Spam Recognition using Linear Regression and Radial Basis Function Neural Network

Tich Phuoc Tran[1] , Min Li[1], Dat Tran[2] and Dam Duong Ton[3]
[1]*Centre for Quantum Computation and Intelligent Systems (QCIS)*
University of Technology, Sydney, NSW 2007, Australia
{tiptran, minli }@it.ut.edu.au

[2]*Faculty of Information Sciences and Engineering*
University of Canberra, ACT 2601, Australia
dat.tran@canberra.edu.au

[3]*Faculty of Computer Science*
University of Information Technology, VNU-HCMC, Vietnam
damdt@uit.edu.vn

Keywords: Spam Recognition, Redial Basis Function, Linear Regression

Abstract:
Spamming is the abuse of electronic messaging systems to send unsolicited bulk messages. It is becoming a serious problem for organizations and individual email users due to the growing popularity and low cost of electronic mails. Unlike other web threats such as hacking and Internet worms which directly damage our information assets, spam could harm the computer networks in an indirect way ranging from network problems like increased server load, decreased network performance and viruses to personnel issues like lost employee time, phishing scams, and offensive content. Though a large amount of research has been conducted in this area to prevent spamming from undermining the usability of email, currently existing filtering methods' performance still suffers from extensive computation (with large volume of emails received) and unreliable predictive capability (due to highly dynamic nature of emails). In this chapter, we discuss the challenging problems of Spam Recognition and then propose an anti-spam filtering framework; in which appropriate dimension reduction schemes and powerful classification models are employed. In particular, Principal Component Analysis transforms data to a lower dimensional space which is subsequently used to train an Artificial Neural Network based classifier. A cost-sensitive empirical analysis with a publicly available email corpus, namely Ling-Spam, suggests that our spam recognition framework outperforms other state-of-the-art learning methods in terms of spam detection capability. In the case of extremely

high misclassification cost, while other methods' performance deteriorates significantly as the cost factor increases, our model still remains stable accuracy with low computation cost.

1. Introduction

Email is widely accepted by the business community as a low cost communication tool to exchange information between business entities which are physically distant from one another. It minimizes the cost of organizing an in-person meeting. It is reported by a recent survey SurePayroll (Surepayroll, 2007), over 80% of small business owners believe email is a key to the success of their business and most people today spend between 20% to 50% of their working time using email, including reading, sorting and writing emails. Due to the very low cost of sending email, one could send thousands of malicious email messages each day over an inexpensive Internet connection. These junk emails, referred to as *spam,* can severely reduce staff productivity, consume significant network bandwidth and lead to service outages. In many cases, such messages also cause exposure to viruses, spyware and inappropriate contents that can create legal/compliance issues, loss of personal information and corporate assets. Therefore, it is important to accurately estimate costs associated with spam and evaluate the effectiveness of countermeasures such as spam-filtering tools. Though such spam prevention capability is implemented in existing email clients, there are some barriers that discourage users from utilizing this feature including error-prone and labor-intensive maintenance of filtering rules. Many researchers have developed different automatic spam detection systems but most of them suffer from low accuracy and high false alarm rate due to huge volume of emails, the wide spectrum of spamming topics and rapidly changing contents of these messages, especially in the case of high misclassification cost (Bayler, 2008). To deal with such challenges, this chapter proposes an anti-spam filtering framework using a highly performing *Artificial Neural Network* (ANN) based classifier. ANN is widely considered as a flexible "model-free" or "data-driven" learning method that can fit training data very well and thus reduce *learning bias* (how well the model fits the available sample data). However, they are also susceptible to the overfitting problem, which can increase generalization variance, i.e. making the predictive model unstable for unseen instances. This limitation can be overcome by combining ANN with a simple Linear Regression algorithm which makes the resulting classification model a stable semi-parametric classifier. Such model combination aims at stabilizing non-linear learning techniques while retaining their data fitting capability. Empirical analysis with the Ling-Spam benchmark confirms our superior spam detection accuracy and low computation cost in comparison with other existing approaches.

This chapter is organized as follow. Firstly, an overview of the spam problem is presented with associated negative impacts and protection techniques. This is followed by the application of Machine Learning (ML) to spam recognition, related works and details of the Ling-Spam corpus. Next, a brief review of several commonly used classification models and our proposed framework is given. The subsequent section compares the performance of our method with other learning techniques using the benchmark corpus under different cost scenarios. Finally, we provide some conclusion remarks for this chapter and future research directions.

2. Spam Recognition and Machine Learning techniques

2.1. Spam Recognition as a Challenging Task

2.1.1. Overview of Spamming

Spamming is one of the biggest challenges facing Internet consumers, corporations, and service providers today. Email spamming, also known as *Unsolicited Bulk Email* (UBE) or *Unsolicited Commercial Email* (UCE), is the practice of sending unwanted email messages, frequently with commercial content, in large quantities to an indiscriminate set of recipients (Schryen, 2007). Due to the increasing popularity and low cost of email, there are more and more spam circulating over the Internet. Spam emails were found to account for approximately 10% of the incoming message to corporate networks (L. F. Cranor & LaMacchia, 1998) and currently costs businesses US $ 13 billion annually (Swartz, 2003). Not only wasting time and consuming bandwidth, undesired emails are extremely annoying to most users due to their unsuitable contents, ranging from advertising vacations to pornographic materials.

Spam is usually classified into two categories which have different effects on Internet users. *Cancellable Usenet spam* is a single message sent to many Usenet newsgroups (Wolfe, Scott, & Erwin, 2004). This spamming attack can overwhelm the users with a barrage of advertising or other irrelevant posts. It also subverts the ability of system administrators and group owners to manage the topics they accept on their systems. The second type of spam is *Email spam* which targets individual users with direct mail messages (Bayler, 2008). Not only causing loss of productivity for email users, it also costs money for Internet Service Providers (ISP) and online services to transmit spam, and these costs are transferred directly to other subscribers. Though there are different types of spam, they all share some common properties. First, the sender's identity and address are concealed. Second, spam emails are sent to a large number of recipients and in high quantities. In fact, spam is economically viable to its senders not only because it is low cost to send an email, but also because spammers have no operating costs beyond the management of their mailing lists. This attracts numerous spammers and the volume of unsolicited mail has become very high. Finally, spam messages are unsolicited, that is, the individuals receiving spam would otherwise not have opted to receive it.

