I J C T A, 10(9), 2017, pp. 751-763 ISSN: 0974-5572
© International Science Press

Neural Networks Optimization Using Integrated Particle Swarm Optimization for E-Mail Spam Identification

Amaresh Sahu* and Sabyasachi Pattnaik**

ABSTRACT

E-mail is popular because of its simplicity and low cost. E-mail spam, known as unsolicited bulk Email (UBE), junk mail, or unsolicited commercial email (UCE), is the practice of sending unwanted e-mail messages, frequently with commercial content, in large quantities to an indiscriminate set of recipients. Spam e-mail has a considerable negative impact on individuals and organizations, and is measured as a serious waste of resources, time and efforts. Hence, classification of spam E-mail is required. In this paper a new Artificial Neural Network (ANN) training method Integrated Particle Swarm Optimization (IPSO) is proposed for e-mail classification. IPSO training method is associated with adaptive weights, adaptive cognitive and social parameter and self adaptive Gaussian mutation. Experiments and results based on spam dataset show that the developed IPSO has high generalization performance compared to other optimization methods used in the literature for e-mail spam detection.

Index Terms: Multilayer Perceptron, Classification, Feature selection, *Particle Swarm Optimization*, self adaptive Gaussian Mutation

1. INTRODUCTION

The internet is an essential part of our everyday life. Email has become one of the fastest and most economical mode of information exchange. Due to the increase of email users have resulted in the remarkable increase of spam emails in communication during the past few years. E-mail spam, also named as junk e-mail or unsolicited bulk e-mail (UBE). It is a subset of spam that delivers nearly identical messages to a large number of recipients by using the e-mail. Although there is a huge development of anti-spam services and technologies, the number of spam messages continues to increase rapidly. In order to tackle this growing problem, each organization must analyze the various tools available to counter spam in its communication environment. Various tools, like the corporate e-mail system, email filtering gateways, contracted anti-spam services, and end-user training, provide a vital weapon store for handling spam in any organization. E-mail spam is growing progressively since the early 1990s. Email addresses are collected by the spammers from chat-rooms, websites, customer lists, newsgroups, and viruses which harvest users' address books, and are sold to other spammers. Since the cost associated with the spam is borne mostly by the recipient, many individual and business people send their bulk messages in the form of spam. The growth of voluminous spam emails turn out a strain to the Information Technology based organizations and creates billions of dollars lose in terms of productivity. In current trend, spam emails lead to serious security threat, and act as a prime medium for phishing of sensitive information [1]. In addition to this problem, it also spread malicious software to various system users. Therefore, now email classification becomes an essential research area to automatically classify spam emails from original emails. Spam email is prone to misuse therefore, now it is

* Computer Science Engineering Department, Siksha 'O' Anusandhan University, Bhubaneswar, Odisha, India, *Email: amaresh_sahu@yahoo.com*

** Information & Communication Technology Department, Fakir Mohan University, Balasore, Odisha 756019, India, *Email: spattnaik40@yahoo.co.in*

a fascinate problem for individuals and organizations. Automatic email spam classification is more challenging [2] because of unstructured information, more number of features and large number of documents. All of these features may adversely affect performance in terms of quality and speed as the usage increases. Many recent algorithms use only relevant features for classification. Even though many classification techniques have been developed for spam classification, still now 100% accuracy of predicting the spam email is quite impossible. So Identification of best spam classification algorithm is itself a tedious task because of features and drawbacks associated with every algorithm. In this paper, spam dataset from UCI machine learning repository [3] is taken in to consideration for input dataset for analyzing the various classification techniques. Many ways of fighting against spam have been proposed. There are “social” methods like legal measures introduced in US used as anti-spam law [4].

