Machine Learning methods for E-mail Classification

W.A. Awad
Math.&Comp.Sci.Dept., Science faculty, Port Said University

S.M. ELseuofi
Information. System Dept., Ras El Bar High inst.

Abstract
The increasing volume of unsolicited bulk e-mail (also known as spam) has generated a need for reliable anti-spam filters. Using a classifier based on machine learning techniques to automatically filter out spam e-mail has drawn many researchers attention. In this paper we review some of the most popular machine learning methods (Bayesian classification, k-NN, ANNs, SVMs, Artificial immune system and Rough sets) and of their applicability to the problem of spam Email classification. Descriptions of the algorithms are presented, and the comparison of their performance on the SpamAssassin spam corpus is presented.

Keywords
Spam, E-mail classification, Machine learning algorithms.

1. INTRODUCTION
In recent years, spam e-mail became a big trouble over the Internet. Spam known as unsolicited commercial/bulk e-mail, is a bane of e-mail communication. There are many serious problems associated with growing volumes of spam. Spam is not only a waste of storage space and communication bandwidth, but also a waste of time to tackle. Automatic e-mail filtering seems to be the most effective method for countering spam at the moment and a tight competition between spammers and spam-filtering methods is going on: the finer the anti-spam methods get, so do the tricks of the spammers. Only several years ago most of the spam could be reliably dealt with by blocking e-mails coming from certain addresses or filtering out messages with certain subject lines. To overcome these spammers began to specify random sender addresses and to append random characters to the end of the message subject. There are two general approaches to mail filtering: knowledge engineering (KE) and machine learning (ML). When using the knowledge engineering, a set of rules is created according to which messages are categorized as spam or legitimate mail. A set of such rules should be created either by the user of the filter, or by some other authority (e.g. the software company that provides a particular rule-based spam-filtering tool). The major drawback of this method is that the set of rules must be constantly updated, and maintaining it is not convenient for most users. The machine learning approach does not require specifying any rules explicitly [1]. Instead, a set of pre-classified documents (training samples) is needed. A specific algorithm is then used to "learn" the classification rules from this data. The subject of machine learning has been widely studied and there are lots of algorithms suitable for this task. They include Naïve Bayes, support vector machines, Neural Networks, K-nearest neighbor, Rough sets and the artificial immune system.

2. Machine Learning in Spam Management
The Machine Learning field evolved from the broad field of Artificial Intelligence, This aims to mimic intelligent abilities of humans by machines. In the field of Machine Learning one considers the important question of how to make machines able to "learn". Learning in this context is understood as inductive inference, where one observes examples that represent incomplete information about some "statistical phenomenon". In unsupervised learning one typically tries to uncover hidden regularities (e.g. clusters) or to detect anomalies in the data (for instance some unusual machine function, Spam Messages or a network intrusion). In e-mail filtering task some features could be the bag of words or the subject line analysis. Thus, the input to a pattern recognition task can be viewed as a two-dimensional matrix, whose axes are the examples and the features. Pattern classifications tasks are often divided into several sub-tasks are Data collection and representation, Feature selection and/or feature reduction and Classification.
Data collection and representation are mostly problem-specific. Feature selection and feature reduction attempt to reduce the dimensionality (i.e. the number of features) for the remaining steps of the task. Finally, the classification phase of the process finds the actual mapping between patterns and labels (or targets). In the following section we will review some of the most popular machine learning methods.

