12

Modeling Spammer Behavior: Artificial Neural Network vs. Naïve Bayesian Classifier

Md. Saiful Islam[1] and Md. Rafiqul Islam[2]
[1]*University of Dhaka,*
[2]*Deakin University,*
[1]*Bangladesh*
[2]*Australia*

1. Introduction

The exponential growth of spam emails in recent years is a fact of life. Internet subscribers world-wide are unwittingly paying an estimated €10 billion a year in connection costs just to receive "junk" emails, according to a study undertaken for the European Commission. Though there is no universal definition of spam, unwanted and unsolicited commercial email as a mass mailing to a large number of recipients is basically known as the junk email or spam to the internet community. Spams are considered to be potential threat to Internet Security. Spam's direct effects include the consumption of computer and network resources and the cost in human time and attention of dismissing unwanted messages. More importantly, these ever increasing spams are taking various forms and finding home not only in mailboxes but also in newsgroups, discussion forums etc without the consent of the recipients. Overflowing mailboxes are overwhelming users, causing newsgroups and discussion forums to be flooded with irrelevant or inappropriate messages. As a consequence, users are getting discouraged not to use them anymore though these systems can provide numerous benefits to them.

Combating spam is a difficult job contrast to the spamming. Millions of spammers around the world are engaged in spreading spams with ever changing tricks and tactis to circumvent the filters deployed by the mailbox providers. As spammers are paid per volume for thier job, they invest thier best effort in reaching everyone by all possible ways. No antispamming technique is hundred percent accurate for spam problem. Antispamming techniques try to make a trade-off between rejecting legitimate e-mail vs. not rejecting all spam, and the associated costs in time and effort.

The simplest and most common approaches are to use filters that screen messages based upon the presence of common words or phrases common to junk e-mail. Other simplistic approaches include *blacklisting* and *whitelisting*.

- *Blacklisting* technique automatically rejects messages received from the addresses of known spammers.
- *Whitelisting* accepts messages received from known and trusted correspondents only.

The major flaw in the first two approaches is that it relies upon complacence by the spammers by assuming that they are not likely to change (or forge) their identities or to alter the style and *vocabulary* of their sales pitches. *Whitelisting* risks the possibility that the

recipient will miss legitimate e-mail from a known or expected correspondent with a heretofore unknown address, such as correspondence from a long-lost friend, or a purchase confirmation pertaining to a transaction with an online retailer. A detail explanation of these techniques is given in (Islam & Chowdhury, 2005). Ramachandran et al. (2007) propose a new technique called *behavioral blacklisting*, which complements existing blacklists by categorizing spammers based on *how* they send email, rather than the IP address (or address range) from which they are sending it. The intuition behind their idea is that, while IP addresses are ephemeral as identifiers, spam campaigns, spam lists, and spamming techniques are more persistent. If one can identify email-sending patterns that are characteristic of spamming behavior, then she can continue to classify IP addresses as spammers even as spammers change their IP addresses.

Machine learning algorithms namely Naïve Bayesian classifier, Decision Tree induction, Artificial Neural Network and Support Vector Machines, based on keywords or tokens extracted from the e-mail's Subject, Content-Type Header and Message Body, have been used successfully in the past (Aery & Chakravarthy, 2005 ; Drucker et al., 1999; Eichler, 2005; Islam & Chowdhury, 2005). Very soon they fall short to filter out spam emails as the spammer changing themselves in the ways that are very difficult to model by simple keywords or tokens (Stuart et al., 2004). The tactics the spammer uses follow patterns and these behavioral patterns can be modeled to combat spam. Actually the more they try to hide, the easier it is to see them (Stuart et al., 2004). Now the question is:

Ques 1. Are the patterns that the spammers follow common to all?
Ques 2. If the spammers follow patterns to spread spams, is it possible to track those patterns?
Ques 3. If one can track the common spammer patterns, is it possible to model them?
Ques 4. Is it possible to model common spammer patterns by machine learning approaches?
Ques 5. What level of accuracy is possible to achieve if one apply machine learning approaches?

Many researchers observe that spammers follow patterns. These patterns can be discovered from many different places: from email corpus (Stuart et al., 2004) by analysing their contents, network-level behavioral patterns (Ramachandran et al., 2006; Sperotto et al., 2009), transfer pattern during transfer sessions (Zhang et al., 2006), resource usage patterns (Xu et al., 2010) and spammer behavior in terms of the chain of machines they use to deliver their messages (Guerra et al., 2009).

