2013 Fifth International Conference on Computational Intelligence, Modelling and Simulation

# Improving Spam Detection Using Neural Networks Trained by Memetic Algorithm

Shaveen Singh
School of Computing, Information
& Mathematical Science
University of the South Pacific
Suva, Fiji Islands
e-mail: singh_sv@usp.ac.fj

Anish Chand
UXC Eclipse Ltd
Suva, Fiji Islands
e-mail: AChand@uxceclipse.com

Sunil Pranit Lal
School of Computing, Information
& Mathematical Science
University of the South Pacific
Suva, Fiji Islands
e-mail: lal.sunil@ieee.org

*Abstract*— **In this paper we train an Artificial Neural Network (ANN) using Memetic Algorithm (MA) and evaluate its performance on the UCI spambase dataset. The Memetic algorithm incorporates the local search capacity of Simulated Annealing (SA) and the global search capability of Genetic Algorithm (GA) to optimize the parameters of the ANN. The performance of the MA is compared with traditional GA in training the ANN. We further explore the different parameters, mechanisms and architectures used to optimize the performance of the network and attain a practical balance between the global genetic algorithm and the local search technique. Classification using ANN trained by MA yielded better results on the spambase dataset compared with other algorithms reported in literature.**

***Keywords****- Genetic Algorithm, Neural Network, Simulated Annealing, Memetic Algorithms, Spam Classification*

## I. Introduction

Email spam has become a major problem for Internet users and providers. The volume of unsolicited (also called spam or junk) email accounted for more than 85% of all emails sent on the Internet. A 2009 report by Ferris Research placed the economic impact of this phenomenon at $130 Billion globally and around $42 Billion to the US economy [1].

In its infancy, spam was fairly random and would be easily handled using selective keyword filters and black lists. However, modern spam is more tricky and sophisticated. An ongoing challenge, therefore, rests within the development and refinement of automatic classifiers that is able to correctly identify the complex patterns within these messages at a commendable degree of precision.

There has been some important advancement in recent years towards using machine learning approaches to spam detection. Most of these works have focused on perfecting the algorithms and techniques to improve the generalization performance of the classifiers.

Different ensemble methods such as AdaBoost using MLP, single Multi-layer Perceptron (MLP), Reinforcement Learning (RL), Mixture of Experts (MOE) and Naïve Bayes have shown positive results in classification of spam in [2].

AdaBoost is popular algorithm used alongside machine learning algorithms. It has also been used with great success in spam classification [3]. It is one of a family of “boosting” algorithms; these techniques attempt to “boost” the accuracy of a “weak” classifier [4]. The algorithm proceeds in a series of rounds, each time training a different version of the weak classifier. The difference is based on a weight value for each training instance; after each round, all the weights are updated in a way that emphasizes the instances that previous versions of the weak classifier had trouble with.

The reinforcement learning (RL) algorithm [5] focuses on the interaction between an agent and the environment it is embedded in. This involves optimizing the expected reward from of all policies obtained by the system. The policy is adapted using Q-learning [3], a method that improves a policy through the iterative approximation of an evaluation function.

The Mixture of Experts (MOE) architecture is made up of a number of experts and a gate. Each of the experts can produce independent and sensible outputs based on the given data, and the gate needs to decide which expert to be called upon for a particular set of inputs. The MOE architecture in [2] uses Expectation Maximization (EM) algorithm and Reinforcement Learning and MLP algorithms as individual experts and the gate keeps a record for each expert’s past outcome. It penalizes the experts who make higher number of wrong predictions by reducing its weight. This results in a dynamic adjustment of the weights at the gate. This adjustment makes the system focus more on the experts who has a history of high prediction accuracy.

The Bayesian decision theory discussed in [5] uses a statistical approach that makes decisions under uncertainty based on posterior probabilities and costs associated with decisions. In the naive Bayesian spam filter, an incoming email is classified as legitimate if the posterior odds ratio exceeds a certain threshold value. Shilton, A. and Lai, D. T. H., [6] highlight the success of an iterative fuzzy support vector machine (I-FSVM) for classification tasks and particularly the spam dataset. The method works by generating membership values based on their positions relative to the SVM decision function.

