Seventh IEEE/ACIS International Conference on Computer and Information Science

Email Categorization Using (2+1)-tier Classification Algorithms

Md. Rafiqul Islam, Wanlei Zhou and Morshed U. Chowdhury
School of Engineering and Information Technology, Deakin University, Australia.
{rmd,wanlei,muc}@deakin.edu.au

Abstract

In this paper we have proposed a spam filtering technique using (2+1)-tier classification approach. The main focus of this paper is to reduce the false positive (FP) rate which is considered as an important research issue in spam filtering. In our approach, firstly the email message will classify using first two tier classifiers and the outputs will appear to the analyzer. The analyzer will check the labeling of the output emails and send to the corresponding mailboxes based on labeling, for the case of identical prediction. If there are any misclassifications occurred by first two tier classifiers then tier-3 classifier will invoked by the analyzer and the tier-3 will take final decision. This technique reduced the analyzing complexity of our previous work [11,12]. It has also been shown that the proposed technique gives better performance in terms of reducing false positive as well as better accuracy.

Key words: email, false positive, (2+1)-tier, classifier.

1. Introduction

Over the last decade, unsolicited bulk email which is called spam has become a major problem for email users. It is used daily by millions of people to communicate around the globe and is a mission-critical application for many businesses. An overwhelming amount of spam is flowing into users' mailboxes daily. Many different approaches for fighting spam have been proposed. A promising approach is the use of content-based filters, capable of discerning spam and legitimate email messages automatically. Machine learning methods are particularly attractive for this task, since they are capable of adapting to the evolving characteristics of spam, and data is often available for training such models. Unlike most text categorization tasks, the cost of misclassification is heavily skewed. Labeling a legitimate email as spam, usually referred to as a *false positive*, carries a much greater penalty than vice-versa. Practically the users are much more concerned about legitimate email than about receiving a few spam emails.

Keeping this in mind, we have proposed a spam filtering approach using (2+1)-tier filtering using different well known classification algorithms. Actually we used two tier classifiers i.e. tier-1 and tier-2 classifier, for categorizing email. If any of the tiers failed to predict with identical labeling then tier-3 classifier will be invoked. In that case the tier-3 labeling will be the final decision for categorizing emails. This approach reduces the FP problems substantially as well as reduces analysing complexity proposed in [11].

The organization of this paper is as follows: Section 2 will describe the related work for spam filtering and section 3 will describe the proposed technique and its detail description. Section 4 presents the algorithm of tier-3 classification technique. Section 5 gives experimental results. Finally, the paper ends with conclusion and references in section 6 and 7 respectively.

2. Related work

Spam will typically have a distinctive content, which should be easy to distinguish from legitimate e-mail. Categorising e-mail based on its content seems like a logical progression from simplistic rule based approaches. This would help reduce error rates as legitimate e-mail would not be blocked even if the ISP (Internet Service Provider) from which it originated, is on a real-time block list. In addition, the presence of a single token should not cause the e-mail to be classified as spam.

This section describes the overview of classification algorithms such as SVM (support vector machine), NB (Naive Bayes) and Boosting, which are used in our proposed model. Each algorithm can be viewed as searching for the most appropriate classifier in a search space that contains all the classifiers it can learn. Classification algorithm needs instance representation and the instances are messages. Each message is transformed into a vector ($x_1, \dots, x_m$), where $x_1, \dots, x_m$ are the values of the attributes $X_1, \dots, X_m$, much as in the vector space model in information retrieval [1,2,3]. In the simplest case, each attribute represents a single token (e.g., "money"), of Boolean variables:

$$X_i = \sum\nolimits_{0-Otherwise}^{1-Contains\_Tokens} \quad (1)$$

Instead of Boolean attributes, another two attribute vector representations such as frequency

978-0-7695-3131-1/08 $25.00 © 2008 IEEE
DOI 10.1109/ICIS.2008.96

attributes and n-gram attributes are considered here [4,10,11,12].

Support vector machine (SVM) is a new learning algorithm which has some attractive features, such as eliminating the need for feature selections, which makes for easier spam classification. SVMs are a range of classification and regression algorithms that have been based on the Structural Risk Minimization (SRM) principle from statistical learning theory formulated by Vapnik [2,6,9,10,11,12]. The SRM is to find an optimal hyperplane that can guarantee the lowest true error. The key concepts of SVMs are the following: there are two classes, $y_i \in \{-1,1\}$, and there are N labelled training examples : $\{x_1, y_1), \ldots, (x_n, y_n), x \in R^d$ where d is the dimensionality of the vector.