2.1 Naïve Bayes classifier method
In 1998 the Naïve Bayes classifier was proposed for spam recognition [2, 3]. It is based on the principle that most events are dependent and that the probability of an event occurring in the future can be inferred from the previous occurrences of that event. This same technique can be used to classify spam. If some piece of text occurs often in spam but not in legitimate mail, then it would be reasonable to assume that this email is probably spam. Bayesian spam filtering has become a popular mechanism to distinguish illegitimate spam

email from legitimate email. Nowadays many mail clients implement Bayesian spam filtering. Bayesian filters must be 'trained' to work effectively. Particular words have certain probabilities (also known as likelihood functions) of occurring in spam email but not in legitimate email. For instance, most email users will frequently encounter the word Viagra in spam email, but will seldom see it in other email. Before mail can be filtered using this method, the user needs to generate a database with words and tokens (such as the $ sign, IP addresses and domains, and so on), collected from a sample of spam mail and valid mail (referred to as 'ham'). For all words in each training email, the filter will adjust the probabilities that each word will appear in spam or legitimate email in its database. After training, the word probabilities are used to compute the probability that an email with a particular set of words in it belongs to either category. If the total of word probabilities exceeds a certain threshold, the filter will mark the email as spam. Users can then decide whether to move email marked as spam to their spam folder or whether to just delete them. Here, only two categories are necessary: spam or ham. Almost all the statistic-based spam filters use Bayesian probability calculation to combine individual token's statistics to an overall score [4], and make filtering decision based on the score. Usually, these filters first go through a training stage that gathers statistics of each token. The statistic we are mostly interested for a token T is its spamminess (spam rating) [10], calculated as follows:

$$S[T] = \frac{C_{Spam}(T)}{C_{Spam}(T) + C_{Ham}(T)}$$

Where $C_{Spam}(T)$ and $C_{Spam}(T)$ are the number of spam or ham messages containing token T, respectively. To calculate the possibility for a message M with tokens $\{T_1,\ldots\ldots, T_N\}$, one needs to combine the individual token's spamminess to evaluate the overall message spamminess. A simple way to make classifications is to calculate the product of individual token's spamminess and compare it with the product of individual token's hamminess

$$(H[M] = \prod_{I=1}^{N} (1 - S[T_I]))$$

The message is considered spam if the overall spamminess product S[M] is larger than the hamminess product H[M]. The above description is used in the following algorithm [5]:

```
Stage1. Training
    Parse each email into its constituent tokens
    Generate a probability for each token W
    $S[W] = C_{spam}(W) / (C_{ham}(W) + C_{spam}(W))$
    Store spamminess values to a database

Stage2. Filtering
For each message M
While (M not end) do
   Scan message for the next token Ti
   Query the database for spamminess S (Ti)
   Calculate accumulated message probabilities
         S[M] and H[M]
Calculate the overall message filtering indication by:
         I[M] = f(S[M] , H[M])
         f is a filter dependent function,
   Such as   $I[M] = \frac{1+S[M]-H[M]}{2}$
If I[M] > threshold
      msg is marked as spam
else
      msg is marked as non-spam
```

2.2 K-nearest neighbor classifier method

The k-nearest neighbor (K-NN) classifier is considered an example-based classifier, that means that the training documents are used for comparison rather than an explicit category representation, such as the category profiles used by other classifiers. As such, there is no real training phase. When a new document needs to be categorized, the k most similar documents (neighbors) are found and if a large enough proportion of them have been assigned to a certain category, the new document is also assigned to this category, otherwise not . Additionally, finding the nearest neighbors can be quickened using traditional indexing methods. To decide whether a message is legitimate or not, we look at the class of the messages that are closest to it. The comparison between the vectors is a real time process. This is the idea of the k nearest neighbor algorithm:

Stage1. Training

Store the training messages.

Stage2. Filtering

Given a message x, determine its k nearest Neighbors among the messages in the training set. If there are more spam's among these neighbors, classify given message as spam. Otherwise classify it as legitimate mail.

We should note that the use of an indexing method in order to reduce the time of comparisons induces an update of the sample with a complexity *O(m)*, where m is the sample size. As all of the training examples are stored in memory, this technique is also referred to as a memory-based classifier [6]. Another problem of the presented algorithm is that there seems to be no parameter that we could tune to reduce the number of false positives. This problem is easily solved by changing the classification rule to the following *l/k*-rule:

If *l* or more messages among the *k* nearest neighbors of *x* are spam, classify *x* as spam, otherwise classify it as legitimate mail.

