An Integrated Model to Email Spam Classification Using an Enhanced Grasshopper Optimization Algorithm to Train a Multilayer Perceptron Neural Network

Sanaa A. A. Ghaleb[1,3,4](✉), Mumtazimah Mohamad[1], Engku Fadzli Hasan Syed Abdullah[1], and Waheed A. H. M. Ghanem[2,3,4]

[1] Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Terengganu, Malaysia
sanaaghaleb.sg@gmail.com
[2] Faculty of Ocean Engineering Technology and Informatics, Terengganu, Malaysia
[3] Faculty of Education-Aden and Faculty of Education-Saber, Aden University, Aden, Yemen
[4] Faculty of Engineering, Aden University, Aden, Yemen

Abstract. Email is an important communication that the Internet has made available. One of the significance is seen in the great ease in which immediate transmission of internet data is done during email transmission. This great ease emerges with a major issue which is the continuous increase in spam emails. Thus, the need for a spam email detector. The versatility and adaptability of the nature of spam influenced past innovations. However, previous techniques have been weakened. This study introduces an email detection model that is designed based on use of an improved version of the grasshopper optimization algorithm to train a Multilayer Perceptron in classifying emails as ham and spam. To validate the performance of EGOA, executed on the spam email dataset are utilized, then the performance was relatively compared with popular search algorithms. The implementation demonstrates that EGOA introduces the best results with high accuracy of up to 96.09%.

Keywords: E-mail · Spam · Grasshopper optimization algorithm · Multilayer perceptron · Classification

1 Introduction

Nowadays, when society develops day by day, people around the world are using an electronic email to communicate by send or receive in a faster way [1]. Email is very familiar in daily life and easier to contact with other people either in short or long distance. In fact, the usage of email is varied in many purposes and needs that may be settled in short period [2, 3]. However, sometimes the user received some email from an unknown person and also unwanted messages [4, 5]. Spam is anonymous and unwanted bulk email [6]. Spam is commonly used for advertising products and services, and this is affecting ordinary Internet users and companies and institutions [7]. The latest statistics

© Springer Nature Singapore Pte Ltd. 2021
M. Anbar et al. (Eds.): ACeS 2020, CCIS 1347, pp. 402–419, 2021.
https://doi.org/10.1007/978-981-33-6835-4_27

in 2019 show that the total number of global email users amounted to 3.9 billion, and It is expected to increase its growth to 4.48 billion users by 2024 [8].

Previously, various methods were used to reduce threat from spam or spam attacks globally. To that effect, knowledge engineering and machine learning [9] are generally the main approaches used in email spam detection [10]. The knowledge engineering method apply network information and IP address methods to decide if an email is ham or spam; this method is called Heuristic or Rule-Based Spam filter. This approach requires a set of rules to be made by some other authority, and these rules established in knowledge engineering approaches [11] to decide message to be flagged as either ham or spam. An example is where a software company provides rules for its spam filtering tools. Additionally, rules set out initially required maintenance regularly and updated, which is time consuming for many users. The Machine learning (ML) approach [12] is better accurate compared to a knowledge engineering approach as it does not need a pre-classified set of rules. Rather, a number of pre-categorized emails are established to be either Spam or not, and then particular algorithms apply to establish and further perform the classification through the pre-categorized emails. Algorithm checks if a message is Spam or ham using the contents of the email and some other characteristics of the email.

2 Related Work

Even with the different approaches and methods being used to combat email spam, the Internet currently still suffers from a very large amount of Spam, and more efforts are needed to resolve this issue that affects millions of people and organizations worldwide. As the fight against Spam is getting more sophisticated, with spammers using new techniques, it is important to take an adaptive approach to fight Spam. Most models emphasize the application and design of calculation algorithms with the aid of simplifying models of various immunological processes. Currently, spam detection is an extensive and weighty research problem that encompasses daily testing and experimentation of numerous algorithms. This study reviews related work that utilized intelligent algorithms that is relevant to the ANNs that addressed the challenge of detecting email spam. Most researchers proposed techniques for training MLPs, based on the production of a diverse random function for sorting out the identified problem in view. The nature-inspired algorithms are established as one of the many method attracting research for the training of neural networks (NN). They are referred to as metaheuristics, and some examples are as follows:

Negative Selection Algorithm (NSA) [13] has proposed to optimize weights of the backpropagation neural network (BPNN). After optimization of BPNN, it is employed into the spam detector. The approach improves the false rate and effective performance. Its performance was evaluated in the dataset on the Spambase. However, it has limitation, such as high false positive detection error. And performance SVM does not better.