This paper outlines the use of a Multilayer Neural Network to approximate the target problem of spam classification. The ANN is initially trained with the commonly used Back Propagation (BP) algorithm and

2166-8531/13 $26.00 © 2013 IEEE
DOI 10.1109/CIMSim.2013.18

Genetic Algorithm (GA). Later we train the neural network with Memetic algorithm, a hybrid approach that leverages on the global search capability of Genetic algorithm combined with local search capability of Simulated Annealing. Potential selection and search schemes of the hybrid solution are discussed and experimented to maintain an acceptable false positive rate. The paper concludes with the performance evaluation of the MA with other research work on spam dataset.

## II. Artificial Neural Network

An Artificial Neural Network (ANN) consists of a number of very simple and highly interconnected processors, also called neurons, which is analogous to the biological neurons of the brain [7]. The neurons are connected by weighted links passing signals from one neuron to another. Each neuron receives a number of input signals through its connection; however, it never produces more than a single output signal. Each neuron is an elementary information-processing unit that computes its activation level based on the weighted sum of input signals passing through a transfer function. A typical Multilayer Perceptron is a feed forward neural network with one or more hidden layers. Each layer is fully connected to the succeeding layer, as shown in Fig 1.

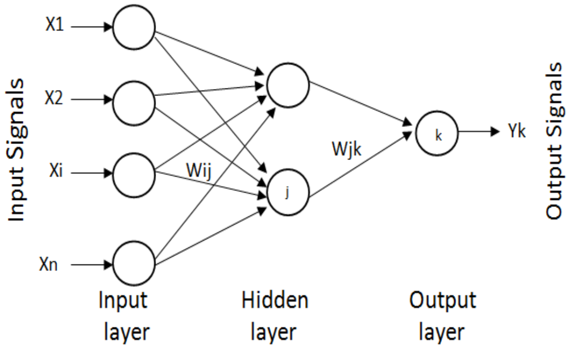

Fig. 1: Three-layer Feed Forward Neural Network

To build an ANN, it is important to first decide the number of hidden layers, the number neurons in each layer and the appropriate activation function. Most discontinuous functions can be represented by a single or two hidden layers, though experimental neural networks may have more than two hidden layers. Having additional layers increase the computational burden exponentially.

In order to optimize the number of hidden neurons, several techniques have been developed in the literature, which correlate it with the number of training samples or the inputs and output layers [8]. If the ANN is using BP learning algorithm, it has been shown that increasing the number of hidden neurons makes it easier to find the global minimum[9,10] however at the expense of over-fitting the training set, which leads to significant deviation in prediction. Other training algorithms exhibit the same problem as well. In this sense, determining the optimum number of neurons means finding a reasonable structure with best generalization [11].

During the training process, the weights of the ANN are updated incrementally. Therefore, besides the network size, training accuracy also depends on the number of training epochs. Too many training epochs lead to overtraining, which is a concept similar to over fitting. To overcome this issue, cross validation is commonly used to check the network quality [12].

## III. Genetic Algorithm

Genetic algorithm is a class of stochastic search algorithm developed by Holland [13]. Inspired from biological evolution, it is a population based search and optimization method inspired by the Neo Darwinism process of reproduction (crossover), mutation, competition and selection. The power of GA comes from its ability to combine both exploration and exploitation operator. The crossover operator basically exploits the convergence of the population to the best solution while the mutation operator explores the search space for promising solutions by introducing diversity.

The genetic algorithm (Fig. 2) proceeds to perform crossover on a selected chromosome array with other suitable selected chromosomes. Random genes in the newly created chromosome arrays are mutated based on the mutation probability before being introduced in the new generation. Through subsequent generations the suitability of the neural network should increase as less fit chromosomes are replaced with better-suited ones. At the end of the run, it is expected to find one or more highly fit chromosomes that would be recognized as weights and thresholds for the ANN.

