A study of boosted evolutionary classifiers for detecting spam

Shrawan Kumar Trivedi
Indian Institute of Technology (ISM) Dhanbad, Dhanbad, India, and
Shubhamoy Dey
Indian Institute of Management Indore, Indore, India

Received 4 June 2019
Revised 5 September 2019
Accepted 5 September 2019

Abstract

Purpose – Email is a rapid and cheapest medium of sharing information, whereas unsolicited email (spam) is constant trouble in the email communication. The rapid growth of the spam creates a necessity to build a reliable and robust spam classifier. This paper aims to presents a study of evolutionary classifiers (genetic algorithm [GA] and genetic programming [GP]) without/with the help of an ensemble of classifiers method. In this research, the classifiers ensemble has been developed with adaptive boosting technique.

Design/methodology/approach – Text mining methods are applied for classifying spam emails and legitimate emails. Two data sets (Enron and SpamAssassin) are taken to test the concerned classifiers. Initially, pre-processing is performed to extract the features/words from email files. Informative feature subset is selected from greedy stepwise feature subset search method. With the help of informative features, a comparative study is performed initially within the evolutionary classifiers and then with other popular machine learning classifiers (Bayesian, naive Bayes and support vector machine).

Findings – This study reveals the fact that evolutionary algorithms are promising in classification and prediction applications where genetic programing with adaptive boosting is turned out not only an accurate classifier but also a sensitive classifier. Results show that initially GA performs better than GP but after an ensemble of classifiers (a large number of iterations), GP overshoots GA with significantly higher accuracy. Amongst all classifiers, boosted GP turns out to be not only good regarding classification accuracy but also low false positive (FP) rates, which is considered to be the important criteria in email spam classification. Also, greedy stepwise feature search is found to be an effective method for feature selection in this application domain.

Research limitations/implications – The research implication of this research consists of the reduction in cost incurred because of spam/unsolicited bulk email. Email is a fundamental necessity to share information within a number of units of the organizations to be competitive with the business rivals. In addition, it is continually a hurdle for internet service providers to provide the best emailing services to their customers. Although, the organizations and the internet service providers are continuously adopting novel spam filtering approaches to reduce the number of unwanted emails, the desired effect could not be significantly seen because of the cost of installation, customizable ability and the threat of misclassification of important emails. This research deals with all the issues and challenges faced by internet service providers and organizations.

Practical implications – In this research, the proposed models have not only provided excellent performance accuracy, sensitivity with low FP rate, customizable capability but also worked on reducing the cost of spam. The same models may be used for other applications of text mining also such as sentiment analysis, blog mining, news mining or other text mining research.

Originality/value – A comparison between GP and GAs has been shown with/without ensemble in spam classification application domain.

Keywords Genetic algorithms, Genetic programming, Support vector machine, Bayesian classifier, Naive bayes, Classification accuracy, *F*-value, False positive rate, Greedy stepwise search, AdaBoost

Paper type Research paper

1. Introduction

In today's electronic world, information sharing between the units of an organization is necessary to be competitive and sustainable in business. Email is an essential and useful

Global Knowledge, Memory and Communication
Vol. 69 No. 4/5, 2020
pp. 269-287

2514-9342
DOI 10.1108/GKMC-05-2019-0060

tool for rapid and cheap communication. It is now a popular medium for connecting people with each other. On the other hand, spam (also known as unsolicited bulk email) is a challenge for the researcher because its size is increasing day by day. A study has estimated that 70 per cent of business emails are spam (Aviv). This rapid growth has caused serious issues, such as user mailboxes filling up unnecessarily, important emails being engulfed by unimportant ones, storage space and bandwidth is limited, and the process of sorting emails being excessively time-consuming (Trivedi and Dey, 2013a, 2013b, 2013c, 2013d).

At present, there is increased research interest in spam classification because of the complexity introduced by spammers, who make it difficult to distinguish between spam (unsolicited emails) and ham (legitimate emails). Complexity may be because of attacks such as tokenization (splitting or modifying a feature, for example, writing "free" as "f r 3 3") and obfuscation (hiding features by adding hyper text markup language or other codes, for example, coding "free" as "frand#101xe" or as "FR3E") that alter information on particular features (Goodman *et al.*, 2007; Lai, 2007).

Various machine learning classifiers have been tested and have performed well in tackling the current classification problems. In particular, support vector machine (SVM), Bayesian filtering and naive Bayes (NB) have performed satisfactorily in detecting spam, although performance accuracy is still a matter of debate. Nowadays, the interest of researchers is moving toward evolutionary algorithms (Cantú-Paz, 2007; Raymer *et al.*, 2000; Trivedi and Dey, 2013a, 2013b, 2013c, 2013d) because of their potential for feature selection and for classification.

In this study, a comparative analysis is performed between two classifiers, genetic algorithm (GA) and genetic programming (GP), with and without boosting. The results of these algorithms are then compared with those of other machine learning classifiers (Bayesian, NB and SVM).

GA is continuously tested in the literature because of its interesting operators that obtain new individuals by combining old individuals with the help of evolutionary principles such as selection (searching the fittest individual) and reproduction (merging and altering old individuals). At the same time, GP, which also uses evolutionary learning techniques, is attracting interest from researchers in this domain (Banzhaf, 1998). GP is a heuristic technique that permits complex representations of patterns (for example, in the form of decision trees [DT]). One characteristic of GP that makes it promising for classification purposes is its flexibility, which allows it to adapt techniques according to the requirements of a particular problem.

Boosting algorithms create a single classifier from a number of weak classifiers (Bauer and Kohavi, 1999). In this study, the AdaBoost technique is used for boosting, and the GreedyStepwise search is used, as it is the feature selection technique that obtains the most informative feature subset.

The rest of this paper has been organized in the following way: Section 2 summarizes previous work connected with this research. Section 3 describes the structure of a spam classifier. Section 4 presents the machine learning classifiers used in this study. Section 5 discusses the results of this research, and Section 6 presents its conclusions.

2. Related work

A number of studies have been carried out in the area of text classification, and various machine learning classifiers have shown their potential. This study focuses on email spam classification, an area where evolutionary algorithms are an appealing choice. This section summarizes the literature related to relevant classifiers that have been tested on various data sets.

A plethora of research has been conducted into methods of robust email spam detection (Fdez-Glez *et al.*, 2016), including SVM (Cortes and Vapnik, 1995), C4.5 DT (Quinlan, 1993), Bayesian networks (Friedman *et al.*, 1997) and random decision forests (RF) (Breiman, 2001). Among these, machine learning algorithms, SVM and RF are the most commonly used models (Patil and Patil, 2015; Jia *et al.*, 2012; Meda *et al.*, 2016).