This study investigates the possibilities of modeling spammer behavioral patterns instead of vocabulary as features for spam email categorization and these behavioral patterns are discovered by analysing email corpus *Subject*, *Content-Type Header* and *Message Body*. The two machine learning algorithms *Naïve Bayesian Classifier* and *Artificial Neural Networks* are experimented to model common spammer patterns and both of them achieve a promising detection rate that can be considered as an improvement of performance compared to the keyword-based contemporary filtering approaches.

2. Methodology

The success of machine learning algorithms in *text categorization* (TC) has led researchers to investigate learning algorithms for filtering spam emails (Sebastiani, 2002). The central purpose of learning is to accurately predict unseen data. There are two types of learning

strategy and those are: *supervised* and *unsupervised*. In *supervised learning* both the data objects and their labels are given, but in unsupervised learning only data objects are provided. *Naïve Bayesian classifier*, *Decision Tree induction*, *Artificial Neural Network* and *Support Vector Machines* are *supervised learning* algorithms. Clustering is a good example of *unsupervised learning* algorithm.

To learn the machine both training and test data are needed. Training data are used to build the model and then test data are used to measure the effectiveness of the model. In *n-fold cross validation* technique initial data are divided into *n* subsets. From these *n* subsets, *n*-1 subsets are used to train the model and the remaining subset is used to test the model. This process continues *n* times and average performance is taken to judge the effectiveness of the model.

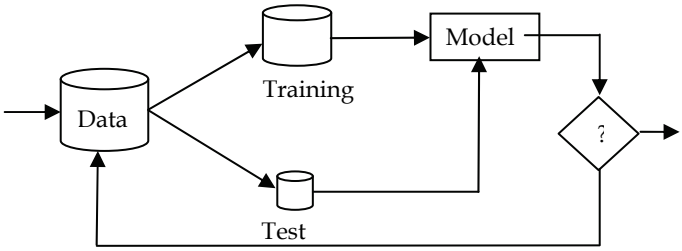


Fig. 1. Learning model

The following two *supervised* machine learning algorithms are exploited to model spammer tricks and techniques in this study.

2.1 Artificial Neural Network

Artificial neural networks (ANN) are non-linear statistical data modeling tools that tries to simulate the functions of biological neural networks. It consists of interconnected collection of simple processing elements or artificial neurons and processes information in a connectionist approach to computation (Han & Kamber, 2001; Stuart et al., 2004). ANN is generally considered to be an adaptive system that changes its structure in response to external or internal information that flows through the network during the learning phase. Fig. 2 shows an example of *multilayer feed forward neural network* (MFFNN).

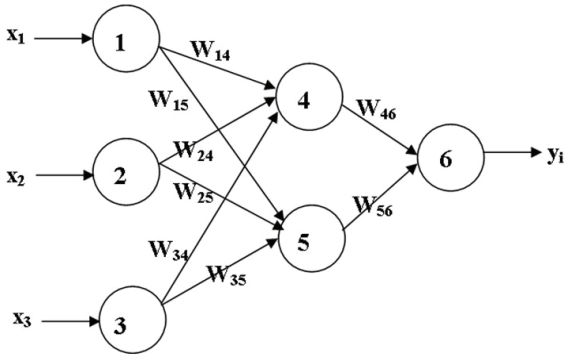


Fig. 2. An example of a multilayer feed-forward artificial neural network

In MFFNN nodes represent neurons are denoted by circle, edges represent connections between neurons are directed edges and labelled with corresponding weight. Nodes in the *input layer* receive inputs (x_i) from external world and nodes in the *output layer* give output (y_i) to the user. Nodes in the *middle layers* (generally called *hidden layers*) receive inputs from the previous layer and deliver it to the next layer. The output of a node in an *artificial neural network* is computed by the equation given below:

$$y_j = \sum_{i=1}^{n} w_{ij} x_i + \theta_j$$

where y_j is the value that will be passed to the next layer from node j, n is the number of incoming edges to node j, x_i's are the inputs coming from previous layer to node j and θ_j is the bias for node j.