To successfully send a spam message, spammers usually undertake two steps: (1) collecting target email addresses and (2) bypassing anti-spam measures (Schryen, 2007). The later task involves cleverly disguising an unsolicited message as a non-spam message with normal appearing subject lines and other ways of getting around anti-spam software. The first task seems easier but in fact could be very challenging. To collect valid email addresses of potential target victims, the following techniques can be deployed (Bayler, 2008):

- Buying lists of addresses from some companies or other spammers.
- Harvesting email addresses from web sites or UseNet News posts with automated programs.
- Stealing users address books on compromised computers.
- Collecting addresses via Internet Relay Chat (IRC) programs.
- Guessing email addresses, then sending email to see if it goes through (Directory Harvest attacks).
- Using false reasons to trick a user into giving up their email address (Social Engineering attacks).

Though spam emails are troublesome, most of them can be easily recognized by human users due to their obvious signatures. For example, spam emails normally relate to specific topics such as prescription drugs, get-rich-quick schemes, financial services, qualifications, online gambling, discounted or pirated software. However, with a huge volume of spam messages received every day, it would not be practical for human users to detect spam by reading all of them manually. Furthermore, spam sometimes comes disguised, with a subject line that reads like a personal message or a non-delivery message. This makes highly accurate spam detection software desirable for encountering spam.

2.1.2 Impacts of Spamming and Preventive Techniques

Even though spam does not threaten our data in the same way that viruses do, it does cause businesses billions of lost dollars worldwide. Several negative impacts of spam are listed as follow (Schryen, 2007):

- Spam is regarded as privacy invasion because spammers illegally collect victim's email address (considered as personal information)
- Unsolicited emails irritate Internet users.
- Non-spam emails are missed and/or delayed. Sometimes, users may easily overlook or delete critical emails, confusing them with spam.
- Spam wastes staff time and thereby significantly reduce enterprises' productivity.
- Spam uses a considerably large bandwidth and uses up database capacity. This causes serious loss of Internet performance and bandwidth.
- Some spam contains offensive content.
- Spam messages can come attached with harmful code, including viruses and worms which can install backdoors in receivers' systems.
- Spammers can hijack other people's computers to send unwanted emails. These compromised machines are referred to as "zombie networks", networks of virus- or worm-infected personal computers in homes and offices around the globe. This ensures spammers' anonymity and massively increases the number of spam messages can be sent.

Various counter-measures to spam have been proposed to mitigate the impacts of unsolicited emails, ranging from regulatory to technical approaches. Though anti-spam legal measures are gradually being adopted, their effectiveness is still very limited. A more direct counter-measure is software-based anti-spam filters which attempt to detect spam from legitimate mails automatically. Most of the existing email software packages are equipped with some form of programmable spam filtering capability, typically in the form of blacklists of known spammers (i.e. block emails that come from a black list; check whether emails come from a genuine domain name or web address) and handcrafted rules (i.e. block messages containing specific keywords and unnecessary embedded HTML code). Because spammers normally use forged addresses, the blacklist approach is very ineffective. Handcrafted rules are also limited due to their reliance on personal preferences, i.e. they need to be tuned to characteristics of messages received by a particular user or groups of users. This is a time consuming task requiring resources and expertise and has to be repeated periodically to account for changing nature of spam messages (L. F. Cranor, LaMacchia, B.A. , 1998).

Spam detection is closely related to *Text Categorization* (TC) due to their text-based contents and similar tasks. However, unlike most TC problems, spamming is the act of blindly mass-

mailing an unsolicited message that makes it spam, not its actual content (Schryen, 2007): any otherwise legitimate message becomes spam if blindly mass-mailed. From this point of view, spamming becomes a very challenging problem to the sustainability of the Internet, given the content of emails the only foundation for spam recoginition. Nevertheless, it seems that the language of current spam messages constitutes a distinctive genre, and that the topics of most current spam messages are rarely mentioned in legitimate messages, making it possible to train successfully a text classifier for spam recognition.

2.2. Machine Learning for Spam Recognition

Recent advances of *Machine Learning* (ML) techniques in *Text Classification* (TC) have attracted immense attention from researchers to explore the applicability of learning algorithms in anti-spam filtering (Bayler, 2008). In particular, a collection of messages is input to a learning algorithm which infers underlying functional dependencies of relevant features. The result of this process is a model that can, without human intervention, classify a new incoming email as spam or legitimate according to the knowledge collected from the training stage. Apart from automation which frees organizations from the need of manually classifying a huge amount of messages, this model can be retained to capture new characteristics of spam emails. To be most useful in real world applications, the anti-spam filters need to have a good generalization capability, that is, they can detect malicious messages which never occur during the learning process. There has been a great deal of research conducted in this area, ranging from simple methods such as propositional learner Ripper with "keyword-spotting rules" (Cohen, 1996) to more complicated approaches such as Bayesian networks using *bags of words* representation and binary coding (Sahami, 1998). In (Androutsopoulos, Koutsias, Chandrinos, Paliouras, & Spyropoulos, 2000), a system implementing Naïve Bayes and a k-NN technique is reported to be able to outperform the keyword-based filter of Outlook 2000 on the Ling-Spam corpus. Ensemble methods also prove their usefulness in filtering spam. For example, staked Naïve Bayes and k-NN can achieve good accuracy (Drucker, Wu, & Vapnik, 1999), and Boosted trees were shown to have better performance than individual trees, Naïve Bayes and k-NN alone (Carreras & Marquez, 2001). A support vector machine (SVM)(Drucker et al., 1999) is also reported to achieve a higher detection rate as well as lower false alarm rate for spam recognition compared with other discriminative classification methods. It is suggested that email headers play a vital role in spam recognition, and to get better results, classifiers should be trained on features of both email headers and email bodies (Androutsopoulos et al., 2000).

2.3 Ling-Spam Benchmark

The Ling-Spam corpus (Androutsopoulos et al., 2000) is used as a benchmark to evaluate our proposed algorithm with other existing techniques, the anti-spam filtering task. Using this publicly available dataset, we can conduct tractable experiments and also avoid complications of privacy issues. While spam messages do not pose this problem as they are blindly distributed to a large number of recipients, legitimate email messages may contain personal information and cannot usually be released without violating the privacy of their recipients and senders.

The corpus contains legitimate messages collected from a moderated mailing list on profession and science of linguistics and the spam messages collected from personal mailboxes:

- 2412 legitimate messages with text added by the list's server removed.
- 481 spam messages (duplicate spam messages received on the same day excluded)

The headers, HTML tags, and attachments of these messages are removed, leaving only the subject line and body text. The distribution of the dataset (16.6% is spam) makes it easy to identify legitimate emails because of the topic-specific nature of the legitimate mails. This dataset is partitioned into 10 stratified subsets which maintain the same ratio of legitimate and spam messages as in the entire dataset. Though some research (Androutsopoulos et al., 2000; Carreras & Marquez, 2001; Hsu, Chang, & Lin, 2003; Sakkis et al., 2003) has been conducted on this data showing their comparative efficiency, most of them suffer from high a false alarm rate which results in a degraded performance when the misclassification cost is high. Overcoming this problem is our major objective in this chapter.