2. RELATED WORK

R. Parimala *et al*. [5] Presented a new feature selection (FS) technique which was guided by F selector Package. They had used nine feature selection techniques in their research such as Correlation based feature selection, Chi-square, Entropy, Information Gain, Gain Ratio, Mutual Information, Symmetrical Uncertainty, One R, Relief and five classification algorithms such as Linear Discriminant Analysis, Random Forest, Rpart, Naïve Byes and Support Vector Machine on spam base dataset. In their evaluation results, they had shown that the methods CFS, Chi-squared, GR, Relief, SU, IG, and one enables the classifiers to accomplish the highest increase in classification accuracy. They concluded that by using FS they can improve the accuracy of Support vector machine classifiers. R. Kishore Kumar *et al*. [6] analyzed spam dataset by using Tanagra data mining tool in his research. Initially, Fisher filtering, Relief, Runs Filtering, Step disc were used for feature selection and feature construction to extract the relevant features. Then classification algorithms such as C4.5, C-PLS, C-RT, CS-CRT, CS-MC4, CS-SVC, ID, K-NN LDA, Log Reg TRIRLS, Multilayer Perceptron, Multi logical Logistic Regression, Naïve Bayes Continuous, PLS-DA, PLSLDA, Rend Tree and SVM were applied over spam base dataset and cross validation test was done for each of these classifiers. W.A. Awad *et al*. [7]reviewed many machine learning methods such as Bayesian classification, k-NN, ANNs, SVMs, Artificial immune system and Rough sets applied on the Spam Assassin spam corpus. They concluded that Naïve bayes method had the highest precision percentage and the k-nearest neighbor had th0e worst precision percentage among all the six algorithms. Also, they shown the rough sets method had very competitive percentage. Rafiqul Islam *et al*. [9] presented a email classification technique based on data filtering method using instance selection method (ISM) to reduce the pointless data instances from training model and then classify the test data. The behavioral features in email like frequency of sending/receiving emails, email attachment, type of attachment, and size of attachment and length of the email were also included for improving performance. In their research, they tested by using five base classifiers Naive Bayes, SVM, IB1, Decision Table and Random Forest on six different types of datasets. A comparative study was performed by Ms.D. Karthika Renuka,et.al.[10], for the classification techniques such as MLP, J48 and Naïve Bayesian, for classifying e-mail spam messages by using WEKA tool. They gathered the datasets from UCI repository. V.Christina,et.al.[8] employed supervised machine learning techniques in his research, namely C4.5 Decision tree classifier, Multilayer perceptron and Naïve Bayes classifier on Spam dataset. Five major features of an e-mail such as: all (A), header (H), body (B), subject (S), and body with subject (B+S), were used to determine the performance of four machine learning algorithms. The training dataset, spam and legitimate message corpus is generated from the mails that they had received in institute mail server for a period of six months. In their experiment, they concluded that by using Multilayer Perceptron classifier they could outperform other classifiers.MLP is a model requires a considerable time for parameter selection and training [11], which has encouraged many researchers to optimize it in several ways. Conventionally, MLP networks are optimized by using gradient based techniques like the Back propagation(BP) algorithm. However, gradient based techniques suffer some major drawbacks like the slow convergence, high dependency on the initial parameters and the high probability of being trapped in local minima [12]. Therefore, researchers proposed stochastic methods for training MLPs. Stochastic methods are based on generating a number of random solutions to solve the problem. Many types of stochastic methods that are getting more interest in training neural networks are the nature-inspired meta-heuristic algorithms. Examples of this type of algorithms are the Genetic Algorithm (GA) [13], Particle Swarm Optimization (PSO) [14], Differential Evolution (DE) [15] and Ant Colony Optimization (ACO) [16] etc. In the context of spam detection, the authors in [13] suggested Genetic Algorithm to train MLP networks to optimize its performance in identifying spam. Their results show that such hybridized method outperform traditional MLP neural network. Improved Particle Swarm Optimization (IPSO) was used for feed forward artificial neural network training and structure optimization[17]. Feed forward Neural Network also trained with hybrid Particle swarm optimization-Back propagation algorithm to improve the speed of convergence in global optimization problems [18]. In this paper, we develop an MLP neural network model trained with the integrated Particle Swarm Optimization (IPSO) algorithm for identifying e-mail spam. In this paper, the proposed method IPSO is compared with other MLPs trained with the Back propagation (BP) algorithm, common meta-heuristic algorithms like Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE) and Ant Colony Optimization (ACO).

3. DATASET DESCRIPTION

3.1. Spambase dataset

It is collected from the UCI Machine Learning Repository, which has the data of 4601 E-mail messages. Spambase dataset has 58 attributes in every instance. The majority of the attributes signifies the frequency of the particular words or else characters into the e-mail to correspond towards the instances. Roughly 39.4% of the emails present in this dataset are spam e-emails. The collected features in this data are based on frequency of some selected words and special characters in the e-mails.

- Word freq w:- 48 attributes is used to define the frequency as well as percentage of the words within e-mails.
- Char freq c:- 6 attributes defines the frequency of the character c as well as percentage of character in emails.
- Char freq cap:- 3 attributes define the longest length, average length and the entire number of capital letters.
- Spam class:- 1 target attribute declared that the e-mail is spam or not.

3.2. Description of the feature extracted

Feature extraction module extract the spam text and the non-spam text, then produce feature dictionary and feature vectors as input of the selected algorithm, the function of feature extraction is to train and test the classifier. For the train part, this module account frequency of words present in the email text, we take words which the time of appearance is more than three times as the feature word of this class. And denote every email in training as a feature vector [33].

4. CONCEPTS AND METHODOLOGIES

4.1. Multilayer Perceptron

Today Multilayer Perceptron (MLP) network is the most widely used neural network classifier. MLP networks are general purpose, flexible, nonlinear models consisting of a number of units or nodes organised into multiple layers. The complexity and architecture of the MLP network can be changed by varying the number of layers and the number of units or nodes in each layer. MLPs can approximate virtually any linear or nonlinear function to any desired accuracy. In other words, MLPs are treated as universal approximators.

MLPs are valuable tools in solving problems, when one has little or no knowledge about the form of the relationship between input vectors and their corresponding outputs in large datasets.

4.2. Particle Swarm Optimization

Particle swarm optimization (PSO) is a nature inspired stochastic algorithm invented by Kennedy and Eberhart [19]. PSO algorithm development has been inspired from bird flocking, fish schooling, and herds of animals.

In PSO, a set of randomly generated solutions/initial swarm propagates in the multidimensional design space towards the optimal solution over a number of iterations/moves based on large amount of information about the multidimensional design space that is assimilates and shared by all members of the swarm. A complete history of the development of the PSO algorithm from simply a motion simulator to heuristic optimization of the PSO algorithm is described in Kennedy and Eberhart [19] [20].

The standard PSO algorithm goes through the computational steps like generation of particles positions and velocities, updating the velocity of each particle and updating the position of each particle.

Here, a particle refers to a potential solution to a problem. A particle $\vec{x}_k$ in d-dimensional design space is represented as $\vec{x}_k$ = (x_{k1}, x_{k2}, x_{k3} ,. . ., x_{kd}), where, k = 1, 2,. . ., n. n is the number of particles in a swarm. Each particle maintains its own velocity and has a memory of its previous best position $\vec{p}_k$ = (p_{k1}, p_{k2}, p_{k3}, . . ., p_{kd}). Let the velocity $\vec{p}_g$ = (p_{g1}, p_{g2}, p_{g3},. . ., p_{gd}) refer to the position found by the kth member of the neighborhood that has done the best performance so far. The particle changes its position in each based on the updates of velocity. In each iteration $\vec{p}_g$ and $\vec{p}_k$ of the current swarm are combined with some weighting and acceleration coefficients to adjust the velocities of the particles in the swarm. The position of the velocity tuning influenced by the particle's previous best position is considered as the cognition component, and the position influenced by the best in the neighborhood is the social component.