The *topology* of an artificial neural network is defined by the number of layers in it and the number of nodes that appear in each layer. Once the topology is given of an artificial neural network, the learning task is to assign proper weight in each connection or edge so that it can correctly identify unknown data. This can be done by the most popular method for learning in multilayer networks: *backpropagation*. In *backpropagation* neural network learning algorithm, initially the weights and the biases are assigned randomly in the range [0, 1]. Then one of the example cases is applied to the network and the network produces some output based on the current state of its synaptic weights. This output is compared to the known-good output and a mean-squared error is calculated. The error value is then propagated backwards through the network and changes are made to the weights in each layer. The whole process is repeated for each of the example cases, then back to the first case again, and so on. The cycle is repeated until the overall error value drops below some pre-determined threshold. After then the network assumed to learn the problem well enough. The pseudo code for backpropagation artificial neural network learning algorithm is given below:

Step 1: Randomly initialize the weights and the biases in the network
Step 2: For each example e in the training set
// forward pass
O = neural-net-output (network, e)
T = teacher output for e
Calculate error (T - O) at the output units

// backward pass
Compute delta_wh for all weights from hidden layer to output layer
Compute delta_wi for all weights from input layer to hidden layer
Update the weights in the network

Step 3: Continue step 3 until all examples classified correctly or stopping criterion satisfied
Step 4: Return the network

The detail about *backpropagation* learning can be found in (Han & Kamber, 2001; Rojas, 1996).

2.2 Naïve Bayesian classifier

Bayesian classifier, the simplest and most widely used for filtering spams, is based on the so-called Bayes' theorem. For a training e-mail E, the classifier calculates for each category,

the probability that the e-mail should be classified under C_i, where C_i is the i^{th} category, making use of the law of the conditional probability:

$$P(C_i \mid E) = \frac{P(C_i)P(E \mid C_i)}{P(E)}$$

where $P(C_i)$ is the prior probability of hypothesis C_i; $P(E)$ is the prior probability of training data E; $P(C_i \mid E)$ is the probability of C_i given E and $P(E \mid C_i)$ is the probability of E given C_i. Assuming class conditional independence, that is, the probability of each word in an e-mail is independent of the word's context and its position in the e-mail, $P(E \mid C_i)$ can be calculated as the product of each individual word W_j's probabilities appearing in the category C_i (W_j being the j^{th} of l words in the e-mail):

$$P(E \mid C_i) = P(w_1 \mid C_i)P(w_2 \mid C_i)...P(w_l \mid C_i)$$

$$= \prod_{j=1}^{l} P(w_j \mid C_i)$$

The category maximizing $P(C_i \mid E)$ is predicted by the classifier (Han & Kamber, 2001; Eichler, 2005).

Bayesian classifier does not require having lots of observations for each possible combination of the variables and are particularly suitable when the dimensionality of input data is high. The effect of a variable value on a given class in Bayesian Classifier is independent of the values of other variable. This assumption simplifies the computation and despite its simplicity, *Naive Bayesian Classifier* can often outperform more sophisticated classification methods (Caruana & Niculescu-Mizil, 2006) makes it particularly popular in commercial and open-source spam filters (Metsis et al., 2006).

2.3 Spammer behavioral patterns

The keyword-based statistical analyzers mostly depend on tokenization of the email content and extracting feature from tokenized keywords to model spammer behavior. Tokenization can be misguided in many several ways as today's email supports character sets other than ASCII, non-text attachments and bodies with multiple parts. For example, the following HTML tricks can be used to do this:

GET<!-- banana -->V<!-- 45-->I<!-- wumpus -->A<!-- dskfj -->G <!-- adf -- >R<!-- free -->A

Thus above nonsense HTML tags only split the special word "viagra" and disguise the tokenizer though it would be shown as "GET VIAGRA" to email client.

Even a word can be replaced with characters of other languages or like same character. For example, "V1DEO" can be sent instead of "VIDEO" and "Fántástìç" instead of "Fantastic". A combination of special characters can used to produce alphabetical characters. For example, char "V" can be represented as the combination of right slash"\" and left slash "/". A grouping or clustering of these techniques is given Table 1 for quick review.

Table 1 has 30 different tricks and one can easily verify that HTML based tactics cover most of them (70%). It can also be shown that 75% of Cascading Style Sheet (CSS) and 50% of Image-based tricks are also covered by HTML-based tactics. It is evident from table 1 that Java Script and MIME (and/or others) based tricks do not overlap with HTML/CSS based tactics.