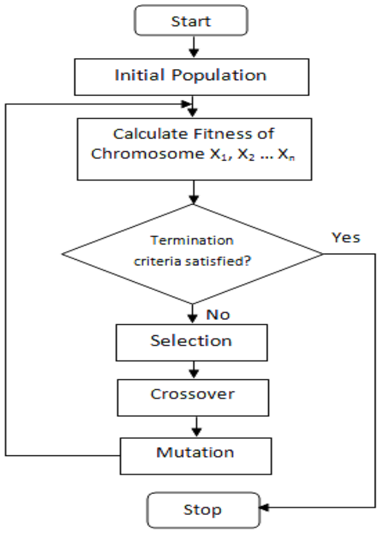
Start
Initial Population
Calculate Fitness of Chromosome $X_1$, $X_2$ … $X_n$
Termination criteria satisfied?
Yes
No
Selection
Crossover
Mutation
Stop

Fig. 2: Basic Structure of Genetic Algorithm

### A. *Chromosome Representation*

In order to train an ANN using GA, the weights and thresholds of ANN are represented in a chromosome (Fig. 3). These weights and thresholds are initialized to random numbers uniformly distributed over a small range [8],

$$\left(-\frac{2.4}{F_i},+\frac{2.4}{F_i}\right) \qquad (1)$$

where $F_i$ is the number of inputs for the given neuron.

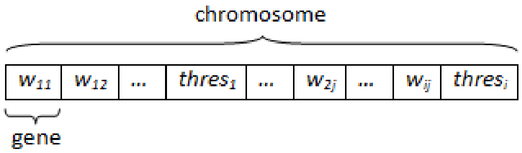

Fig. 3: Representation of the ANN as a chromosome of weights (*w*) and thresholds (*thres*) using real-valued encoding scheme

### B. *Calculating Fitness*

The fitness is a measure of how accurately the ANN is able to represent the training pattern. For classification problems such as the one attempted in this case, mean square error (MSE) is a good indicator of this value. The fitness of a chromosome is inversely proportional to the classification error.

$$Fitness(p)=\frac{1}{Error_p} \qquad (2)$$

where $Error_p$ is the MSE of the chromosome at epoch $p$.

### C. *Selecting Chromosomes*

Most commonly used selection techniques in GA include the roulette wheel selection and elite based selection. In roulette wheel selection, the probability of selection is proportional to the fitness of the chromosomes. The elitist based method however is biased towards selecting fit chromosomes from a pool of top *n* percent of the population pool.

### D. *Crossover of Genes*

The crossover operator produces new offspring from two parent chromosomes, by exchanging selected genes from each parent. Point crossover schemes choose a crossover point where the two parent strings 'break' and then exchanges the chromosome parts after that point.

A second crossover scheme commonly used is uniform crossover. Uniform crossover combines genes sampled uniformly from two parents. Each gene is chosen at random, independent of the others to be included in the offspring. The resulting offspring is a more uniform representation of the parents.

### E. *Mutation Operator*

Mutation has low probability of occurrence, and is a means to introduce diversity in a population by introducing random variations. Mutation flips a randomly selected gene in the chromosome. For a real coded GA, this would mean adding a random noise to a random gene. The noise would be a random value bounded by the mutation amplitude.

## IV. Memetic Algorithm (Hybrid GA)

Memetic algorithms (MA) represent a synergy of evolutionary population-based approach with a local improvement procedure [14].

In such a hybrid systems, GA is complemented by the local search method to explore the promising regions [15]. This accelerates the convergence and drastically reduces the time needed to reach optimum solution.

To achieve this, a simplified Simulated Annealing (SA) technique is incorporated within GA to explore the local promising region of the search space. SA is a smart heuristic for optimization having some similitude with the metal annealing analogy. Given a cost function in a search space, SA replaces the current solution by a random "nearby" solution generated as a function of the global parameter *T* (temperature). The new solution is chosen with a probabilistic measure that depends on the difference between the corresponding function values and on a global parameter *T*. The temperature is gradually decreased during the annealing cycle. This fit chromosome produced after annealing is introduced back into the GA population and is expected to improve the fitness of the entire population.

A model for Memetic Algorithm (Hybrid GA) algorithm is represented in Fig. 4.