3. Spam Recognition Methods

This section discusses commonly used learning algorithms for spam recognition problems.

3.1. Naïve Bayes

Naive Bayes is a well-known probabilistic classification algorithm which has been used widely for spam recognition (Androutsopoulos et al., 2000). According to Bayes' theorem, we can compute the probability $Prob(C = c|\vec{X} = \vec{x})$ that a message with vector $\vec{x} = \{x_1, \dots, x_n\}$ belongs to a class $c \in \{legit, spam\}$:

$$P\left(C = c\middle|\vec{X} = \vec{x}\right) = \frac{P(C = c).\, Prob(\vec{X} = \vec{x}|C = c)}{\sum_{k\in\{spam,legit\}} P(C = k).\, P(\vec{X} = \vec{x}|C = k)}$$

The calculation of $P(\vec{X} = \vec{x}|C = c)$ is problematic because most all novel messages are different from training messages. Therefore, instead of calculating probability for messages (a combination of words); we can consider their words separately. By making the assumption that $x_1, \dots, x_n$ are conditionally independent given the class c, we have:

$$Prob\left(C = c\middle|\vec{X} = \vec{x}\right) = \frac{P(C = c).\prod_{i=1}^{n} P(X_i = x_i|C = c)}{\sum_{k\in\{spam,legit\}} P(C = k).\prod_{i=1}^{n} P(X_i = x_i|C = k)}$$

3.2. Memory Based Learning

In (Androutsopoulos et al., 2000), an anti-spam filtering technique using Memory-Based Learning (MBL) that simply stores the training messages. The test messages are then classified by estimating their similarity to the stored examples based on their *overlap* metric which counts the attributes where the two messages have different values. Given two instances $\vec{x}_i = \{x_{i1}, \dots, x_{in}\}$ and $\vec{x}_j = \{x_{j1}, \dots, x_{jn}\}$, their overlap distance is:

$$d\left(\vec{x}_i, \vec{x}_j\right) = \sum_{r=1}^{n} \delta(x_{ir}, x_{jr})$$

Where $\delta(x, y) = 0$ if $x = y$ or *1* otherwise.

The confidence level that a message $\vec{x}$ belongs to a class c is calculated based on the classes of other neighbor instances $C(\vec{x}_i)$:

$$W_c(\vec{x}) = \sum \big(1 - \delta(c, C(\vec{x}_i))\big)$$

MBL's performance can be significantly improved by introducing some weighting schemes.

3.3. Distance Weighting

Depending on how far a test instance is away from its neighborhood, its' confidence level is estimated:

$$W_c(\vec{x}) = \sum \frac{1}{d^3(\vec{x}, \vec{x}_i)} \big(1 - \delta(c, C(\vec{x}_i))\big)$$

3.3.1 Attribute Weighting

Unlike the basic k-neighborhood classifiers where all attributes are treated equally, MBL assigns different weights to the attributes; depending on how well they discriminate between the categories, and adjust the distance metric accordingly. In particular, an attribute X_i has a weight of M_i which is the reduction of entropy *H(C)* (uncertainty on any category *C* of a randomly selected instance) and the expected value of entropy $H(C|X = x)$ (uncertainly on any category *C* given the value of attribute *X*). This means an attribute would have a higher weight if knowing its value reduces uncertainty on category *C*.

$$M_i = H(C) - \sum_{x \in \{0.1\}} P(X = x).H(C|X = x)$$

Where

$H(C) = \sum_{c \in \{spam, legit\}} P(C = c).log_2 P(C = c)$

$H(C|X = x) = \sum_{c \in \{spam, legit\}} P(C = c|X = x).log_2 P(C = c|X = x)$

The distance between two instances is recalculated as below:

$$d(\vec{x}_i, \vec{x}_j) = \sum_{r=1}^{n} M_r \delta(x_{ir}, x_{jr})$$

3.4. Boosted Decision Tree

Boosted Tree (BT) is a popular method implemented in many anti-spam filters with great successes (Carreras & Marquez, 2001). It uses the ADA-Boost algorithm (Schapire & Singer, 2000) to generate a number of Decision Tress classifiers which are trained by different sample sets drawn from the original training set. Each of these classifiers produces a hypothesis from which a learning error can be calculated. When this error exceeds a certain level, the process is terminated. A final composite hypothesis is then created by combining individual hypotheses.

3.5. Support Vector Machine

Support Vector Machines (SVM) (Drucker et al., 1999) have become one of the popular techniques for text categorization tasks due to their good generalization nature and the ability to overcome the curse of dimensionality. SVM classifies data by a set of representative support vectors. Assume that we want to find a discriminant function *f(x)* such that $y_i = f(x_i)$. A possible linear discriminant function can be presented as $f(x) =$

$sgn\big((w.x)+b\big)$ where $(w.x)+b=0$ is a separating hyperplane in the data space. Consequently, choosing a discriminant function is to find a hyperplane having the maximum separating margin with respect to the two classes. A SVM model is constructed by solving this optimization problem.

3.6. Artificial Neural Network

Artificial neural network (ANN) has gained strong interests from diverse communities due to its ability to identify the patterns that are not readily observable. Multi-Layer Perceptron (MLP) is the most popular neural network architecture in use today. This network uses a layered feed-forward topology in which the units each perform a biased weighted sum of their inputs and pass this activation level through a transfer function to produce their output (Rumelhart & McClelland, 1986). Though many applications have implemented MLP for superior learning capacity, its performance is unreliable when new data is encountered. A recently emerging branch of ANN, the RBF networks, is also reported to gain great successes in diverse applications. In this chapter, MLP is implemented as typical ANN models for spam recognition.

4. Classification Framework for Spam Recognition

Although letting undetected spam pass through a filter is not as dangerous as blocking a legitimate message, one can argue that among one million incoming emails, a few thousand unsolicited message that are misclassified as normal is still very costly. Hence, anti-spam filters really need to be accurate, especially when they are used in large organizations. Advanced ML techniques have been used to improve performance of spam filtering systems. Amongst those methods, *Artificial Neural Network* (ANN) has been gaining strong interests from diverse communities due to its ability to identify the patterns that are not readily observable. Despite recent successes, ANN based applications still have some disadvantages such as extensive computation and unreliable performance. In this study, we use a Modified Probabilistic Neural Network (MPNN) which is developed by Zaknich (Zaknich, 1998).