SVM is based on the idea that every solvable classification problem can be transformed into a linearly separable one by mapping the original vector space into a new one, using non-linear mapping functions. More formally, SVMs learn generalized linear discriminant functions of the following form:

$$f(\vec{x}) = \sum_{i=1}^{m'} w_i . h_i(\vec{x}) + w_0 \qquad (2)$$

where m′ is the dimensionality of the new vector space, and $h_i(\vec{x})$ are the non-linear functions that map the original attributes to the new ones. The higher the order of the $h_i(\vec{x})$ functions, the less linear the resulting discriminant. The type of $h_i(\vec{x})$functions that can be used is limited indirectly by the algorithm's search method, but the exact choice is made by the person who configures the learner for a particular application. The function $f(\vec{x})$ is not linear in the original vector space, but it is linear in the transformed one.

The Naive Bayes (NB) learner is the simplest and most widely used algorithm that derives from Bayesian Decision Theory [4,7,8,10,11,12]. A Bayesian classifier is simply a Bayesian network applied to a classification task. It contains a node C representing the class variable and a node X_i for each of the features. From Bayes' theorem and the theorem of total probability $P(C = c_k \mid X = x)$ for each possible class c_k, the probability that a message with vector $\bar{x} = (x_1, \ldots, x_m)$ belongs in category c is:

$$P(C=c \mid \vec{X}=\vec{x}) = \frac{P(C=c)P(\vec{X}=\vec{x} \mid C=c)}{\sum_{c' \in \{c_L, c_S\}} P(C=c')P(\vec{X}=\vec{x} \mid C=c')}. \qquad (3)$$

The boosting algorithms, like SVMs, learn generalized linear discriminates of the form of equation

$$f(\vec{x}) = \sum_{i=1}^{m'} w_i . h_i(\vec{x}) + w_0 \qquad (4)$$

In boosting algorithms, however, the mapping functions $h_i(\vec{x})$are themselves learnt from data by another learning algorithm, known as weak learner. A common weak learner is decision stump induction [5,9,10,11,12], which constructs a one-level decision tree that uses a single attribute from the original attribute set to classify the instance $\vec{x}$ to one of the two categories. In the case of continuous attributes, the decision tree is a threshold function on one of the original attributes.

Furthermore, the mapping functions $h_i(\vec{x})$ are learnt by applying iteratively (for $i = 1, \ldots, m'$) the weak learner, in our case regression stump induction, to an enhanced form of the training set, where each training instance $\vec{x}_j$ carries a weight $v_i(\vec{x}_j)$. At each iteration, the weights of the training instances are updated, and, hence, applying the weak learner leads to a different mapping function $h_i(\vec{x})$. This iterative process is common to all boosting methods, where each $h_i(\vec{x})$ can be thought of as a weak classifier that specializes in classifying correctly training instances that the combination of the previous weak classifiers $h_i(\vec{x}_k)(k = 1, \ldots, i-1\}$ either misclassifies or places close to the classification boundary. This is similar to the behaviour of SVMs, which focus on instances that are misclassified or support the tangential hyper planes [4,5,6,7,10.11.12].

3. Proposed (2+1)-tier classification technique for spam filtering.

In this section, the proposed technique of (2+1)-tier filtering system has been illustrated. In every tier we have proposed different classifier with serial procedural approach. We have been investigated on different individual classifiers and found that the output of different individual classifier varies one another with same email corpora. Sometimes one particular classifier gives good result but not other one and vice versa. It has also been shown that some classifier gives good result for particular data sets but not in other data sets. It is because the spam data are dynamic rather than static because the spammers are always changing the strategy to sending spam. Considering the above we have proposed our (2 + 1)-tier filtering system using three different well known classifiers. Graphically the architecture of our proposed (2+1)-tier filtering system is demonstrated in figure 1.