Krill Herd Algorithm (KH) [14] have presented a spam detection model using a feedforward neural network, after which it trained KH Algorithm to identify spam and non-spam. A comparison of the model to the conventional backpropagation algorithm. sows that it gave a faster convergence speed and higher accuracy. Its performance was evaluated in the dataset on the SpamAssassin. However, a conspicuous limitation of this algorithm was that sometimes unable escape not the trap of local optima.

Biogeography Based Optimization (BBO) algorithm [15] have proposed a model for detecting spam based on Feedforward NN. That is, after which it trained BBO to identify spam e-mails. The model demonstrated high accuracy compared to the other approaches. Its performance was evaluated in two datasets on the SpamAssassin and Spambase. However, it has limitation, such as poor in exploiting the solutions.

Bat Algorithm [16] have proposed a spam detection model of an artificial neural network (ANN) trained using enhanced Bat Algorithm (EBAT). Its performance was evaluated in two datasets on SpamBase and UK-2011 Webspam training datasets.

A Memetic Algorithm (MA) [17] has proposed a method for detection of spam based on an ANN classifier that can be applied for spam detection even in an offline environment. However, MA is an extension of the traditional genetic algorithm (GA) and is utilized to optimize the interconnection weights of the NN for spam detection. Its performance was evaluated on the UCI Spambase dataset. However, a conspicuous limitation of this algorithm was that poor in local.

Genetic Algorithm (GA) [18] have developed a detection model for training of BPNNs, to optimize the weights of the BPNN to improve the classification of email spam. Its performance was evaluated on his inbox. However, a conspicuous limitation the author was Its performance was evaluated on his inbox.

Genetic Algorithm (GA) [19] have equally presented a new model for spam detection, which was applied to the standard artificial immune system (AIS) and ANN. GA modifications on AIS were used in determining the stop criterion and the number of iteration variables than using fixed variables, while modifications on ANN allowed neurons replacement of useless layers. The main disadvantage of GA is that it cannot guarantee an optimal solution, However, One of the limitations, it high false positive detection error.

Recently, the work by Saremi [20], proposes a new nature-inspired metaheuristic algorithm. GOA is a novel kind of inspired metaheuristic algorithm that is used for optimization. The life cycle of the grasshopper swarms (GS) consists of two stages: nymphs and adults. The nymph grasshopper moves slowly over a small distance, which helps to exploit their living area and eat all plants on their paths. On the other hand, the adult grasshopper has two main tasks: finding food and migration. It can jump high and move over a large distance to find the food and therefore has a larger area to explore. The GOA is able to exploit the search space. But it can fall into local optima that affect its ability on global search task. Hence, the original GOA algorithm is enhanced by integrating several of the mutation operator's technique to improve it power to effectively explore, exploit the search space and rapidly attain optimal point.

In this paper we designed MLP model that was trained using EGOA which achieved higher generalization when relatively compared to other optimization approaches. That is, MLP was trained using EGOA based on a spam dataset and compared to other MLPs trained using common metaheuristic algorithms such as DA, MBO, GOA, HS, ABC, CS, PSO, and SCA. This paper was organized such that Sect. 3 provide a wide discussion on an enhanced grasshopper optimization algorithm, Training algorithm (EGOA-MLP), GOA for training Multilayer Perceptron (MLP), and the datasets that are used to evaluate the MLP-GOA approach. Section 4 shows the experiments, analyzes used, and the results, finally, Sect. 5 discusses conclusion.