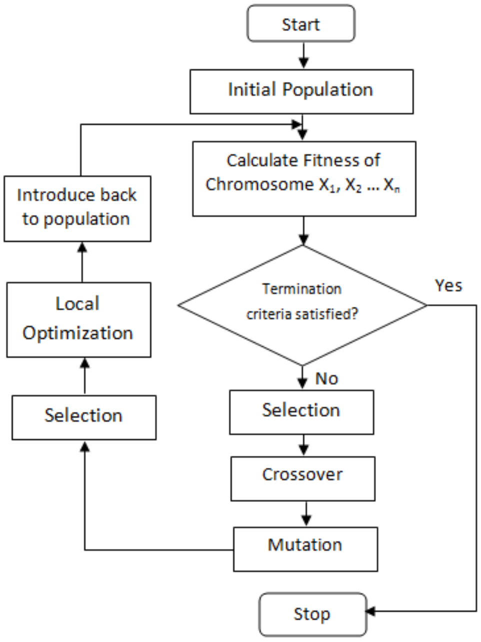
Start
Initial Population
Calculate Fitness of Chromosome $X_1, X_2 \dots X_n$
Introduce back to population
Termination criteria satisfied?
Yes
Local Optimization
No
Selection
Selection
Crossover
Mutation
Stop

Fig. 4: Structure of Hybrid Genetic Algorithm

### A. *Balance between Global and Local Search*

In MA, the main aim of the global search is to guide the search to the basin of attraction, thereby isolating the promising regions of search space, or even hitting the global optimum. The local search method (SA) exploits the information gathered by the global GA method. Hence mutation takes a more exploratory role in this situation for the proposed MA, and as suggested by Land [16], using relatively large mutation values will allow exploration from one search basin to another.

### B. *Frequency of Local Search*

This refers to the number of continuous uninterrupted generations that GA performs before applying the local search algorithm. Lobo and Goldberg [17] addressed the problem of when to employ the global search and local search in order to make the most of either technique. Utilizing the local search too frequently may not necessarily have significant improvement in the convergence but will incur more computational overhead. One way to balance this would be to adopt a technique of probability matching, where local search is employed depending on the efficiency of both genetic and local techniques as the search progresses.

Hacker et al. [15] on the other hand proposed that local search should be ignored till the global search algorithm has defined its basin of attraction. The homogeneity of the population is then checked using the values of coefficient of variance. If the variance is small (population has converged to small area), the local search is employed else the global search is continued. For ANN training, we consider this to be an effective technique to employ.

### C. *Duration of Local Search*

It is important to note that in a hybrid system, long local search duration will require fewer generations of the global algorithm to be executed than a hybrid system with shorter local search duration. Short local search are more likely to keep the population diverse. However, the depth of the search is very much dependent on the type of problem at hand.

A study by Hart [18] found that using short duration local search performed best results for Griewank function [19], whereas long local search produced better results for Rastrigin functions [20]. Hart et. al. [21] concluded that hybrid systems with long local search will be most effective for complex problems.

### D. *Probability and Selection of Local Search Chromosome*

A local search can be applied to either every individual in the population or only a few individuals. Traditional hybrid genetic logarithms apply local search to every individual in the population. This approach however leads to high computational overhead. Hart [18] investigated the impact of the fraction of population that undergo local search with respect to the overall performance of the hybrid algorithm. He concluded that a more selective use of local search produced better results. We adopted this approach to minimize the computation burden of training large networks.

## V. Experimentation

In this section we evaluate the performance of a fully connected multilayer perceptron with sigmoid activation function trained using GA and MA on UCI Spambase dataset.

### A. *Experimental design*

The dataset used in this experiment is from the UCI Machine Learning Repository [1] and consists of 4601 instances. The class distribution comprised of 1813 Spam messages (39.4%) and 2788 Ham messages (60.6%). The attribute information of the dataset is provided in Table I.

To carry out the experiment, the 4601 instances corpus is split into training and testing set in 80:20 ratio. Table II shows the decomposition of the training and testing set respectively. The training set and testing set is randomly selected and the same data has been maintained for all experimentation.

The performance of the ANN Spam Filter in the experiment is evaluated on precision and recall measure [22].

TABLE I
Attributes of Spambase Dataset

| Num of Attributes | Data type | Range | Description |
|---|---|---|---|
| 48 | Real | [0,100] | Word frequency expressed as a percentage |
| 6 | Real | [0,100] | Char frequency expressed as a percentage |
| 1 | Real | [1,…] | Average length of uninterrupted sequences of capital letters |
| 1 | Integer | [1,…] | Average length of uninterrupted sequences of capital letters |
| 1 | Integer | [1,…] | Total number of capital letters in the e-mail |
| 1 | Nominal | {0,1} | Class attribute<br>{0=Ham, 1= Spam} |
| 58 Total Attributes | | | |

TABLE II
Decomposition of Training and Test Data

| Description | Ratio | Num of Instances | Num of Spam | Num of Ham |
|---|---|---|---|---|
| Training Set | 80% | 3680 | 1450 | 2230 |
| Testing Set | 20% | 921 | 363 | 558 |
| Total Corpus | 100% | 4601 | 1813 | 2788 |

Note: The 80:20 split of training and testing data retains the original composition ratio of Spam (39.4%) and Ham at (60.6%).