3.1 Description of the proposed system

Figure 1 shows the block diagram of (2+1)-tier classification. In this approach, firstly email message $\mathcal{M}$ passes to the base classifier (C_1) which is called "*TierOneClassifer*" and identified the label of the messages either $\mathcal{M}l_1$ or $\mathcal{M}s_1$. The labelled message then passes to the tier-2 classifier, known as "*TierTwoClassifer*", and the message again labelled by any of the four combinations:

$$\begin{aligned} &\text{i)} && \mathcal{M}l_1 \cap \mathcal{M}l_2 \\ &\text{ii)} && \mathcal{M}l_1 \cap \mathcal{M}s_2 \\ &\text{iii)} && \mathcal{M}s_1 \cap \mathcal{M}l_2 \ \& \\ &\text{iv)} && \mathcal{M}s_1 \cup \mathcal{M}s_2 \end{aligned} \tag{5}$$

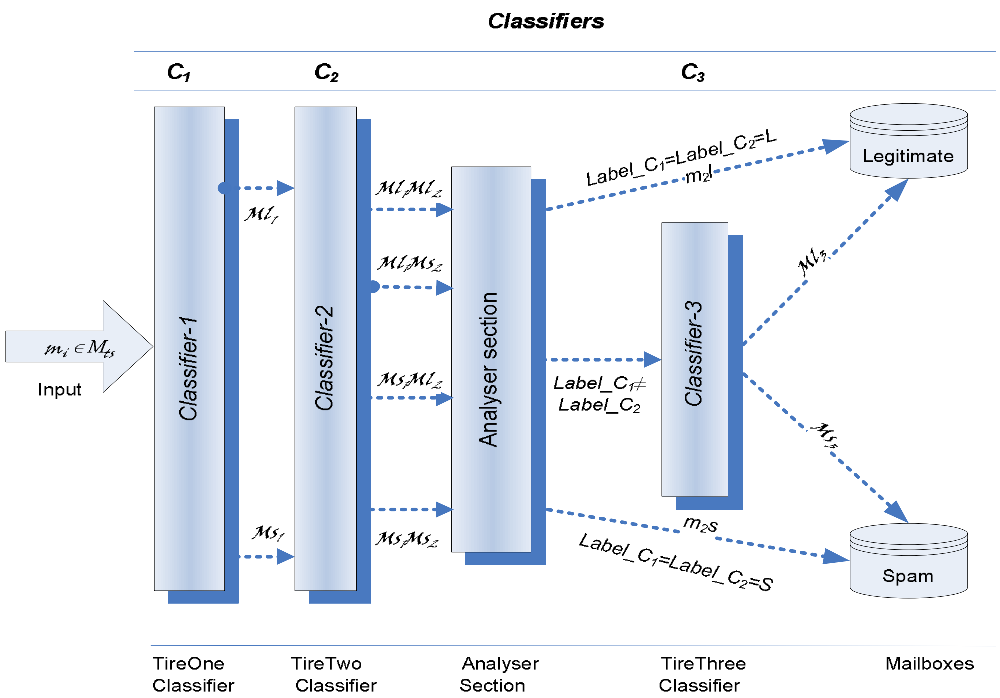

Figure 1. Block diagram for (2+1)-tire spam filtering system.

After tier-2 classification, the classifiers output will appear to the analyser section. In this section the analyser will analyse the output message based on the identification of tier-1 and tier-2 classifiers whether it needs further classification using tier-3 classifier or not. If tier-1 and tier-2 classifier identified the message with indistinguishable label, i.e. the combination of i & iv in equation 5, then the message does not need to classify further using tier-3 classifier. The analyser will send it to the corresponding mailboxes according to the recognition label recognized by the classifiers. On the other hand, if tier-1 and tier-2 classifier identified the message with dissimilar label then the analyser will send this message to the tier-3 classifier, known as "*TierThreeClassifer*".

From the above figure 1, only two combinations, (i.e. the combination of *ii* & *iii* in equation 5), of the output from tier-2 message will appear to the tier-3 classifier. In tier-3 the message will again labelled either $\mathcal{M}l_3$ or $\mathcal{M}s_3$. In this stage the message will directly store to the corresponding mailboxes based on the identification of the tier-3 classifier. There is no comparison with previous label given by the classifiers.

