Hybrid GA-SVM for Efficient Feature Selection in E-mail Classification

Fagbola Temitayo* Olabiyisi Stephen Adigun Abimbola

Department of Computer Science & Engineering, Ladoke Akintola University of Technology,
PMB 4000, Ogbomoso, Oyo State, Nigeria
* E-mail of the corresponding author: cometoty@yahoo.com

Abstract

Feature selection is a problem of global combinatorial optimization in machine learning in which subsets of relevant features are selected to realize robust learning models. The inclusion of irrelevant and redundant features in the dataset can result in poor predictions and high computational overhead. Thus, selecting relevant feature subsets can help reduce the computational cost of feature measurement, speed up learning process and improve model interpretability. SVM classifier has proven inefficient in its inability to produce accurate classification results in the face of large e-mail dataset while it also consumes a lot of computational resources. In this study, a Genetic Algorithm-Support Vector Machine (GA-SVM) feature selection technique is developed to optimize the SVM classification parameters, the prediction accuracy and computation time. Spam assassin dataset was used to validate the performance of the proposed system. The hybrid GA-SVM showed remarkable improvements over SVM in terms of classification accuracy and computation time.

Keywords: E-mail Classification, Feature-Selection, Genetic algorithm, Support Vector Machine

1. Introduction

E -mail is one of the most popular, fastest and cheapest means of communication. It has become a part of everyday life for millions of people, changing the way we work and collaborate (Whittaker et al 2005). Its profound impact is being felt on business growth and national development in a positive way and has also proven success in enhancing peer communications. E-mail is one of the many technological developments that have influenced our lives. However, the downside of this sporadic success is the constantly growing size of unfiltered e-mail messages received by recipients. Thus, E-mail classification becomes a significant and growing problem for individuals and corporate bodies. E-mail classification tasks are often divided into several sub-tasks. First, Data collection and representation of e-mail messages, second, e-mail feature selection and feature dimensionality reduction for the remaining steps of the task (Awad & ELseuofi 2011). Finally, the e-mail classification phase of the process finds the actual mapping between training set and testing set of the e-mail dataset.

The spam email problem is well-known, and personally experienced by anyone who uses email. Spam is defined as junk e-mail message that is unwanted, delivered by the internet mail service (Chan 2008). It could also be defined as an unsolicited, unwanted email that was sent indiscriminately, directly or indirectly, by a sender having no current relationship with the recipient. Some of the spam e-mails are unsolicited commercial and get-rich messages while others could contain offensive material. Spam e-mails can also clog up the e-mail system by filling-up the server disk space when sent to many users from the same organization. The goal of Spam Classification is to distinguish between spam and legitimate mail messages. Technical solutions to detecting spam include filtering the sender’s address or header content but the problem with filtering is that a valid message may be blocked sometimes; hence e-mail is better classified as spam or non-spam based on features (Chan 2008).

The selection of features is flexible such as percentage of words in the e-mail that match specified word, percentage of words in the e-mail that match specified character, average length of uninterrupted sequences of capital letters etc. Thus, the feature selection process can be considered a problem of global combinatorial optimization in machine learning, which reduces the number of features, removes irrelevant, noisy and redundant data, and results in acceptable classification accuracy. Feature subset selection can involve random or systematic selection of inputs or formulation as an optimization problem which involves searching the solution space of subsets for an optimal or near-optimal subset of features, according to a specified criterion.

Feature selection methods seeking optimal subsets are usually directed toward one of two goals:

(1) minimize the number of features selected while satisfying some minimal level of classification capability or

(2) maximize classification ability for a subset of prescribed cardinality. The feature selection process can be made more efficient by optimizing its subset selection techniques through the use of some well-known optimizers.

Techniques for feature selection are characterized either as filters, which ignore the classifier to be used, or wrappers, which base selection directly on the classifier. Computationally more efficient than wrappers, a filter approach performs subset selection based only on the feature qualities within the training data. Since the classifier is ignored, there is no interaction between the biases of the feature selector and the classifier, but the quality of the best filter subset is typically not as effective as a subset selected using a wrapper model (John et al. 1994). Filters rate features based on general characteristics, such as interclass distance or statistical independence, without employing any mining algorithms (Guyon & Elisseef 2003). Wrappers select a feature subset based directly on the classifier. The training data are used to train the classifier using different feature subsets; each is then evaluated using the testing data to find the best subset.

As a new approach to pattern recognition, SVM is based on Structural Risk Minimization (SRM), a concept in which decision planes define decision boundaries. A decision plane is one that separates a set of objects having different class memberships; the SVM finds an optimal hyperplane with the maximal margin to separate the two classes and then classify the dataset. It is suitable for dealing with magnitude features problems with a given finite amount of training data only. Youn & McLeod (2006) used four classifiers including Neural Network, SVM, Naïve Bayesian, and J48 to filter spams from the dataset of emails. All the emails were classified as spam (1) or not (0). The authors reported that Neural Network and SVM did not show good result compared with J48 or Naïve Bayesian classifier on large features. Based on this fact, the authors concluded that Neural Network and SVM are not appropriate for classifying large email dataset. The result of the study by Priyanka et al. (2010) on SVM for large dataset classification showed that SVM is time and memory consuming when size of data is enormous.