Probabilistic classifiers (Bayesian and NB) are popular in the area of classification. They provide a cost-sensitive evaluation by observing the degree of confidence of the classification. Sahami *et al.* (1998) incorporated an NB classifier and a bag-of-words (BoW) approach to represent features of an email data set. They included some complex features (such as "FREE!" and "f r 3 3"), domain name features, and non-alphabetic features to improve classification accuracy. The iFile filter (Rennie, 1998) uses NB to develop appropriate folders for email; it constructs three different folders for each email according to the features selected by the use of stemming, removal of stop words and implementation of document frequency thresholds. The SpamCop system (Pantel and Lin, 1998) also uses an NB approach to spam filtering. Parveen and Halse (2016) carried out a comparative analysis of various data mining techniques in the spam classification application domain, and they suggest that an NB classifier provides good accuracy in comparison to the other classifiers tested in their study.

SVM is a promising classifier for text and email. A comparative study by Drucker *et al.* (1999) compared SVM with various machine learning classifiers and predicted that SVM and boosted DT were the best classifiers in terms of performance accuracy and speed (although boosted DT was slower than SVM). Trivedi and Dey (2013a, 2013b, 2013c, 2013d) observed the effects of different kernel functions for improving the learning capability of SVM and identified NormalizedPolyKernel as the best.

Nowadays, evolutionary classifiers GA (Liu *et al.*, 2003; Li, 2009) and GP (Shengen *et al.*, 2011) are being widely tested in classification research because of their capacities for feature selection and classification. These classifiers work with interesting rules performed by the reproduction, mutation and crossover operators. A comparative study by Trivedi and Dey (2013a, 2013b, 2013c, 2013d) compared GA with various other machine learning classifiers. Also, in 2013, Trivedi and Dey implemented an enhanced GP approach and compared it with various machine learning classifiers; in this case, enhanced GP was found to be the best.

A further study by Trivedi and Dey (2013a, 2013b, 2013c, 2013d) focused on boosting approaches and their effects on probabilistic classifiers; they found that this method helps classifiers by boosting their performance, even with a small subset of the most informative features. Datta *et al.* (2015) worked on video traffic classification with J48 and AdaBoost, where they did experiments on Google Hangout data with 2.5 packet collection and obtained 99.99 per cent accuracy. Gashti (2017) tested various DT-Classification & Regression Trees (CART), SVM, NB and multilayer perceptron for spam email classification and obtained accuracy rates of between 87.05 and 100 per cent for the CART algorithm. Work by Shah and Kumar (2018) incorporating the ID3 DT and other machine learning models obtained good accuracy of up to 83.92 per cent. Goh and Singh (2015) proposed RF integrated with the real AdaBoost algorithm (Schapire *et al.*, 1998) and used it in the classification of webspam; the results showed that performance had been greatly improved by the integration of AdaBoost.

3. Machine learning classifiers

3.1 Boosting with AdaBoost

The origin of boosting techniques is bootstrapping (Trivedi and Dey, 2013a, 2013b, 2013c, 2013d; Wu, 1986). The fundamental purpose of bootstrapping is to reassess the accuracy of

an estimate. It is a statistical sample-based method that consists of drawing randomly from a data set with replacement. In the classification domain, some boosting algorithms have shown their potential in terms of strengthening the accuracy of the classifiers.

Adaptive boosting works to reweight the data as an alternative to random sampling. This technique develops a concept of building ensembles for improving the performance of the classifiers. AdaBoost (Trivedi and Dey, 2013a, 2013b, 2013c, 2013d) learn with the collection of output M_x of weak classifiers $G_t^{m_x}$ and then predict a decision, which forms the final classifier G_t^x.

3.2 Algorithm for boosting classifiers

Input: Training set $T_y = t_1, t_2, t_3 \ldots t_n$ with $t_i = (x^i, y^i)$; number of sample version of training set B

Output: An appropriate classifier for the training set G_t^x

I. Initialize the weights $w_i^t = \frac{1}{N}, i \in \{1, 2, 3, \ldots N\}$.

II. From $m = 1, 2, 3, \ldots M_x$

a) train the weak classifier $G_t^{m_x}$ with the training outset using weights w_i^t

b) calculate the error term $E_{rror}^m = \dfrac{\sum_{i=1}^{N} w_i^t I\left(y_i \neq G_t^{m_{xi}}\right)}{\sum_{i=1}^{N} w_i^t}$

c) calculate weight contribution $\theta_m = 0.5 \log\left(\frac{1-E_{rror}^m}{E_{rror}^m}\right)$

d) substitute $w_i^t \leftarrow w_i^t Exp\left(-\theta_{(m)} I\left(y_i \neq G_t^{m_{xi}}\right)\right)$ and Renormalize $\sum_i w_i^t = 1$.

III. The final classifier is:

$$G_t^x = \theta_m sign\left(\sum_{m=1}^{M_x} G_t^{m_x}\right). \tag{1}$$

3.3 Genetic algorithm-based classifiers

GA uses the concept of iterative learning that was initially proposed by Holland (1975). This algorithm works on a principle similar to that of genetic models of natural systems. Initially, it uses individuals from a constant population to search a sample of space. A fitness function is developed to evaluate each individual. Thereafter, the following operators are used to produce new individuals:

Selection operator: This operator selects outperforming individuals from the set of individuals to produce "offspring" (Holland, 1975).

Crossover operator: This operator selects a random point within the two-parent gene structure and exchanges parts of the parent to create new individuals. This operator thus generates two high-performing individuals by combining the properties of two old individuals.

Mutation operator: This operator obtains a new individual by arbitrarily altering the properties of an old individual. It works on the concept of population perturbation theory, which involves providing new information to the population. It also helps to evade any stagnation that may occur during the search process.

3.4 Genetic programming-based classifiers

GP is usually represented by a tree-based structure (Kishore *et al.*, 2000) in which a binary tree is used to represent an individual. A non-leaf node of the tree serves as the operation carried out on terminals of leaf nodes. Two types of the terminal can be identified: feature terminals and numeric/constant terminals. Feature terminals are basically identified as transformed link-based features selected from the training set of ham and spam web emails, such as log of in degree, log of out-degree or page rank. Numeric or constant terminals are some predefined number in the range from 0.0 to 1.0. The simple arithmetic operations $+$, $-$, $\times$ and $\div$ are performed by internal nodes where $+$, $-$ and $\times$ operators simply perform according to their basic meanings but the $\div$ operator is reserved for "protected" division (i.e. when divided by zero, the result is zero). Some additional operators can be used, such as *log, sin* and *cosine*. It has been shown that the use of simple arithmetic operators achieves higher accuracy and significantly reduces the cost of computation (Kishore *et al.*, 2000).

Tree-based GP follows predefined rules that are divided into two parts: antecedent and consequent. The antecedent works with a collection of conditions for predicted features and the consequent includes predicted classes. Generally, a condition can be performed by binary relational operators ($=$, $\neq$, $<$, $>$, $\leq$ and $\geq$) and used to compare the value of one feature with another feature. GP individuals can use any kind of operator, and hence, a complex condition can be handled easily. GP uses a discriminant function that is defined by some mathematical expression in which different kinds of operator and function are applied to the features of the data instances that are to be classified. Mutation and crossover operators are used to recombine individuals to create new individuals.