The total number of output email E_{out} from tier-1 classifier can be represents mathematically as $E_{out} \Rightarrow n\,(\mathcal{M}l_1 \cup \mathcal{M}s_1)$. But in the case of tier-2 classifier, the outputs can be categorized following three different sets, which is graphically demonstrated in figure 2:

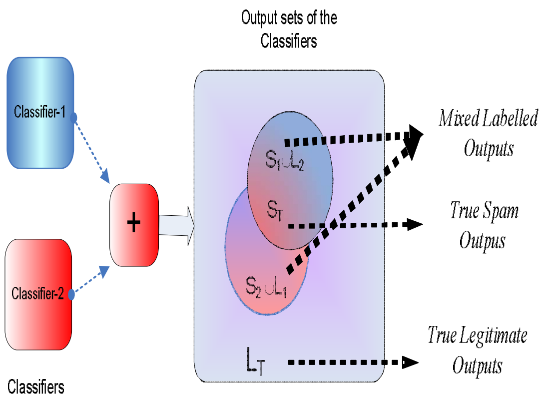

Figure 2. Output sets of tire-2 classifier.

True legitimate outputs L_T: This is the common legitimate output from tier-1 and tier-2 classifier and this type of output is considered as true positive (TP). Mathematically the number of outputs can be represented as L_T =>n $(\mathcal{M}l_1 \cap \mathcal{M}l_2)$.

True spam outputs S_T: This is the common spam output from tier-1 and tier-2 classifier. This sort of output is considered as true negative (TN). The total number of S_T => n $(\mathcal{M}s_1 \cap \mathcal{M}s_2)$

Mixed labelled output: These are the mixed outputs from tier-1 and tier-2 classifiers, which mean any of the classifier truly classified but another is misclassified. These sorts of output are considered neither true positive nor true negative. The total number of outputs are as $E_{out(mixed)}$=>n$(\mathcal{M}l_1 \cap \mathcal{M}s_2)$+n$(\mathcal{M}s_1 \cap \mathcal{M}l_2)$.

For the case of mixed labelled outputs, the tier-3 classifier will be invoked for further classification. Whatever the pronouncements comes from tier-3 classifier, the emails goes to the corresponding mailboxes based on the label of the tier-3 classifier. The reason behind this approach is that any of the two tiers classifier among the 3-tier classifiers, the decisions are unique. So, there is less probability to misclassification.

4. Algorithms for (2+1)-tier classification

This section illustrated the algorithms of our proposed (2+1)-tier filtering system.

Algorithm : 3-tier filtering

1. INPUT: Messages $\mathcal{M}(=\mathcal{M}l \cup \mathcal{M}s)$
2. OUTPUT: $\mathcal{M}_l$ & $\mathcal{M}_s$
3. Split $\mathcal{M}$ into two sets; training set $\mathcal{M}_{tr}s=\mathcal{M}l \cup \mathcal{M}s)$ and test set $\mathcal{M}_{ts}(=\mathcal{M}l \cup \mathcal{M}s)$
4. Train all classifiers, $\mathcal{F}_i$ $(i=1..3)$, using $\mathcal{M}_{tr}(=\mathcal{M}l \cup \mathcal{M}s)$
5. Index all $\mathcal{M}_{ts}(=\mathcal{M}l \cup \mathcal{M}s)$ messages// only //test sets
6. **begin**
7. **for** $m_i \in \mathcal{M}_{ts}$ **do**
8. **array** *MesTag[1..3] of* **string**;*// array //variable for storing the message //label*
9. T_1, T_2, T_3 *Boolean; // Three variables //for identifying the classifier*
10. $T_1 \leftarrow$*True;* $T_2=T_3=$*False;i* $\leftarrow$*1;//initialize //the variables*
11. **repeat**
12. *MesTag[i]* $\leftarrow$ *ThreeTierClassification*(m_i, *i*)
13. **if** T_2 **then**
14. **if** *MesTag[i-1]=MesTag[i]* **then**
    MessageStore(*MesTag[i]*, m_i);
    exit();
    else
    T_2=False; T_3=True;
    endif
    endif
15. **if** T_3 ***then***
    MessageStore(*MesTag[i]*, m_i);
16. **endif**
17. $i \leftarrow i+1$;
18. **until** (i<3)
19. **endfor**
20. **end**

