users and for ISPs. The causes are growth of value of electronic communications on the one hand and improvement of spam sending technology on the other hand. A. Bratko & B. Filipic (2005). Spam causes several problems, some of them resulting in direct financial losses. More precisely, spam causes misuse of traffic, storage space and computational power. W. Yerazunis (2004); spam makes users look through and sort out additional email, not only wasting their time and causing loss of work productivity, but also irritating them and, as many claim, violating their privacy rights; finally, spam causes legal problems by advertising pornography, pyramid schemes, etc. Evangelos Moustakas (2005).

Distributed application programs consist of many parts which are capable of running in different virtual machines at the same time. The efficiency of an application implemented in a distributed environment lies is the time factor which gets reduced. Reduction in the time is due to the simultaneous execution of various modules of the application. Big data is a large and complex datasets. Handling of big data is a difficult task due to the high level complexity involved in understanding and processing of these data. Apache Hadoop is open source software available to processes these big data in a very efficient manner. Hadoop's Map Reduce Framework parallelizes the execution of various modules in a distributed manner. This leads to the efficient processing of

Corresponding Author: Karthika Renuka Dhanaraj, Anna University, Department of IT, PSG College of Technology, Coimbatore-641004, India.
Tel: +91 9976128726 E-mail: karthirenu@gmail.com

big data to get the desired results out of it. Karmasphere is the software used to implement the distributed environment.

In this paper, we have proposed an efficient technique to classify the email spam using firefly and naïve bayes classifier. Initially, the input email data is given to the feature selection to select the suitable feature for spam classification. The traditional firefly algorithm is taken and the optimized feature space is chosen with the best fitness. Once the best feature space is identified through firefly algorithm, the spam classification is done using the Naïve Bayes classifier. The results for the spam classification technique are validated through evaluation metrics namely, sensitivity, specificity, accuracy and computation time.

The rest of the paper is organized as follows: A brief account of the related research papers are given in section 2. Section 3 describes the problem statement and contribution. The proposed technique is presented in section 4. The detailed experimental results and discussions are given in Section 5. The conclusions are summed up in Section 6.

Related researchers: a brief review:

Several spam detection and handling techniques have been proposed to mitigate the impact of spam on e-mail and Internet users. Here, the review of recent works from these topics is presented. Wenqing Zhao & Yongli Zhu (2005) have developed a DTRS (Decision-Theoretic Rough Set Theory) based email classification model was developed, which was classified the incoming emails into three categories spam, non-spam and suspicious rather than just classifying the incoming emails as spam and non-spam. By comparing with popular classification methods like Naïve Bayes classification, anti-Spam filter model reduced the error ratio that a non-spam was discriminated to spam, and they found potential security problems of some email systems. The experimental results showed that DTRS based model was reduced the error rate that discriminating a non-spam to spam.

Hongrong Cheng *et al.* (2010) have presented a framework named BFMLC (Binary Filtering with Multi-Label Classification) to take both spam image filtering and user preferences into account. The BFMLC framework comprised two-stage classification tasks: the filter-oriented binary classification and user-oriented multi-label classification. They implemented a spam image filtering system based on the BFMLC framework and conduct experiments in public personal datasets. In addition, they showed how to implement a filter based on BFMLC framework and conduct experiments to illustrate the effectiveness of the framework. Rasim M *et al.*(2011) have formalized the problem of clustering of spam messages collection. The criterion function was a maximization of similarity between messages in clusters, which was defined by k-nearest neighbor algorithm. Genetic algorithm including penalty function for solving clustering problem was offered. Classification of spam messages coming to the bases of antispam system was also given. After classification, the knowledge extraction from divided classes was considered.

Dae-Neung Sohn *et al*. (2012) have developed an approach to spam classification of extremely short messages using not only lexical features that reflect the content of a message but stylistic features that indicated the manner in which the message was written. Experiments on two mobile phone message collections in two different languages showed that the approach outperformed previous content-based approaches significantly, regardless of language. Mu-Chun Su *et al.* (2010) have presented a approach to constructing a neural tree to integrate the advantages of decision trees and neural networks. The presented neural tree, called a quadratic-neuron-based neural tree (QUANT), was a tree-structured neural network composed of neurons with quadratic neural-type junctions for pattern classification. A quadratic neuron was capable of forming a hyper-ellipsoid that can be varied in sizes and in locations on the space spanned by the input variables. Via a batch-mode training algorithm, the QUANT grows a neural tree containing quadratic neurons in its nodes. To demonstrate the performance of the presented QUANT, one pattern recognition problem and the spam e-mail detection problem were tested.

Sarah Jane Delany *et al*. (2012) have presented the state of the art in SMS spam filtering and have reviewed a number of different approaches to the problem which have been suggested and tested. The paper also discussed the issues with data collection and availability for furthering research in that area, analyses a large corpus of SMS spam, and provided some initial benchmark results. Muhammad N *et al.* (2009) have developed a spam detection technique, at the packet level (layer 3), based on classification of e-mail contents. Their project targets spam control implementations on middle boxes. E-mails were first pre-classified (pre-detected) for spam on a per-packet basis, without the need for reassembly. This, in turn, allows fast e-mail class estimation (spam detection) at receiving e-mail servers to support more effective spam handling on both inbound and outbound (relayed) e-mails. In their paper, the naïve Bayes classification technique was adapted to support both pre-classification and fast e-mail class estimation, on a per-packet basis. They focused on evaluating the accuracy of spam detection at layer 3, considering the constraints on processing byte-streams over the network, including packet re-ordering, fragmentation, overlapped bytes, and different packet sizes.

Australian Journal of Basic and Applied Sciences, 8(17) November 2014, Pages: 118-130

Problem definition and contribution of the paper:

Recently, various researchers present several algorithms for email spam classification based on classification methods. But, the challenge is not only in finding the spam e-mails and also, how the dimensionality and scalability is taken into consideration for spam classification because, in reality, the processing is with a large and high dimensional data. So, (i) curse of dimensionality By handling these criteria, an email spam classification technique is urgently needed for improving the classification accuracy. By solving the above challenge in this research, the feature selection is the main aspect for our research. The feature selection method can solve the curse of dimensionality by identifying the suitable features.