The personal best of the kth particle and the global best position can be computed as follows.

$$\vec{p}_k(t+1)\begin{cases}\vec{p}_{k^{(t)}} & if\ f\ \left(\vec{x}_{k^{(t+1)}}\right) \geq f\left(\vec{p}_{k^{(t)}}\right)\forall k, k=1,2...N\\ \vec{x}_{k^{(t+1)}} & \text{otherwise}\end{cases} \tag{1}$$

$$\vec{p}_g(t+1)\begin{cases}\vec{p}_{g^{(t)}} & if\,\forall kf\left(\vec{x}_{k^{(t+1)}}\right) \geq f\left(\vec{p}g^{(t)}\right)\\ \vec{x}_{k^{(t+1)}} & if for any j \quad f\left(\vec{x}_{k^{(t+1)}}\right) < f\left(\vec{p}g^{(t)}\right) \quad j=1,2....N\end{cases} \tag{2}$$

In standard PSO algorithm at iteration t, the velocity and the position can be updated by using Eqs. (3) and (4) respectively.

$$\vec{v}_{k^{(t+1)}} = w \circledast \vec{v}_{k^{(t)}} + \vec{c}_1 \circledast \vec{r}_1(t) \times \left(\vec{p}_k(t) - \vec{x}_k(t)\right) + \vec{c}_2 \circledast \vec{r}_2(t) \circledast \left(\vec{p}_g(t) - \vec{x}_k(t)\right) \tag{3}$$

$$\vec{x}_k(t+1) = \vec{x}_k(t) + \vec{v}_k(t+1) \tag{4}$$

In the equations symbol $\circledast$ denotes point by point vector multiplication. The inertia momentum factor w, (0< w ≤1), self confidence factor c_1 and swarm confidence factor c_2 are non-negative real constants. Randomness (useful for good state space exploitation) is introduced via the vectors of random numbers $\vec{r}_1$ and $\vec{r}_2$. They are usually selected as a uniform random numbers in the range [0, 1]. The original PSO algorithm uses w = 1, $c_1 = 2$ and $c_2 = 2$. Over years many researchers have fined-tuned these useful parameters and found out a very standard optimized values [21].

The steps velocity update, position update, and fitness computations are repeated until a desired convergence criterion is met. The stopping criterion is represented by the maximum change in the best fitness should be smaller than the specified tolerance for a specified number of iterations, It, as shown in Eq. (5). Alternatively, the algorithm execution can be stopped when the velocity updates are close to zero over a number of iterations.

$$\left| f\left(\vec{p}_{g(t)}\right) - f\left(\vec{p}_{g(t+1)}\right) \right| \le \in, t = 2, 3, \ldots, It \tag{5}$$

While empirical evidence has accumulated that the standard PSO algorithm works properly, e.g., it is a useful tool for global optimization problems, In order to address this, a generalized model has been proposed in Clerc and Kennedy [22]. Consequently, the convergence and the stability of the standard PSO has been proposed by many researches [23]; [22]; [24]; [25].

4.3. Adaptive PSO (APSO)

The improved PSO algorithm is based on the standard version of the PSO. In the previous variants of PSO, the main drawbacks of PSO with respect to inefficiency in fine tuning solutions, and a very slow searching around the global optimum. These problems inspired our modifications.

The PSO can be described as follows:

$$\vec{v}_{k(t+1)} = \lambda w \circledast \vec{v}_{k(t)} + \vec{c}_1 \circledast \vec{r}_1(t) \times \left(\vec{p}_k(t) - \vec{x}_k(t)\right) + \vec{c}_2 \circledast \vec{r}_2(t) \circledast \left(\vec{p}_g(t) - \vec{x}_k(t)\right) \tag{6}$$

$$\vec{x}_k(t+1) = \vec{x}_k(t) + \vec{v}_k(t+1) \tag{7}$$

Where, λ *w* is the defined as new adaptive inertia weight. The algorithm, adjusts the parameter λ *w* by reducing it gradually as the generation increases. In APSO, the search space will reduce gradually as the generation increases. So the APSO algorithm is more effective, because the search space is reduced step by step. The search step length for the parameter also reduces correspondingly. Similar to genetic algorithms (GAs) [26], after each generation, the best particle of the swarm in the last generation will replaces the worst particle in the swarm of the current generation, thus the better result can be achieved. The literature [27];[28] are given with several selection strategies of inertia weight. Generally, in the initial stages of algorithm implementation the initial weight should be reduced rapidly, while around the optimum at final stages, the initial weight should be reduced slowly. So in this paper, we adopted the following procedure:

4.3.1. Adaptive Inertia Weight Reduction

We have applied two types of selection strategy sequentially for inertia weight, one is linear selection and the other is the non-linear selection. In linear selection, λ*w* should reduce rapidly, while around the optimum ëw will reduce slowly [18]. Mathematically, it can be described as follows.