[1] *http://archive.ics.uci.edu/ml/datasets/Spambase*

### *B. Performance of ANN trained using GA*

The ANN was trained on the spam dataset using GA to optimize the synaptic weights in the fully connected MLP network. Each training run cycle is comprised of 1000 epochs.

Table III, shows the results of GA trained ANN. The GA parameters were set as follows:

Population size = 50
Selection ratio = 10%
Mutation rate = 4%.

The best architecture consisting of 56 inputs layer neurons, 24 hidden layer neurons and 1 output layer neuron was identified using cross validation process.

TABLE III
GA CLASSIFICATION RESULTS (5 RUN AVERAGE)

| Neurons in Hidden Layer | Spam Precision | Legitimate Precision | Spam Recall | Legitimate Recall |
|---|---|---|---|---|
| 19 | 91.06% | 93.43% | 89.81% | 94.27% |
| 24* | 92.31% | 93.16% | 89.26% | 95.16% |
| 29 | 89.86% | 93.71% | 90.06% | 93.37% |
| 34 | 90.17% | 93.30% | 89.67% | 93.64% |

TEST DATA SUMMARY (24 NEURONS)

| | Total Instances | Correctly Classified | Incorrectly Classified | Classification Error |
|---|---|---|---|---|
| Spam | 363 | 32 | 30 | 8.26%(FN) |
| Legitimate | 558 | 515 | 43 | 7.71% (FP) |
| Test Corpus | 921 | 848 | 73 | 7.93% (MC) |

*BEST ARCHITECTURE
FP (FALSE POSITIVE) – LEGITIMATE CLASSIFIED AS SPAM.
FN (FALSE NEGATIVE) – SPAM CLASSIFIED AS LEGITIMATE
MC (MISCLASSIFICATION) – TOTAL MISCLASSIFIED INSTANCES

### *C. Performance of ANN trained using MA*

In training the ANN using the hybrid Memetic Algorithm the only GA parameter changed was the mutation rate, which was increased to 30%. In fact, the experiment confirmed the claim by Rosin et al [23] that the mutation rate in MA can be more adventurous in its role. This aspect allowed us to significantly reduce the overhead of SA to only accepting fitter solution. (The simplified form of SA we used eliminated the probability based hill-climbing).

Empirical results showed that GA exploration slows after 120 generations, thus the local search algorithm SA is activated on every 100$^{th}$ generation. This is in line with Hacker et al. [15] who stated that an effective methodology to employ local optimization is when the population becomes too homogeneous. Then a roulette wheel based selection is used to pick a single chromosome. As opposed to annealing the entire population, this selective approach will incur substantially less computational overhead.

The SA parameters controlled the local search process. The duration of the search is set at 100 iterations with the initial temperature, *T,* for annealing set at 10. *T* is then gradually decreased to 0.01 (stop temperature) by a logarithmic cooling rate over the entire training process. This ensures higher randomness being added to the chromosome initially and lower randomness as the network stabilizes. Our training process runs over 1000 epochs. The primary objective is to introduce new and fitter chromosomes in the population by replacing the weakest individual. This technique ensures the population of good chromosomes is consistently maintained.

Different MLP networks with varying number of hidden layer neurons were trained with the hybrid GA (Fig. 5). Through cross validation process we selected a network consisting of 56 input, 19 hidden, and 1 output layer neuron. This network produced smaller misclassification error (Table IV) and better convergence characteristics (Fig. 6) compared to the best-selected ANN trained using traditional GA.