- Curse of dimensionality problem is solved by identifying the suitable features. Here, firefly algorithm is used for this suitable feature selection process.
- Also the time complexity can be reduced if the algorithm is implemented in a distributed environment such as Hadoop.

Hadoop is open source software designed for reliable, scalable, distributed computing data. It works on the principle of Google's Map Reduce programming model for distributed computing where the master node takes the input, divides it into smaller sub-problems, and distributes them to worker nodes. Map/Reduce is a programming paradigm that expresses a large distributed computation as a sequence of distributed operations on data sets of key/value pairs. In Map-reduce algorithm the master node takes the input, divides it into smaller sub-problems, and distributes them to worker nodes. The worker node processes the smaller problem, and passes the answer back to its master node. The master node then collects the answers to all the sub-problems and combines them to form the output.

A Map/Reduce computation has two phases, a map phase and a reduce phase. The input to the computation is a data set of key/value pairs. In the map phase, the framework splits the input data set into a large number of fragments and assigns each fragment to a map task. The framework also distributes the many map tasks across the cluster of nodes on which it operates. Each map task consumes key/value pairs from its assigned fragment and produces a set of intermediate key/value pairs. Following the map phase is the reduce phase, each reduce task consumes the fragment of tuples assigned to it. For each such tuple it invokes a user-defined reduce function that transmutes the tuple into an output key/value pair.

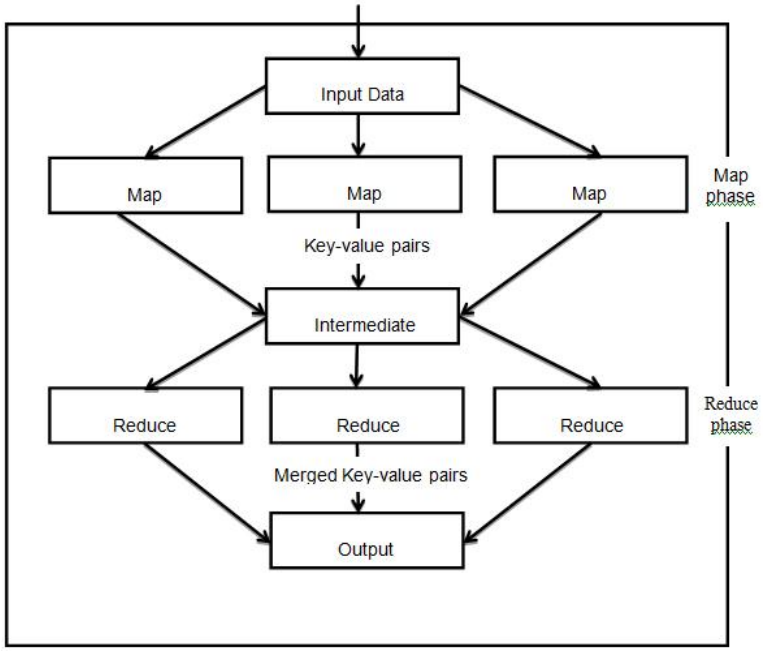

Fig. 1: Map Reduce Framework of Apache Hadoop.

Efficient Email Spam Classification Through Proposed Feature Extraction Algorithm For Naïve Classifier:

The ultimate target of this research is to design and develop a technique for email classification using Naïve Bayes classifier. The Naïve Bayes algorithm spam filtering is a probabilistic classification technique of email filtering which is based on Bayes theorem with naïve independence assumptions. Let us consider each of the email can be illustrated by a set of features (attributes) $\{a_n\}$, where $1 < n > N$. Filtering of spam mails with Naïve Bayes by considering of all features is very difficult also it need more time. In order solve this problem in this paper; we propose an efficient algorithm to select the significant features from the available to filter the

Australian Journal of Basic and Applied Sciences, 8(17) November 2014, Pages: 118-130

spam in efficient manner. The overall model of the proposed-mail spam classification system is given in the figure 2 and each part of the framework is elucidated concisely in the following sections.

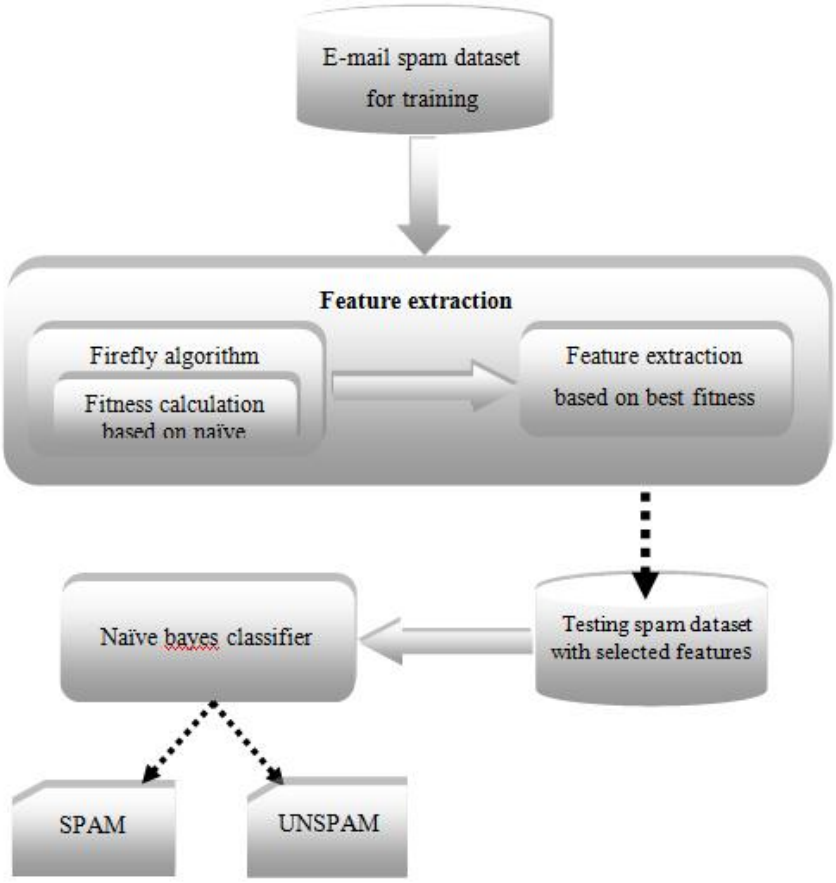

Fig. 2: Overall model of the proposed-mail spam classification system.