If there exists a corresponding scalar output y_n for each local region (cluster) which is represented by a center vector $\underline{c}_i$, MPNN can be modeled as follow (Zaknich, 2003):

$$\hat{y}(\underline{x}) = \frac{\sum_{i=0}^{M} Z_i y_i f_i(\underline{x}-\underline{c}_i,\delta)}{\sum_{i=0}^{M} Z_i f_i(\underline{x}-\underline{c}_i,\delta)}$$

With Gaussian function $f_i(\underline{x}) = exp\frac{-(\underline{x}-\underline{c}_i)^T(\underline{x}-\underline{c}_i)}{2\delta^2}$

Where

$\underline{c}_i$ = center vector for cluster i in the input space

y_i = scalar output related to $\underline{c}_i$

Z_i = number of input vectors x_j within cluster $\underline{c}_i$

δ = single smoothing parameter chosen during network training

M = number of unique centers $\underline{c}_i$

Though MPNN is reported to provide acceptable accuracy and affordable computation, it, just like other ANN, cannot classify reliably when an unusual input which differs from their training data emerges. As a result, it is essential that some degree of generalization capacity

must be incorporated in the MPNN based classifiers. A possible approach to this problem is to incorporate MPNN with a linear model which offers stability, analyzability and fast adaptation (Hayes, 1996).

4.1 Description

Figure 1 shows the overall filtering framework proposed for spam recognition problem. There are 4 main phases.

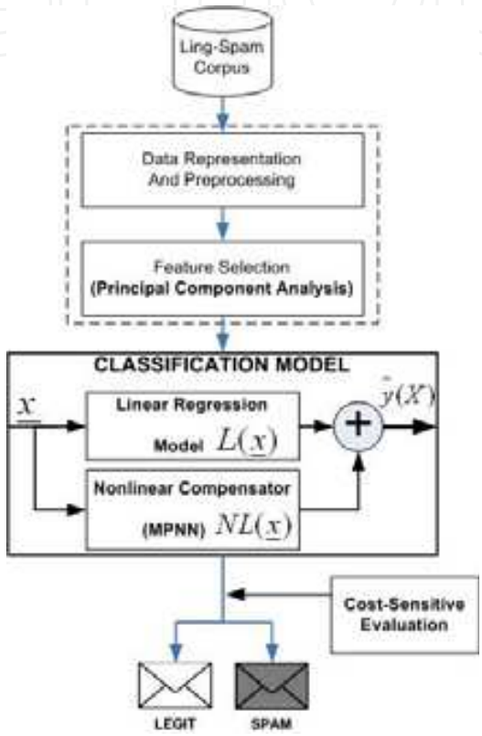


Fig. 1. Proposed anti-spam filtering framework

4.1.1. Phase 1: Data Representation and Preprocessing

The purpose of data preprocessing is to transform messages in the mail corpus into a uniform format that can be understood by the learning algorithms. Features found in mails are normally transformed into a vector space in which each dimension of the space corresponds to a given feature in the entire corpus. Each individual message can then be viewed as a feature vector. This is referred to as the "*bag of words*" approach. There are two methods to represent elements of the feature vector: (1) *multi-variate presentation* assigns a binary value to each element showing that the word occurs in the current mail or not and (2) *multi-nomial presentation* represents each element as a number that shows the occurrence frequency of that word in the current mail. A combination of "bag of words" and multi-variate presentation is used in our experiments (the order of the words is neglected). To construct the feature vectors, the important words are selected according to their *Mutual Information (MI)* (Sakkis et al., 2003):

$$MI(X,C) = \sum_{x\in\{0,1\},c\in\{spam,legit\}} Prob(X=x,C=c).log_2\frac{Prob(X=x,C=c)}{Prob(X=x).Prob(C=c)}$$

The words with the highest *MI* values are selected as the features. Assume that there are n features to be chosen, each mail will be represented by a feature vector $\vec{x} = \{x_1, \dots, x_n\}$ where $x_1, \dots, x_n$ are the values of binary attributes $X_1, \dots, X_n$, indicating the presence or absence of an attribute (word) in current message.

Moreover, w*ord stemming* and *stop-word removal* are two important issues that need to be considered in parsing emails. Word stemming refers to converting words to their morphological base forms (e.g. "gone" and "went" are reduced to root word "go"). Stop-word removal is a procedure to remove words that are found in a list of frequently used words such as "and, for, a". The main advantages of applying the two techniques are the reduction of feature space dimension and possible improvement on classifiers' prediction accuracy by alleviating the data sparseness problem(Androutsopoulos et al., 2000). The Ling-Spam corpus has four versions, each differs from each other by the usage of a *lemmatizer* and a *stoplist* (removes the 100 most frequently used words). We use the version with lemmatizer and stoplist enabled because it performs better when different cost scenarios are considered (Androutsopoulos et al., 2000). Words that appear less than 4 times or longer than 20 characters are discarded.

Also, it is found that phrasal and non-textual attributes may improve spam recognition performance (Androutsopoulos et al., 2000). However, they introduce a manual configuration phase. Because our target was to explore fully automatic anti-spam filtering, we limited ourselves to word-only attributes.

Finally, some data cleaning techniques are required after converting raw data into appropriate format. In particular, to deal with missing values, the simplest approach is to delete all instances where there is at least one missing value and use the remainder. This strategy has the advantage of avoiding introducing any data errors. Its main problem is that discard of data many damage the reliability of the resulting classifier. Moreover, the method cannot be used when a high proportion of instances in the training set have missing values. Together, these weaknesses are quite substantial. Although it may be worth trying when there are few missing values in the dataset, this approach is generally not recommended. Instead, we use an alternative strategy in which any missing values of a categorical attribute are replaced by its most commonly occurring value in the training set. For continuous attributes, missing values are replaced by its average value in the training set.

4.1.2. Phase 2: Feature Transformation

The tremendous growth in computing power and storage capacity has made today's databases, especially for text categorization tasks, contain very large number of attributes. Although faster processing speeds and larger memories may make it possible to process these attributes, this is inevitably a losing struggle in the long term. Besides degraded performance, many irrelevant attributes will also place an unnecessary computational overhead on any data mining algorithm. There are several ways in which the number of attributes can be reduced before a dataset is processed. In this research, a *dimension reduction* (also called *feature pruning* or *feature selection*) scheme called *Principal Component Analysis* (PCA) (Jolliffe, 2002) is performed on the data to select the most relevant features. This is necessary given the very large size and correlated nature of the input vectors. PCA

eliminates highly correlated features and transforms the original data into lower dimensional data with most relevant features. From our observation, the selected features are words that express the distinction between spam and non-spam groups, i.e. they are either common in spam or legitimate messages, not in both. Several punctuation and special symbols (e.g. "$", "@") are also selected by PCA, and therefore, they are not eliminated during preprocessing.