Let λw0 is the initial value of inertia weight, λw1 is the end point of linear selection, Generation1 is the number of generations for linear selection and Generation2 is the number of generations of non-linear selection. Then, according to the proposed algorithm for 1 to Generation1 number of generations the inertia weight for PSO will be calculated as given in Equation (8).and from Generation1 to Generation2 will be calculated as given in Equation (9)

$$= \lambda w = \lambda w0 - ((\lambda w1 / \text{Generation1}) \times i) \text{ where, } 1 \le i \le \text{Generation1}. \tag{8}$$

$$= w = (\lambda w0 - \lambda w1) \times \exp(((\text{Generation1} + 1) - i) / i) \text{ where, Generation1} \le i \le \text{Generation1} \tag{9}$$

In particular the value of Generation1 and Generation2 are selected according to empirical knowledge. Although PSO performs very fast for global search as it is capable of finding and exploring promising regions in the search space, quickly searching near global optimum is very slow. The self adaptive evolutionary strategy (ES) is suited for local optimization due to its high probability of generating small Gaussian and Cauchy perturbation [29];[30]. Thus it is capable of fine-tuning those solutions found by PSO. When the global best position of PSO cannot be improved for some successive generations, the self-adaptive ES [31] is used an enhancement operation of $\hat{p}_i$ and $\hat{p}_g$. Thus the self adaptive ES facilitates the convergence of PSO towards global optima. In this study we adapted [30] proposal to use self-adaptive ES like the self adaptive Gaussian and Cauchy mutations for evolving weight parameters of MLP.

4.3.2. Adaptive Cognitive and Social Components

In PSO the search toward the optimum solution is guided by two Stochastic components is used for finding optimal solution. Ratnaweera et al. [34] proposed a version of PSO based on time-varying acceleration coefficient, which decreases the cognitive component and increases the social component by changing the acceleration coefficients c_1 and c_2 with time. It encourages that, by using a large cognitive component and a small social component at the beginning of the search to guarantee particles' moving around the search space to avoid particles' moving toward the population best position. On the other hand, by using a small cognitive component and a large social component allow the particles to converge to the global optima in the latter of the search. The varying scheme of c_1 and c_2 is given as:

$$c_1 = (c_{1f} - c_{1i}) \times (i / (\text{Generation2})) + c_{1i}. \quad (10)$$

$$c_2 = (c_{2f} - c_{2i}) \times (i / (\text{Generation2})) + c_{2i}. \quad (11)$$

where, c_{1f} ,$c_{1i,}$ c_{2f} and c_{2i} are constants, i is the current iteration number and maximum allowable iteration is (Generation2). c_1 varies from 2.5 to 0.5 c_2 varies from 0.5 to 2.5 over the full range of search.

4.3.3. Self Adaptive Evolutionary Strategy (ES)

Although PSO performs very fast for global search as it is capable of finding and exploring promising regions in the search space, quickly searching near global optimum is very slow. The self adaptive evolutionary strategy (ES) is suited for local optimization due to its high probability of generating small Gaussian and Cauchy perturbation [29]; [30]. Thus it is capable of fine-tuning those solutions found by PSO. When the global best position of PSO cannot be improved for some successive generations, the self-adaptive ES [31] is used an enhancement operation of $\hat{p}_i$ and $\hat{p}_g$. Thus the self adaptive ES facilitates the convergence of PSO towards global optima. In this study we adapted [30] proposal to use self-adaptive ES like the self adaptive Gaussian and Cauchy mutations for evolving weight parameters of MLP.

4.3.3.1. Self-adaptive Gaussian mutation

Mutation is carried out by first mutating the velocity and then the position of the particle by the methods given below.

$$v_{ki}(t+1) = v_{ki} \times \exp\left(y' \times N_{gi}(0,1) + y \times N_{ki}(0,1)\right) \quad (12)$$

$$x_{ki}(t + 1) = x_{ki}(t) + v_{ki}(t + 1) \quad (13)$$

Where, $N_{gi}(0, 1)$ is the standard Gaussian density function with respect to the ith dimension of the global best position of the particle. Similarly $N_{ki}(0, 1)$ is the standard Gaussian density function of the ith dimension of the best position found by the particle so far. For this work we follow [31] in setting the values of $y = \sqrt[1/]{2n}$ and $y = \sqrt[1/]{2\sqrt{n}}$ respectively.

4.3.3.2 Self-adaptive Cauchy mutation

A random variable is said to have the Cauchy distribution (C(t)), if it has the following density function:

$$c(t) = \frac{t}{\Pi\left(t^2 + x^2\right)}, -\infty < x < +\infty \tag{14}$$

We will define self adaptive Cauchy mutation as follows:

$$v_{ki}(t+1) = v_{ki} \times \exp\left(y' \times c_{gi}(0,1) + y \times c_{ki}(0,1)\right) \tag{15}$$

$$x_{ki}(t + 1) = x_{ki}(t) + v_{ki}(t + 1), (16)$$

Where, t = 1. In practice, Cauchy mutation is able to make a larger perturbation than Gaussian mutation. This implies that the Cauchy mutation has a higher probability of escaping from the local minima than does Gaussian mutation.

5. PROPOSED INTEGRATED PSO (IPSO)

In PSO, each particle should be kept in a restricted space corresponding to the parameter limitations. This results in decreasing the diversity of the particle. Stagnation occurs in the population if the global best particle does not change its gbest position after some iteration, then the solution may be a local optimal solution. That is, due to its stochastic behavior, it is not possible to get a way to find the global optimum. The major drawback of PSO is that the swarm may converge prematurely. The underlying principle behind this problem is that for the global best PSO, particles converge to a single point, which is in between the boundary of global best and the personal best positions. This point is not guaranteed for finding a local optimum. An additional reason for this problem is that due to the fast rate of information flow between particles, resulting in the creation of similar type of particles with a loss in diversity that increases the chance of being trapped in the local optima. A further drawback is that the performance of stochastic approaches are mostly problem dependent. This dependency usually results from the improper parameter settings in each algorithm. The different proper parameter settings for a stochastic search algorithm result in high performance variances. In general, no single parameter setting can be applied to all different problems. By Increasing the inertia weight (w) in PSO will increase the particle speed, resulting in more exploration (global search) than exploitation (local search). On the other hand, by reducing the inertia weight will decrease the particle speed, resulting in more exploitation than exploration. Thus, finding the best value for the parameter is not an easy task and is problem dependent. Therefore, from the above discussion, it can be concluded that the PSO performance is problem dependent. The problem dependent performance can be addressed through a hybrid mechanism that combines different approaches in order to get benefit from the advantages of each approach.