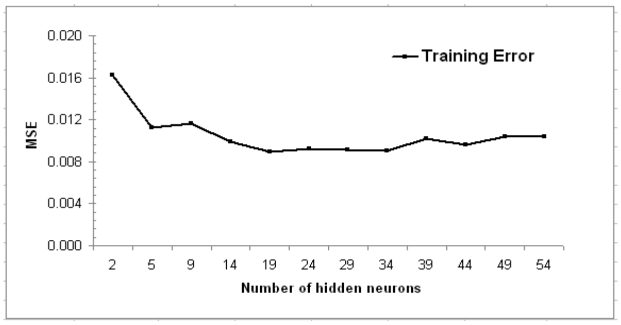

Fig. 5: Relationship between number of hidden layers neurons and MSE

TABLE IV
MA CLASSIFICATION RESULTS (5 RUN AVERAGE)

| Neurons in Hidden Layer | Spam Precision | Legitimate Precision | Spam Recall | Legitimate Recall |
|---|---|---|---|---|
| 14 | 93.12% | 93.75% | 90.19% | 95.66% |
| 19* | 93.14% | 94.63% | 91.67% | 95.61% |
| 24 | 93.21% | 94.06% | 90.70% | 95.70% |
| 29 | 92.46% | 93.56% | 89.92% | 95.23% |
| 34 | 92.27% | 94.06% | 90.74% | 95.05% |
| 39 | 92.58% | 93.63% | 90.03% | 95.30% |
| 44 | 92.07% | 94.07% | 90.80% | 94.91% |
| 29 \| 15 ** | 91.95% | 93.17% | 89.31% | 94.91% |

TEST DATA SUMMARY (19 NEURONS)

| | Total Instances | Correctly Classified | Incorrectly Classified | Classification Error |
|---|---|---|---|---|
| Spam | 363 | 333 | 30 | 8.26% (FN) |
| Legitimate | 558 | 534 | 24 | 4.30% (FP) |
| Test Corpus | 921 | 867 | 54 | 5.86% (MC) |

*OPTIMAL STRUCTURE AS PER MSE (FIG 5)
** ARCHITECTURE WITH 2 HIDDEN LAYERS
FP (FALSE POSITIVE) – LEGITIMATE CLASSIFIED AS SPAM.
FN (FALSE NEGATIVE) – SPAM CLASSIFIED AS LEGITIMATE.
MC (MISCLASSIFICATION) – TOTAL MISCLASSIFIED INSTANCES

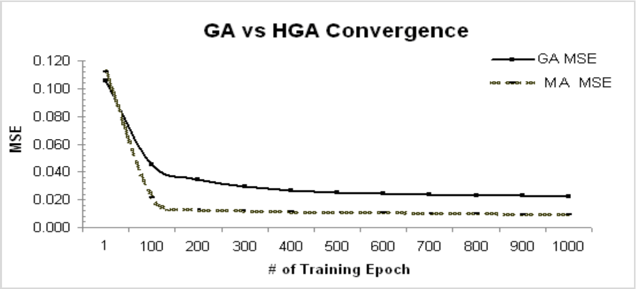

Fig. 6: MSE graph for the best selected ANN trained with GA and MA.

## VI. Discussion

ANN trained using Memetic Algorithm (MA) outperformed other algorithms reported in literature (Table V). This can be attributed to the Memetic algorithm's ability to leverage local search capacity of Simulated Annealing (SA) and global search capability of Genetic Algorithm (GA) to better optimize the parameters of the ANN. Its accuracy in modeling the spam patterns and precisely classifying new instances affirms the potential that hybrid algorithms have in solving complex real world problems.

TABLE V

Algorithm comparison on Spambase datset

| Algorithm | Misclassification |
|---|---|
| Adaptive Boost (AdaBoost)* | 6.48% |
| Reinforcement Learning (RL)* | 7.41% |
| Mixtures of Experts (MOE) architecture* | 7.75% |
| Single MLP* | 8.33% |
| Genetic Algorithm (Table III) | 7.93% |
| Memetic Algorithm (MA) (Table IV) | 5.86% |

*SOURCE [2]

## VII. Conclusion

In this paper, we have attempted to improve spam classification using a hybrid learning algorithm. The MA deployed in this research combined Genetic Algorithm (GA) with Simulated Annealing (SA). Results have confirmed that Memetic Algorithm effectively fights the genetic drift

Furthermore, this hybridization technique accelerates the search towards global optimum and hence guarantees better convergence and finer performance with lesser number of training epochs.

## References

[1] R. Jennings, The Cost of Spam, 2009 : Web page : http://www.ferris.com/research-library/industry-statistics, 2009.