In this study, a model has been developed exploiting two machine learning algorithms to capture common spammer patterns instead of keyword analysis. The 21 handy crafted features from each e-mail message extracted from subject header, priority & content-type headers and body shown in Table 2 simulate all possible common spammer tricks. These features have also been optimized in their capability of classifying spam emails. The rationale of these features can be verified by their statistics both in spam and non-spam emails. For example, whether a content-type header appeared within the message headers or whether the content type had been set to "text/html" is a common feature of spam, as our investigation revealed. The corpus that has been used in our experimentation, we observed that 98% spam emails include this feature. Similarly, color element (both CSS and HTML format) is also a frequent feature of spam emails. Colorful images those are generally included in the email for X-rated and unwanted internet marketing groups send to catch users' attention. The use of color elements in non-spam mails is very low. We found that 56% spam emails contain color elements whereas it exists only for 10% non-spam emails. The inclusion of this feature in our classification has improved performance considerably, which shows its practicality. We also added feature 19-21 as in Table 2, which are significant features of recent spams.

	Java Script	Image	CSS	HTML	MIME/Others
Title Case				Y	
Sticky Finger				Y	
Accent					Y
Readable Spell				Y	
Dot Matrix			Y	Y	
Right-to-Left				Y	
HTML Numbers				Y	
Comments				Y	
Styles			Y		
Invisible Ink			Y	Y	
Matrix			Y	Y	
Encoding of MSG					Y
Encrypted Message Bodies	Y				
Copperfield			Y		
Invisible Image		Y			
Zero Image		Y	Y	Y	
Slice and Dice		Y	Y	Y	
Cross Word			Y	Y	
Honorary Title				Y	
Image Chopping		Y			
Cramp				Y	
Framed				Y	
Big Tag				Y	
Fake Text				Y	
Slick Click				Y	
Phishing				Y	
False Click				Y	
Pump & Dump					Y
I'm Feeling Lucky				Y	

Table 1. Common spammer tricks

2.4 Email corpus

Classification based spam filtering systems have two major drawbacks. Firstly, building a perfect data set free from noise or imperfection as noise adversely affect the classifier's performance (Islam et al., 2009). The nature of spam email is very dynamic and the content of email is textually misleading due to obfuscation. This remains a continuous challenge for spam filtering techniques. Secondly, most training models of the classifier have limitations on their operations (Ranawana & Palade, 2006). Classifiers often produce uncorrelated training errors due to the dimension of feature space; a dissimilar output space is generated for changing feature space from small dimension to complex high dimension.

In this work a corpus of 1,000 emails received over a period of several months is used for experimentation. The distribution of both spam and non-spam emails in this collection is equal. The equal distribution is preferred to make the classifier to eliminate the biasness towards a particular category. That is, out of 1,000 emails 500 is spam and 500 is non-spam. The collection of this corpus is selected over a time and latest trend in spamming is kept in mind. Also the author's experience with spam research and statistical selection methodology is applied to the selection, which made this email bank very much representative of current spamming.

Feature	Category 1: Features From the Message Subject Header
1	Binary feature indicating 3 or more repeated characters
2	Number of words with all letters in uppercase
3	Number of words with at least 15 characters
4	Number of words with at least two of letters J, K, Q, X, Z
5	Number of words with no vowels
6	Number of words with non-English characters, special characters such as punctuation, or digits at beginning or middle of word
	Category 2: Features From the Priority and Content-Type Headers
7	Binary feature indicating whether the priority had been set to any level besides normal or medium
8	Binary feature indicating whether a content-type header appeared within the message headers or whether the content type had been set to "text/html"
	Category 3: Features From the Message Body
9	Proportion of alphabetic words with no vowels and at least 7 characters
10	Proportion of alphabetic words with at least two of letters J, K, Q, X, Z
11	Proportion of alphabetic words at least 15 characters long
12	Binary feature indicating whether the strings "From:" and "To:" were both present
13	Number of HTML opening comment tags
14	Number of hyperlinks ("href=")
15	Number of clickable images represented in HTML
16	Binary feature indicating whether a text color was set to white
17	Number of URLs in hyperlinks with digits or "&", "%", or "@"
18	Number of color element (both CSS and HTML format)
19	Binary feature indicating whether JavaScript has been used or not
20	Binary feature indicating whether CSS has been used or not
21	Binary feature indicating opening tag of table

Table 2. Features extracted from each e-mail

2.5 Feature construction

Each email is parsed as text file to identify each header element to distinguish them from the body of the message. Every substring within the subject header and the message body that was delimited by white space was considered to be a token, and an alphabetic word was defined as a token delimited by white space that contains only English alphabetic characters (A-Z, a-z)or apostrophes.