5.1. Proposed Integrated PSO Algorithm

1. DIVISION OF DATASET

   Divide the dataset into two parts as training set and testing set.

2. MAPPING OF INPUT PATTERNS

   Perform features selection and map each pattern from lower to higher dimension, i.e., expand each selected feature value according to predefined set of functions.

3. RANDOM INITIALIZATION

   Initialize each particle values randomly with small values from the domain [–1, 1].

```
4. WHILE (Termination Criterion Not Met)
         FOR entire swarm
            FOR each particle in the swarm
               FOR each sample of training sample
               Calculate the weighted sum and feed as an input to the node of the output layer.
               Calculate the error and accumulate it.
               END
               Fitness of the particle is equal to the accumulated error. If fitness value is better than
               the best fitness value in history, set current value as the new personal best,
         END
      Choose the particle with best fitness value among all the particles as the global best.
         FOR each particle
         Update weight ($\lambda$ $w$) by using equations(8) and (9).
         c1 and c2 by applying equations (10) and (11).
         calculate particle velocity according to Equation (6).
         Update Particle position according to Equation (7).
         END
   END
   MUTATION
   Apply Cauchy and Gaussian mutation if the position of the global best solution is not changed for a
   successive number of pre-defined generations alternatively by using Equations. (12) and (15).
5. WHILE END
```

6. EXPERIMENTS AND RESULTS

The proposed IPSO-MLP classifier is evaluated using the spam dataset and compared with Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE), Ant Colony Optimization (ACO) and Back Propagation (BP) algorithms. All parameters of algorithms including IPSO are tuned as listed in Table 1.

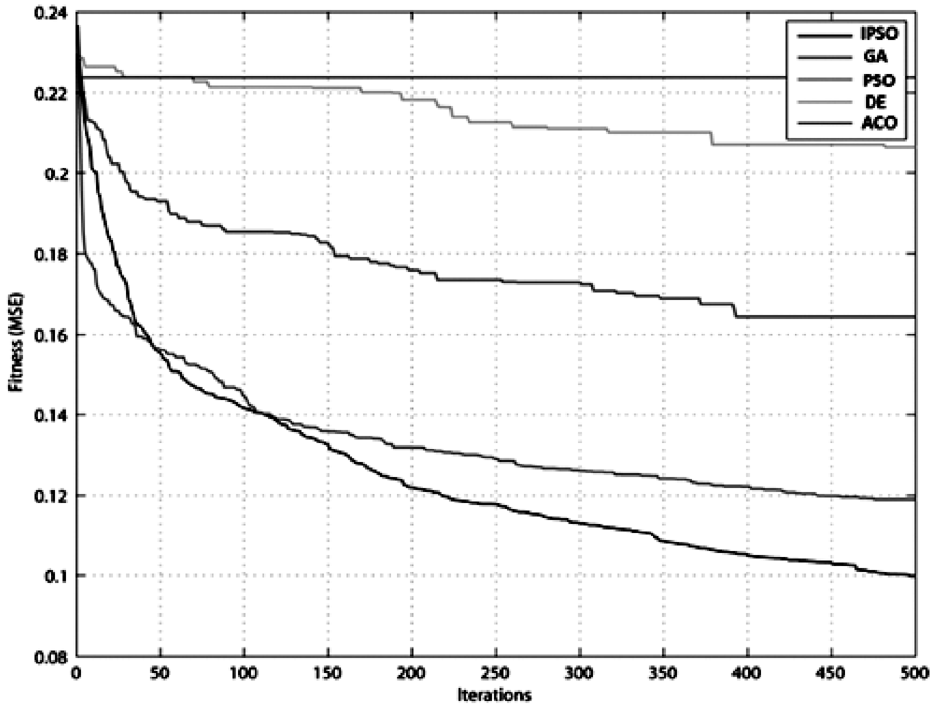
Each individual represents the connection weights of the neural network that connect the input layer to the hidden layer, the weights from the hidden layer to the output layer and the set of bias terms. In our experiments, we tried with different numbers of neurons in the hidden layer of the trained networks: 5, 10, 15 and 20 neurons respectively. Dataset is equally split into two parts: one is used for training and the other is used for testing. Each experiment was repeated 10 times in order to get statistically significant results.

Figure 1. shows the convergence curves for the metaheuristic algorithms based on the Spambase dataset. It can be noticed in the figure that the IPSO trainer has achieved the fastest and lowest convergence curves while GA and PSO come second and ACO is the worst.