SVMs (Chapelle et al. 1999; El-Naqa et al. 2002; Kim et al. 2002; Kim et al. 2003; Liyang et al. 2005a; Liyang et al. 2005b; Song et al. 2002; Vapnik 1995) are much more effective than other conventional nonparametric classifiers (e.g., the neural networks, nearest neighbor, k-NN classifier) in terms of classification accuracy, computational time and stability to parameter setting, it is weak in its attempt to classify highly dimensional dataset with large number of features (Andrew 2010). Liu et al. (2005) in the result of their study on Support Vector Machine (SVM) showed that the skewed distribution of the Yahoo! Directory and other large taxonomies with many extremely rare categories makes the classification performance of SVMs unacceptable. More substantial investigation is thus needed to improve SVMs and other statistical methods for very large-scale applications. These drawbacks deteriorated the success of SVM strongly. Sequel to this, an optimizer is required to reduce the number of feature vectors before the resultant feature vectors are introduced to SVM and this forms the focus of this study.

Computational studies of Darwinian evolution and natural selection have led to numerous models for solving optimization problems (Holland 1975; Fogel 1998). Genetic Algorithm comprises a subset of these evolution-based optimization problems techniques focusing on the application of selection, mutation, and recombination to a population of competing problem solutions (Holland 1975; Goldberg 1989). GA's are parallel, iterative optimizers, and have been successfully applied to a broad spectrum of optimization problems, including many pattern recognition and classification tasks. De-Jong et al. (1993) produced *GABIL (Genetic Algorithm-Based Inductive Learning)*, one of the first general-purpose GAs for learning disjunctive normal form concepts. *GABIL* was shown to produce rules achieving validation set accuracy comparable to that of decision trees induced using *ID3* and *C4.5*.

Furthermore, GAs have been applied to find an optimal set of feature weights that improve classification accuracy (Chandra & Nandhini 2010) and has proven to be an effective computational method, especially in situations where the search space is uncharacterized, not fully understood, or/and highly dimensional (Ishibuchi & Nakashima 2003). Thus, GA is employed as an optimizer for the feature selection process of SVM classifier in this study. The approach for feature selection can be divided into wrappers, filters and embedded methods essentially. In this study, the spam problem is treated as a classification problem in a wrapper manner; a GA-Based feature selection optimization technique for Support Vector Machine classifier is proposed and developed to classify e-mail dataset as spam or ham.

2. Related Works

E-mail classification has been an active area of research. Cohen (1996) developed a propositional learning algorithm RIPPER to induce ‘‘keyword-spotting rules’’ for filing e-mails into folders. The multi-class problem was transformed into several binary problems by considering one folder versus all the others. Cohen argued that keyword spotting rules are more useful as they are easier to understand, modify and can be used in conjunction with user-constructed rules. Sahami (1998) applied NB for spam e-mail filtering using bag of words to represent e-mail corpora and binary encoding. The performance improved by the incorporation of hand-crafted phrases and domain-specific features such as the domain type of the sender and the percentage of non-alphabetical characters in the subject. Rennie (2000) used Naïve-Bayes to file e-mails into folders and suggested the three most suitable folders for each message. The system applies stemming, removes stop words and uses document frequency threshold as feature selector.

Pantel et al. (1998) developed SpamCop: a spam classification and organization program. SpamCop is a system for spam e-mail filtering also based on Naïve-Bayes. Both stemming and a dynamically created stop word list are used. The authors investigated the implications of the training data size, different rations of spam and non-spam e-mails, use of trigrams instead of words and also showed that SpamCop outperformed Ripper. MailCat et al. (1999) uses a nearest-neighbor (k-NN) technique and tf-idf representation to file e-mails into folders. K-NN supports incremental learning but requires significant time for classification of new e-mails. Androutsopoulos et al. (2000) found that Naïve-Bayes and a k-NN technique called TiMBL clearly outperform the keyword-based spam filter of Outlook 2000 on the LingSpam corpora. Ensembles of classifiers were also used for spam filtering. Sakkis et al. (2001) combined a NB and k-NN by stacking and found that the ensemble achieved better performance. Carrera & Marquez (2001) showed that boosted trees outperformed decision trees, NB and k-NN. Rios & Zha (2004) applied RF for spam detection on time indexed data using a combination of text and metadata features. For low false positive spam rates, RF was shown to be overall comparable with SVM in classification accuracy.

Koprinska et al. (2007) worked on supervised e-mail classification by:

(1) applying RF for both filing e-mails into folders and filtering spam e-mails, comparing RF with a number of state-of-the-art classifiers and showing that it is a very good choice in terms of accuracy, running and classification time, and simplicity to tune,

(2) introducing a new feature selector that is accurate and computationally efficient,

(3) studying the portability of an anti-spam filter across different users, and

(4) comparing the performance of a large number of algorithms on the benchmark corpora for spam filtering using the same version of the data and the same pre-processing.