The tokens were evaluated to create a set of 21 hand-crafted features from each e-mail message (Table 2) of which features 1-17 are proposed in (Stuart et al., 2004). In addition of these 17 features this study proposes other four features 18-21. The study investigates the suitability of these 21 features in classifying spam emails. To do this, each email is represented by a vector of dimension 21 in the vector space model (Salton et al., 1975). To learn the neural network the input layer of *Multilayer Perceptron* will have 21 nodes. These nodes receive values from each dimension of the vector representing the email. For Naïve Bayesian Classifier, each dimension corresponds to spammer behavioral pattern instead of keyword.

2.6 Evaluation measures

Estimating classifier accuracy is important since it allows one to evaluate how accurately a given classifier will classify unknown samples on which the classifier has not been trained. The effectiveness of a classifier is usually measured in terms of accuracy, precision and recall (Makhoul et al., 1999). These measures are calculated using the confusion matrix given below:

Category C_i	Correct	
Predicted ↓	YES	NO
YES	TP_i	FP_i
NO	FN_i	TN_i

TP=true positives
FP=false positives
FN=false negatives
TN=true negatives

Table 3. Confusion matrix

Accuracy of a classifier is calculated by dividing the number of correctly classified samples by the total number of test samples and is defined as:

$$Accuracy = \frac{number\ \ of\ \ correctly\ \ classified\ \ samples}{total\ \ number\ \ of\ \ test\ \ samples}$$

$$= \frac{TP + TN}{TP + FP + FN + TN}$$

Precision measures the system's ability to present only relevant items while recall measures system's ability to present all relevant items. These two measures are widely used in TREC evaluation of document retrieval (Makhoul, 1999). Precision is calculated by dividing the number of samples that are true positives by the total number of samples classified as positives and is defined as:

$$\Pr ecision = \frac{number\ \ of\ \ true\ \ positives}{total\ \ number\ \ of\ \ samples\ \ classified\ \ as\ \ positives}$$

$$= \frac{TP}{TP + FP}$$

Analogously, recall is calculated by dividing the number of samples that are true positives by the total number of samples that classifier should classified as positives and is defined as:

$$\text{Re}call = \frac{number\ \ of\ \ true\ \ positives}{total\ \ number\ \ of\ \ postive\ \ samples}$$

$$= \frac{TP}{TP + FN}$$

3. Experimental results

Table 4 summarizes the comparative results of Naïve Bayesian Classifier and Artificial neural Networks. These algorithms are tested on Weka 3.6.0 suite of machine learning software written in Java, developed at the University of Waikato (Holmes et al., 1994). Before simulating on *Weka Tool* we transform the vectors obtained from our dataset into *Attribute-Relation File Format* (ARFF).

ARFF files have two distinct sections. The first section is the *Header* information, which is followed the *Data* information. The *Header* of the ARFF file contains the name of the relation, a list of the attributes (the columns in the data), and their types. An example header for our dataset is given below:

@RELATION Email_Classification

@ATTRIBUTE	feature01	{yes, no}
@ATTRIBUTE	feature02	NUMERIC
@ATTRIBUTE	feature03	NUMERIC
@ATTRIBUTE	feature04	NUMERIC
@ATTRIBUTE	feature05	NUMERIC
@ATTRIBUTE	feature06	NUMERIC
@ATTRIBUTE	feature07	{normal, medium}
@ATTRIBUTE	feature08	{yes, no}
@ATTRIBUTE	feature09	NUMERIC
@ATTRIBUTE	feature10	NUMERIC
@ATTRIBUTE	feature11	NUMERIC
@ATTRIBUTE	feature12	{yes, no}
@ATTRIBUTE	feature13	NUMERIC
@ATTRIBUTE	feature14	NUMERIC
@ATTRIBUTE	feature15	NUMERIC
@ATTRIBUTE	feature16	{yes, no}
@ATTRIBUTE	feature17	NUMERIC
@ATTRIBUTE	feature18	NUMERIC
@ATTRIBUTE	feature19	{yes, no}
@ATTRIBUTE	feature20	{yes, no}
@ATTRIBUTE	feature21	{yes, no}
@ATTRIBUTE	isHam	{yes, no}