3 Proposed Approach

3.1 An Enhanced Grasshopper Optimization Algorithm (EGOA)

The EGOA is described succinctly in this section. It is based on the original GOA algorithm introduced in the previous Sect. 1. More details on the algorithm can be explored by interested readers in [20]. The original GOA algorithm has the capacity to exploit the search space. Notwithstanding, it falls into the trap of local optima in some cases. Such anomaly impacts its performance as regards to the global search. In this context, the original GOA algorithm is enhanced by integrating several of the mutation operator's strategy to improve the potentiality to deeply traverse and exploit the search space and swiftly achieve the optimal value. The detailed pseudocode for the proposed EGOA is given in Algorithm 1. Randomization is required in the GOA component for increasing the diversity of the solutions. In spite of the fact that coefficient c in GOA has a comparative job, it is restricted to certain local jump alterations and hence, corresponds to local search. Utilizing randomization can cause the algorithm to continue further to investigate multifarious regions with high diversity so as to search for the global optimal solution. Furthermore, the probabilistic operator is the mutation operator that randomly adjusts the grasshopper movement in search space, based on the grasshopper's prior probability of existence, also based on the best and worst grasshopper. An increase in the population diversity is experienced by the mutation operator. The mutation operator's strategy in this work is applied with the GOA's exploitation stage to balance 50% of the domain space computed by the GOA. Those strategies allow the original search space to attain the optimal solution swiftly and renovate the out-of-range solutions. Summarily, this proposed algorithm contains two main phases: 1) the initial phase, and 2) the updating phase. In the iteration process of the strategy, the model examines ageing degree of probability values for individual to determine the type of search phase (initial phase and updating phase) to adopt.

Algorithm 1 *Grasshopper optimization algorithm (EGOA)*

```
Initialize all the parameters such as:
Maximum No. of iterations ($iter_{max}$), $c_{max}$, $c_{min}$, and number of population (N);
Generate a random population ($X_i^d$):
(i = 1,2,3...,N) and (d = 1,2,.... dim ➔ no. of dimensions);
Calculate the fitness of each grasshopper;
$\widehat{T}_d$ = the best grasshopper;
While (iter< $iter_{max}$)
    Update the parameter c using Eq. (8);
    for i=1 to N (grasshopper in population) do
      if $\varepsilon_1 \le p$ then
        GOA phase ();
      else
        if $\varepsilon_2 \le limit_1$ then
          Randomly select a grasshopper in population($r_1$);
          $X_{ir_1}^{t+1} = X_{r_1}^t$;
          Randomly select a grasshopper in population($r_2$);
          $X_{ir_2}^{t+1} = X_{r_2}^t$;
          if ($r_1 \ne r_2$) then
            $X_i^{t+1} = w \times (X_{ir_1}^t - X_{best}^t) \times 2\times (rand-1)$;
          else
            $X_i^{t+1} = w \times (X_{ir_2}^t - X_{best}^t) \times 2\times (rand-1)$;
          end if
        else
          $X_i^{t+1} = X_{min}^t + rand \times ( X_{max}^t - X_{min}^t)$;
        end if
      end if
    end for
Update T if there is a better solution;
iter = iter + 1;
end while
Return the best solution of T;
```

The strategy of this mutation operator is a set of random-based modifications aimed at increasing diversity of GOA algorithm and allowing for more mutations in solutions that have been examined in grasshopper search, to assist in jumping out of the local optimum traps. Particularly, the enhanced GOA can exploit the solution in the neighborhood, which is buttressed by the capability to find new areas in the search space. As exploration and exploitation are two crucial characteristics in building an effective optimization algorithm [21]. The paramount enhancement in the algorithm is to incorporate the mutation operator to elevate the diversity of the population for improving the search efficiency and accelerate the convergence to the optimal level. There are two phases in EGOA: initialization and updating phase. In the initialization phase, the search space solutions are identified, and the algorithm assigns values to several parameters. The EGOA officially adopts all the parameters of the GOA and adds two control parameters: *p* and *limit1*. This variable plays a key role in balancing exploitation and exploration of the mutation operator around the optimum value. And ε_1 and ε_2 in [0, 1] are random numbers taken from the same distribution. If $\varepsilon_1 \le p$, the solution will be obtained by using the GOA phase. If $\varepsilon_1 > p$, the solution will be obtained by using the mutation operator. The new mutation operator has one control parameter, limit1, such that if $\varepsilon_2 \le limit_1$, two individual grasshoppers, $x_{r_1}^t and x_{r_2}^t$ are chosen randomly from the population of grasshopper. N is population size, (r_1) or (r_2) are integer of numbers in [1, N]. If (r_1) is not equal to (r_2), then x_i^{t+1} is modified by Eq. (1) else x_i^{t+1} is modified by