Okunade & Longe (2011) developed an improved electronic mail classification using hybridized root word extractions. They employed word Stemming/Hashing combined with Bayesian probability as an approach to ensure accurate classification of electronic mails in the electronic mail infrastructure. The authors argued that Content based spam filters only work well if the suspicious terms are lexically correct; and that spammers deceive content based filters by rearranging suspicious terms to fool such filters.

In order to effectively rack such deceptions, the suspicious terms used by the authors were valid terms with correct spelling that is grammatically correct; which otherwise could result into false positive. They developed the hybridized technique which could detect the modified suspicious terms of the harmful messages by examining the base root of the misspelled or modified word and reconverting them to the correct tokens or near correct tokens. The implementation of the technique results indicated the reduction of false positives thus improving e-mail classification.

3. Materials and Method

This section presents the statement of problem, feature representation concept, support vector machine, genetic algorithm, genetic algorithm for support vector machine parameters optimization, the e-mail dataset features, performance evaluation metrics and the simulation tool.

3.1 Problem Statement

E-mail classification (spam filtering) is a supervised learning problem. It can be formally stated as follows. Given a training set of labeled e-mail documents $D_{train} = \{(d_1, c_1), \ldots, (d_n, c_n)\}$, where d_i is an e-mail document from a document set D and c_i is the label chosen from a predefined set of categories C. It should be noted that effective feature selection is essential to making the learning task more efficient and faster.

Sequel to this, the goal of this study is to optimize the feature selection technique of the SVM hypothesis (classifier) $h: D \rightarrow C$ to accurately classify new, unseen e-mail documents D_{test}, D_{test} □ D_{train} in which C contains two labels: spam and non-spam (legitimate).

3.2 Feature Representation

A feature is a word. In this study, ω refers to a word, ***x*** is a feature vector that is composed of the various words from an e-mail message formed by analyzing the contents of the e-mail. There is one feature vector per e-mail message. There are various alternatives and enhancements in constructing the ***x*** vectors.

(a) Term Frequency

The *i*th component of the feature vector is the number of times that word w_i appears in that email message. In this study, a word is a feature only if it occurs three or more times in a message or messages. This prevents misspelled words and words used rarely from appearing in the dictionary.

(b) Use of a Stop List

Stop list was formed by using words like “of,” “and,” “the,” etc., are used to form a stop list. While forming a feature vector, words on the stop list were not used. This is because common words are not very useful in classification.

However, the learning algorithm itself determined whether a particular word is important or not. The choice of words put on the stop list is a function of the classification task and as such, the use of word stemming: words such as “work”, “worker” and “working” are shortened to the word stem “work.” Doing this, the size of the feature vector becomes reduced as redundant and irrelevant features are being removed.

3.3 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a machine learning algorithm that is useful in solving classification problems. A classification problem typically involves a number of data samples, each of which is associated with a *class* (or label) and some *features* (or attributes). Given a previously unseen sample, the problem is to *predict* its class by looking at its features. A support vector machine solves this problem by first building a model from a set of data samples with known classes (i.e., the training set), and use the model to predict classes of data samples that are of unknown classes.

To evaluate the performance of an SVM, a testing set of data samples is only given with the features.

The *accuracy* of the SVM can be defined as the percentage of the testing samples with correctly predicted class labels. In essence, given the training set with class labels and features, a support vector machine treats all the features of a data sample as a point in a high dimensional space, and tries to construct hyperplanes to divide the space into partitions. Points with different class labels are separated while those with the same class label are kept in the same partition.

There are two classes:

$$y_i \in \{-1, \qquad (1)$$

and there are ***N*** labeled training examples:

$$(x_1, y_1), \ldots, (x_N, y_N), x \in R^d \qquad (2)$$

where d is the dimensionality of the vector.

If the two classes are linearly separable, then one can find an optimal weight vector $\dot{w}$ such that $\|\dot{w}\|^2$ is minimum; and:

$$(\dot{w}) * x_i - b \geq 1 \text{ if } y_i = +1, \qquad (3)$$

$$(\dot{w}) * x_i - b \leq 1 \text{ if } y_i = -1 \qquad (4)$$

or equivalently y_i such that:

$$y_i(\dot{w}) * x_i - b \geq \qquad (5)$$

Training examples that satisfy the equality are termed support vectors. The support vectors define two hyperplanes, one that goes through the support vectors of one class and one goes through the support vectors of the other class. The distance between the two hyperplanes defines a margin and this margin is maximized when the norm of the weight vector $\|\dot{w}\|$ is minimum.

This minimization can be performed by maximizing the following function with respect to the variables $\propto_j$:

$$\text{W} \quad (\propto) = \sum_{i=1}^{N} \propto_i - 0.5 \sum_{i=1}^{N} X \sum_{j=1}^{N} \propto_i \propto_j (x_i * x_j) y_i \qquad (6)$$

subject to the constraint: $0 \leq$ where it is assumed that there are ***N*** training examples, is one of the training vectors, and * represents the dot product. If $\propto_j >$ then is termed a support vector and X is a multiplication factor.