3.5 Fitness function

The fitness function is calculated to evaluate the fitness of each individual; hence, an individual must be capable of classifying a set of instances. The fitness function $F(\mathrm{I}^x)$ for any individual I^x can be represented as $A_{ccuracy}(\mathrm{I}^x, \mathrm{D}^{Emails})$ and defined in the following equation (2):

$$A_{ccuracy}\left(\mathrm{I}^x, \mathrm{D}^{Emails}\right) = \frac{The \text{ number of examples of } D^{Emails} \text{ that are correctly classified by } \mathrm{I}^x}{|D^{Emails}|} \tag{2}$$

where D^{Emails} is the email data set.

3.6 Algorithm for genetic programming

Input: Training set T^x, number of individuals N^x, maximum depth of binary tree D^x and number of generations R^x

Output: A best individual with a unique discriminant function

I. Initialize population P^x from randomly generated individuals with respect to N^x and D^x.

II. Assign values for operators mutation ($m^x = 0.07$), crossover ($c^x = 0.9$), new program ($N_p = 0.03$), and reproduction ($r_p = 0.0$).

III. Compute fitness function $F(\mathrm{I}^x)$ for individuals I^x, where $\mathrm{I}^x \in P^x$ with training set T^x.

IV. Execute genetic operators:

a. Reproduction operator: choose the two fittest individuals from the population P^x and include them in the new population P_n^x.

b. Mutation operator: apply to a randomly selected individual to compute its fitness for mutation, compare it with the best fit individual, and put it in the new population P_n^x.
c. Crossover operator: Apply to the two fittest individuals selected above. The properties of the parent individuals are exchanged to form two offspring. Compute the fitness value of these new offspring, compare, and then put them in the new population P_n^x.
d. Repeat until $|P^x| < N^x$.

V. Let $P^x = P_n^x$, $P_n^x = \phi$, and $r_p = r_p + 1$.
VI. Repeat until $r_p < R_x$.
VII. Evaluate and compare the fitness function of every individual in population P_x with training set Tx to observe the best output.

3.7 Probabilistic classifiers

The idea of probabilistic classifiers was introduced by Lewis (1998), who suggested a term $P\left(\frac{c^i}{d^j}\right)$, which is the probability of document vector $d^j = \left\{w_1^j, w_2^j, w_3^j, \ldots, w_n^j\right\}$ words falling within a certain category c^i and is measured by an equation based on Bayes' theorem:

$$P\left(\frac{c^i}{d^j}\right) = \frac{P(c^i) * P\left(\frac{d^j}{c^i}\right)}{P(d^j)} \tag{3}$$

Where $P(d^j)$ is the probability of arbitrarily selected document d^j and $P(c^i)$ is the probability of arbitrarily selected document d^j falling in the specific category c^i This classification method is known as a "Bayesian classifier."

In case of a high-dimensional data vector d^j, the Bayesian classifier is limited in its capability. It makes an assumption in which two randomly selected words/features of the data vector d^j are considered to be independent of each other. The given equation is the modified classifier with the said assumption:

$$P\left(\frac{d^j}{c^i}\right) = \prod_{l=1}^{n} P\left(\frac{w_l^j}{c^l}\right). \tag{4}$$

This equation, which is popular in the field of classification, is called a modified Bayesian classifier or NB.

3.8 Support vector machine classifiers

SVM is a popular classifier that works on the principles of statistical learning theory and structural maximization (Vapnik, 1999). It is a well-accepted classifier because of its potential for dealing with high-dimensional data using a specific kernel function.

SVM separates the classes (positive and negative) by the maximum margin created by a hyperplane. Consider a training set $X = \{x^i, y^i\}$, where $x^i \in R^m$ and $y^i \in \{+1, -1\}$, which implies a unique class for the i^{th} training sample. This study takes $+1$ for the class of spam and -1 as the class of ham. The final classification output is obtained from the following equation (5):

$$y = w.x - b \tag{5}$$

where y is the final classification output, w is the normal vector comparable to those in the feature vector x, and b is the bias parameter observed by the training process. The classes are separated by the following equations (6) and (7):

$$\text{minimize } \frac{1}{2}\|w\|^2 \tag{6}$$

$$\text{subject to } y^i(w.x - b) \geq 1, \forall i. \tag{7}$$

3.9 Kernel selection for support vector machine

An appropriate selection of kernel function is essential for unique applications of SVM-based classification. A good choice of kernel function affords to learn potential to SVM. Various kernel functions have been discussed in the literature; our research incorporates a normalized polynomial kernel.

4. Structure of an email classifier

The information for an email file is divided into the header (general information, such as subject, sender, and recipient) and the body (the main content of the email) (Figure 1). This study focuses only on the email body.

4.1 Preprocessing

Preprocessing (Sergienko *et al.*, 2017) is performed on incoming email files to transform strings of characters into a representation suitable for the classification algorithm (Figure 1, Tables I and II).

Figure 1 clearly depicts the structure of a spam classifier in which an email file with header and body goes for preprocessing. In this research, only the body is included, as this part of the email contains the main content. Preprocessing consists of tokenization (extraction of the words from the email file), removal of stop words (removal of words such as articles and prepositions that often occur in email files) and lemmatization (reducing a word to its normal form, for example, reducing "boosting" and "boosted" to "boost") (Patel and Patel, 2017). After the feature extraction process, feature selection is used to select the

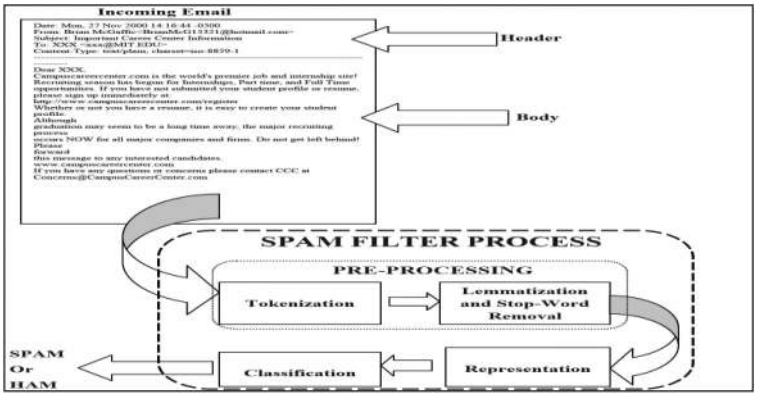

Figure 1.
Structure of a spam classifier

informative features of the words. In this study, the GreedyStepwise feature subset selection method is used. After the selection of all the informative features, feature representation is carried out. In this study, the binary feature representation method is used to construct a term-document matrix (TDM). Machine learning models are trained and validated through several performance evaluation metrics. The subsequent sections describe in detail the component steps of a spam classifier.