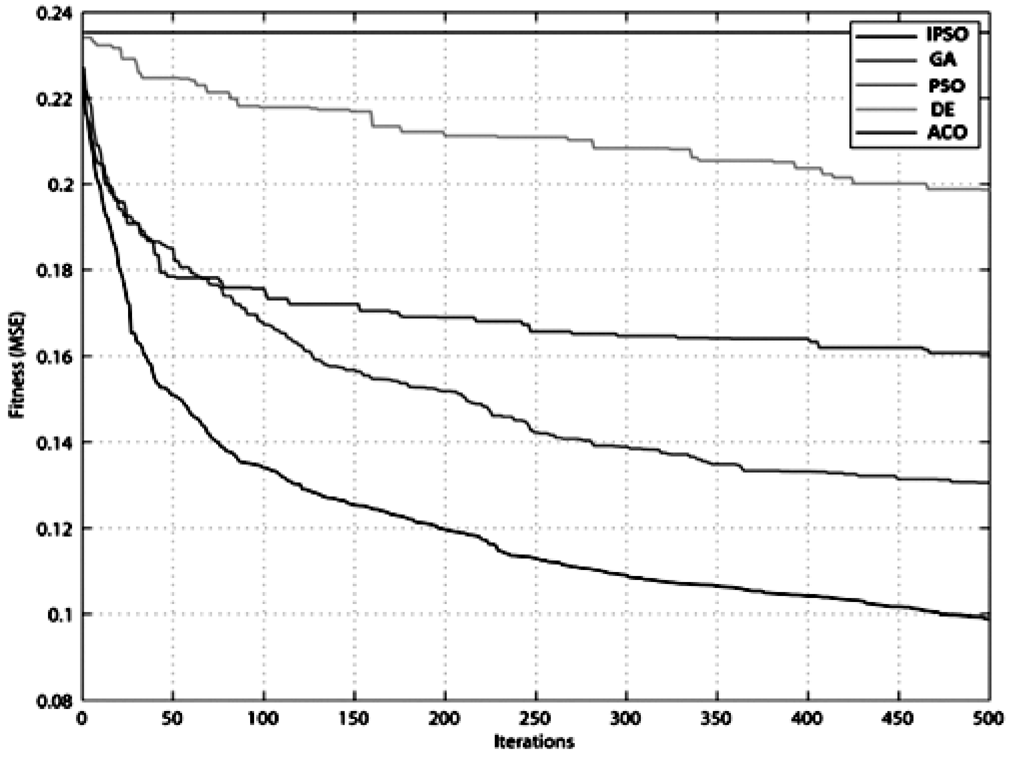
For each one of the training algorithms (GA, PSO, DE, ACO, BP, and IPSO), we find the best representative that work on MLP model and then evaluate it based on the accuracy rate, which is the

Table 1
Parameters settings of the meta-heuristic algorithms.

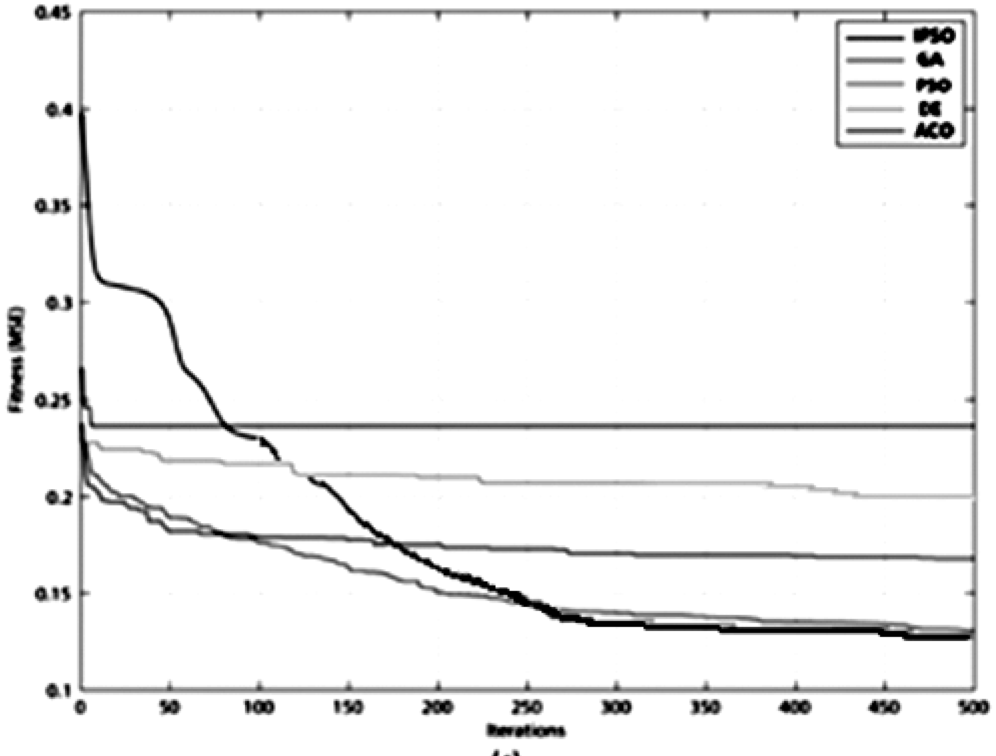
Algorithm	Parameter	Value
IPSO		
	· Inertial value of inertia weight (⋋ 0)	0.9
	· Inertia weight value of the end point of linear section (⋋ 1)	0.4
	· Cognitive parameter(c1)	2.5..0.5
	· Social parameter (c2)	0.5..2.5
GA		
	• Crossover probability	0.9
	• Mutation probability	0.1
	• Selection mechanisim	Stochastic universal sampling
PSO		
	• Acceleration constants	[2.1,2.1]
	• Intertia weights	[0.9,0.6]
DE		
	• Crossover probability	0.9
	• Differential weight	0.5
ACO		
	• Initial pheromone (τ)	1e”06
	• Pheromone update constant (Q)	20
	• Pheromone constant (q)	1
	• Global pheromone decay rate (g p)	0.9
	• Local pheromone decay rate (t p)	0.5
	• Pheromone sensitivity (α)	1
	• Visibility sensitivity (β)	5



(a)



(b)



(c)

number of correctly classified instances divided by the total number of instances. **Table 2** shows the best, average and standard deviation values achieved by each approach for the Spambase dataset. According to the tables, it can be clearly seen that the IPSO trainer achieved the highest averages and best accuracy rates results for spambase dataset. It can be also noticed that 5 neurons in the hidden layer was good enough to train the MLP network. Most of metaheuristic trainers did not achieve any better results with the 10, 15 and 20 neurons in the hidden layer.