Function: ThreeTierClassification

1. **function** *ThreeTierClassification* (message m, int tier) as string
2. **begin**
3. **if** tier=1 **then**
4. **return** *TierOneClassifer(m);*
5. **elseif** tier=2 **then**
6. **return** *TierTwoClassifer(m);*
7. **else**
8. **return** *TierThreeClassifier(m);*
9. **end**

Procedure : MessageStore

1. **procedure** MessageStore(String Label, message m)
2. **if** Label = *L* **then**
3. MailBox(*L*) $\leftarrow m_i$;
4. **else**
5. MailBox(*S*) $\leftarrow m_i$;
6. **endif**
7. **end**

5. Experimental Results

The experimental result of our proposed (2+1)-tier filtering system has been presented here. We have used three classification algorithms such as, tier-1 as support vector machine (SVM), tier-2 as Boosting (AdaBoost) and tier-3 as Bayesian (Naïve Bayas) in our simulation. We have monitored the outputs of every tier classifiers in our simulation and compared it to its previous tier. Finally a comparative analysis has been shown with tier-1-2-3 outputs. In our experiment, we have used the public data sets PU1-2-3[3] for our experiments and converted the data sets based on our experimental design and environment. Firstly we have encoded the whole data sets both train and test sets, then indexed every email for test data sets and finally recorded the output according to the index value.

5.1 Tier-1 classification

Table 1 shows the tier-1 classifier outputs. It has been shown that the average TP of this tier is 0.74283 (~74%) and TN is 0.9093 (~91%). There is lots of misclassified emails because the average rate of false positive is 0.136 (~14%) and false negative is 0.181(~18%) and the final accuracy achieved 0.8781 (~88%) which is lower in the case of spam filtering.

Table 1. Shows tier-1 classification results

	TP	FP	TN	FN	Acc
PUD1	1	0	0.90909	0.091	0.954555
PUD2	0.909	0.091	1	0	0.9545
PUD3	0.091	0.181	1	0	0.85771
PUD4	0.819	0.181	0.909	0.091	0.864
PUD5	0.819	0.181	0.819	0.181	0.819
PUD6	0.819	0.181	0.819	0.181	0.819
AVG	0.74283	0.136	0.90935	0.0907	0.8782

5.2 Tier-2 classification

Table 2 shows the result of tier-2 classifier. It has been shown that the average TP of this tier is 0.9545 (~95%) and TN is 0.96967 (~97%) which is better than the output of tier-1 classifier as shown in Table 1. On the other hand, the average false positive rate is 0.046 (~4.6%) and false negative is 0.03(~3%) which is much lower compared to tier-1 classifier, as shown in Table 1. The final accuracy achieved in tier-2 is 0.962083 (~96%) which is much higher compared to tier-1 accuracy. In tier-2 classifier there are some misclassified emails which are not considered here for calculating confusing matrix. It is therefore, the output of tier-2 shows significant performance compared to tier-1 which is more convincing.

Table 2. Shows tier-2 classification results

	TP	FP	TN	FN	Acc
PUD1	1	0	0.909	0.091	0.9545
PUD2	1	0	1	0	1
PUD3	0.909	0.091	0.909	0.091	0.909
PUD4	1	0	1	0	1
PUD5	0.909	0.091	1	0	0.9545
PUD6	0.909	0.091	1	0	0.9545
AVG	**0.9545**	**0.046**	**0.96967**	**0.03**	**0.96208**

5.3 Tier-3 classification

The Table 3 shows the tier-3 classifier outputs. It is to be noted that the tier-3 classifier will be invoked only when the different result comes from the analyser based on the output of tier-1 and tier-2. The tier-3 classifier output will be the final prediction of those emails. From Table 3, it has been shown that the average TP of this tier is 1.0 (~100%) and FP is 0 (~0%) which is a significant performance of our experiment. Zero FP is a substantial performance considered in spam filtering technique. Because one misclassified legitimate email may cause a huge problem for the user. Furthermore, there is lots of misclassified emails in the false negative side which is 0.075166(~7.5%) and the true negative is 0.92467 (~92.46 %). So the final accuracy achieved 0.96242 (~96%) which is almost similar to tier-2 outputs. But only the difference we achieved using tier-3 is lower false positive and higher true positive. It is to be noted here that in tier-2 classifier we did not mentioned the misclassified emails which plays an important role in tier-3 results.