4.2 Dimensionality reduction

The major problem in spam classification is the high dimensionality (Trivedi and Dey, 2016) of the feature space, where one dimension of a unique word is found in many files. This large set of feature space creates difficulty for standard classification methods because the computation process is costly and the results are unreliable. Therefore, this large feature space will have to be reduced. The reduction process is known as dimensionality reduction and is carried out by a process of feature selection.

4.2.1 Feature subset selection. This technique (Trivedi and Dey, 2016; Tripathi and Trivedi, 2016) is applied directly after the dimensionality reduction of the TDM. GreedyStepwise search is used in this study because of its capacity to search informative features.

4.2.2 GreedyStepwise subset search. The greedy algorithm (Caruana and Freitag, 1994; Trivedi and Dey, 2016) takes the form of an iterative process and features are measured iteratively in each step. The aim of this process is to produce the best informative feature for the model. Stepwise regression is used for the evaluation process. Three different methods are available for the selection of the best feature: forward selection (adding the best informative features), backward selection (removing the worst features) and the mixed method (a combination of forwarding and backward selection). For the termination of the feature selection process, some measures such as p-value are used to determine whether all the best-observed features have been added to the model or whether any valuable feature

Table I.
Most informative feature selection process (Enron data set)

Tokenization	After pre-processing	After feature selection
Attached	Attach	Enron
Schedules	Schedule	Deal
Awarded	Award	Award
Deals	Deal	Revise
Enron	Enron	Draft
The	Revise	Request
Requested	Request	Document
. . ..	. . ..	. . ..

Table II.
Most informative feature selection process (SpamAssassin data set)

Tokenization	After pre-processing	After feature selection
Delivered	Deliver	SpamAssassin
Exim	Exim	Taint
Habeas	Habeas	Communication
For	Justin	Geneva
Habeas	Business	Jalapeno
If	Package	Users
Justin	Cash	Laugh
. . ..	. . ..	. . ..

remains to be added. In this study, 49 informative features from the Enron data set (Table I) and 36 informative features from the SpamAssassin data set (Table II) are selected.

4.3 Feature representation

An important component in text classification research is feature representation (Tripathi and Trivedi, 2016; Wu *et al.*, 2016). In email files, the text is generally obtained from the body of the message but sometimes the header and the subject line may also be considered. One popular representation technique is the BoW model, also known as the vector space model. Other approaches, such as character N-gram, are also used for representation but have not been tested specifically for the purposes of spam classification. Selected features may be represented using a binary representation method in which email files and words together form the binary matrix known as a TDM. This method is known as the term weighting method. The binary matrix contains binary values (1 and 0), where 1 indicates the presence of the particular feature or word in a specific email file and 0 indicates its absence.

The term weighting method is the choice of this research. Let us suppose that each email file is represented as a column vector D^x defined by the words extracted from the email files (i.e. $D^x = (W^1, W^2, W^3, \ldots)$ where w^j is the i^{th} word/feature of the email document d^x) (Aas and Eikvil, 1999). The combination of all email documents and words forms an $M \times N$ matrix, where M represents the number of distinct features and N represents the number of email instances. Table III represents the term-document relationship as a matrix a^{ij} that is defined as the degree of relationship between word i (column) and email file j (row).

4.4 Experimental design

4.4.1 Data sets. This research includes two different data sets produced from two different sources. It focuses on the Enron email data set for analysis; thereafter, the SpamAssassin data set is used for validation of the results obtained from the first data set.

4.4.1.1 Enron email. The main data set of this study is the Enron email data set, which contains 6,000 email files with a 50 per cent spam rate. Of the six existing versions of the Enron data set, versions five and six are selected for this study on the basis that the

1	Step	Submit	Taint	Thousands	Top	Total	Trading	Transaction
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	1
9	0	0	0	0	0	0	0	0
10	1	1	0	0	0	0	0	0
11	0	1	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0

Table III.
Term-document binary representation

complexity of the attacks found in the features of these email files allows the strength of the classifiers to be thoroughly tested.

4.4.1.2 SpamAssassin. The second data set of this study, SpamAssassin (Trivedi and Dey, 2013a, 2013b, 2013c, 2013d, 2014) is used to validate the results obtained from the first data set. SpamAssassin was obtained from a collection of unsolicited email, some older and some newer. In total, 4,700 files with a 50 per cent spam rate are selected at random for this study. The data set contains some easy email files (i.e. simple to classify) and hard email files (i.e. files with complexities). In this study, these files are mixed to generate 2,350 ham and 2,350 spam files.

4.4.2 Instruments for evaluation. This study incorporates three performance metrics: accuracy, *F*-value and false positive (FP) rate.

4.4.2.1 Accuracy. Accuracy is defined as the ratio of total correctly classified text emails to the total text emails, represented by the following equation (8):

$$Accuracy = \frac{\text{Total Correctly Classified Emails}}{\text{Total Emal Files}}. \tag{8}$$

4.4.2.2 *F*-value. The *F*-value is calculated from the harmonic sum of precision (the fraction of retrieved classified emails that are relevant) and recall (the fraction of accurate classified emails that are retrieved), represented as follows:

$$F(value) = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{9}$$

4.4.2.3 False positive rate. The FP rate measures the sensitivity of accurate classification and tells us how many positive instances are misclassified. It is represented as follows:

$$False\ Positive = \frac{\text{Misclassified Legitimate Emails}}{(Misclassified + Correctoly\,\text{Classified})\ \text{Legitimate Emails}.} \tag{10}$$

4.4.3 Design parameters. In this study, we use a JAVA and MATLAB environment on a Windows 7 operating system to test the selected classifiers. After preprocessing (feature extraction and feature search), the 49 most informative features for the Enron corpus and the 36 most informative features for the SpamAssassin corpus are selected for training the classifiers. The whole corpus is split so that 66 per cent of the files are used for training and 34 per cent for testing.

The manual method used in this study to tune the parameters for the evolutionary algorithms is given in Table IIV.

5. Results and discussion

To evaluate the performance of the classifiers, tests were run on the Enron (versions 5 and 6) and SpamAssassin data sets. Tables V and VI and Figures 2 and 3 present a comparative analysis of the GA and GP classifiers with and without boosting. Tables VII and VIII and Figures 4 and 5 present a comparative analysis of evolutionary classifiers with other popular classifiers (Bayesian, NB and SVM). The performance is shown in terms of accuracy, *F*-value and FP rate.