[2] C. Dimitrakakis, and S. Bengio, "Online policy adaptation for ensemble classifier", Neurocomputing, vol 64 , pp. 211-221, 2005.

[3] X. Carreras, and L. M´arquez, "Boosting trees for anti-spam email filtering". In Proceedings of 4th International Conference on Recent Advances in Natural Language Processing, Tzigov Chark, BG, 2001.

[4] R. E. Schapire, "A brief introduction to boosting", In Proceedings of the International Joint Conference on Artificial Intelligence, pp. 1401–1406, 1999.

[5] R.O. Duda, and P. E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.

[6] A. Shilton, and D. T. H. Lai, "Iterative Fuzzy Support Vector Machine Classification", FUZZ-IEEE, pp. 1–6,2007.

[7] M. Negnevitshy, Artificial Intelligence: A Guide to Intelligent Systems, Addison Wesley (Second edition), 2005.

[8] K. Swingler, Applying Neural Networks: A Practical Guide. London, U.K.: Academic, 1996.

[9] S. Lawrence, C. L. Giles, and A. C. Tsoi, "What size neural network gives optimal generalization? Convergence properties of backpropagation," Inst. Adv. Comput. Studies, Univ. Maryland, College Park, MD, Tech. Rep. UMIACS-TR-96-22 and CS-TR-3617, Jun. 1996.

[10] S. Lawrence, C. L. Giles, and A. C. Tsoi, "Lessons in neural network training: Overfitting may be harder than expected," in Proc. 14th Nat. Conf. Artif. Intell., pp. 540–545, 1997.

[11] E. J. Teoh, K. C. Tan, and C. Xiang, "Estimating the number of hidden neurons in a feedforward network using the singular value decomposition," IEEE Trans. Neural Netw., vol. 17, no. 6, pp. 1623–1629, 2006.

[12] R. Setiono, "Feedforward neural network construction using cross validation," Neural Comput., vol. 13, pp. 2865–2877, 2001.

[13] J. Holland, Adaptation in Natural Artificial Systems, University of Michigan Press (Second edition ; MIT Press), 1992.

[14] Pastorino, M. "Reconstruction Algorithm for Electromagnetic Imaging". IEEE Transactions on Instrumentation and Measurement. Volume 53. pp. 692-699, 2004.

[15] K. A. Hacker, J. Eddy, and K. E. Lewis, "Efficient global optimization using hybrid genetic algorithms," In proceedings of 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, USA, 2002.

[16] M. Land, "Evolutionary algorithms with local search for combinatorial optimization," Doctoral Dissertation. San Diego: University of California, 1998.

[17] F. G. Lobo and D. E. Goldberg, "Decision making in a hybrid genetic algorithm," in IEEE International Conference on evolutionary Computation. Piscataway, USA: IEEE Press, pp. 122-125, 1997.

[18] W. E. Hart, "Adaptive global optimization with local search," Doctoral Dissertation. San Diego: University of California, 1994.

[19] A. O. Griewank, "Generalized descent for global optimization," Journal of Optimization Theory and Applications, vol. 34, pp. 11-39, 1981.

[20] A. Törn and A. Zilinskas, "Global optimization," in Lecture Notes in Computer Science, Springer-Verlag, vol. 350: 1989.

[21] W. E. Hart, C. R. Rosin, R. K. Belew, and G. M. Morris, "Improved evolutionary hybrids for flexible ligand docking in AutoDock," in Optimization in Computational Chemistry and Molecular Biology, C. A. Floudas and P. M. Pardalos, Eds.: Springer-Verlag, pp. 209-230, 2000.

[22] J. Clark, I. Koprinska, and J. Poon "A Neural Network Based Approach to Automated E-Mail Classification", IEEE/WIC International Conference on Web Intelligence, 2003.

[23] C. D. Rosin, R. S. Halliday, W. E. Hart, and R. K. Belew, "A comparison of global and local search methods in drug docking," in the Seventh International Conference on Genetic Algorithms, T. Bäck, Ed. Michigan, USA: Morgan Kaufmann, pp. 221-228, 1997.

[24] D. Thierens, D. Goldberg, and P. Guimaraes, "Domino convergence, drift, and the temporal-salience structure of problems," in 1998 IEEE International Conference on Evolutionary Computation Anchorage, USA: IEEE, pp. 535-540, 1998.