The *k* nearest neighbor rule has found wide use in general classification tasks. It is also one of the few universally consistent classification rules.

2.3 Artificial Neural Networks classifier method

An artificial neural network (ANN), also called simply a "Neural Network" (NN), is a computational model based on biological neural networks. It consists of an interconnected collection of artificial neurons. An ANN is an adaptive system that changes its structure based on information that flows through the artificial network during a learning phase. The ANN is based on the principle of learning by example. There are, however the two classical kind of the neural networks, perceptron and the multilayer perceptron. Here we will focus on the perceptron algorithm. The idea of the perceptron is to find a linear function of the feature vector $f(x) = w^T x + b$ such that $f(x)>0$ for vectors of one class [7], and $f(x) < 0$ for vectors of other class. Here w = (w_1 $w_{2,...}w_m$) is the vector of coefficients (weights) of the function, and b is the so-called bias. If we denote the classes by numbers +1 and -1, we can state that we search for a decision function $d(x)$ = sign ($w^T x + b$) [8]. The perceptron learning is done with an iterative algorithm. It starts with arbitrarily chosen parameters (w_0,b_0) of the decision and updates them iteratively. On the *n-th* iteration of the algorithm a training sample (x,c) is chosen such that the current decision function does not classify it correctly (i.e. sign ($w_n x + b_n$) $\neq c$). The parameters (w_n,b_n) are then updated using the rule:

$$w_{n+1} = w_n + cx \qquad b_{n+1} = b_n + c$$

The algorithm stops when a decision function is found that correctly classifies all the training samples. The above description is used in the following algorithm [9].

```
Stage1. Training
Initialize w and b ( to random values or to 0).
Find a training example (x,c) for which sign( $w^T$ x+ b).
If there is no such example, then training is completed
    Store the final w and stop.
        Otherwise go to next step
      Update (w,b): w := w + cx, b := b + c.
    Go to previous step.

Stage2. Filtering
Given a message x, determine its class as sign $(w^Tx+b)$
```

2.4 Support Vector Machines classifier method

Support Vector Machines are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships [10], the SVM modeling algorithm finds an optimal hyperplane with the maximal margin to separate two classes, which requires solving the following optimization problem.

Maximize

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i , x_j)$$

Subject to

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

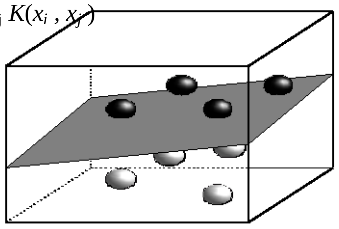

Figure 1 An SVM separating black and white points in 3 dimensions

Where α_i is the weight assigned to the training sample x_1. If $\alpha_i > 0$, x_1 is called a support vector C is a "regulation parameter" used to trade-off the training accuracy and the model complexity so that a superior generalization capability can be achieved. K is a kernel function, which is used to measure the similarity between two samples. A popular radial basis function *(RBF)* kernel functions, as shown in [10].

$$K(x_i , x_j) = exp(-\gamma|| x_i - x_j||^2), \gamma > 0$$

After the weights are determined [11], a test sample x is classified by

$$y = Sign \left(\sum_{i=1}^{n} \alpha_i y_i K(x_i , x_j) \right),$$

$$Sign (a) = \begin{cases} +1, & if\, a > 0 \\ -1, & otherwise \end{cases}$$

To determine the values of < γ, C >, a Cross Validation (CV) process is usually conducted on the training dataset [3]. CV is also used to estimate the generalization capability on new samples that are not in the training dataset. A k-fold CV randomly splits the training dataset into k approximately equal-sized subsets, leaves out one subset, builds a classifier on the remaining samples, and then evaluates classification performance on the unused subset. This process is repeated k times for each subset to obtain the CV performance over the whole training dataset. If the training dataset is large, a small subset can be used for CV to decrease computing costs. The following algorithm [13] can be used in the classification process.

```
Input:  sample x to classify
    Training set $T = \{(x_1,y_1),(x_2,y_2),......(x_n,y_n)\}$;
    Number of nearest neighbors k.
Output:  decision $y_p \in \{-1,1\}$
    Find k sample $(x_i,y_i)$ with minimal values
            of $K(x_i,x_i) - 2 * K(x_i,x)$
      Train an SVM model on the k selected samples
      Classify x using this model, get the result $y_p$
      Return $y_p$
```