The last attribute is the class column. The *Data* of the ARFF file looks like the following (only few samples out of 1000 are given here):

no, 0, 0, 0, 1, 2, medium, yes, 2, 0, 118, yes, 1, 13, 0, no, 0, 0, no, yes, no, yes
no, 8, 0, 0, 1, 2, medium, yes, 1, 219, 223, no, 0, 0, 0, no, 0, 0, no, no, no, yes
no, 0, 1, 0, 0, 1, medium, yes, 5, 4, 48, no, 0, 4, 1, no, 2, 0, no, yes, no,yes
no, 0, 0, 0, 0, 1, medium, yes, 3, 0, 144, yes, 1, 69, 0, no, 0, 3, no, yes, no, yes
no, 0, 1, 1, 0, 1, medium, yes, 4, 0, 7, yes, 1, 1, 0, no, 0, 0, no, yes, no,yes
no, 0, 0, 0, 1, 1, medium, yes, 0, 265, 271, no, 0, 0, 0, no, 0, 0, no, no, no, yes
no, 0, 1, 0, 0, 1, medium, no, 0, 0, 0, no, 0, 0, 0, no, 0, 0, no, no, no,yes
no, 0, 0, 0, 1, 0, medium, yes, 0, 0, 1, no, 0, 1, 0, no, 1, 1, no, yes, no,yes
yes, 0, 0, 0, 3, 5, medium, yes, 0, 0, 2, no, 0, 0, 0, no, 0, 0, no, no, no,yes
yes, 0, 0, 0, 3, 6, medium, no, 0, 0, 0, no, 0, 0, 0, no, 0, 0, no, no, no,yes
no, 0, 0, 0, 0, 0, medium, yes, 10, 0, 55, no, 0, 11, 1, yes, 0, 2, no, yes, yes, no
no, 0, 0, 0, 0, 1, medium, yes, 12, 0, 45, no, 0, 0, 1, no, 0, 15, no, yes, yes, no
no, 0, 0, 0, 0, 0, medium, yes, 6, 0, 43, no, 0, 11, 1, no, 0, 0, no, yes, yes, no
no, 0, 0, 0, 0, 1, medium, yes, 2, 0, 3, no, 0, 0, 0, no, 0, 0, no, no, no,no
no, 0, 0, 0, 0, 0, medium, yes, 2, 0, 21, no, 0, 0, 0, no, 0, 2, no, yes, yes, no
no, 0, 0, 0, 0, 1, medium, yes, 8, 0, 12, no, 0, 29, 0, no, 0, 0, no, no, no,no
no, 0, 0, 0, 0, 0, medium, yes, 14, 0, 57, no, 0, 16, 1, yes, 0, 2, no, yes, yes, no
no, 0, 0, 0, 0, 1, medium, yes, 1, 0, 39, no, 0, 14, 1, no, 0, 0, no, yes, yes, no
yes, 1, 0, 0, 4, 6, medium, yes, 2, 0, 17, no, 0, 2, 0, no, 0, 0, no, no, no, no
yes, 4, 0, 0, 4, 5, medium, yes, 3, 0, 8, no, 0, 7, 0, no, 0, 0, no, no, no, no

Features	Naïve Bayesian Classifier (NaiveBayes)			ANN (Multilayer Perceptron)		
	Accuracy	Precision	Recall	Accuracy	Precision	Recall
Category 1 Only	56.5 %	55.7%	56.5%	67.8%	73.1%	67.8%
Category 2 Only	65.2%	75.0%	65.2%	65.2%	75.0%	65.2%
Category 3 Only	88. 7 %	88.7%	88.7%	86.1%	86.1%	86.1%
Category 1+Category 2	66.9 %	67.3%	67.0%	73.1%	77.2%	73.0%
Category 2+ Category3	92.2 %	92.2%	92.2%	87.8%	88.1%	87.8%
Category 1+Category 3	80.8 %	80.9%	80.9%	74.7%	75.4%	74.8%
Category1+ Category 2 + Category 3	86.9 %	87.0%	87.0%	84.3%	85.1%	84.3%

Table 4. Comparison results for Naïve Bayesian classifier and Artificial Neural Network

The highest level of accuracy that can be achieved by Naïve Bayesian classifier is 92.2% (shown in Table 4) using features from category 2 and 3. The accuracy that can be achieved by any learning algorithms using features from category 1 is negligible. Features from category 2 and 3 contribute mostly in classifying spam emails from non-spam emails for all machine learning algorithm experimented in this study.

Highest number of features is always desirable only if their inclusion increase classifier's accuracy significantly. Growing number of features not only hinders multidimensional indexing but also increases overall execution time. So, this study starves to find an optimal number of features that can be effectively used to learn a classifier without degrading the level of accuracy.

Applying best first forward attribute selection method the study gets only 10 features from category 2 and category 3 useful for classifying the spam and non-spam emails without sacrificing the accuracy as shown in Table 5. The set includes features 8, 9, 10, 12, 13, 14, 15, 16, 17, and 18 of which feature 18 is identified in this study. The Naïve Bayesian classifier again outperforms the Artificial Neural Networks. The optimal feature set obtained by applying best first forward attribute selection method for the features proposed in (Stuart et al., 2004) includes only features 8, 9, 10, 12, 13, 14, 15, 16 and 17, a total of 9 features. In this case ANN outperforms than Naïve Bayesian classification algorithm as shown in Table 5.