Spam dataset:

The spam dataset is taken from the UCI machine learning repository and which is formed by Mark Hopkins, Erik Reeber, George Forman, Jaap Suermondt. Hewlett-Packard Labs. Their collection of spam e-mails came from their postmaster and individuals who had filed spam and their collection of non-spam e-mails came from filed work and personal e-mails, and hence the word 'George' and the area code '650' are indicators of non-spam. This spam dataset consists of 4601 instances and 58 attributes in which 57 continuous real attributes and 1 nominal attribute. From the spam dataset, 80% (3681 instances) of instances are taken for finding the significant features among the available (58) attributes for the training process and the other 20% (920 instances) of instances are taken for the testing process with the significant attributes. The descriptions of those attributes are given in the table 1.

Table 1: The description about the attributes of spam dataset.

Attribute number	Category of attribute	Description of attribute
A1 to A48	Word_freq_WORD	percentage of words in the e-mail that match WORD
A49 to A54	char_freq_CHAR	Percentage of characters in the e-mail that match CHAR
A55	capital_run_length-_average	average length of uninterrupted sequences of capital letters
A56	capital_run_length-_longest	Length of longest uninterrupted sequence of capital letters
A57	capital_run_length-_total	Total numberof capital letters in the e-mail
A58	Class attribute	Denotes if the e-mail was spam (1) or not (0)

Significance feature extraction algorithm:

In order to select the significance feature, here we utilized the firefly algorithm, each firefly is represents the set of attributes and we calculate the fitness for each subsequently, we select the best firefly and their

corresponding attributes from the training dataset through the fitness function. The selected significant attribute data is given to the Naïve classifier to classify the email is spam or not. The firefly algorithm starts its process from initialization matrix by get the population of firefly population and dimension of the firefly. The population of firefly N is given by the user and the dimension (attributes) of each the firefly $M-1$(58-1=57) of the spam dataset. With the intention of select the significant features, here we calculate the fitness for each firefly; the fitness calculation is done through calculation of accuracy. The accuracy is calculated by compare the 58th attribute of training spam dataset with the result of the Naïve Bayes algorithm. The Naïve Bayes algorithm analyzes the input data of 57 attributes of training spam dataset and it finds the 58th attribute of each instances. The value k represents the number of iterations, initially the value of k is zero $k=0$ and it will increase counter by counter at each iteration. The following table 1 represents the initialization matrix of firefly algorithm.

Table 2: Represent the initialization matrix of the firefly algorithm.

	a_1	a_2	a_3	⋯	a_m	⋯	a_{56}	a_{57}
ff_1^k	1	0	0	1	0	1	0	0
ff_2^k	0	0	1	1	0	0	0	1
ff_3^k	1	0	1	0	0	1	0	0
-	1	1	0	0	1	0	0	1
-	0	1	0	1	1	0	1	1
ff_n^k	1	0	0	1	0	1	0	0
.	0	1	1	0	1	0	1	0
.	0	1	0	1	0	1	0	1
ff_{N-1}^k	1	0	1	0	0	0	1	1
ff_N^k	1	0	0	0	1	1	0	1

From the table 1 each firefly has 58 dimensions each of which represents the attributes of spam dimension which is loaded by '0' and '1' where the value of '0' represents the corresponding attribute is not taken into the account and the value of '1' represents the corresponding attribute is taken into the account. We consider those attributes a wherever the '1' is present subsequently we apply those attributes a from the training spam dataset into the naïve bayes and we calculate the fitness for each firefly.

Algorithm procedure:

Input: spam dataset with M number of attributes
Output: classified mail
Parameters
k = number of iteration
N = total number of firefly population
M = number of attributes
$NB(I)$ = instance of naïve bayes
$Pos(I|S)$ = posterior of spam of instance I

Pseudo code
Begin
1. set k= 0
2. get population of firefly N
3. get the number of attributes M
4. initialize the firefly population
5. for each firefly
4.1 call naïve bayes for training
4.2 call fitness
6. end for
7. select the firefly which has best fitness
8. select corresponding attributes from the testing part of the spam database
9. call naïve bayes for testing
10. k=k+1

Australian Journal of Basic and Applied Sciences, 8(17) November 2014, Pages: 118-130

11. update each firefly (equations 12 & 13)
12. go to step 4
End
Subroutine: Naïve bayes
1. select those attributes from the spam dataset
2. generate naïve bayes instance matrix $NB(I)$ *for selected attributes from spam dataset*
(for training select 80% of instances I_b *where* $1 \le b \ge B$
for testing select 20% of instances I_c *where* $1 \le c \ge C$ *)*
3. for each attribute from the matrix of $NB(I)$
3.1. calculate mean and variance [equations 7 & 8]
4. for each instance from the $NB(I)$
4.1 calculate $P(a_i(I)|(S))$ *&* $P(a_i(I)|(NS))$
(for training equations 4 and 5
for testing equations 17 & 18)
4.2 calculate $evidence(I)$
(for training equations 6; for testing equation 19)
4.3 calculate $Pos(I|S)$ *&* $Pos(I|NS)$
(for training equations 2 & 3
for testing equation 15 & 16)
5. end for
6. end for
7. if $Pos(I|S) > Pos(I|NS)$
7.1 then $I \rightarrow Spam$
8. else
8.1 then $I \rightarrow Non-Spam$
Subroutine: Fitness
1. compare the outcome result of naïve bayes with the spam dataset
2. calculate accuracy as fitness $F(ff_n^k)$ *(equation 11)*

Probability calculation using Naïve Bayes Algorithm:

For instance the suspected message contains the word "replica". Most people who are used to receiving e-mail know that this message is likely to be spam, more precisely a proposal to sell counterfeit copies of well-known brands of watches. The spam detection software, however, does not "know" such facts; all it can do is compute probabilities.