For an unknown vector x_j classification then corresponds to finding:

$$F(x_j) = sign\{\dot{w} * x_j - b\} \qquad (7)$$

where

$$\dot{w} = \sum_{i=1}^{r} \propto_i y_i x_i \qquad (8)$$

and the sum is over the r nonzero support vectors (whose $\propto'$s are nonzero). There are two reasons for believing that the training time for SVM's with binary features can be reduced. The first is that the vectors are binary and the second is that the feature vectors are sparse (typically only 4% of a vector is nonzero). This allows the dot product in the SVM optimization to be replaced with faster non-multiplicative routines.

3.4 Genetic Algorithm (GA)

There are three major design decisions to consider when implementing a GA to solve a particular problem. A representation for candidate solutions must be chosen and encoded on the GA chromosome, fitness function must be specified to evaluate the quality of each candidate solution, and finally the GA run parameters must be specified, including which genetic operators to use, such as crossover, mutation, selection, and their possibilities of occurrence. In general, the initial population is generated from the spamassassin email dataset with 6,000 email messages.

GA Parameters

GA approach was used to select a set of good finite feature subset for SVMs classifier. The parameters used with their corresponding default value are presented in table 1.

3.5 SVM Parameters Optimization using GA

In addition to the feature selection, proper parameters setting can improve the SVM classification accuracy. The choice of C and the kernel parameter is important to get a good classification rate. In the most case these parameters are tuned manually. In order to automatize this choice we use genetic algorithms. The SVM parameters, C and γ are real, we have to encode them with binary chains; we fix two search intervals, one for each parameter,

$$C_{max} \le C \le C_{min} \qquad (9)$$

$$\text{and } \gamma_{max} \le \gamma \le \gamma_{min} \qquad (10)$$

Thus, a 32 bits encoding scheme of C is given by $C_{b1},\ldots,C_{b32}$ where

$$C_b = \sum_{i=1}^{32} C_{bi} 2^{i-1} \qquad (11)$$

and γ is given by $\gamma_{b1,\dots,}\gamma_{b32}$ where

$$\gamma_b = \sum_{i=1}^{32} \gamma_{bi} 2^{i-1} \qquad (12)$$

$$\text{with } C_b = g_{max}(C - C_{min})/(C_{max} - C_{min}) \qquad (13)$$

$$\text{and } \gamma_b = g_{max}(\gamma - \gamma_{min})/(\gamma_{max} - \gamma_{min}) \qquad (14)$$

$$\text{where } g_{max} = 2^{32} - 1 \qquad (15)$$

Fitness Function

Fitness function was developed to evaluate the effectiveness of each individual in a population; it has an individual as an input and then returns a numerical evaluation that must represent the goodness of the feature subset. The fitness function that was used to evolve the chromosomes population is the SVM classification accuracy. The goal was to see if the GA would maximize this function effectively. Some reasons why SVM must use combined feature selection include its high computational cost and inaccurate classification result on large datasets. In addition, SVMs are formulated as a quadratic programming problem and, therefore, it is difficult to use SVMs to do feature selection directly.

The main steps of the proposed GA-SVM are as follows:

1. E-mail feature extraction.
2. Using Genetic Algorithms to generate and select both the optimal feature subset and SVM parameters at the same time.
3. Classification of the resulting features using SVM.

3.6 The E-Mail Dataset

In this study, SpamAssassin dataset consisting of 6000 emails with the spam rate of 67.04% was used to test the performance of the proposed system. The dataset has two folders containing spam and ham messages. The corpus was divided into training and testing set. Each training set contained 75% of the original set while each testing set contains the rest 25%. The emails are in a bag-of-words vector space representation known as feature vector dictionary. Attributes are the term frequencies of the words. Words occurring more than three times or suspicious are removed in the data set resulting in a dictionary size of about 120,000 words. The data set files are in the sparse data format and are so manually modified to the form suitable for this study. These E-mail datasets are freely downloadable at www.spamassassins.org.

Algorithm for Processing the Dataset

Step1**:** The raw mails (both training and testing) are converted to .xlsx format. In which each row corresponds to one email and each cell contains the index number of the feature vector as indicated in the feature vector dictionary.

Step2**:** The appropriate label (-1, 1) for each row of the entire email dataset as either spam or non-spam (-1 for non-spam, 1 for spam) are determined.

Step3**:** 6000 most frequent words from both spam and ham mails are taken and mixed together to form around 7000 most frequent words.

Step4**:** A unique integer was assigned to each word to act as our features for classification.

Step5**:** 75% of the entire dataset is randomly generated for training and the remaining 25% for testing.

Step6**:** The dataset is passed to the appropriate classifier as required.

3.7 Performance Evaluation Metrics

The two performance evaluation metrics considered in this study are classification accuracy and the computation time.