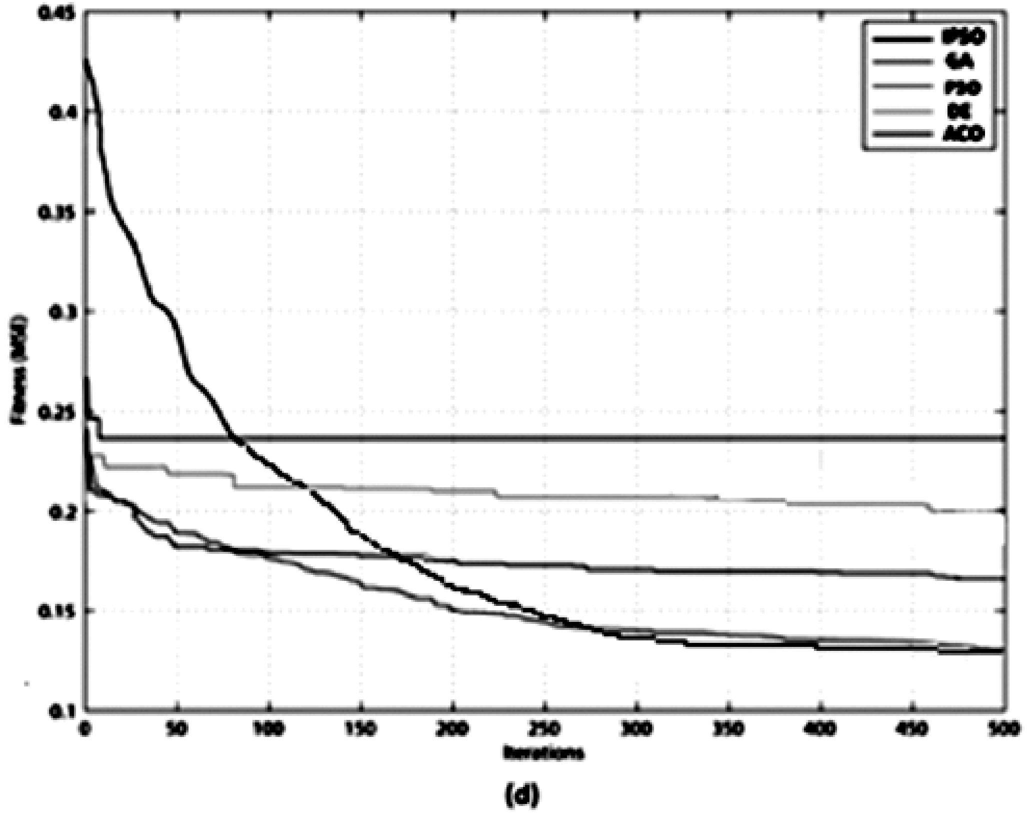


Figure 1: Convergence curves of IPSO, GA, PSO, DE and ACO in training the MLP neural network with 5, 10, 15 and neurons in the hidden layer respectively for the Spambase dataset. (a) MLP-5; (b) MLP-10; (c) MLP-15(d) MLP-15

Table 2
Results of Spambase dataset.

		Number of neurons in the hidden layer			
		5	10	15	20
IPSO	Average	0.885	0.882	0.886	0.885
	Stdv	0.011	0.011	0.012	0.012
	Best	0.905	0.901	0.908	0.910
GA	Average	0.857	0.832	0.832	0.840
	Stdv	0.009	0.011	0.011	0.012
	Best	0.873	0.867	0.902	0.901
PSO	Average	0.784	0.784	0.769	0.767
	Stdv	0.018	0.021	0.010	0.009
	Best	0.817	0.811	0.905	0.901
DE	Average	0.683	0.695	0.710	0.711
	Stdv	0.043	0.037	0.023	0.021
	Best	0.741	0.740	0.752	0.756
ACO	Average	0.642	0.622	0.617	0.612
	Stdv	0.042	0.028	0.062	0.067
	Best	0.753	0.673	0.710	0.715
BP	Average	0.665	0.675	0.730	0.732
	Stdv	0.052	0.060	0.047	0.042
	Best	0.748	0.768	0.792	0.792

7. CONCLUSION

In this work, a nature inspired metaheuristic algorithm named Integrated Particle Swarm Optimization is used to train the Multilayer Perceptron neural network for the purpose of email spam identification. The developed approach is evaluated and compared with four metaheuristic algorithms (GA, PSO, DE and ACO) and the gradient decent based Backpropagation (BP) algorithm. Spam dataset with their extracted features based on the content of the emails are deployed. The IPSO based training approach showed significant improvement in the accuracy of identifying spam e-mails compared to the other metahuristic and gradient decent approaches. The results of the experiments support the conclusion that IPSO is very effective and efficient in avoiding local minima and have a relatively fast convergence.

REFERENCES

[1] M. Chandrasekaran, K. Narayanan, and S. Upadhyaya, "Phishing email detection based on structural properties," in *Proc. 9th Annual NYS Cyber Security Conf.*, Jun. 2006

[2] R.Deepa Lakshmi, N.Radha," Supervised Learning Approach for Spam Classification Analysis using Data Mining Tools", (IJCSE) International Journal on Computer Science and Engineering. Vol.02, No. 08, 2010, 2760-2766.

[3] UCI Machine Learning Repository – Spambase Dataset http://archive.ics.uci.edu/ml/datasets/Spambase

[4] S. Mason. New Law Designed to Limit Amount of Spam in E-Mail.http://www.wral.com/technology/2732168/detail.html

[5] "A Study of Spam E-mail classification using Feature Selection package", R.Parimala, Dr. R. Nallaswamy, National Institute of Technology, Global Journal of Computer Science and Technology, Volume 11 Issue 7 Version 1.0 May 2011.