The formula used by the software to determine that is derived from Bayes' theorem

$$\Pr(S|W) = \frac{\Pr(W|S) \cdot \Pr(S)}{\Pr(W|S) \cdot \Pr(S) + \Pr(W|H) \cdot \Pr(H)}$$

where:

- Pr(S|W) is the probability that a message is a spam, knowing that the word "replica" is in it;
- Pr(S) is the overall probability that any given message is spam;
- Pr(W|S) is the probability that the word "replica" appears in spam messages;
- Pr(H) is the overall probability that any given message is not spam (is "ham");
- Pr(W|H) is the probability that the word "replica" appears in ham messages.

Australian Journal of Basic and Applied Sciences, 8(17) November 2014, Pages: 118-130

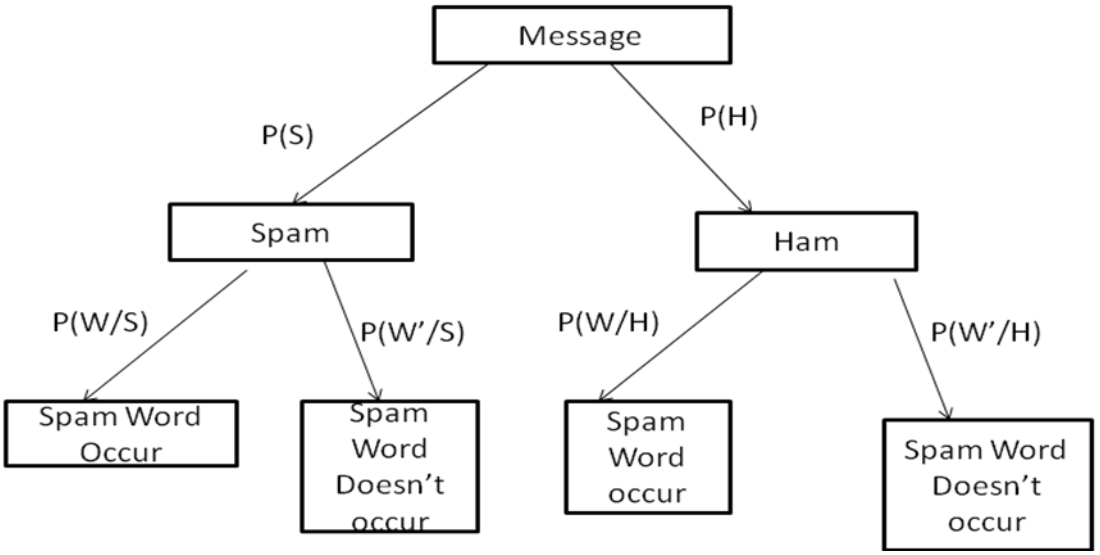

Fig. 3: Naive Bayes Algorithm.

Fitness calculation:

With the intention of extraction of significant features, we select the various random features through the firefly algorithm also we calculate the fitness of each set of features (firefly). The fitness of each firefly is calculated through the Naïve Bayes algorithm. For each firefly, the Naïve Bayes algorithm taken the 57 attributes from the training spam dataset wherever the value '1' present. For each firefly, the Naïve Bayes algorithm selects the entire instances from the training spam dataset with only the selected significance attributes. Let us consider the n^{th} firefly ff_n^k which consists of m number of '1' subsequently the corresponding attributes and all the instances from the training spam dataset are elected which is represented as instances of Naïve Bayes $NB(I_b)$ where $1 \le b \ge B$. The value of B represents the last instances of training spam dataset (3681st instance). The following table represents the instance of Naïve Bayes of training dataset ff_n^k.

Table 3: Training spam dataset instances of naïve bayes.

	a_1	a_2	a_i	a_{m-1}	a_m
I_1	d_{11}	d_{12}	d_{1i}	d_{1m-1}	d_{1m}
I_2	d_{21}	d_{22}	d_{2i}	d_{2m-1}	d_{2m}
I_b	d_{b1}	d_{b2}	d_{bi}	d_{bm-1}	d_{bm}
I_{B-1}	d_{B-11}	d_{B-12}	d_{B-1i}	d_{B-1m-1}	d_{B-1m}
I_B	d_{B1}	d_{B2}	d_{Bi}	d_{Bm-1}	d_{Bm}

After the corresponding attributes (features) obtain from the training spam datasets, the next process of the Naïve Bayes is to calculate the 58th attribute of each instance. To do that, for each instance, algorithm calculates the posterior (spam) $Pos(S)$ and posterior (non-spam) $Pos(NS)$. The algorithm compares and determines which posterior is greater. If the posterior (spam) of the instance I_b is greater, then the instance corresponding instance related to spam mail otherwise it related to non-spam which is represented in the equation 1. The following equations (2 to 10) helps to find $Pos(S)$ & $Pos(NS)$.

$$if\ Pos(I_b|S) > Pos(I_b|S)\ then\ I_b \to Spam$$
$$else\quad I_b \to Non-Spam \tag{1}$$