Accuracy (A) is the percentage of all emails that are correctly categorized. Simply put,

$$A = \frac{Nos\ of\ e-mails\ correctly\ categorized}{Total\ nos\ of\ e-mails} = \frac{N_{ham\rightarrow ham} + N_{spam\rightarrow spam}}{N_{ham} + N_{spam}} \qquad (16)$$

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

Computational time is total finite time in seconds for a program to compile, run and display results after being loaded.

3.8 Simulation Tool

The programming tool used to implement the algorithms is MATLAB. This is because MATLAB is a very powerful computing system for handling calculations involved in scientific and engineering problems. The name MATLAB stands for MATrix LABoratory. With MATLAB, computational and graphical tools to solve relatively complex science and engineering problems can be designed, developed and implemented. Specifically, MATLAB 2007b was used for the development.

4 Results

In this study, two classification methods (Support Vector Machine classifier and the optimized Support Vector Machine classifier) were evaluated. The performance metrics used include accuracy and computation time. The dataset used contained 6,000 emails which is large enough for the study. 4,500 emails were used to train the SVM classifier and 1,500 emails were used to test it. The results obtained are presented in Fig. 2, 3, 4 and 5. After the evaluation of SVM and GA-SVM, the result obtained is summarized and presented in Table 2. GA-SVM yielded a higher classification accuracy of 93.5% within a lesser computational time of 119.562017 seconds while SVM yielded a classification accuracy of 90% within a computational time of 149.9844 seconds. This shows that the hybrid GA-SVM algorithm yielded better classification accuracy than SVM within a reasonable computational time which eventually has eliminated the drawbacks of SVM. These results confirmed the results obtained by Priyanka et al. (2010) and Andrew (2010) on SVM drawbacks; and also the work of Ishibuchi & Nakashima (2000) and Chandra & Nandhini (2010) on GA's ability to find an optimal set of feature weights that improve classification rate, and as an effective computational search method especially in situations where the search space is uncharacterized, not fully understood, or highly dimensional.

5 Conclusion and Future Works

In this study, two classifiers, SVM and GA-SVM were tested to filter spams from the spamassassin dataset of emails. All the emails were classified as spam (1) or legitimate (-1). GA is applied to optimize the feature subset selection and classification parameters for SVM classifier. It eliminates the redundant and irrelevant features in the dataset, and thus reduces the feature vector dimensionality drastically. This helps SVM to select optimal feature subset from the resulting feature subset. The resultant system is called GA-SVM. GA-SVM achieves higher recognition rate using only few feature subset. The hybrid system has shown a significant improvement over SVM in terms of classification accuracy as well as the computational time in the face of a large dataset. Future research work should extend GA-SVM to allow for filtering multi-variable classification problems. Also, a number of other optimization algorithms should be introduced to SVM and other classifiers to investigate the performance of these optimizers on the classification accuracy and computation time of the resulting system over large dataset. Performance of such integrated systems can be evaluated when compared with the existing ones. Evaluation of the performance of such integrated systems on varying sizes of features could also be carried out. An ensemble of classifiers can also be integrated together and optimized using any optimizer of choice. Performance evaluation of the ensemble of classifiers can also be investigated, with or without the optimizer and put to test over small and large dataset, to evaluate the classification accuracy and as well as the computational time.

References

Andrew, W. (2010), "*Statistical Pattern Recognition*", London: Oxford University Press.

Androutsopoulos, I., Palioras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C., & Stamatopoulos, P. (2000), "Learning to filter spam e-mail: a comparison of a Naive Bayesian and memory-based approach", in: Proc. 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), 1–13.

Awad, W., A., & ELseuofi, S., M. (2011), "Machine Learning Methods for Spam E-mail Classification", International Journal of Computer Science & Information Technology (IJCSIT) **3**(1).

Carrera, X. & Marquez, L. (2001), "Boosting trees for anti-spam email filtering", in: Proc. 4th International Conference on Recent Advances in Natural Language Processing.

Chan, R. (2008), "A novel evolutionary data mining algorithm with applications to GA prediction", *IEEE Transactions on Evolutionary Computation* **7**(6), 572-584.

Chandra, E., & Nandhini, K. (2010), "Learning and Optimizing the Features with Genetic Algorithms", International Journal of Computer Applications **9**(6).

Chapelle, O., Haffner, P., & Vapnik, V. (1999), "Support Vector Machines for Histogram-based Image Classification", IEEE Transactions on Neural Networks **10**(5), 1055–1064.

Cohen, W. (1996), "Learning rules that classify e-Mail", in: Proc. AAAI Symposium on Machine Learning in Information Access, 18–25.

De-Jong, K., A., Spears, W., M., & Gordon, F., D. (1993), "Using genetic algorithms for concept learning", *Machine Learning* **13**, 161-188.

El-Naqa, I., Yongyi, Y., Wernick, M., N., Galatsanos, N., P., & Nishikawa, R., M. (2002), "A support Vector Machine approach for Detection of Microcalcifications", Medical Imaging **21**(12), 1552–1563.