Eq. (2). However, if $\varepsilon_2 > limit_1$, x_i^{t+1} is randomly modified from the range of feasible solution. x_{best}^t represents the current best grasshopper in the study population, where t is the current number of generation. A random number generated from same distribution in [0, 1] is a variable rand.

$$x_i^{t+1} = w \times \left(x_{ir_2}^t - x_{best}^t\right) \times 2 \times (rand - 1); \tag{1}$$

$$x_i^{t+1} = w \times \left(x_{ir_1}^t - x_{best}^t\right) \times 2 \times (rand - 1); \tag{2}$$

As presented in our previous work [22–26], the major contribution of the EGOA is to plug the new mutation operator into the GOA to improve population diversity in an endeavor and to improve the performance of GOA to accelerate the convergence to the optimal point.

3.2 Training Algorithm (EGOA-MLP)

In recent time, researchers used a metaheuristic algorithm to train the multilayer perceptron NN, as presented in our previous work [27–30]. The replaced the conventional algorithm with the metaheuristic algorithm, that demonstrated better results compared to conventional algorithm. There are three techniques of utilizing the metaheuristic algorithm for training MLPNNs namely:

1. It is utilized for obtaining biases and weights which enable small error for MLPNN.
2. It is utilized to get an appropriate structure for an MLPNN in a problem.
3. Also, it is utilized to obtain an evolutionary algorithm (EA) to change the parameters of a gradient-based algorithm for learning, namely, the rate of learning and momentum.

During utilizing a multilayer perceptron neural network, the initial stage that must be done is knowing the fixed structure for the NN, that require training. The goal is to locate the right values for connection biases and weights, to reduce error in MLPNNs. Likewise, is the likelihood that a training algorithm is utilized to quantify the best structure for a given problem. By controlling the relationship between neurons, that is, the number of hidden layers, and a number of hidden neurons in MLPNN. Figure 1 indcates the representation of grasshopper in EGOA-MLP.

3.2.1 Two-Layered Multilayer Perceptron NN

This study is based on training a multilayer perceptron NN, to obtain weights and biases that could yield minimal error in MLPNN. The applied algorithms are ABC-MLP, DA-MLP, GOA-MLP, CS-MLP, HS-MLP, MBO-MLP, PSO-MLP and SCA-MLP, respectively. MLPNN has a stable two-layer structure since the input layer consist of n neurons; The hidden layer comprises of hidden neurons and the output layer consist of the resulting neurons. When the hidden transfer function is a sigmoid function (SF), and the output transfer function (OTF) is a linear activation function (LAF), then a corresponding fitness function (FF) is achieved. The FF through the error of the MLPNN is defined to ascertain the fitness and for encoding the weights and biases of the MLPNN. Those elements are detailed below:

3.2.2 Fitness Function

An FF is a type of objective function that is used to calculate the fitness function Eq [3, 4]. In that MLPNNs with two layers contain one input, one hidden, and one output layer; the number of input neurons is equal to (n), the number of hidden neurons is equal to (h), and the number of output neurons is (m). The output of the ith hidden node is calculated as follows:

$$f(s_i) = 1/(1 + exp(-(\sum\nolimits_{i=1}^{n} w_{kf}.x_i - \theta_k))) \tag{3}$$

$$\text{Where} \quad s_f = \sum\nolimits_{i=1}^{n} w_{kf}.x_i - \theta_k \quad f = 1, 2, \ldots., h$$

$$P_{bast} = \frac{fit}{\text{fit}_{fbest+}\text{fit}_{hest}} \tag{4}$$

After computing outputs of the hidden neurons, the output can be given as:

$$o_k = \sum\nolimits_{i=1}^{n} w_{kf}.f(s_i) - \theta_k \tag{5}$$

$$\text{Where} \quad w_{kf}, \ \text{k} = 1, 2, \ldots\ldots, \ \text{n}$$

Conclusively, the learning error which is the FF is computed as:

$$E_k = \sum\nolimits_{i=1}^{m} (o_i^k - d_i^k)^2 \tag{6}$$

$$E = \sum\nolimits_{k=1}^{q} \frac{E_k}{q} \tag{7}$$

Where is the number of training samples, is the expected output of the jth input unit where the kth training sample is utilized and is the actual output of the ith input unit when the kth training sample is applied. Hence, the fitness function of the ith training sample can be obtained as:

$$Fitness\,(x_i) = E(x_i) \tag{8}$$

3.2.3 Encoding Strategy

Encoding strategy is a procedure for representing the biases and weights of the MLPNN. That means its use to represent the biases and weights for nine algorithms such as ABC-MLP, DA- MLP, GOA-MLP, CS-MLP, HS-MLP, MBO-MLP, PSO-MLP, SCA-MLP, and EGOA-MLP. Each agent represents the biases and weights of the MLPNNs architecture. Three procedures represent the biases and weights of MLPNNs for each agent in EA. Encoding strategy consist of binary, vector, and matrix. Each agent encoded as vector for MLPNN training in vector coding. Also, each proxy is encoded as an array in an array encoding, and agents are encoded as binary bits in binary encoding. The effectiveness of using a matrix encoding strategy is suitable for neural network training operations to enable easy implementation of the decoding of NNs.

$$\mathrm{Agent\ (i)} = \left[\mathrm{w}_1,\ \mathrm{b}_{1,}\mathrm{w}_{2,}\mathrm{b}_2\right] \tag{9}$$

$$\mathrm{W}_1 = \begin{bmatrix} w_{12} & w_{22} \\ w_{14} & w_{24} \end{bmatrix},\quad \mathrm{b}_1 = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix},\quad \mathrm{w}_2 = \begin{bmatrix} w_{35} \\ w_{45} \end{bmatrix},\quad \mathrm{b}_2 = [\theta_2] \tag{10}$$

Where, w_1 denote weight matrix for the hidden layer, b_1 referred to as bias matrix for the hidden layer. Then, w_2 and b_2 are the weight and bias in the output layer.

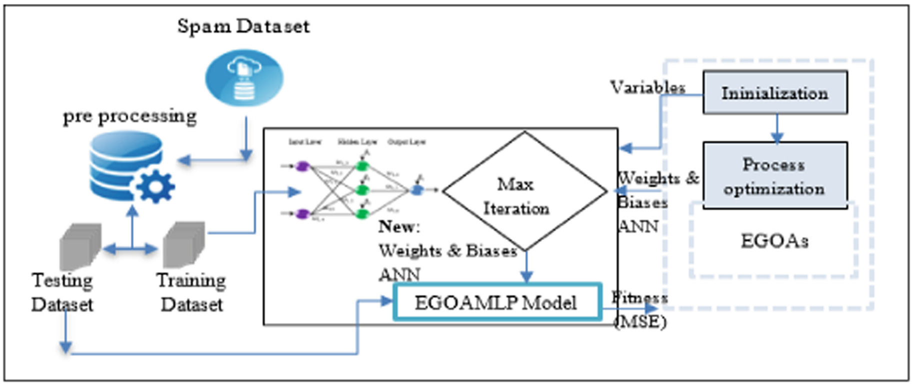


Fig. 1. The EGOAMLP Model

4 Performance Evaluation and Discussion

4.1 Experimental Setup

In this article, the proposed EGOA-MLP is evaluated in the identification of spam email using the SpamAssassin dataset, which is gotten from the UCI Machine Learning (ML) Repository compared with existing techniques. Experiments were implemented and evaluated in MATLAB 2014 on a personal computer (PC) with Core i5 2.7 GHz CPU and 8 gigabyte of RAM. Details of the datasets utilized are depicted in Table 1.

The developed EGOAMLP approach used on a spam dataset which contains 5797 messages (instances) and 140 attributes. The data is obtained from the Spam Assassin dataset and then each data designated as spam or ham. The data are 1897 spam emails and 3900 ham emails. An approximate of 31.4% was obtained as the percentage of the spam email, which results to data imbalance and is more challenging [31]. Randomly selected is the training set and testing set, and the similar data maintained for experimentation. To obtain reliable comparison, common control parameters of all the algorithms have been set as thesame values. Detail of all algorithm's parameters shown in Table 2.