5.1 Analysis of classifiers based on evolutionary algorithms

Table V and Figure 2 show that initially when the classifiers were tested on the Enron data set, both evolutionary algorithm-based classifiers (i.e. GA and GP) gave weak observations but the result for GA was better than the result for GP. In this case, performance accuracy rose to 87.6 per cent for GA and 76.1 per cent for GP. To boost the performance of the classifiers further, this study performed numerous iterations and incorporated a boosting strategy (i.e. AdaBoost). After several iterations, both classifiers gave contrary indications, showing the potency of GP and proving that GA is a weak classifier. GP achieved the best result after 40 iterations, with 94.1 per cent accuracy; GA results were very poor, with an accuracy of 80.7-87.6 per cent.

Parameters	Value
Target fitness [$F(I^x)$]	90%
Maximum generation (R^x)	20
Maximum tree depth (D^x)	5
Mutation rate (m^x)	7%
Crossover rate (c^x)	90%
New program generation (N_b)	3%
No. of classifiers for ensemble generation (M_x)	1-50

Table IV. Manual tuning of parameters for evolutionary algorithms

	Boosting with AdaBoost			
Enron data set	GA		GP	
Performed iterations	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)
1	*87.6*	*87.5*	76.1	76.0
5	83.1	83.1	91.1	91.1
10	81.5	81.5	92.4	92.4
20	80.7	80.7	93.6	93.6
30	81.5	81.5	93.7	93.7
40	82.5	82.5	*94.1*	*94.1*
45	82.5	82.6	93.9	93.9
50	81.4	81.4	93.8	93.8

Table V. Accuracy and *F*-value of classifiers tested on Enron data set (with boosting)

	Boosting with AdaBoost			
SpamAssassin data set	GA		GP	
Performed iterations	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)
1	*96.5*	*96.6*	92.7	92.7
5	85.3	85.3	95.6	95.6
10	95.9	96.0	96.3	96.4
20	95.9	95.9	97.9	98.0
30	92.3	92.3	98.1	98.2
40	95.9	95.9	*98.2*	*98.2*
45	96.0	95.9	97.9	97.8
50	93.2	93.2	97.8	97.8

Table VI. Accuracy and *F*-value of classifiers tested on SpamAssassin data set (with boosting)

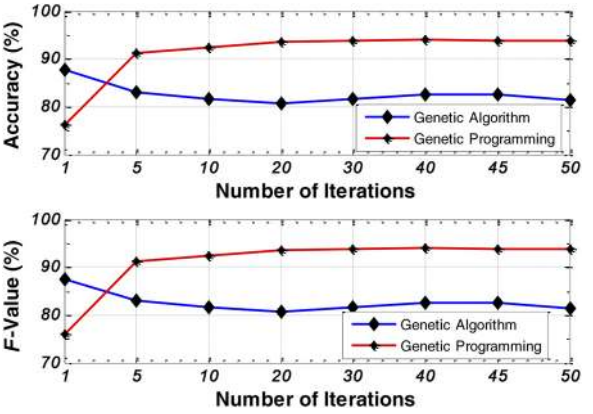

Figure 2.
Accuracy and *F*-value for Enron data set (with boosting)

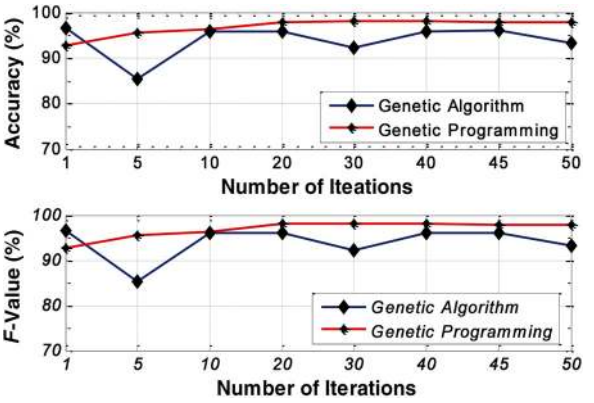

Figure 3.
Accuracy and *F*-value for SpamAssassin data set (with boosting)

	Greedy search			
In (%)	Enron		SpamAssassin	
Classifiers	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)
Bayesian	93.0	93.1	97.0	97.1
NB	92.9	92.9	96.7	96.7
SVM	93.8	93.8	97.8	97.8
GA	87.6	87.5	96.5	96.5
BGA	82.5	82.5	95.9	95.9
GP	84.8	84.8	92.3	92.3
BGP	94.1	94.1	98.2	98.2

Table VII.
Accuracy and *F*-value of all classifiers (both data sets)

The results for the classifiers tested on the SpamAssassin data set validate the results from the Enron data set (Table VI and Figure 5). In this case, GP initially showed worse performance than GA, with an accuracy of 92.7 per cent for GP and 96.5 per cent for GA. After numerous iterations, however, the performance of GP increased dramatically, whereas

the performance of GA deteriorated. Over 40 iterations, GP gave the best result, with 98.2 per cent accuracy, and the GA results were disappointing, with accuracy of 85.3-96.5 per cent.

5.2 Comparison with various machine learning classifiers

A comparative study was performed between the evolutionary classifiers (GA, GP, boosted GA (BGA) and boosted GP [BGP]) and some other popular classifiers (Bayesian, NB and SVM).

Table VII and Figure 4 present the results of all the classifiers that were tested on the Enron data set. The results show that the BGP classifier performed better than the others, with 94.1

In (%) Classifiers	Greedy search FP rate Enron	SpamAssassin
Bayesian	1.8	3.6
NB	4.4	4.5
SVM	2.8	1.0
GA	22.5	2.6
BGA	14.9	1.6
GP	1.3	1.9
BGP	2.7	1.0

Table VIII. FP rates for all classifiers (both data sets)

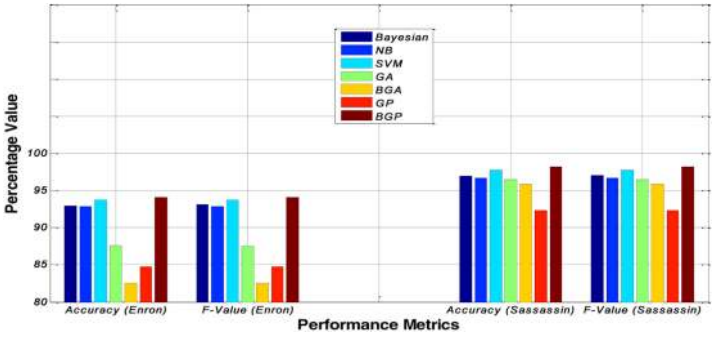

Figure 4. Accuracy and *F*-value for all classifiers (for both data set)

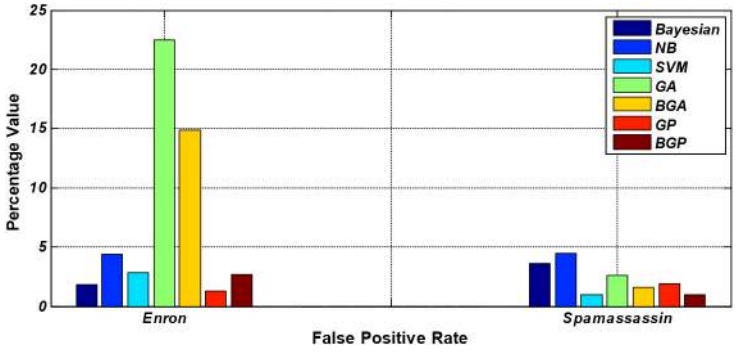

Figure 5. FP rates for all classifiers (both data sets)

per cent performance accuracy. The BGA was the worst classifier, with 82.5 per cent accuracy. The SVM and Bayesian classifiers were the second and third best, respectively.