Fogel, D., B. (1998), "*Evolutionary Computation: The Fossil Record*", New York: IEEE Press.

Goldberg, D. (1989), "*Genetic Algorithms in Search, Optimization, and Machine Learning Reading*", MA: Addison-Wesley.

Guyon, I., & Elisseef, A. (2003), "An Introduction to Variable and Feature Selection", *Journal of Machine Learning Research*, vol. 3, 1157-1182.

Holland, J., H. (1975) "*Adaptation in Natural and Artificial Systems*", Ann Arbor, MI: University of Michigan Press.

Ishibuchi, H., & Nakashima, T. (2000), "Multi-objective pattern and feature selection by a genetic algorithm", Proc. of Genetic and Evolutionary Computation Conference (Las Vegas, Nevada, U.S.A.), 1069-1076.

John, G., H., Kohavi, R., & Pfleger, K. (1994), "Irrelevant features and the subset selection problem", in *Proc. 11th Int. Conf on Machine Learning*, 121-129.

Kim, K., I., Jung, K., Park, S. H. & Kim, J. H. (2002), "Support Vector Machines for Texture Classification", IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(11), 1542–1550.

Kim, K., I., Jung, K., & Kim, J., H. (2003), "Texture-Based Approach for Text Detection in Images using Support Vector Machines and Continuously Adaptive Mean Shift Algorithm", IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(12), 1631–1639.

Koprinska, I., Poon, J., Clark, J. & Chan, J. (2007), "Learning to classify e-mail", Information Sciences 177, 2167–2187.

Liu, T., Yang, Y., Wan, H., Zeng, H., Chen, Z., & Ma, W. (2005), "Support Vector Machines Classification with a Very Large-scale Taxonomy", SIGKDD Explorations **7**(1), 36 -43.

Liyang, W., Y., Yongyi, R., M., Nishikawa, M., N., & Wernick, A., E. (2005a), "Relevance vector machine for automatic detection of clustered microcalcifications", IEEE Transactions on Medical Imaging **24**(10), 1278–1285.

Liyang, W., Y., Yongyi, R., M., Nishikawa, M., N., & Yulei, J. (2005b), "A study on several machine-learning methods for classification of malignant and benign clustered microcalcifications", IEEE Transactions on Medical Imaging, **24**(3), 371–380.

MailCat, M., Kuposde, L., & Sigre, T. (1999), "K-NN technique for email filing", Communications of the ACM, **41**(8):86-89.

Okunade, O., & Longe, O., B. (2011), "Improved Electronic Mail Classification Using Hybridized Root Word Extractions," Proceedings of ICT4A, 23-26 March, Nigeria: Covenant University & Bells University of Technology, Ota, Ogun State, Nigeria. **3**(1), 49-52.

Pantel, P., Met, S., & Lin, D. (1998), "SpamCop: a spam classification and organization program", in: Proc. AAAI Workshop on Learning for Text Categorization.

Priyanka, C., Rajesh, W., & Sanyam, S. (2010), "Spam Filtering using Support Vector Machine", Special Issue of IJCCT **1**(2, 3, 4), 166-171.

Rennie, J. (2000), "An application of machine learning to e-mail filtering", in: Proc. KDD-2000 Text Mining Workshop.

Rios, G., & Zha, H. (2004), "Exploring support vector machines and random forests for spam detection", in: Proc. First International Conference on Email and Anti Spam (CEAS).

Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998), “A Bayesian approach to filtering junk e-mail”, in: Proc. AAAI Workshop on Learning for Text Categorization.

Sakkis, G., Androutsopoulos, I., Palioras, G., Karkaletsis, V., Spyropoulos, C., & Stamatopoulos, C. (2001) “Stacking classifiers for anti-spam filtering of e-mail”, in: Proc. 6th Conference on Empirical Methods in Natural Language Processing, 44–50.

Song, Q., Hu, W., & Xie, W. (2002), “Robust support vector machine with bullet hole image classification”, IEEE Transactions on System, Man and Cybernetics, Part C, **32**(4), 440–448.

Vapnik, V. (1995), “The nature of statistical learning theory”, New York: Springer Verlag.

Whittaker, Bellotti, V., & Moody, P. (2005), “Introduction to this special issue on revisiting and reinventing e-mail”, Human-Computer Interaction **20,** HCI 1–9.

Youn, & McLeod (2006), “A Comparative Study for Email Classification Group”, doi: 10.1.1.21.1027, Sir-lab.usc.edu.