Australian Journal of Basic and Applied Sciences, 8(17) November 2014, Pages: 118-130

$$Pos(I_b|S) = \frac{P(S_{trn})\prod_{i=1}^{m} P\left(a_i(I_b)\middle|(S)\right)}{evidence(I_b)} \tag{2}$$

$$Pos(I_b|NS) = \frac{P(NS_{trn})\prod_{i=1}^{m} P\left(a_i(I_b)\middle|(NS)\right)}{evidence(I_b)} \tag{3}$$

$$P\left(a_i(I_b)\middle|(S)\right) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(\frac{-\left(d_{bi}-\mu(a_i)\right)^2}{2\sigma^2}\right) \tag{4}$$

$$P\left(a_i(I_b)\middle|(NS)\right) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(\frac{-\left(d_{bi}-\mu(a_i)\right)^2}{2\sigma^2}\right) \tag{5}$$

$$evidence(I_b) = P(S)\prod_{i=1}^{m} P\left(a_i(I_b)\middle|(S)\right) + P(NS)\prod_{i=1}^{m} P\left(a_i(I_b)\middle|(NS)\right) \tag{6}$$

$$Mean:\ \mu(a_i) = \frac{1}{m}\sum_{i=1}^{m} a_i \tag{7}$$

$$Variance:\ \sigma(a_i) = \frac{1}{m}\sum_{i=1}^{m}\left(a_i-\mu(a_i)\right) \tag{8}$$

The value of probability of spam $P(S_{trn})$ and probability of non-spam $P(NS_{trn})$ is calculated from the training spam dataset.

$$P(S_{trn}) = \frac{N(S_{trn})}{N(I_{trn})} \tag{9}$$

$$P(NS_{trn}) = \frac{N(NS_{trn})}{N(I_{trn})} \tag{10}$$

From the above equations (9 and 10) $N(S_{trn})$ represents the number of occurrences of spam in the 58th attribute of training dataset, $N(NS_{trn})$ represents the number of occurrences of spam in the 58th attribute of training dataset and $N(I_{trn})$ represents the total number of instances in the training spam dataset. Now we classified the each instances is spam or not through the above equations. At this time, we compare the outcome result with the training spam dataset and we calculate the accuracy through the following equation (11) as fitness function for each firefly.

$$F\left(ff_n^k\right) = \frac{number\ of\ true\ positives + number\ of\ true\ negatives}{number\ of\ true\ positives + false\ negatives + true\ negatives + false\ positives} \tag{11}$$

Firefly Updation:

In this paper, the updation of the each firefly is based on its fitness; the following equations are utilized to calculate the fitness of each firefly.

$$\left(ff_n^{k+1}\right) = ff_n^k + \beta\exp\left[-\gamma\ r_{nq}^2\right]\left(ff_n^k - ff_q^k\right) + \alpha\left(rand - \frac{1}{2}\right) \tag{12}$$

$$r_{nq} = \left\| ff_n^k - ff_q^k \right\| = \sqrt{\sum_{m=1}^{M}(ff_{nm} - ff_{qm})^2} \tag{13}$$

From the above equations (12 and 13) r_{nq} is the distance between the two fireflies where $(1 \le n \ge M-1)$ and $(1 \le q \ge M)$ and the $\gamma, \sigma, \beta, rand$ are the random values between 0 to 1. Among the two fireflies, the firefly gets update which has more fitness among them. Likewise entire fireflies get updates through the above equations (12 and 13). At each updation of firefly the value of k become increase counter by counter and this process get stop when it achieves k= K where K is the total number of iteration. After the completion of K number of iterations on the firefly, we select the firefly which has the best fitness (highest fitness value) among the available and we select the corresponding attributes and instances from the testing dataset. The selected attributes are the significance attributes among the 57 attributes, the classification process is sort the mail based on the selected significant attributes. Finally the selected attributes from the testing dataset is given to the Naïve Bayes algorithm to classify each instance become spam or not.

Mail classification:

At this moment, the algorithm has takes testing data for classify the mails become spam or not, in this phase for classifying the testing spam dataset we utilize the Naïve Bayes classifier to classify the mails. Let us consider the n^{th} firefly ff_n^k which consists of maximum fitness and m number of '1' subsequently the corresponding attributes and all the instances from the testing spam dataset are elected which is represented as instances of Naïve Bayes $NB(I_c)$ where $1 \le c \ge C$. The value of C represents the last instances of testing spam dataset (920th instance). The following table represents the instance of Naïve Bayes of ff_n^k.

Table 4: Testing spam dataset instances of naïve bayes

	a_1	a_2	a_i	a_{m-1}	a_m
I_1	d_{11}	d_{12}	d_{1i}	d_{1m-1}	d_{1m}
I_2	d_{21}	d_{22}	d_{2i}	d_{2m-1}	d_{2m}
I_c	d_{c1}	d_{c2}	d_{ci}	d_{cm-1}	d_{cm}
I_{C-1}	d_{C-11}	d_{C-12}	d_{C-1i}	d_{C-1m-1}	d_{C-1m}
I_C	d_{C1}	d_{C2}	d_{Ci}	d_{Cm-1}	d_{Cm}

For each instances in the testing spam dataset with the selected significant attributes, algorithm calculates the posterior (spam) $Pos(S)$ and posterior (non-spam) $Pos(NS)$. The algorithm compares and determines which posterior is greater. If the posterior (spam) of the instance I_c is greater, then the instance corresponding instance related to spam mail otherwise it related to non-spam which is represented in the equation 1. The above equations helps to find $Pos(S)$ & $Pos(NS)$.