Table VII and Figure 4 show the results of the same classifiers tested on the SpamAssassin data set. The results from this data set strongly corroborate the observations from the first data set. In this case, BGP again showed its potential, with 98.2 per cent performance accuracy. BGA was disappointing, with 95.9 per cent accuracy.

5.3 False positive rates

Table VIII and Figure 5 demonstrate the FP rates. For a good classifier, the value of this metric should be as low as possible. In the case of the Enron data set, if we consider both metrics (accuracy and FP rate) simultaneously, BGP is seen to be excellent, with a 2.7 per cent FP rate. The GP and Bayesian classifiers are shown to be good on the basis of their FP rates but they had the worst accuracy. GA and BGA again performed poorly, with FP rates of 22.5 and 14.9 per cent, respectively.

The same classifiers tested on the SpamAssassin data set produced results that disagreed with those from the first data set. In this case, the BGA classifier was the best in terms of both metrics, with an FP rate of 1.0 per cent. The reason for this discrepancy may be the complexity of the Enron data set in terms of attacks incorporated by spammers.

5.4 Tenfold cross-validation of accuracy and F-values

Cross-validation is a technique for assessing classification models by splitting the data set into a training set to train the models and a test set to evaluate them. In the tenfold cross-validation method (Moreno-Torres *et al.*, 2012), the data set is randomly split into 10 subsamples of equal size. From these 10 subsamples, a single subsample is used as the validation data for testing the model and the remaining nine subsamples are taken as training data. The cross-validation process is then repeated 10 times, with each of the 10 subsamples used exactly once as validation data. The 10 results are then averaged to obtain a single estimation. The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once.

Table IX shows the results for accuracy and *F*-value of the evolutionary classifiers and the other classifiers obtained after the tenfold cross-validation process was carried out on the Enron and SpamAssassin email data sets. The results of the cross-validation strongly support the results obtained in the 66/34 per cent training and testing data split method. Following analysis of the results, four main observations can be made.

Observation 1: GP with AdaBoost performed better than other classifiers tested in this study, with 94.2 per cent accuracy for the Enron data set and 98.5 per cent accuracy for the

Table IX.
Accuracy and *F*-value of all classifiers (both data sets, tenfold cross-validation)

	Enron		SpamAssassin	
Classifiers	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)
Bayesian	93.4	93.3	97.8	97.7
NB	93.2	93.2	97.1	97.1
SVM	93.9	93.9	98.3	98.2
GA	88.1	88.1	96.9	96.9
BGA	83.1	83	96.2	96.2
GP	85.2	85.1	93	93
BGP	94.2	94.1	98.5	98.5

SpamAssassin data set. However, SVM results were comparable at their best, with 93.9 per cent accuracy for the Enron data set and 98.3 per cent accuracy for the SpamAssassin data set.

Observation 2: When GA and GP are compared in the absence of boosting, GA turns out to be the better classifier, with 88.1 per cent accuracy for the Enron data set and 96.9 per cent accuracy for the SpamAssassin data set, compared to GP's 85.2 per cent for Enron and 93.0 per cent for SpamAssassin.

Observation 3: When GA and GP are compared in the presence of AdaBoost, BGP is shown to be a better classifier than BGA, with 94.2 per cent accuracy for the Enron data set and 98.5 per cent accuracy for the SpamAssassin data set, compared to BGA's 83.1 per cent for Enron and 96.2 per cent for SpamAssassin.

Observation 4: With boosting, the training of GP was shown to be good, and its accuracy increased to approximately 5.1 per cent for the Enron data set and 2.8 per cent for the SpamAssassin data set. However, for GA after boosting, performance went down by 3.0 per cent for Enron data set and 0.4 per cent for SpamAssassin data set.

5.5 Wilcoxon signed-rank test (for matched data set)

This study also used the Wilcoxon signed-rank test (Hu *et al.*, 2016) to further verify the predictive accuracy of the evolutionary classifiers and the other classifiers used in this study. The Wilcoxon signed-rank test was used to check whether the results of the two classifiers are significantly different or not (Demšar, 2006). The null hypothesis is that the predictive capabilities of the two classifiers are the same, in which case the mean difference of the accuracy of the classifiers would be zero. In this study, all the machine learning classifiers were tested on each data set; the Wilcoxon signed-rank test was first performed on the Enron data set and then on the SpamAssassin data set. Tables X and XI show the *p*-values of pairwise machine learning classifiers based on the *F*-values for both data sets, with

Table X. Wilcoxon's signed-rank test between each pair of classifiers on the Enron data set (*F*-values)

p-values*	NB	SVM	GA	BGA	GP	BGP
Bayesian	*0.34*	2.9E−5	1.86E−09	2.0E−6	1.86E−09	8.01E−08
NB		2.0E−6	1.86E−09	2.0E−6	1.86E−09	3.73E−09
SVM			2.0E−6	2.0E−6	1.86E−09	0.001483
GA				1.86E−09	1.86E−09	2.0E−6
BGA					3.73E−09	2.0E−6
GP						2.0E−6

Note: *Wilcoxon's signed-rank test is performed on the 95 per cent confidence level

Table XI. Wilcoxon's signed-rank test between each pair of classifiers on the SpamAssassin data set (*F*-values)

p-values*	NB	SVM	GA	BGA	GP	BGP
Bayesian	1E−04	3E−05	5E−05	2E−06	2E−06	1.86E−09
NB		2E−06	0.09166	7E−06	2E−06	2E−06
SVM			*1.90E−09*	2E−06	2E−06	5E−06
GA				0.021	1.90E−09	2E−06
BGA					2E−06	2E−06
GP						2E−06

Note: *Wilcoxon's signed-rank test is performed on the 95 per cent confidence level

the p-values that are significant (i.e. $p > 0.05$) shown in bold. Five observations can be made on the basis of this analysis.

Observation 1: In terms of F-values for the two data sets, most of the pairwise machine learning classifiers showed significant differences from the others ($p < 0.05$), with only a small number of exceptions ($p > 0.05$).

Observation 2: For the Enron data set, the differences between the Bayesian and NB classifiers were not significant ($p > 0.05$), and hence, the performance capability of both classifiers is comparable. For the SpamAssassin data set, the differences between GA and NB were not significant ($p > 0.05$), and hence, the performance capability of both classifiers is comparable.