4.1.3. Phase 3: Email Classification

The data after being processed by the Feature selection module is input to train the Classification Model. The resulting model is then used to label emails as either "legit" or "spam", indicating whether a message is classified as legitimate or a spam email. To implement the Classification Model, we propose an intelligent way of combining the linear part of the modeling with a simple non-linear model algorithm. In particular, MPNN is adapted in the nonlinear compensator which will only model higher ordered complexities while linear model will dominate in case of data far away from training clusters. It is described in the following equation.

$$\hat{y}(X) = L(\underline{x}) + NL(\underline{x}) = [w_0 + \underline{Wx}] + [\frac{\sum Z_i \alpha_i y_i f_i(\underline{x} - \underline{c_i}, \delta)}{\sum Z_i f_i(\underline{x} - \underline{c_i}, \delta)}]$$

Where

$L(\underline{x})$ = Linear Regression Model

$NL(\underline{x})$= Nonlinear Residual Compensator (MPNN)

w_0 = initial offset

w_i = weights of the linear model

$\alpha_i = \frac{y_{Ni}}{d_1}$ = Compensation factor

$y_{Ni} = y_i - [w_0 + W_i \underline{x_i}]$ = difference between the linear approximation and the training output

$d_i = ||\underline{x} - \underline{c_i}||$ = distance from the input vector to cluster *i* in the input space

The combination of linear regression model and MPNN is referred to as Linear Regression – Modified Probabilistic Neural Network (LR-MPNN). The piecewise linear regression model is firstly approximated by using all available training data in a simple regression fitting analysis. The MPNN is then constructed to compensate for higher ordered characteristics of the problem. Depending on different portions of the training set and how far the test data is away from the training data, the impact of nonlinear residual function is adjusted such that the overall Mean Square Error is minimized. This adjustment is formulated by the *compensation factor* α_i. In particular, $\alpha_i$is computed based on how well the linear model performs on a training examples and distances from a test vector to clusters of training data.

Firstly, the goodness of the linear model $L(\underline{x})$ on a particular training data is measures by y_{Ni} which is defined as the difference between the linear approximation and the actual output of the training data. A very small value of y_{Ni} means that $L(\underline{x})$ fits the data well in this case and therefore it should have higher priority or the impact of the nonlinear model $NL(\underline{x})$ is minimized. In contrast, large value of y_{Ni} indicates that $NL(\underline{x})$ should compensate more for the degraded accuracy of $L(\underline{x})$.

Secondly, to determine how far a given test vector is away from the available training data, a distance d_i from that vector to each training cluster $\underline{c_i}$ is calculated. For any data which is

far away from the training set, i.e. d_i is large, the value of α_i will be minimized. As the result, $NL(\underline{x})$will have minimal residual effect and $L(\underline{x})$ will dominate. This is because $L(\underline{x})$ has more stable generalization than $NL(\underline{x})$ for new instances.

4.1.4. Phase 4: Evaluation

To evaluate the overall performance of the framework, the Cost-sensitive Evaluation module computes several performance metrics and also takes into consideration different cost scenarios.

4.2. Performance Evaluation

4.2.1. Performance Measures

To measure the performance of different learning algorithms, the following measures are used:

$$SPAM\ RECALL\ (SR) = \frac{n_{S\to S}}{n_{S\to S} + n_{S\to L}}$$

$$SPAM\ PRECISION\ (SP) = \frac{n_{S\to S}}{n_{S\to S} + n_{L\to S}}$$

$$FAR(False\ Alarm\ Rate) = \frac{n_{S\to L}}{N_S}$$

$$Miss\ Rate(MR) = \frac{n_{L\to S}}{N_L}$$

From the above equations, Spam Recall (SR) is, in fact, the percentage of spam messages ($N_S = n_{S\to S} + n_{S\to L}$) that are correctly classified ($n_{S\to S}$) while Spam Precision (SP) compares the number of correct spam classifications ($n_{S\to S}$) to the total number of messages classified (correctly and incorrectly) as spam ($n_{S\to S} + n_{L\to S}$). As the Miss Rate (MR) increases, the number of misclassifications of legitimate emails increases while the False Alarm Rate (FAR) increases, the number of misclassifications of spam emails (passing from the filter) increases. Therefore, both of FAR and MR should be as small as possible for a filter to be effective (should be 0 for a perfect filter).

4.2.2. Cost-Sensitive Analysis

a) Cost Scenarios

Depending on what action is taken by a spam filter in response to a detected spam message, there are three major misclassification cost scenarios. The *no-cost* case is when the filter merely flags a detected spam message. This notification of spam does not risk losing any legitimate mail due to misclassification error (no misclassification cost), but it still takes time for the human users to check and delete the spam messages manually. To minimize the user efforts on eliminating spam, the filter can automatically detect and remove the suspicious messages. However, the total cost of misclassification in this case can be extremely high due to the seriousness of falsely discarding legitimate mails. This refers to the *high-cost* scenario.
Beside the above approaches, the filter may not either flag or completely eliminate the detected spam messages. Instead, it might resend the message to the sender. This approach, referred to as *moderate-cost*, combats spamming by increasing its cost via *Human Interactive Proofs* (HIP) (L. F. Cranor & LaMacchia, 1998). That is, the sender is required to give a proof of humanity that matches a puzzle before his message is delivered. The puzzles could be, for

example, images containing some text that is difficult to automatically analyze by pattern recognition software. Alternatively, for anti-spam programs, simple questions (e.g. "what is one plus one") can be used instead of graphical puzzles.

The concept of HIP has been implemented in many security related applications. For example, certain web-based email systems use HIP to verify that password cracking software is not systematically brute-forcing to guess a correct password for email accounts. When a user types his password wrong three times, a distorted image is presented that contains a word or numbers and the user must verify before being allowed to continue. A human can easily convert the image to text, but the same task is extremely difficult for a computer. Some email client programs have anti-spam filtering heuristics using HIP implemented. When such programs receive an email that is not in the white-list of the user, they send the sender a password. A human sender can then resend the email containing the received password. This system can effectively defeat spammers because spam is *bulk*, meaning that the spammers do not bother to check replies manually or commonly use a forged source email address. The cost of creating and verifying the proofs is small, but they can be computationally impossible for automated mass-mailing tools to analyze. Though spammers can still use human labor to manually read and provide the proofs and finally have their spam message sent. HIP actually restricts the number of unsolicited messages that the spammer can send for a certain period of time due to the inability to use cheap automated tools (Carreras & Marquez, 2001). This barrier for spammers effectively introduces additional cost to sending spam messages.