$$\begin{aligned} &if\ Pos(I_c|S) > Pos(I_c|S)\ then\ I_c \to Spam \\ &\qquad else\quad I_c \to Non-Spam \end{aligned} \tag{14}$$

$$Pos(I_c|S) = \frac{P(S_{tst}) \prod_{i=1}^{m} P\left(a_i(I_c)|(S)\right)}{evidence(I_c)} \tag{15}$$

$$Pos(I_c|NS) = \frac{P(NS_{tst}) \prod_{i=1}^{m} P\left(a_i(I_c)|(NS)\right)}{evidence(I_c)} \tag{16}$$

$$P\left(a_i(I_c)|(S)\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-\left(d_{bi} - \mu(a_i)\right)^2}{2\sigma^2} \right) \tag{17}$$

$$P\left(a_i\left(I_c\right)\middle|\left(NS\right)\right)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(\frac{-\left(d_{ci}-\mu\left(a_i\right)\right)^2}{2\sigma^2}\right) \tag{18}$$

$$evidence(I_c)=P\left(S\right)\prod_{i=1}^{m}P\left(a_i\left(I_c\right)\middle|\left(S\right)\right)+P\left(NS\right)\prod_{i=1}^{m}P\left(a_i\left(I_c\right)\middle|\left(NS\right)\right) \tag{19}$$

$$Mean:\ \mu\left(a_i\right)=\frac{1}{m}\sum_{i=1}^{m}a_i \tag{7}$$

$$Variance:\ \sigma\left(a_i\right)=\frac{1}{m}\sum_{i=1}^{m}\left(a_i-\mu\left(a_i\right)\right) \tag{8}$$

The value of probability of spam $P\left(S_{tst}\right)$ and probability of non-spam $P\left(NS_{tst}\right)$ is calculated from the training spam dataset.

$$P\left(S_{tst}\right)=\frac{N\left(S_{tst}\right)}{N\left(I_{tst}\right)} \tag{20}$$

$$P\left(NS_{tst}\right)=\frac{N\left(NS_{tst}\right)}{N\left(I_{tst}\right)} \tag{21}$$

From the above equations (9 and 10) $N\left(S_{tst}\right)$ represents the number of occurrences of spam in the 58th attribute of testing dataset, $N\left(NS_{tst}\right)$ represents the number of occurrences of spam in the 58th attribute of testing dataset and $N\left(I_{tst}\right)$ represents the total number of instances in the testing spam dataset.

RESULTS AND DISCUSSION

This section presents the results obtained from the experimentation and its detailed discussion about the results. The proposed approach of email spam classification is experimented with the Spambase, CSDMC2010 SPAM corpus Datasets and the result is evaluated with the sensitivity, specificity and accuracy and computation time.

Database description:

Spambase Dataset: The spambase dataset is taken from UCI machinery. The dataset contains 4601 instances of email, of which about 39% is spam. Each instance corresponds to a single email, and is represented with 57 attributes plus a class label (1 if spam, 0 if not). The data files contain one instance per line, and each line has 58 comma delimited attributes, ending with the class label. Most of the attributes indicate whether a particular word or character was frequently occurring in the e-mail; frequency is encoded as a percentage in [0,1]. A few attributes measure the length of sequences of consecutive capital letters.

CSDMC2010 SPAM corpus Dataset: CSDMC2010 SPAM corpus datasets includes 4327 labelled emails where 2949 emails tagged as ham and 1378 emails tagged as spam.

Evaluation Metrics:

The evaluation of proposed technique in email dataset is carried out using the following metrics as suggested by below equations,

$$sensitivy=\frac{number\ of\ true\ positives}{number\ of\ true\ positives+number\ of\ \ false\ negatives} \tag{22}$$

$$Specificity=\frac{number\ of\ true\ \ negatives}{number\ of\ true\ negatives+number\ of\ \ false\ \ positives} \tag{23}$$

$$Accuracy=\frac{number\ of\ true\ positives+number\ of\ true\ \ negatives}{number\ of\ true\ positives+\ false\ \ negatives+true\ \ negatives+\ false\ \ positives} \tag{24}$$

Table 5: Confusion matrix.

		Predicted Label	
		Positive	Negative
Known Label	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Performance analyze on spam base dataset:

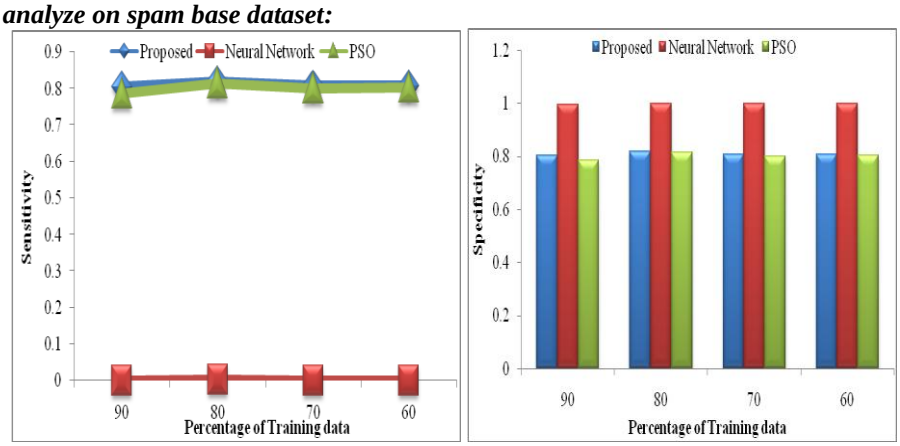

Fig. 4: comparative analysis of sensitivity and specificity on spam base dataset.

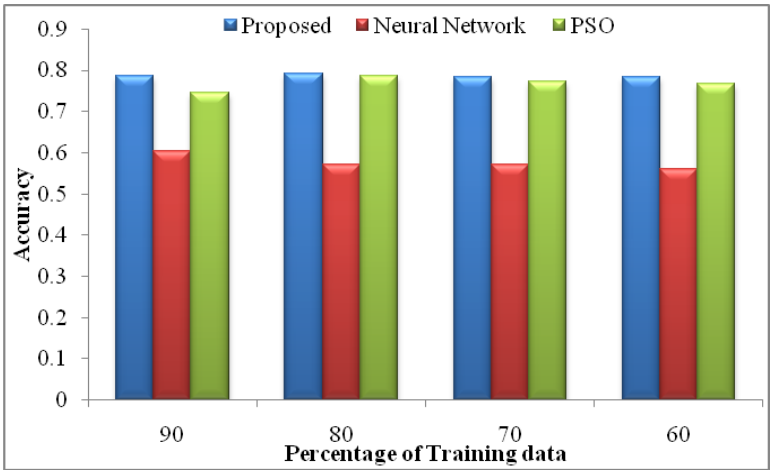

Fig. 5: Comparative analysis of accuracy on spam base dataset.