2.5 Artificial Immune System classifier method

Biological immune System has been successful at protecting the human body against a vast variety of foreign pathogens. A role of the immune system is to protect our bodies from infectious agents such as viruses, bacteria, fungi and other parasites. On the surface of these agents are antigens that allow the identification of the invading agents (i.e., pathogens) by the immune cells and molecules, thus provoking an immune response Recognition in the immune system is performed by lymphocytes. Each lymphocyte expresses receptor molecules of one particular shape on its surface (called antibody). An elaborate genetic mechanism involving combinatorial association of a number of gene segments underlies the construction of these receptors. The overall immune response involves three evolutionary methods: gene library evolution generating effective antibodies, negative selection eliminating inappropriate antibodies and clonal selection cloning well performing antibodies.

In gene library evolution, antibodies recognize antigens by the complementary properties that belong only to antigens, not self-cells. Thus, some knowledge of antigen properties is required to generate competent antibodies. Because of this evolutionary self-organization process, in spam management the gene libraries act as archives of information on how to detect commonly observed antigens. An important constraint that the immune has to satisfy is not to attack self cells. Negative selection eliminates inappropriate and immature antibodies which bind to self. Clonal selection clones antibodies performing well. In contrast, antibodies performing badly die off after a given lifetime. Thus, according to currently existing antigens, only the fittest antibodies survive. Similarly, instead of having the predefined information about specific antigens, it organizes the fittest antibodies by interacting with the current antigens. The above description is used in the following algorithm [14]:

Artificial Immune System algorithm (an email message m)

```
For (each term t in the message) do {
If (there exists a detector p, based on base
        String r, matches with t) then {
If (m is spam) then {
  Increase r's spam score by s-rate;
} else {
  Increase r's ham score by ns-rate;
}
} else {
If (m is spam) then {
  If (detector p recognizes t and edmf (p, t) >
             threshold) then {
The differing characters are added to its
corresponding entry in the library of
character generalization rules;
    } else {
      A new base string t is added into the
library of base strings;
          }
        }
      }
  Decrease the age of every base string by a-rate;
  }
```

2.6 Rough sets classifier method

In 1982 Rough set (RS) theory was developed by Pawlak. The most advantage of rough set is its great ability to compute the reductions of information systems. In an information system there might be some attributes that are irrelevant to the target concept (decision attribute), and some redundant attributes. Reduction is needed to generate simple useful knowledge from it. A reduction is the essential part of an information system. It is a minimal subset of condition attributes with respect to decision attributes. The Rough set scheme is provided as follows.

Step 1: With the incoming emails, first thing we need to do is to select the most appropriate attributes to use for classification. Then the input dataset is transformed into a decision system L, which is then split into the training dataset (TR) and the testing dataset (TE). A classifier will be induced from the TR and applied to the TE to obtain performance estimation. For TR, do Step 2 and Step 3.

Step 2: Because the decision system has real values attributes, Boolean reasoning algorithm [15] should be used to finish the discretization strategies.

Step 3: Genetic algorithms [16] should be used to get the decision rules. Then For TE, continue to Step 4.

Step 4: First, discretizes the TE employing the same cuts computed from step 2. Then the rules generated in Step 3 are used to match every new object in TE to make decision.

Let $b = 0.15 \in (0, \frac{1}{2})$ be the threshold for positive region, therefore, these b − lower and b − upper approximations divide the whole emails in tree regions, called 0.15−positive, 0.15−boundary and 0.15−negative regions, The algorithm is described as follows:

Input: Dis_T E, RUL,b.