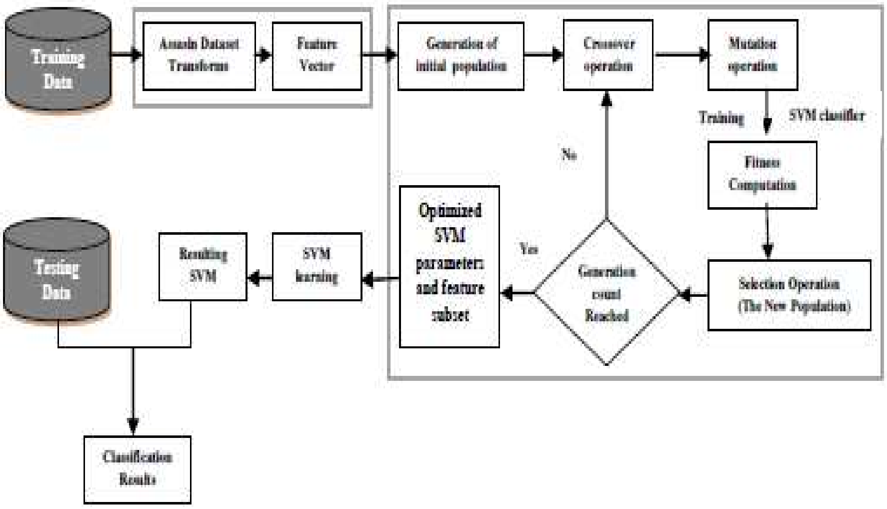
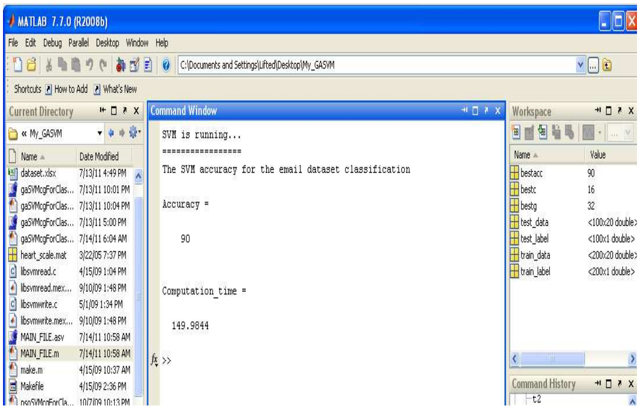


Figure 1. Proposed GA-SVM System for Feature Subset Selection and Classification

Figure 1 is the proposed GA-SVM conceptual model for generating optimal feature subset and best-fit SVM parameters.


MATLAB 7.7.0 (R2008b)
File Edit Debug Parallel Desktop Window Help
C:\Documents and Settings\Lifted\Desktop\My_GASVM
Shortcuts How to Add What's New
Current Directory
« My_GASVM
Command Window
Workspace
Command History
t2

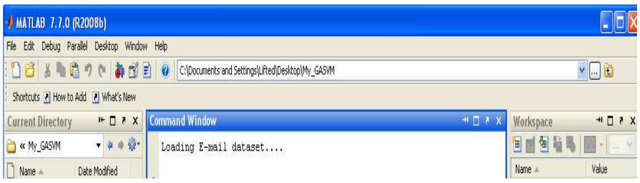
Name	Date Modified
dataset.xlsx	7/13/11 4:49 PM
gaSVMcgForClas...	7/13/11 10:01 PM
gaSVMcgForClas...	7/13/11 10:04 PM
gaSVMcgForClas...	7/13/11 5:00 PM
gaSVMcgForClas...	7/14/11 6:04 AM
heart_scale.mat	3/22/05 7:37 PM
libsvmread.c	4/15/09 1:04 PM
libsvmread.mex...	9/10/09 1:48 PM
libsvmwrite.c	5/1/09 1:34 PM
libsvmwrite.mex...	9/10/09 1:48 PM
MAIN_FILE.asv	7/14/11 10:58 AM
MAIN_FILE.m	7/14/11 10:58 AM
make.m	4/15/09 10:37 AM
Makefile	4/15/09 2:36 PM
psoSVMcgForCla...	10/7/09 10:13 PM

```
SVM is running...
=================
The SVM accuracy for the email dataset classification

Accuracy =

    90


Computation_time =

  149.9844

>>
```

Name	Value
bestacc	90
bestc	16
bestg	32
test_data	<100x20 double>
test_label	<100x1 double>
train_data	<200x20 double>
train_label	<200x1 double>

Figure.2. Results obtained using SVM.

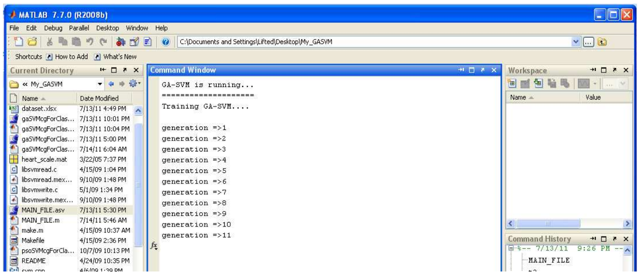
Figure 2 shows the results obtained using SVM. The accuracy and the computation time obtained are 90% and 149.9844 respectively.


MATLAB 7.7.0 (R2008b)
File Edit Debug Parallel Desktop Window Help
C:\Documents and Settings\Lifted\Desktop\My_GASVM
Shortcuts How to Add What's New
Current Directory
« My_GASVM
Name
Date Modified
Command Window
Workspace
Name
Value

```
Loading E-mail dataset....
```

Figure 3. GA-SVM Loading Email Dataset.