Performance analyze on CSDMC2010 SPAM corpus dataset:

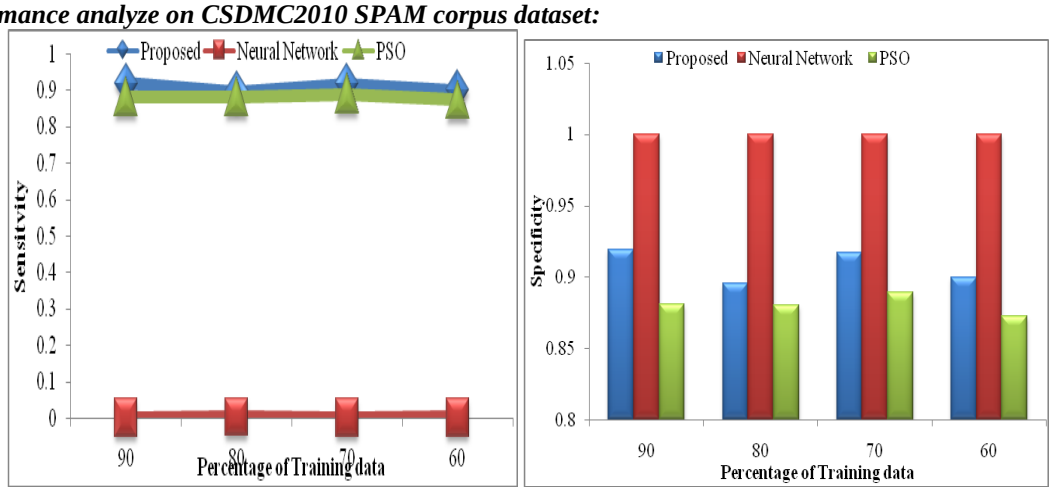

Fig. 6: Comparative analysis of sensitivity and specificity CSDMC2010 SPAM corpus dataset.

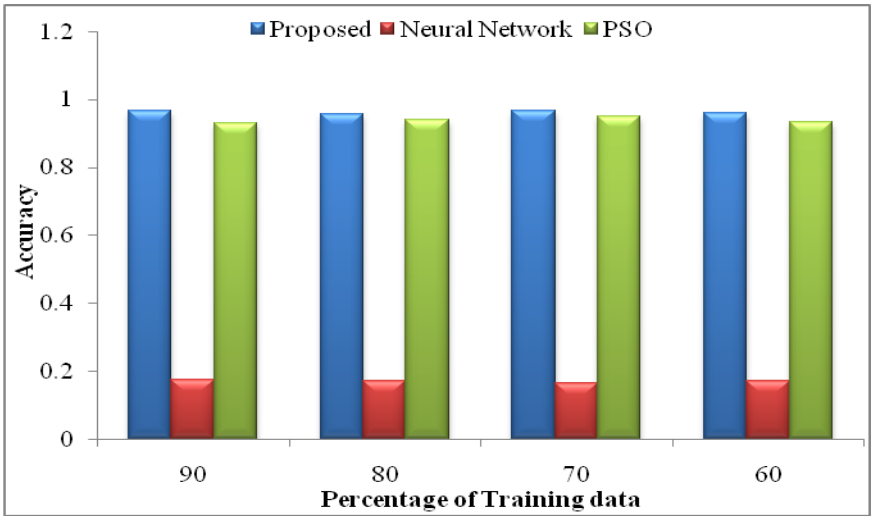

Fig. 7: Comparative analysis of accuracy CSDMC2010 SPAM corpus dataset.

Comparative analysis:

In this paper, we have compared proposed technique of improved email spam classification against and existing techniques which are particle swarm optimization, neural network. The performance analysis has been made by plotting the graphs of evaluation metrics such as sensitivity, specificity and accuracy. By analyzing the plotted graph, the performance of the proposed email spam classification technique has significantly improved. The evaluation graphs of the sensitivity, specificity and the accuracy graph are shown in figure 4 to 7 for two datasets such as spam base and CSDMC2010 SPAM corpus. By analyzing the above figure 5 represents the comparative analysis of accuracy on spam base dataset from which in terms of accuracy our proposed algorithm performs 26.7 % better than neural network and 2.4% better than PSO algorithm. By analyzing the above figure 7 represents comparative analysis of accuracy CSDMC2010 SPAM corpus dataset from which in terms of accuracy our proposed algorithm performs 82.29% better than Neural Network and 2.56% better than PSO algorithm.

Conclusion:

In this paper, we have presented an efficient technique to classify the email spam using firefly and Naïve Bayes classifier. Initially, the input email data is given to the feature selection to select the suitable feature for spam classification. The traditional firefly algorithm is taken and the optimized feature space is chosen with the best fitness. Once the best feature space is identified through firefly algorithm, the spam classification is done using the Naïve Bayes classifier. Here, these two processes are effectively distributed based on the concept given in Map-Reduce framework. The results for the spam detection are validated through evaluation metrics namely, sensitivity, specificity, accuracy and computation time. For comparative analysis, proposed spam classification is compared with the existing works such as particle swarm optimization and neural network for two datasets such as spam base and CSDMC2010 SPAM corpus.

REFERENCES

Bratko, A., B. Filipic, 2005. Spam filtering using compression models. Technical Report IJS-DP-9227: Department of Intelligent Systems, Jozef Stefan Institute, Ljubljana, Slovenia.