Observation 3: BGP turns out to be a good classifier in comparison to the other classifiers included in this study, as the p-value of the BGP classifier is higher than the other paired classifiers. Hence, in respect to F-value, the performance capability of BGP classifier is stronger than that of the other classifiers.

Observation 4: When the test of significance is carried out between the GA and GP classifiers in the absence of boosting, the p-value is high for both classifiers. Hence, in respect of F-values, GA without boosting is comparable with GP for both data sets.

Observation 5: When the test of significance is carried out between the GA and GP classifiers in the presence of boosting, the p-value is high. Hence, in respect of F-values, BGP is better than BGA for both data sets.

6. Conclusion, research implications and limitations

Evolutionary algorithm-based classifiers are now very familiar in the literature because of their interesting rules. This study has shown the strength of wrapper feature search algorithms; the GreedyStepwise feature search method obtained as a smaller number of best informative features and the GA classifier demonstrated its potential for accurate classification. GA and GP classifiers were tested in this study, with and without boosting. Over numerous iterations, GP showed excellent classification strength with the help of the boosting strategy; GA initially gave satisfactory results compared to GP but, after numerous iterations, its performance deteriorated and did not approach the level of performance of GP. When comparison was made with various other classifiers, BGP again proved its potential with excellent performance, with the SVM and Bayesian classifiers the second and third best, respectively. BGP also showed its strength insensitivity of classification; in each iteration, its FP rate was always the lowest value.

To validate the performance of the evolutionary classifiers and the other machine learning classifiers, tenfold cross-validation of accuracy and F-measures was performed. In the validation process, the results strongly corroborated the results obtained from the 66/34 per cent training and testing split. Further, to check the significance of the differences in performance accuracy and F-measure between the evolutionary classifiers and the other machine learning classifiers, the Wilcoxon signed-rank test of matched data were performed for both data sets. The results suggest that most of the differences were significant at the 95 per cent confidence level ($p < 0.05$), with only a small number of exceptions ($p > 0.05$). This study, therefore, concludes that GP, with AdaBoost algorithms and boosting with resampling, is good for classifying spam and ham emails.

In the future, the same classifiers could usefully be tested on other data sets. Further studies might also be carried out to check the credibility of GP by means of comparative analysis with other machine learning classifiers.

The primary research implication of this study relates to reductions in costs incurred because of spam/unsolicited bulk email. Email is fundamentally necessary for sharing information

between a number of units in an organization and for being competitive with business rivals. It is a continual challenge for internet service providers to provide the best email services to their customers. Although internet service providers and organizations are continuously adopting novel spam filtering approaches to reduce the number of unwanted emails, the desired effect is frequently not achieved because of the cost of installation, issues of customizability and the risk of misclassification of important emails. This study deals with many of the issues and challenges faced by organizations and internet service providers. It found that the proposed models not only provided excellent performance accuracy, sensitivity, low FP rates and customizability but also contributed to reducing the cost of spam. The same models may be used for other text mining applications such as sentiment analysis, blog mining or news mining.

This research focuses on the classification of spam emails that are written in the English language, simply because of English a universally accepted language that is widely used to write emails. No other languages have been included in this study. The body of an email file is particularly suitable for content-based filtering, as this method works with the content (words/features) of the email data that is found in the body; hence, this research used only the body of the emails. A critical category of spam is image spam, in which the content of spam is absorbed into the image file; however, image spam has not been included in this research. Only email messages have been incorporated, and no other message types such as short message service or multimedia messaging service were a part of this study.

References

Aas, K. and Eikvil, L. (1999), *Text Categorisation: A Survey*.

Banzhaf, W. (Ed.) (1998), "Genetic programming: first European workshop", *Proceedings of EuroGP'98, Paris, France, 14-15 April*, Vol. 1, *Springer Science and Business Media*.

Bauer, E. and Kohavi, R. (1999), "An empirical comparison of voting classification algorithms: bagging, boosting, and variants", *Machine Learning*, Vol. 36 Nos 1/2, pp. 105-139.

Breiman, L. (2001), "Random forests", *Machine Learning*, Vol. 45 No. 1, pp. 5-32.

Cantú-Paz, E. (2007), "Parameter setting in parallel genetic algorithms", *Parameter Setting in Evolutionary Algorithms*, Springer, Berlin, Heidelberg, pp. 259-276.

Caruana, R. and Freitag, D. (1994), "Greedy attribute selection", in *Machine Learning Proceedings, 1994*, Morgan Kaufmann, pp. 28-36.

Cortes, C. and Vapnik, V. (1995), "Support-vector networks", *Machine Learning*, Vol. 20 No. 3, pp. 273-297.

Datta, J., Kataria, N. and Hubballi, N. (2015), "Network traffic classification in encrypted environment: a case study of google hangout", *Twenty-First National Conference on Communications (NCC)*, *IEEE*, pp. 1-6.

Drucker, H., Wu, S. and Vapnik, V.N. (1999), "Support vector machines for spam categorization", *IEEE Transactions on Neural Networks*, Vol. 10 No. 5, pp. 1048-1054.

Fdez-Glez, J., Ruano-Ordás, D., Laza, R., Méndez, J.R., Pavón, R. and Fdez-Riverola, F. (2016), "WSF2: a novel framework for filtering web spam", *Scientific Programming*, Vol. 2016, p. 1.

Friedman, N., Geiger, D. and Goldszmidt, M. (1997), "Bayesian network classifiers", *Machine Learning*, Vol. 29 No. 2/3, pp. 131-163.

Gashti, M.Z. (2017), "Detection of spam email by combining harmony search algorithm and decision tree", *Engineering, Technology and Applied Science Research*, Vol. 7 No. 3, pp. 1713-1718.

Goh, K.L. and Singh, A.K. (2015), "Comprehensive literature review on machine learning structures for web spam classification", *Procedia Computer Science*, Vol. 70, pp. 434-441.

Goodman, J., Cormack, G.V. and Heckerman, D. (2007), "Spam and the ongoing battle for the inbox", *Communications of the ACM*, Vol. 50 No. 2, pp. 24-33.

Holland, J.H. (1975), *Adaptation in Natural and Artificial Systems. An Introductory Analysis with Application to Biology, Control, and Artificial Intelligence*, University of MI Press, Ann Arbor, MI, pp. 439-444.

Hu, Z., Chiong, R., Pranata, I., Susilo, W. and Bao, Y. (2016), "Identifying malicious web domains using machine learning techniques with online credibility and performance data" *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pp. 5186-5194.

Jia, Z., Li, W., Gao, W. and Xia, Y. (2012), "Research on web spam detection based on support vector machine", *International Conference on Communication Systems and Network Technologies (CSNT)*, *IEEE*, pp. 517-520.