/* Dis_T E: Discretized TE using cuts obtained from step2 and RUL – the rules generated in Step 3. Rel( ) denotes an object x is relevant to non-spam. CER_x denotes the sum predicts number for object x. $b = 0.15 \in (0, \frac{1}{2})$ */

Output: the three categories – non spam, spam and suspicious.

```
For x ∈ Dis_T E do
     While RUL(x) = 0 do
          suspicious = suspicious ∪ {x};
     End
```

Let all r $\in$ *RUL(x)* cast a number in favor of the non-spam class. The number of predicts a rule gets to cast is actually the membership degree based on the decision rules;

R = r $\in$ *RUL(x)*|r predicts non-spam;

Estimate Rel (Dis_T E | x $\in$ non-spam);

```
Rel (Dis_T E | x $\in$ non-spam) = $\sum r \in R$
        Predicts (non-spam);
Certainty = $1/cer$ $\times$ Rel (Dis_T E | $x \in$ non-spam);
While Certainty$_x$ $\geq 1 - b$ do
        suspicious = suspicious $\cup$ {x};
End
        spam = spam $\cup$ {x};
End
```

3. Machine learning methods performance

3.1 Experiment Implementation

In order to test the performance of above mentioned six methods, some corpora of spam and legitimate emails had to be compiled; there are several collections of email publicly available to be used by researchers. SpamAssassin will be used in this experiment, which contains 6000 emails with the spam rate 37.04%. Thus we have divided the corpora into training and testing sets keeping, in each such set, the same proportions of ham (legitimate) and spam messages as in the original example set. Each training set produced contained 62.96% of the original set; while each test set contain 37.04% as Table 1.

Table 1. Corpora of Spam and Ham Messages

Message collection	Training Set	Testing Set
Ham Messages	2378	1400
Spam Messages	1398	824
Total Messages	3776	2224

In addition to the body message of an email, an email has another part called the header. The job of the header is to store information about the message and it contains many fields like the field (From) and (Subject), we decided to divide the email into 3 different parts. The first part is the (Subject) that can be considered as the most important part in the email, it noticed that most of the new incoming emails have descriptive Subjects that can be used to clearly identify whether that email is Spam or Ham. The second part is (From) which is the person that taking the responsibility of the message, this field we store it in a database and use it after the decision of the classifier has been taken, that is the way to compare the field (From) stored in the database to the field (From) in the new incoming email, if they are the same so the decision of the new incoming email is Spam. The third part is the (Body) which is the main part of the message. Furthermore we applied two procedures in the preprocessing stage. Stopping is employed to remove common word. Case-change is employed to change the (Body) into small letters. The experiment is performed with the most frequent words in spam email; we select 100 of them as features.

3.2 Detailed algorithm steps

Step 1: Email preprocessing

The content of email is received through our software, the information is extracted then as mentioned above, then the information (Feature) extracted is saved into a corresponding database. Every message was converted to a feature vector with 21700 attributes (this is approximately the number of different words in all the messages of the corpus). An attribute n was set to 1 if the corresponding word was present in a message and to 0 otherwise. This feature extraction scheme was used for all the algorithms.

Step 2: Description of the feature extracted

Feature extraction module extract the spam text and the ham text, then produce feature dictionary and feature vectors as input of the selected algorithm, the function of feature extraction is to train and test the classifier. For the train part, this module account frequency of words in the email text, we take words which the time of appearance is more than three times as the feature word of this class. And denote every email in training as a feature vector.

Step 3: Spam classification

Through the steps above, we take standard classification email documents as training document, pretreatment of email, extract useful information, save into text documents according to fix format, split the whole document to words, extract the feature vector of spam document and translate into the form of vector of fix format. We look for the optimal classification using the selected algorithm which is constructed using the feature vector of spam documents.