Features	Naïve Bayesian Classifier(Naïve Bayes)			ANN (Multilayer Perceptron)		
	Accuracy	Precision	Recall	Accuracy	Precision	Recall
Best first: 8, 9, 10, 12, 13, 14, 15, 16, 17, and 18 [This study]	92.2%	92.2%	92.2%	90.4%	90.6%	90.4%
Best first: 8, 9, 10, 12, 13, 14, 15, 16, and 17 (Stuart et al., 2004)	86.1 %	87.4%	86.1%	91.3%	91.4%	91.3%

Table 5. Comparison results for Naïve Bayesian classifier and Artificial Neural Network

The study presented in (Stuart et al., 2004) uses neural network for modeling spammer common patterns and achieved similar performance, but the limitation of neural network is its longer training time and inherent complexity of explaining its derivation (less comprehensibility). On the contrary, Bayesian Classifier has the advantage of incremental inclusion of features and beforehand calculation. Therefore, Naïve Bayes is suitable for adapting itself in modeling new spammer patterns.

4. Conclusion

This research studies the modeling of spammer behavior by Artificial Neural Networks and Naïve Bayesian Classifier algorithms for spam email classification. Based on examining different features and two different learning strategies, the following conclusions can be drawn from the study presented in this study:

Lesson 1. Spammer behavior can be modeled using features extracted from Content-Type header and message Body only.

Lesson 2. The contribution of features extracted from subject header in spam email detection is negligible or insignificant.

Lesson 3. Naïve Bayesian classifier models the spammer behavior best than Artificial Neural Networks.

Lesson 4. It is possible to get an optimal number of features that can be effectively applied to learning algorithms to classify spam emails without sacrificing accuracy

The preliminary result presented in this study seems promising in modeling spammer common behavioral patterns compared to similar research. The contribution of this study is threefold: it shows why keyword based spam email classifier may fail to model spammers' altering tricks, common patterns adopted by spammers and the rationale of using these patterns against them to combat spam; suitability of modeling spammer common patterns using ANN and Naïve Bayes and finally, establishment of the four concluding remarks.

5. References

Aery, M. & Chakravarthy, S. (2005). eMailSift: email classification based on structure and content, *Proceedings of 5th IEEE International Conference on Data Mining*, pp. 1-8, 0-7695-2278-5, Houston, Texas, January 2005, IEEE Computer Society, Los Alamitos, CA

Caruana, R. & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. *Proceedings of the 23rd International Conference on Machine Learning*, pp. 161-168, Pittsburgh, Pennsylvania, 1-59593-383-2, ACM New York, NY, USA

Drucker, H.; Wu, D. & Vapnik, V. N. (1999). Support vector machines for spam categorization, *IEEE Transactions on Neural Networks*, Vol. 10, No. 5, September 1999, pp. 1048-1054, 1045-9227

Eichler, K. (2005). Automatic Classification of Swedish Email Messages, B.A Thesis, Eberhard-Karls-Universitat Tubingen, August 2005

Guerra, P. H. C., Guedes, D., Meira, W., Hoepers, C., Chaves, M. H. P. C., Jessen, K. S. (2009). Spamming chains: a new way of understanding spammer behavior, *Sixth Conference on Email and Anti Spam*, July 1617, 2009, Mountain View, California USA

Han, J. & Kamber, M. (2001). Data Mining Concepts and Techniques, Academic Press, ISBN 81-7867-023-2

Holmes, G; Donkin, A. & Witten, I. H. (1994). Weka: A machine learning workbench, *Proceedings of 2nd Australia and New Zealand Conference on Intelligent Information Systems*, pp. 357-361, 0-7803-2404-8, Brisbane, Australia, November-December 1994

Islam, M. S. & Amin, M. I. (2007). An architecture of active learning SVMs with relevance feedback for classifying E-mail, *Journal of Computer Science*, Vol. 1, No. 1, June 2007, pp. 15-18, 1994-6244

Islam, M. R. & Chowdhury, M. U. (2005). Spam filtering using ML algorithms, *Proceedings of IADIS International Conference on WWW/Internet*, pp. 419-426, 972-8924-02-X, Lisbon, Portugal, October 2005, International Association for Development of the Information Society