In this chapter, spam recognition experiments are conducted in a cost-sensitive manner. As emphasized previously, misclassifying a legitimate message as spam is generally more severe than mistakenly recognizing a spam message as legitimate. Let $L \rightarrow S$ (legitimate classified as spam) and $S \rightarrow L$ (spam classified as legitimate) denote the two types of error, respectively. We invoke a decision-theoretic notion of cost, and assume that $L \rightarrow S$ is λ times more costly than $S \rightarrow L$. A mail is classified as spam if the following criterion is met (Androutsopoulos et al., 2000):

$$\frac{Prob(C = spam|\vec{X} = \vec{x})}{Prob(C = legitimate|\vec{X} = \vec{x})} > \lambda$$

In the case of anti-spam filtering:

$$Prob\left(C = spam\middle|\vec{X} = \vec{x}\right) = 1 - Prob\left(C = legitimate\middle|\vec{X} = \vec{x}\right)$$

The above criterion becomes:

$Prob\left(C = spam\middle|\vec{X} = \vec{x}\right) > t$,with $t = \frac{\lambda}{1+\lambda}, \lambda = \frac{t}{1-t}$

Depending on which cost scenarios are considered, the value of λ is adjusted accordingly.

- No-cost scenario (e.g. flagging spam messages): $\lambda = 1$
- Moderate-cost scenario (e.g. semi-automatic filter which notifies senders about blocked messages): $\lambda = 9$
- High-cost scenario (e.g. automatically removing blocked messages): $\lambda = 999$

b) Total Cost Ratio

Accuracy and error rates assign equal weights to the two error types ($L \rightarrow S$, $S \rightarrow L$) and are defined:

$Acc = \frac{n_{L\rightarrow L}+n_{S\rightarrow S}}{N_L+N_S}$ $\quad$ $Err = \frac{n_{L\rightarrow S}+n_{S\rightarrow L}}{N_L+N_S}$

However, in the cost-sensitive contexts, the accuracy and error rates should be made sensitive to the cost difference, i.e. each legitimate message is counted for λ times. That is,

when a legitimate message is misclassified, this counts as λ errors; and when it passes the filter, this counts as λ successes. This leads to the definition of *weighted accuracy* and *weighted error* (*WAcc and WErr*):

$$WAcc = \frac{\lambda.n_{L\to L}+n_{S\to S}}{\lambda.N_L+N_S} \qquad WErr = \frac{\lambda.n_{L\to S}+n_{S\to L}}{\lambda.N_L+N_S}$$

The values of performance measures (weighted or not) are misleadingly high. To get a true picture of the performance of a spam filter, its performance measures should be compared against those of a "baseline" approach where no filter is used. Such a baseline filter never blocks legitimate messages while spam emails always pass through the filter. The weighted accuracy and error rates for baseline are:

$$WAcc^b = \frac{\lambda.N_L}{\lambda.N_L+N_S} \qquad WErr^b = \frac{N_S}{\lambda.N_L+N_S}$$

Total cost ratio (*TCR*) is another measure which evaluates performance of spam filter to that of a baseline.

$$TCR = \frac{WErr^b}{WErr} = \frac{N_S}{\lambda.\, n_{L\to S} + n_{S\to L}}$$

Greater *TCR* values indicate better performance. For *TCR* < 1, the baseline is better. If cost is proportional to wasted time, a *TCR* is intuitively equivalent to measuring how much time is wasted to manually delete all spam messages when the filter is used (N_S) compared to the time wasted to manually delete any spam messages that passed the filter ($n_{S\to L}$) plus the time needed to recover from mistakenly blocked legitimate messages ($\lambda.\, n_{L\to S}$)

5.Experiment Results

5.1. Experiment Design

The proposed spam recognition framework is tested on the Ling-Spam corpus to compare with other existing learning methods including Naïve Bayes (NB), Weighted Memory Based Learning (WMBL), Boosted Trees (BT), Support Vector Machine (SVM) and Neural Network models (Multilayer Perceptron – MLP. Unlike other text categorization tasks, filtering spam messages is cost sensitive (Cohen, 1996), hence evaluation measures that account for misclassification costs are used. In particular, we define a cost factor λ with different values corresponding to three cost scenarios: first, no cost considered ($\lambda = 0$) e.g. marking messages as spam; second, semi-automatic filtering ($\lambda = 9$) e.g. issuing a notification about spam; and fully automatic filtering ($\lambda = 999$), e.g. discarding the spam messages.

The rate at which a legitimate mail is misclassified as spam is calculated by False Alarm Rate (*FAR*) and it should be low for a filter to be useful. Spam Recall (*SR*) measures the *effectiveness* of the filter, i.e. the percentage of messages correctly classified as spam, while Spam Precision (*SP*) indicates the filter's *safety*, i.e. the degree to which the blocked messages are truly spam. Because *SR* can be derived from *FAR* (e.g. *FAR = 1 – SR*), we will use *SR*, *SP*, and Total Cost Ratio (TCR) for evaluation. Besides comparing how accurately the filters perform, their computation is also measured using the computation time (in seconds) required for each classifier. Particularly, the total computation time is a summation of the time that a classifier needs to perform cross validation, testing on data and to calculate the relevant performance metrics (e.g. misclassification rate, accuracy …).

Stratified tenfold cross validation is employed for all experiments. That is, the corpus is partitioned into 10 stratified parts and each experiment was repeated 10 times, each time

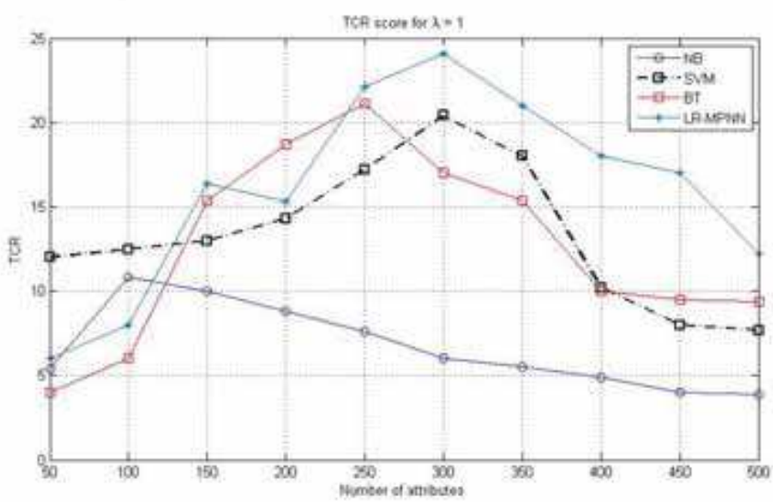
reserving a different part as the testing set and using the remaining 9 parts as the training set. Performance scores are then averaged over the 10 iterations.