Step 4: Performance evaluation

In order to test the performance of above mentioned six methods, we used the most popular evaluation methods used by the spam filtering researchers. Spam Precision (SP), Spam Recall (SR), Accuracy (A). Spam Precision (SP) is the number of relevant documents identified as a percentage of all documents identified; this shows the noise that filter presents to the user (i.e. how many of the messages classified as spam will actually be spam)

$$SP = \frac{\text{\# of Spam Correctly Classified}}{\text{Total \# of messages classifies as spam}}$$

$$= \frac{N_{spam \rightarrow spam}}{N_{spam \rightarrow spam} + N_{ham \rightarrow spam}}$$

Spam Recall (SR) is the percentage of all spam emails that are correctly classified as spam.

$$SR = \frac{\text{\# of Spam Correctly Classified}}{\text{Total \# of messages}}$$

$$= \frac{N_{spam \rightarrow spam}}{N_{spam \rightarrow spam} + N_{spam \rightarrow ham}}$$

Accuracy (A) is the percentage of all emails that are correctly categorized

$$A = \frac{\text{\# of e-mails correctly categorized}}{\text{Total \# of e-mails}} = \frac{N_{ham\rightarrow ham} + N_{spam\rightarrow spam}}{N_{ham} + N_{spam}}$$

Where $N_{ham\rightarrow ham}$ and $N_{spam\rightarrow spam}$ are the number of messages that have been correctly classified to the legitimate email and Spam email respectively;

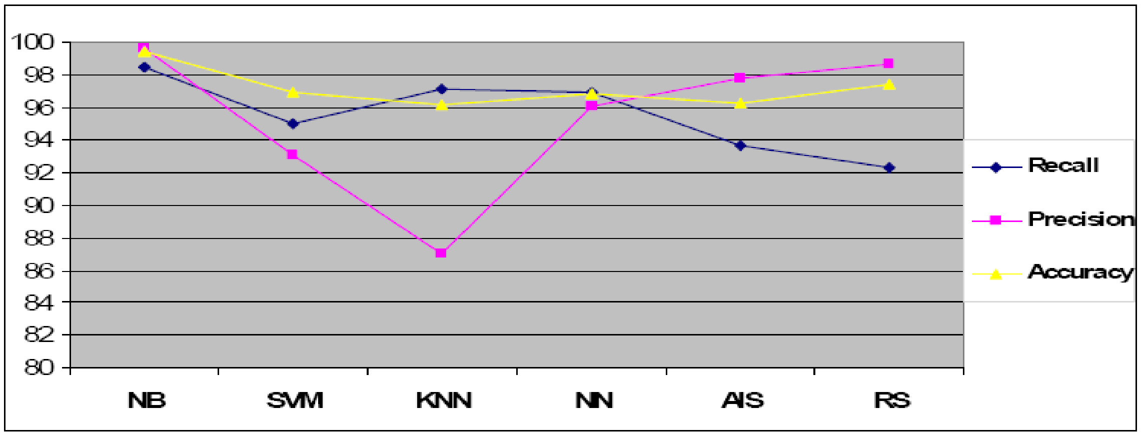

Fig 1: Spam Recall, Precision and Accuracy of six classifiers

$N_{ham\rightarrow spam}$ and $N_{spam\rightarrow ham}$ are the number of legitimate and spam messages that have been misclassified; N_{ham} and N_{spam} are the total number of legitimate and spam messages to be classified.

3.3 Performance Comparison

We summarize the performance result of the six machine learning methods in term of spam recall, precision and accuracy. Table 1 and Figure 2 summarize the results of the six classifiers by selecting the top 100 features (the most relevant word). In term of accuracy we can find that the Naïve bayes method is the most accurate while the artificial immune System and the k-nearest neighbor give us approximately the same lower percentage, while in term of spam precision we can find that the Naïve bayes method has the highest precision among the six algorithms while the k-nearest neighbor has the worst precision percentage and surprisingly the rough sets method has a very competitive percent, and finally we can find that the recall is the less percentage among the six classifiers while the Naïve bayes still has the highest performance but considered low when compared to precision and accuracy while the rough sets has the worst performance.