Islam, M. R.; Zhou, W.; Xiang, Y. & Gao, M. (2009). An innovative analyser for multi-classifier email classification based on grey list analysis, *The Journal of Network and Computer Applications*, Elsevier, Vol. 32, No. 2, March 2009, pp. 357-366, 1084-8045

Makhoul, J.; Kubala, F.; Schwartz, R. & Weischedel, R. (1999). Performance measures for information extraction, *Proceedings of DARPA Broadcast News Workshop*, pp. 249-252, Herndon, VA, February 1999

Metsis, V., Androutsopoulos, I. & Paliouras, G. (2006). Spam filtering with Naive Bayes – Which Naive Bayes?, *Third Conference on Email and Anti Spam*, July 2728, 2006, Mountain View, California USA

Ramachandran, A. & Feamster, N. (2006), Understanding the network level behavior of spammers, *SIGCOMM'06*, September 1116, 2006, pp., 291-302, Pisa, Italy, ACM New York, NY, USA

Ramachandran, A., Feamster, N. & Vempala, S. (2007). Filtering spam with behavioral blacklisting, *CCS'07,* October 29–November 2, 2007, pp. 342-351, Alexandria, Virginia, USA

Ranawana, R. & Palade, V. (2006). Multi-classifier systems - review and a roadmap for developers, *International Journal of Hybrid Intelligent Systems*, Vol. 3, No. 1, January 2006, pp. 35-61, 1448-5869

Rojas, R. (1996). The backpropagation algorithm, *Neural Networks - A Systematic Introduction*, Chapter 7, ISBN 978-3540605058, Springer-Verlag, Berlin, New-York, 1996

Salton, G.; Wong, A. & Yang, C. S. (1975). A Vector Space Model for Automatic Indexing, *Communications of the ACM*, Vol. 18, No. 11, pp. 613–620

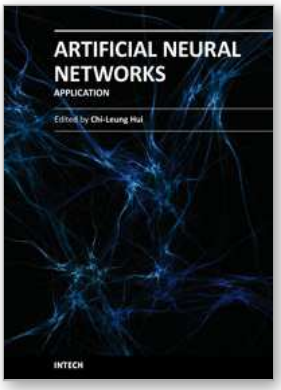
Sebastiani, F. (2002). Machine learning in automated text categorization, *ACM Computing Surveys*, Vol. 34, No. 1, March 2002, pp. 1-47, 0360-0300

Sperotto, A., Vliek, G., Sadre, R. & Pras, A. (2009). Detecting spam at the network level, M. Oliver and S. Sallent (Eds.): *EUNICE 2009*, LNCS 5733, pp. 208–216, Springer-Verlag Berlin Heidelberg 2009

Stuart, J. I.; Cha, S. & Tappert, C. (2004). A neural network classifier for junk E-mail, *Lecture Notes in Computer Science*, Vol. 3163, pp. 442-450, 0302-9743

Xu, K. S., Kliger, M. & Hero, A. O. (2010). Identifying spammers by their resource usage patterns, *Seventh annual Collaboration, Electronic messaging, Anti Abuse and Spam Conference,* July 13-14, 2010, Redmond, Washington, USA

Zhang, X., Liu, J., Zhang, Y. & Wang, C. (2006). Spam behavior recognition based on session layer data mining, *FSKD 2006*, LNAI 4223, pp. 1289–1298, Springer-Verlag Berlin Heidelberg 2006



Artificial Neural Networks - Application

Edited by Dr. Chi Leung Patrick Hui

ISBN 978-953-307-188-6
Hard cover, 586 pages
Publisher InTech
Published online 11, April, 2011
Published in print edition April, 2011

This book covers 27 articles in the applications of artificial neural networks (ANN) in various disciplines which includes business, chemical technology, computing, engineering, environmental science, science and nanotechnology. They modeled the ANN with verification in different areas. They demonstrated that the ANN is very useful model and the ANN could be applied in problem solving and machine learning. This book is suitable for all professionals and scientists in understanding how ANN is applied in various areas.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Saiful Islam and Rafiqul Islam (2011). Modeling Spammer Behavior: Artificial Neural Network vs. Naïve Bayesian Classifier, Artificial Neural Networks - Application, Dr. Chi Leung Patrick Hui (Ed.), ISBN: 978-953-307-188-6, InTech, Available from: http://www.intechopen.com/books/artificial-neural-networks-application/modeling-spammer-behavior-artificial-neural-network-vs-nai-ve-bayesian-classifier

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821