In addition to the studies conducted by other researchers on the same Ling-Spam corpus (NB (Androutsopoulos et al., 2000), WMBL (Sakkis et al., 2003), SVM (Hsu et al., 2003), BT (Carreras & Marquez, 2001)), we also reproduced their experiments (based on the average value of *TCR* of three cost scenarios) to confirm and determine the parameters' values that give best performance for different learning methods. The optimal attribute size of these methods can be found in Figure 2. An MLP wth 15 neurons in hidden layer is deployed using the Matlab Neural Network toolbox.

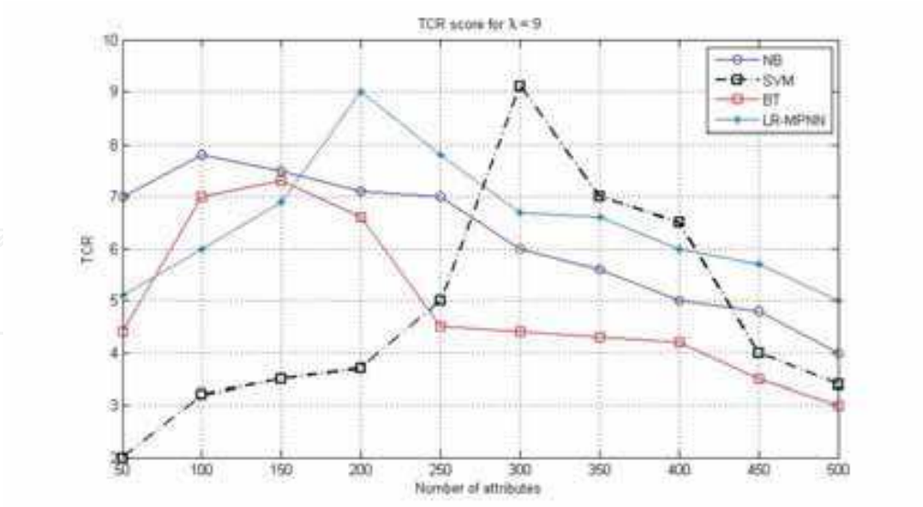
5.2. Experiment Result

5.2.1. TCR and Attribute Selection

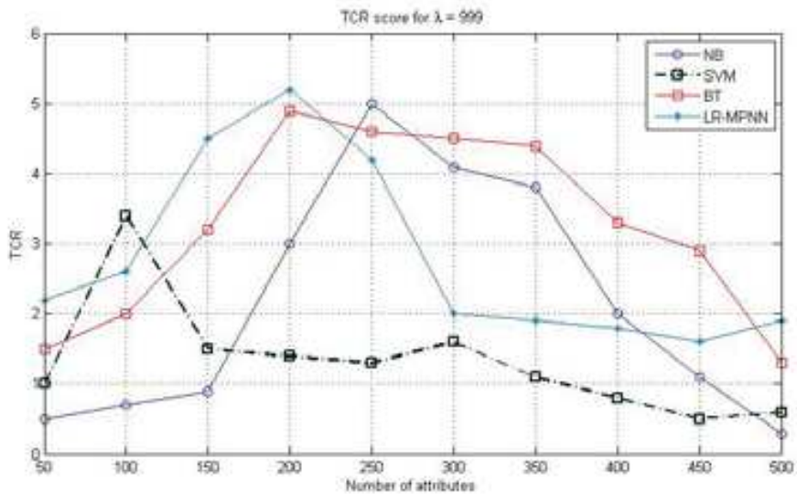
From Figure 2, for $\lambda = 1$ and $\lambda = 9$, most of filters demonstrate a stable performance, with TCR constantly greater than 1. These filters differ from one another in terms of their sensitivity on attribute selection and the number of attributes which give maximum TCR. Our LR-MPNN is found to be moderately sensitive to attribute selection and it obtains the highest TCR for $\lambda = 1$ with 300 attributes selected. When $\lambda = 9$, LR-MPNN achieves very competitive TCR compared to SVM but with less number of attributes (200 attributes) and hence involves less computation overheads.



a)



b)



c)

Fig. 2. TCR score of spam recognition methods

For $\lambda = 999$, all classifiers have their TCR reduced significantly for the effect of very high misclassification cost. The difference between low and high values of misclassification cost λ is the increased performance of the baseline filter when λ increases. That is, without a filter in use (baseline), all legitimate mails are retained, preventing the baseline from misclassifying those legitimate mails as spam. Therefor, large λ benefits the baseline and make it hard to be defeated by other filters. Recall that TCR is the measure of performance that a filter improves on the baseline case. As a result, TCR generally reduces when λ increases. Another important observation is that, the performance of most classifiers, except for BT and LR-MPNN, fall below the base case (TCR<1) for some numbers of selected attributes. This is due to the relative insensitivity of BT and LR-MPNN to attribute selection. In this case, the LR-MPNN is considered to be the best performing filter with the highest TCR.

5.2.2. Spam Precision and Spam Recall

In this experiment, the classifiers are run iteratively by a tenfold cross-validation process. The SP ans SR rates are recorded in Table 1. We observe that, for the no-cost scenario ($\lambda = 1$), our method, LR-MPNN, is found to have best SP while its SR (0.958) is very similar to the highest SR of NB (0.959). For $\lambda = 9$, LR-MPNN obtains the highest SR (0.869) and second highest SP (0.991) after BT algorithm. Finally, in the case of extremely high misclassification cost ($\lambda = 999$), LR-MPNN significantly outperforms other methods with all evaluation metrics are of highest values.