Table 3. Shows tier-3 classification results

	TP	FP	TN	FN	Acc
PUD1	1	0	0.909	0.09	0.95498
PUD2	1	0	1	0	1
PUD3	1	0	0.819	0.181	0.9095
PUD4	1	0	1	0	1
PUD5	1	0	0.91	0.09	0.955
PUD6	1	0	0.91	0.09	0.955
AVG	**1**	**0**	**0.92467**	**0.07517**	**0.96242**

5.4 Accuracy curve for (2+1)-tier classifications

Figure 3 shows the final accuracy of our experiment. It has been shown that the average accuracy of our proposed system (~96.242 %) is always better compared to existing filtering techniques [3,5]. It is shown that the tier-2 accuracy is much better than tier-1 accuracy but tier-2 and tier-3 accuracy is almost similar. But in tier-3 result we have achieved lower FP rate.

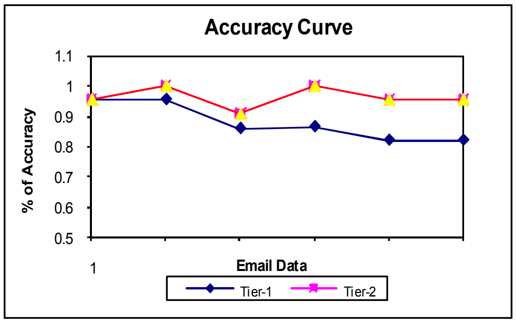

Figure 3. Final accuracy of (2+1)-tier classifier

6. Conclusion

This paper presents an innovative technique for filtering spam using (2+1)-tier filtering approach. In our proposed filtering technique, emphasis has been given to reduce the FP problems based on different aspects of anti-spam filtering and reducing the analysing complexity proposed in [11]. It has been shown that many machine learning techniques for spam filtering can achieve very high accuracy with some amount of FP tradeoffs which are generally expensive in real world. Our experimental result proves the success of our proposed technique in terms of reducing FP and minimizing the complexity of analyser proposed in [11]. However, there is also some complexity in the analyser section which reduces the processing speed. In our future work, we will analyse it and also analyse the rearranging the classifier among the classification tiers.

7. Reference

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

[2]. M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A bayesian approach to filtering junk e-mail. In Learning for Text Categorization: Papers from the Workshop, Madison, Wisconsin, AAAI Technical Report WS-98-05, 1998.

[3]. Androutsopoulos, I., et.,al,. Learning to filter spam e-mail: A comparison of a Naive Bayesian and a memory-based approach. In Proceedings of the Workshop on Machine Learning and Textual Information Access, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases. Lyon, France, 1–13, 2000.

[4]. H. Drucker, B. Shahrary and D. C. Gibbon, "Support vector machines: relevance feedback and information retrieval," Inform. Process. Manag. 38, 3, 305–323, 2003.

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

[6]. Cohen, W. and Singer, Y. Context-sensitive learning methods for text categorization. ACM Transactions on Information Systems 17, 2, pp. 141–173, 1999.

[7]. Cristianini, N. and Shawe-Taylor, J.. An introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press, 2000.

[8]. Kaitarai, H., Filtering Junk e-mail: A performance comparison between genetic programming and naïve bayes, Tech. Report, Department of Electrical and Computer Engineering, University of Waterloo, November 1999

[9]. Liu Huan and Yu Lei, Toward Integrating Feature Selection Algorithms for Classification and Clustering, IEEE Transaction on Knowledge and Data Engg. Vol. 17, No. 4, 2005.

[10]. Islam, M. and Chowdhury, M. (2005) "Spam Filtering Using Ml Algorithms, *The Proceedings of the IADIS International Conference WWW/Internet, International Association for Development of the Information Society Press*, USA. 2005, pp. 419-426.

[11]. Islam, M. and Zhou, W. (2007). "Architecture of Adaptive Spam Filtering Based on Machine Learning Algorithms" *Springer-Verlag, ICA3PP 2007, LNCS* 4494, 2007, pp. 458–46.

[12]. Islam, M., Zhoui, W. And Choudhury, M. (2007) Dynamic feature selection for spam filtering using Support Vector Machine, *IEEE/ACIS ICIS'07, pp (757-762)*.