Table 2. Performance of six machine learning algorithms by selecting top 100 features

Algorithm	Spam Recall (%)	Spam Precision (%)	Accuracy (%)
NB	98.46	99.66	99.46
SVM	95.00	93.12	96.90
KNN	97.14	87.00	96.20
NN	96.92	96.02	96.83
AIS	93.68	97.75	96.23
RS	92.26	98.70	97.42

Performance of the k-nearest neighbor classifier appeared to be nearly independent of the value of k. In general it was poor, and it has the worst precision percentage. The performance of the neural networks was the most simple and fastest algorithm, while the rough sets method is the most complicated and has to be hybrid with the genetics algorithm to get the decision rules. Artificial immune system surprisingly give a very satisfying results which give us a hope to get a better performance when it hybrid with the rough sets method.

4. Conclusion

Spam recall percentage in the six methods has the less value among the precision and the accuracy values, while in term of accuracy we can find that the Naïve bayes and rough sets methods has a very satisfying performance among the other methods, more research has to be done to escalate the performance of the Naïve bayes and Artificial immune system either by hybrid system or by resolve the feature dependence issue in the naïve bayes classifier, or hybrid the Immune by rough sets. Finally hybrid systems look to be the most efficient way to solve the spam challenge nowadays.

5. REFERENCES

[1] M. N. Marsono, M. W. El-Kharashi, and F. Gebali, "Binary LNS-based naïve Bayes inference engine for spam control: Noise analysis and FPGA synthesis", IET Computers & Digital Techniques, 2008

[2] Patrick Pantel and Dekang Lin. Spamcop, "A spam classification & organization program. In Learning for Text Categorization", Workshop. AAAI Technical Report WS-98-05,, 1998

[3] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. "A bayesian approach to filtering junk e-mail. In Learning for Text Categorization", Workshop.AAAI Technical Report WS-98-05, 1998

[4] Guzella, T. S. and Caminhas, W. M. "A review of machine learning approaches to Spam filtering." Expert Syst. Appl., 2009

[5] Li, K. and Zhong, Z., "Fast statistical spam filter by approximate classifications", In Proceedings of the Joint international Conference on Measurement and Modeling of Computer Systems (Saint Malo, France), 2006

[6] Khorsi. "An overview of content-based spam filtering techniques", Informatica, 2007

[7] Muhammad N. Marsono, M. Watheq El-Kharashi, Fayez Gebali "Targeting spam control on middleboxes: Spam detection based on layer-3 e-mail content classification" Elsevier Computer Networks, 2009

[8] Simon Haykin"Neural Networks: A Comprehensive Foundation", Englewood Cliffs, NJ: Prentice-Hall, 2nd ed., 1999

[9] Carpinteiro, O. A. S., Lima, I., Assis, J. M. C., de Souza, A. C. Z., Moreira, E. M., & Pinheiro, C. A. M. "A neural model in anti-spam systems.", Lecture notes in computer science.Berlin: Springer, 2006

[10] V. N. Vapnik, "Statistical Learning Theory" New York: John Wiley and Sons, 1998.

[11] L. Breiman, "Random forests, Machine Learning", Springer, 2001

[12] Yuchun Tang, Sven Krasser, Yuanchen He, Weilai Yang, Dmitri Alperovitch"Support Vector Machines and Random Forests Modeling for Spam Senders Behavior Analysis" IEEE GLOBECOM, 2008

[13] Hao Zhang, Alexander C. Berg, Michael Maire, and Jitendra Malic. "SVM-KNN: Discriminative nearest neighbour classification for visual category recognition", IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006

[14] Wu, C. "Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks" Expert Syst., 2009

[15] Skowron, A. and Son, N., "Boolean Reasoning Scheme with Some Applications in Data Mining", In: Proc. Principles of Data Mining and Knowledge Discovery PKDD'99, Prague, Czech Republic, LNAI 1704, Springer Verlag, Berlin, 1999

[16] Wrblewski, J. "Finding Minimal Reducts Using Genetic Algorithms", Proc. of the Second Annual Joint Conference on Information Sciences. Wrightsvillle Beachm, NC, 1995