Method	λ =1		λ =9		λ =999	
	SR	**SP**	**SR**	**SP**	**SR**	**SP**
NB	**0.959**	0.973	0.861	0.975	0.790	0.984
WMBL	0.860	0.917	0.790	0.982	0.601	0.857
SVM	0.954	0.981	0.847	0.983	0.671	0.995
BT	0.957	0.980	0.864	**0.993**	0.768	0.996
MLP	0.852	0.975	0.782	0.977	0.623	0.979
LR-MPNN	*0.958*	***0.986***	***0.869***	*0.991*	***0.793***	***0.998***

Table 1. Precision/Recall evaluation on Ling-Spam data

5.2.3. Computational Efficiency

Apart from comparing precision, recall and TCR scores between classifiers, we also measure their computational efficiency. Table 2 shows that WMBL had the minimum computation time (2.5 mins), followed by NB, LR-MPNN, SVM, MLP, BT respectively. LR-MPNN can achieve comparative spam precision and recall with a shorter computation time (3.5 mins) compared with BT (11.5 mins) and SVM (5 mins). Moreover, considering TCR scores, the models that require less time (WMBL, NB) than LR-MPNN do not perform as accurately as LR-MPNN.

Method	Computation Time (mins)	λ =1	λ =9	λ =999
		TCR	TCR	TCR
NB	3	10.80	7.80	5.02
WMBL	**2.5**	7.11	5.62	1.33
SVM	5	20.47	9.11	3.42
BT	11.5	21.18	7.35	4.91
MLP	7	12.20	4.50	0.25
LR-MPNN	3.5	24.17	8.99	5.25

Table 2. Computation Time, Memory size evaluation on Ling-Spam data

In summary, the most important finding in our experiment is that the proposed LR-MPNN model can achieve very accurate classification (high TCR, SP, SR) compared to other conventional learning methods. Such superior performance of LR-MPNN was observed most clearly for $\lambda = 999$ though it always obtains the highest TCR and very competitive SP, SR rates for other cases of λ. Our algorithm also requires relatively small computation time to obtain comparable or even higher predictive accuracy to other methods.

6. Conclusions and Future Work

In this chapter, we proposed a novel anti-spam filtering framework in which appropriate dimension reduction schemes and powerful classification models are employed. Particularly, Principal Component Analysis transforms data to a lower dimensional space. At the classification stage, we combine a simple linear regression model with a lightweight nonlinear neural network in an adjustable way. This learning method, referred to as *Linear Regression Modified Probabilistic Neural Network* (LR-MPNN), can take advantage of the virtues of both. That is, the linear model provides reliable generalization capability while the nonlinear can compensate for higher order complexities of the data. A cost-sensitive evaluation using a publicly available corpus, Ling-Spam, has shown that our LR-MPNN classifier compares favorably to other state-of-the-art methods with superior accuracy, affordable computation and high system robustness. Especially for extremely high misclassification cost, while other methods' performance deteriorates as λ increases, the LR-MPNN demonstrates an absolutely superior outcome but retains low computation cost. LR-MPNN also has significant low computational requirement, i.e. its training time is shorter than other algorithms with similar accuracy and cost. Though our proposed model achieves good results in the conducted experiments, it is not necessarily the best solution for all problems. However, comparatively high predictive accuracy along with low computational complexity distinguish it from other state-of-the-art learning algorithms, and particularly suitable for cost-sensitive spam detection applications.

7. References

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., Paliouras, G., & Spyropoulos, C. D. (2000). An Evaluation of Naive Bayesian Anti-Spam Filtering. Paper presented at the Proc. of the ECML.

Bayler, G. (2008). Penetrating Bayesian Spam Filters: VDM Verlag Dr. Mueller e.K.

Carreras, X., & Marquez, L. (2001). Boosting Trees for Anti-Spam Email Filtering. Paper presented at the RANLP, Tzigov Chark, Bulgaria.

Cohen, W. (1996). Learning rules that classify email. AAAI Sump. On Machine Learning in Inf. Access, 18-25.

Cranor, L. F., & LaMacchia, B. A. (1998). Spam! Paper presented at the Communications of ACM.

Cranor, L. F., LaMacchia, B.A. . (1998). Spam! Paper presented at the Communications of ACM.

Drucker, H., Wu, D., & Vapnik, V. N. (1999). Support Vector Machines for Spam Categorization. IEEE Transactions On Neural Networks, 1048-1054.

Hayes, M. H. (1996). Statistical Digital Signal Processing and Modeling: John Wiley & Sons, Inc.

Hsu, C. W., Chang, C. C., & Lin, J. C. (2003). LIBSVM: a library for support vector machines. from http://ww.csie.ntu.edu.tw/~cjlin/libsv

Jolliffe, I. T. (2002). Principle Component Analysis (2 ed.). New York, USA: Springer.

Rumelhart, D. E., & McClelland, J. L. (1986). Parallel Distributed Processing: Explorations in the Microstructure of Cognition (Vol. 2): The MIT Press. .

Sahami, M., S. Dumais, D. Heckerman, and E. Horvitz. (1998). A Bayesian Approach to Filtering Junk E-Mail. Paper presented at the Learning for Text Categorization – AAAI Technical Report WS-98-05.

Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D., & Stamatopoulos, P. (2003). A Memory-Based Approach to Anti-Spam Filtering for Mailing Lists. Paper presented at the Information Retrieval.

Schapire, R. E., & Singer, Y. (2000). BoosTexter: a Boosting-Based System for Text Categorization. .Machine Learning, 39(2), 135–168.

Schryen, G. (2007). Anti-Spam Measures: Analysis and Design: Springer.

Surepayroll. (2007). More than 80 Percent of Small Business Owners Consider E-mail Essential to Success, SurePayroll Insights Survey.

Swartz, N. (2003). Spam costs businesses $13 billion annually. Information Management Journal, 37(2).

Wolfe, P., Scott, C., & Erwin, M. (2004). Anti-Spam Tool Kit: McGraw-Hill Osborne Media.

Zaknich, A. (1998). Introduction to the modified probabilistic neural network for general signal processing applications. IEEE Transactions on Signal Processing, 46(7), 1980-1990.

Zaknich, A. (2003). Neural Networks for Intelligent Signal Processing. Sydney: World Scientific Publishing.

IntechOpen

Pattern Recognition
Edited by Peng-Yeng Yin

ISBN 978-953-307-014-8
Hard cover, 568 pages
Publisher InTech
Published online 01, October, 2009
Published in print edition October, 2009

For more than 40 years, pattern recognition approaches are continuingly improving and have been used in an increasing number of areas with great success. This book discloses recent advances and new ideas in approaches and applications for pattern recognition. The 30 chapters selected in this book cover the major topics in pattern recognition. These chapters propose state-of-the-art approaches and cutting-edge research results. I could not thank enough to the contributions of the authors. This book would not have been possible without their support.

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

IntechOpen