Li, W. (2009), "The e-mail filtering system based on improved genetic algorithm", *International Workshop on Information Security and Application (IWISA 2009)*, p. 26.

Kishore, J.K., Patnaik, L.M., Mani, V. and Agrawal, V.K. (2000), "Application of genetic programming for multicategory pattern classification", *IEEE Transactions on Evolutionary Computation*, Vol. 4 No. 3, pp. 242-258.

Lai, C.C. (2007), "An empirical study of three machine learning methods for spam filtering", *Knowledge-Based Systems*, Vol. 20 No. 3, pp. 249-254.

Lewis, D.D. (1998), "Naive (Bayes) at forty: the independence assumption in information retrieval", *Machine learning: ECML-98*, *Springer*, *Berlin Heidelberg*, pp. 4-15.

Liu, B., McKay, B. and Abbass, H.A. (2003), "Improving genetic classifiers with a boosting algorithm", *The 2003 Congress on Evolutionary Computation, 2003. CEC'03*, Vol. 4, pp. 2596-2602, *IEEE*.

Meda, C., Ragusa, E., Gianoglio, C., Zunino, R., Ottaviano, A., Scillia, E. and Surlinelli, R. (2016), "Spam detection of Twitter traffic: a framework based on random forests and non-uniform feature sampling", *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, *IEEE*, pp. 811-817.

Moreno-Torres, J.G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N.V. and Herrera, F. (2012), "A unifying view on dataset shift in classification", *Pattern Recognition*, Vol. 45 No. 1, pp. 521-530.

Parveen, P. and Halse, P.G. (2016), "Spam mail detection using classification", *International Journal of Advanced Research in Computer Science and Electronics Engineering*, Vol. 5 No. 6, pp. 347-349.

Pantel, P. and Lin, D. (1998), "Spamcop: a spam classification and organization program", *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, pp. 95-98.

Patel, B.B. and Patel, H.M. (2017), "Survey on offline character recognition for handwritten Gujarati text", *International Journal of Computer Applications*, Vol. 177 No. 6.

Patil, R.C. and Patil, D.R. (2015), "Web spam detection using SVM classifier", *IEEE 9th International Conference on Intelligent Systems and Control (ISCO)*, *IEEE*, pp. 1-4.

Quinlan, J.R. (1993), *C4. 5: Programming for Machine Learning*, Vol. 38, Morgan Kauffmann, Burlington, MA, p.48.

Raymer, M.L., Punch, W.F., Goodman, E.D., Kuhn, L.A. and Jain, A.K. (2000), "Dimensionality reduction using genetic algorithms", *IEEE Transactions on Evolutionary Computation*, Vol. 4 No. 2, pp. 164-171.

Rennie, J. (1998), "An application of machine learning to e-mail filtering", *Proceeding KDD Workshop on Text Mining*.

Shah, N.F. and Kumar, P. (2018), "A comparative analysis of various spam classifications", *Progress in Intelligent Computing Techniques: Theory, Practice, and Applications*, Springer, Singapore, pp. 265-271.

Sahami, M. Dumais, S. Heckerman, D. and Horvitz, E. (1998), "A Bayesian approach to filtering junk e-mail, learning for text categorization: Papers from the 1998 workshop", Vol. 62, pp. 98-105.

Schapire, R.E., Freund, Y., Bartlett, P. and Lee, W.S. (1998), "Boosting the margin: a new explanation for the effectiveness of voting methods", *The Annals of Statistics*, Vol. 26 No. 5, pp. 1651-1686.

Sergienko, R., Shan, M. and Schmitt, A. (2017), "A comparative study of text preprocessing techniques for natural language call routing", in *Dialogues with Social Robots*, Springer, Singapore, pp. 23-37.

Shengen, L., Xiaofei, N., Peiqi, L. and Lin, W. (2011), "Generating new features using genetic programming to detect link spam", *International Conference on Intelligent Computation Technology and Automation (ICICTA)*, Vol. 1, *IEEE*, pp. 135-138.

Tripathi, A. and Trivedi, S.K. (2016), "Sentiment analyis of Indian movie review with various feature selection techniques" in *2016 IEEE International Conference on Advances in Computer Applications (ICACA), IEEE*, pp. 181-185.

Trivedi, S.K. and Dey, S. (2013a), "Interplay between probabilistic classifiers and boosting algorithms for detecting complex unsolicited emails", *Journal of Advances in Computer Networks*, Vol. 1 No. 2, pp. 132-136.

Trivedi, S.K. and Dey, S. (2013b), "Effect of various kernels and feature selection methods on SVM performance for detecting email spams", *International Journal of Computer Applications*, Vol. 66 No. 21.

Trivedi, S.K. and Dey, S. (2013c), "Effect of feature selection methods on machine learning classifiers for detecting email spams", *Proceedings of the 2013 Research in Adaptive and Convergent Systems, ACM*, pp. 35-40.

Trivedi, S.K. and Dey, S. (2013d), "An enhanced genetic programming approach for detecting unsolicited emails", *16th International Conference on Computational Science and Engineering (CSE), IEEE*, pp. 1153-1160.

Trivedi, S.K. and Dey, S. (2014), "Interaction between feature subset selection techniques and machine learning classifiers for detecting unsolicited emails", *ACM SIGAPP Applied Computing Review*, Vol. 14 No. 1, pp. 53-61.

Trivedi, S.K. and Dey, S. (2016), "A novel committee selection mechanism for combining classifiers to detect unsolicited emails", *VINE Journal of Information and Knowledge Management Systems*, Vol. 46 No. 4, pp. 524-548.

Vapnik, V.N. (1999), "An overview of statistical learning theory", *IEEE Transactions on Neural Networks*, Vol. 10 No. 5, pp. 988-999.

Wu, C.F.J. (1986), "Jackknife, bootstrap and other resampling methods in regression analysis", *The Annals of Statistics*, Vol. 14 No. 4, pp. 1261-1295.

Wu, S., Chen, S.Y. and Hou, H. (2016), "Exploring the interactive patterns of concept map-based online discussion: a sequential analysis of users' operations, cognitive processing, and knowledge construction", *Interactive Learning Environments*, Vol. 24 No. 8, pp. 1778-1794.

Further reading

Joachims, T. (1998), *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*, Springer, Berlin Heidelberg, pp. 137-142.

Tang, Y., Krasser, S., He, Y., Yang, W. and Alperovitch, D. (2008), "Support vector machines and random forests modeling for spam senders behavior analysis", *Global Telecommunications Conference, IEEE GLOBECOM 2008, IEEE*, pp. 1-5.

Vafaie, H. and Imam, I.F. (1994), "Feature selection methods: genetic algorithms vs. greedy-like search", *Proceedings of International Conference on Fuzzy and Intelligent Control Systems.*

Corresponding author

Shrawan Kumar Trivedi can be contacted at: shrawaniimsirmaur@gmail.com