Paulo, C., L. Clotilde, S. Pedro, 2009. Symbiotic data mining for personalized spam filtering. Proceedings of the International Conference on Web Intelligence and Intelligent Agent Technology, (IEEE/WIC/ACM): pp: 149–156.

Siefkes, C., F. Assis, S. Chhabra, W.S. Yerazunis, 2004. Combining winnow and orthogonal sparse bigrams for incremental spam filtering. Principles and Practice of Knowledge Discovery in Databases (PKDD'04):Lecture Notes in Computer Science, LNCS 3201, Springer, Berlin, pp: 410–421.

Ray, C. and H. Hunt, 2006. Tightening the net: a review of current and next generation spam filtering tools.Computers and Security, 25(8): 566–578.

Dae-Neung Sohn,Jung-Tae Lee,Kyoung-Soo Han,Hae-Chang Rim., 2012. Content-based mobile spam classification using stylistically motivated features. Pattern Recognition Letters, 33: 364-369.

Evangelos Moustakas, C. Ranganathan and Penny Duquenoy, 2005. Combating spam through legislation A comparative analysis of us and European approaches. In Proceedings of Second Conference on Email and Anti-Spam, CEAS.

Mehrnoush, F.S. and B. Hamid, 2008. Spam detection using dynamic weighted voting based on clustering.Proceedings of the 2nd International Symposium on Intelligent Information Technology Application,(IITA '08): pp: 122–126, Shanghai, China.

Hongrong Cheng, Zhiguang Qin, Chong Fu, and Yong Wang, 2010. Novel Spam Image FilteringFramework with Multi-Label Classification. International Conference on Communications, Circuits and Systems (ICCCAS), pp:282-285.

Wen-Feng, H. and C. Te-Min, 2008. An incremental cluster-based approach to spam filtering. Expert Systems with Application, 34(3): 1599-1608.

Drucker, H., B. Shahrary and D.C. Gibbon, 2002. Support vector machines: relevance feedback and information retrieval. Inform. Process. Manag, 38(3): 305–323.

Islam, R., M. Chowdhury, W. Zhou, 2005. An Innovative Spam Filtering Model Based on Support Vector Machine. Proceedings of the IEEE International Conference on Intelligent Agents, Web Technologies and Internet Commerce, 2(28-30): 348-353.

Androutsopolous, I., J. Koutsias, K.V. Chandrinos, G. Paliouras, C.D. Spyropolous, 2000. An evaluation of Naive Bayesian anti-spam filtering, Proceedings of the Workshop on Machine Learning in the New Information Age. 11th European Conference on Machine Learning: Barcelona, Spain, pp: 9–17.

Ahmed., Kh., 2007. An overview of content-based spam filtering techniques. Informatica, 31(3): 269–277.

Mu-Chun, Su., Hsu-Hsun Lo, Fu-Hau Hsu., 2010. A neural tree and its application to spam e-mail detection. Expert Systems with Applications, 37: 7976-7985.

Muhammad, N. Marsonoa, M. Watheq El-Kharashi, Fayez Gebali, 2009. Targeting spam control on middleboxes Spam detection based on layer-3 e-mail content classification. Computer Networks, 53: 835-848.

Marsono, M.N., M.W. El-Kharashi, F. Gebali, 2008. Prioritized e-mail servicing to reduce non-spam delay and loss - a performance analysis. Wiley International Journal of Network Management, 18(4): 323–342.

Tran, M., G. Armitage, 2004. Evaluating the use of spam-triggered TCP/IP rate control to protect SMTP servers. Proceedings of the Australian Telecommunications Networks and Applications Conference (ATNAC 2004): Sydney, Australia, 2004, pp: 329–335.

Marsono, M.N., M.W. El-Kharashi, F. Gebali, 2009. A spam rejection scheme during SMTP sessions based on layer 3 e-mail classification. Elsevier Journal of Network and Computer Applications, 32(1): 236-257.

Sang, M.L., S.K. Dong and S.P. Jong, 2010. Spam detection using feature selection and parameters optimization. Proceedings of the 4th International Conference on Complex, Intelligent and Software Intensive Systems, (CISIS '10): pp. 883–888, Krakow, Poland.

Rasim, M. Alguliev, M. Ramiz Aliguliyev and A. Saadat Nazirova, 2011. Classification of Textual E-Mail Spam Using DataMining Techniques. Applied Computational Intelligence and Soft Computing.

Twining, R.D., M.M. Williamson, M. Mowbray, M. Rahmouni, 2004. Email prioritization: reducing delays on legitimate mail caused by junk mail. Technical Report HPL-2004-5(R.1): HP Digital Media Systems Laboratory, Bristol, UK.

Clayton., R., 2004. Stopping spam by extrusion detection. Proceedings of the First Conference on Email and Anti-Spam (CEAS): Mountain View, CA, USA.

Sarah Jane Delany, Mark Buckley, Derek Greene, 2012. SMS spam filtering: Methods and data. Expert Systems with Applications, 39: .9899-9908.

Minorum, S. and Sh. Hiroyuki, 2005. Spam detection using text clustering. Proceedings of the International Conference on Cyber worlds, (CW '05): pp: 316–319, Singapore.

Xiao Mang Li, Ung Mo Kim, 2012. A Hierarchical Framework for Content-based Image Spam Filtering. 8th International Conference on Information Science and Digital Content Technology (ICIDT), 1: 149-155.

Wenqing Zhao, Yongli Zhu. 2005. An Email Classification Scheme Based on Decision-Theoretic Rough Set Theory and Analysis of Email Security. Transactions of the IRE Professional Group, pp: 1-6,2005.

Yerazunis., W., 2004. The spam filtering plateau at 99.9% accuracy and how to get past it. Proceedings of the MIT Spam Conference: Cambridge, MA, USA.

Zhang, J., 2003. Modified logistic regression-An approximation to SVM and its applications in large-scale text categorization. Proceedings of the 20th International Conference on Machine Learning: AAAI Press, pp: 888–895.