Table 1. Description of Spam Assassin dataset

Type of attribute	No. attribute	Attribute description
Spam	500	Obtained from non-spam trap sources
Easy_Ham	2500	Messages that are non-spam but is easy to differentiate from spam. Not having any spam signature (E.g. HTML)
Easy_Ham_2	1400	Messages that are non-spam
Spam_2	1397	Spam messages
Total	5797	33% Spam ratio

Table 2. Test optimization functions

NO	Name	Equation	Low	Up	Opt
1	Cosine mixture	$f(x) = 0.1\sum_{i=1}^{n}\cos(5\pi x_i) - \sum_{i=1}^{n} x_i^2$	-1	1	0
2	Stepint	$f(x) = 25 + \sum_{i=1}^{n}(\lfloor x_i \rfloor)$	-5.12	5.12	0
3	Mishra's (amgm)	$f(x) = \left[\frac{1}{n}\sum_{i=1}^{n}\lvert x_i\rvert - \left(\prod_{i=1}^{n}\lvert x_i\rvert\right)^{1/n}\right]^2$	0	10	0
4	Alpine No. 1	$f(x) = \sum_{i=1}^{n}\lvert x_i \sin x_i + 0.1x_i\rvert$	0	10	0

4.2 Algorithms and Parameters and Benchmark Functions

The proposed EGOA-MLP performance was assessed by a number of experiments to address global optimization issues. Evaluation is performed using sets of a benchmark as objective functions. To evaluate the proposed EGOA against the original GOA and ABC, CS, PSO, DA, GOA, HS, MBO, PSO, and SCA, 4 standard benchmark functions in Table 2 are applied. These benchmark functions are classified into four characteristics based on the modality (M) into unimodal (U), multimodal (M), separable (S), and non-separable (N). In addition, data on individual function with graphical plots and execution codes in MATLAB are found at www.sfu.ca/~ssurjano/optimization.html). A lot of algorithms laid out all the important parameters as depicted in Table 3.

Table 3. The initial settings of used parameters in the proposed algorithm

Metaheuristic	Parameter	Symbol/Abbr.	Value
(ABC) Artificial Bee Colony	Limit	Limit	100
(CS) Cuckoo Search	Rate of alien eggs/solutions	Pa	0.25
(DA) Dragonfly Algorithm	–	r1,r2	[0, 1]
	–	B	1.5
	Separation weight	S	0.1
	Inertia weight	W	0.2-0.9
	Alignment weight	A	0.1
	Cohesion weight	C	0.7
	Food factor	F	1
	Enemy factor	E	1
(GOA) Grasshopper Optimization Algorithm	C_{min}	–	0.00004
	C_{max}	–	1
	Number of search agents	–	5
(HS) Harmony Search	Harmony memory size	HMS	50
	Harmony memory consideration rate	HMCR	0.95
	Pitch adjustment rate	PAR	0.1
(MBO) Monarch Butterfly Optimization	Butterfly adjusting rate	BAR	0.4167
	Max step	Smax	1.0
	Migration period	Peri	1.2
	Migration ratio	P	0.4
(PSO) Particle Swarm Optimization	Inertial constant	–	0.3
	Cognitive constant	–	1
	Social constant for swarm interaction	–	1
(SCA) Sine Cosine Algorithm	Random number	r_1, r_2, r_3, r_4	[0, 1]
	Linear decreased	A	2

4.3 Evaluation Criteria for Spam Detection

The proposed model has been compared and evaluated. Table 4 shows the confusion matrix, commonly used tool to describe the achievement of NN classifiers. The performance metrics is sows in Table 5.

Table 4. Confusion matrix for classification

Predicted\Actual	Ham	Spam	Total
Ham	TN	FN	TN + FN
Spam	FP	TP	FP + TP
Total	TN + FP	FN + TP	

Table 5. Performance matrix for classification

Measure	Definition	
Accuracy (ACC)	$ACC = \frac{TP+TN}{TP+TN+FP+FN}$	(11)
False alarm rate (FAR)	$FAR = \frac{FP}{FP+TN}$	(12)
Detection rate (DR)	$DR = \frac{TP}{TP+FN}$	(13)

5 Results and Discussion

A number of tests were performed to assess performance on global optimization challenge in order to investigate the advantages of EGOA. The initial investigation was compared to the performance of the EGOA to the first GOA, while the subsequent experiment compared its performance to eight optimization techniques, that is, ABC, CS, DA, GOA, HS, MBO, PSO, and SCA. And the third experiment applied the performance EGOA with eight optimizations of the spam email dataset.