Figure 3 shows GA-SVM algorithm loading the spam email assassin dataset for training and testing


MATLAB 7.7.0 (R2008b)
File Edit Debug Parallel Desktop Window Help
C:\Documents and Settings\Lifted\Desktop\My_GASVM
Shortcuts How to Add What's New
Current Directory
« My_GASVM
Command Window
Workspace
Name
Value
Command History

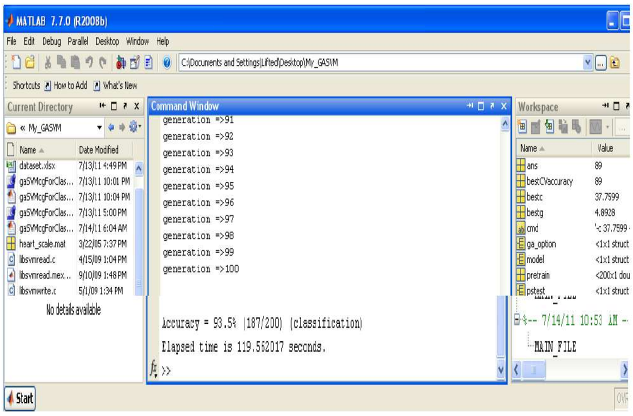
Name	Date Modified
dataset.xlsx	7/13/11 4:49 PM
gaSVMcgForClas...	7/13/11 10:01 PM
gaSVMcgForClas...	7/13/11 10:04 PM
gaSVMcgForClas...	7/13/11 5:00 PM
gaSVMcgForClas...	7/14/11 6:04 AM
heart_scale.mat	3/22/05 7:37 PM
libsvmread.c	4/15/09 1:04 PM
libsvmread.mex...	9/10/09 1:48 PM
libsvmwrite.c	5/1/09 1:34 PM
libsvmwrite.mex...	9/10/09 1:48 PM
MAIN_FILE.asv	7/13/11 5:30 PM
MAIN_FILE.m	7/14/11 5:46 AM
make.m	4/15/09 10:37 AM
Makefile	4/15/09 2:36 PM
psoSVMcgForCla...	10/7/09 10:13 PM
README	4/24/09 10:35 PM

```
GA-SVM is running...
====================
Training GA-SVM....

generation =>1
generation =>2
generation =>3
generation =>4
generation =>5
generation =>6
generation =>7
generation =>8
generation =>9
generation =>10
generation =>11
```

```
%-- 7/13/11  9:26 PM --
MAIN_FILE
```

Figure 4. GA-SVM running.

Figure 4 shows GA-SVM's run time generation of results.


MATLAB 7.7.0 (R2008b)
File Edit Debug Parallel Desktop Window Help
C:\Documents and Settings\Lifted\Desktop\My_GASVM
Shortcuts How to Add What's New
Current Directory
« My_GASVM
Command Window
Workspace
No details available
Start
OVR

Name	Date Modified
dataset.xlsx	7/13/11 4:49 PM
gaSVMcgForClas...	7/13/11 10:01 PM
gaSVMcgForClas...	7/13/11 10:04 PM
gaSVMcgForClas...	7/13/11 5:00 PM
gaSVMcgForClas...	7/14/11 6:04 AM
heart_scale.mat	3/22/05 7:37 PM
libsvmread.c	4/15/09 1:04 PM
libsvmread.mex...	9/10/09 1:48 PM
libsvmwrite.c	5/1/09 1:34 PM

```
generation =>91
generation =>92
generation =>93
generation =>94
generation =>95
generation =>96
generation =>97
generation =>98
generation =>99
generation =>100


Accuracy = 93.5% (187/200) (classification)
Elapsed time is 119.562017 seconds.
>>
```

Name	Value
ans	89
bestCVaccuracy	89
bestc	37.7599
bestg	4.8928
cmd	'-c 37.7599
ga_option	<1x1 struct
model	<1x1 struct
pretrain	<200x1 dou
pstest	<1x1 struct

```
%-- 7/14/11 10:53 AM --
MAIN_FILE
```

Figure 5. Results obtained using GA-SVM.

Figure 5 shows the results obtained using GA-SVM. The accuracy and the computation time obtained are

93.5% and 119.5620 respectively.

Table 1. Parameters Used for the Genetic Process

Parameter	Default Value	Meaning
Population Size	2250	Number of Chromosomes created in each generation
Crossover Rate	0.8	Probability of Crossover
Mutation Rate	0.1	Probability of Mutation
Number of Generations	100	Maximum Number of Generations

Table 1 shows the GA parameters used for the optimal finite feature selection for SVM classifier with their corresponding default value.

Table 2. Summary of Results Obtained

Classifier	Classification Accuracy (%)	Computation time (s)
SVM	90	149.9844
GA-SVM	93.5	119.562017

Table 2 presents the summarized results obtained after the evaluation of SVM and GA-SVM