[6] "Comparative Study on Email Spam Classifier using Data Mining Techniques", R. Kishore Kumar, G. Poonkuzhali, P. Sudhakar, Member, IAENG, Proceedings of the International Multiconference of Engineers and Computer Scientists 2012 Vol I, IMECS 2012, March 14-16, Hong Kong.

[7] "Machine Learning Methods for Spam E-mail Classification", W.A. Awad and S.M. ELseuofi, International Journal of Computer Applications (0975 – 8887) Volume 16– No.1, February 2011.

[8] "Email Spam Filtering using Supervised Machine Learning Techniques", V. Christina, S. Karpagavalli, G. Suganya, (IJCSE) International Journal on Computer Science and EngineeringVol. 02, No. 09, 2010, 3126-3129.

[9] "Email Classification Using Data Reduction Method", Rafiqul Islam and Yang Xiang, member IEEE, School of Information Technology Deakin University, Burwood 3125, Victoria, Australia.

[10] "Spam Classification based on Supervised Learning using Machine Learning Techniques", Ms.D Karthika Renuka, Dr.T.Hamsapriya, Mr.M.Raja Chakkaravarthi, Ms. P. Lakshmi Surya, 978-1-61284-764-1/11/$26.00 ©2011 IEEE.

[11] Yu, B. and Xu, Z.-B. (2008) A Comparative Study for Content-Based Dynamic Spam Classification Using Four Machine Learning Algorithms. Knowledge-Based Systems, 21, 355-362. http://dx.doi.org/10.1016/j.knosys.2008.01.001

[12] Mirjalili, S. (2015) How Effective Is the Grey Wolf Optimizer in Training Multi-Layer Perceptrons. Applied Intelligence, 43, 150-161. http://dx.doi.org/10.1007/s10489-014-0645-7

[13] Arram, A., Mousa, H. and Zainal, A. (2013) Spam Detection Using Hybrid Artificial Neural Network and Genetic Algorithm. 2013 13th International Conference on Intelligent Systems Design and Applications (ISDA), IEEE, 336-340. http://dx.doi.org/10.1109/isda.2013.6920760

[14] Idris, I., Selamat, A., Nguyen, N.T., Omatu, S., Krejcar, O., Kuca, K. and Penhaker, M. (2015) A Combined Negative Selection Algorithm-Particle Swarm Optimization for an Email Spam Detection System. Engineering Applications of Artificial Intelligence, 39, 33-44. http://dx.doi.org/10.1016/j.engappai.2014.11.001

[15] Idris, I., Selamat, A. and Omatu, S. (2014) Hybrid Email Spam Detection Model with Negative Selection Algorithm and Differential Evolution. Engineering Applications of Artificial Intelligence, 28, 97-110.

[16] El-Alfy, E.-S.M. (2009) Discovering Classification Rules for Email Spam Filtering with an Ant Colony Optimization Algorithm. IEEE Congress on Evolutionary Computation, Trondheim, 18-21 May 2009, 1778-1783. http://dx.doi.org/10.1109/cec.2009.4983156,

[17] Yu, J., Xi, L., Wang, S. (2007) An improved Particle Swarm Optimization for evolving feed forward artificial neural networks. Neural Processing Letters 26, 217-231.

[18] Zhang, J., Lok, T., Lyu, M.(2007) A hybrid particle swarm optimization-back propagation algorithm for feed forward neural network training. Applied Mathematics and Computation 185(2), 1026-1037.

[19] Kennedy, J., Eberhart, R.C., (1995) Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948.

[20] Kennedy, J., Eberhart, R.,(2001) Swarm Intelligence Morgan Kaufmann, 3rd edition. Academic Press, New Delhi, India.

[21] Jiang, M., Luo, Y.P., Yang, S.Y., (2007) Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm. Information Processing Letters 102 (1), 8–16.

[22] Clerc, M., Kennedy, J., (2002) The particle swarm—explosion, stability, and conver-gence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 6 (1), 58–73.

[23] Bergh, F.V.D., Engelbrecht, A.P., (2006) A study of particle swarm optimization particle trajectories. Information Sciences 176 (8), 937–971.

[24] Jiang, M., Luo, Y.P., Yang, S.Y., (2007) Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm. Information Processing Letters 102 (1), 8–16.

[25] Trelea, I.C., (2003) The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters 85 (6), 317–325.

[26] Goldberg, David E., (1989) Genetic Algorithms in Search, Optimization and Machine Learning, 1st edition. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[27] Eberhart, R.C., Shi, Y., 2000. Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 Congress on Evolu-tionary Computation 2000. volume 1, pp. 84–88.

[28] Shi, Y., Eberhart, R.C., (1999) Empirical study of particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC 99, volume 3, pp. 6–9.

[29] Rudolph, G., (1997) Local convergence rates of simple evolutionary algorithms with cauchy mutations. IEEE Transactions on Evolutionary Computation 1 (4), 249–258.

[30] Schwefel, H.P., (1981) Numerical Optimization of Computer Models. John Wiley & Sons, Inc., New York, NY, USA.

[31] Yang, J.M., Kao, C.Y., (2001) A robust evolutionary algorithm for training neural net-works. Neural Computing & Applications 10, 214–230.

[32] Bäck, T., Schwefel, H.P., (1993) An overview of evolutionary algorithms for parameter optimization. Evolutionary Computing 1, 1–23.

[33] El-Sayed, M., El-Alfy, Radwan, E., Abdel-Aal(2011) Using GMDH-based networks for improved spam detection and email feature analysis.Applied Soft Computing, Volume 11, Issue 1

[34] Ratnaweera, A., Saman K., Watson HC (2004) Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans Evol Comput 8(3) : 240-255.