5.1 Experiment 1: Performance of EGOA Against the Standard GOA Algorithm

The experimental test of experiment 1 verified the effectiveness of the proposal for an enhanced GOA against the original GOA. The results of comparing the proposed algorithm and original GOA algorithm to solve four of global numerical optimization problems are given in Table 6 below. Table 6 includes number of optimization functions with (10, 30, 60 and 90) dimensions.

The results of these tables represent the best obtained by the algorithm at "best result" implies the closest result to the exact ideal level of the function (reference to Table 2). The results from each benchmark function comprise a group of lines (row), each corresponds to a various proportions of the function as presented in the Table 2. The Table 2 introduces the result of the minimal value, that is, closest value to the global optimum obtained by the models after the fifty generations with run repeated 30 times.

To highlight the best performance, minimum value (i.e. best result) are distinguish for every benchmark function in bold. Table 6 clearly shows that from all the 8 experiments in Table 2 that is 4 benchmark functions $\times$ 4 dimensions for individual function. The proposed algorithm realized the best optimum value except all dimensions of function

Table 6. The best results from the EGOA and GOA on optimization functions test in (10, 30, 60 and 90) dimensions

Fun	Dim.	GOA	Fun	GOA	Fun.	EGOA	Fun.	E-GOA
F1	10	**−1.81E + 00**	F3	1.32E−11	F1	−2.00E + 00	F3	**0.00E + 00**
	30	**−5.22E + 00**		3.60E−05		−6.00E + 00		**0.00E + 00**
	60	**−9.68E + 00**		6.61E−04		−1.20E + 01		**0.00E + 00**
	90	**−1.34E + 01**		5.54E−03		−1.80E + 01		**0.00E + 00**
F2	10	**−3.00E + 01**	F4	8.74E−01	F2	−3.50E + 01	F4	**0.00E + 00**
	30	−1.00E + 02		1.36E + 01		**−1.55E + 02**		**0.00E + 00**
	60	−1.49E + 02		4.53E + 01		−3.35E + 02		**0.00E + 00**
	90	−1.62E + 02		9.52E + 01		**−5.15E + 02**		**0.00E + 00**
Rank	10	2				2		
	30	1				3		
	60	1				2		
	90	1				3		
Total wins		5				10		

F1 and function F2 in dimension 10, and 60. The GOA algorithm shows better best value on function (F1) in dimension 10,30,60 and 90 on function (F2) in dimension 10. The last four rows in Table 6 show the general ranking for the algorithms with overall dimension variations. One results out of sixteen comparisons demonstrate the most convergent curves of the EGOA versus the original GOA. Figure 2 shows the results obtained when EGOA and GOA applied to solve the F2 Cosine Mixture function. Figure 2, revealed that EGOA is superior to GOA regarding optimization performance for both convergence speed and final result.

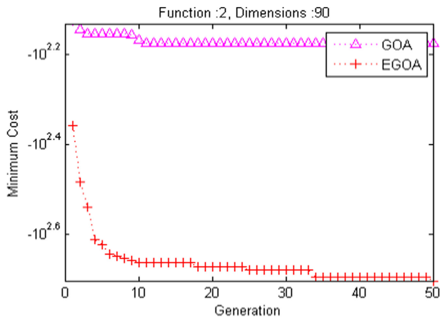


Fig. 2. Convergence curves of GOA, and EGOA against function (F2) with 90 dimensions over 50 generations

5.2 Experiment 2: Performance of EGOA Versus Existing Optimization Methods

The experimental test of experiment 2 verified the effectiveness of the proposal for an enhanced EGOA against old and new optimization algorithm, which are ABC, CS, DA, HS, MBO, PSO, and SCA. The comparisons are benchmarked with similar group of 4 test optimization functions presented in Table 7. The experiments involve four set of results based on the function dimensions that are set to 10, 30, 60, and 90, like the previous experiments (Sec. 4.2). The benchmark function dimensions are updated to determine the modularity of the new model against optimization functions of higher dimension. Utilizing 50 generations with random seeds, the algorithms were run exactly 30 times. In this same manner, the set of test optimization function is kept the same as earlier mentioned. However, the compared algorithm are seven metaheuristic algorithms selected from field swarm intelligence. To put the performance of the proposal for enhancing EGOA in perspective and demonstrate its advantage between similar metaheuristic approaches, therefore, we compared its performance on global numeric optimization issues. The fundamental parameters for the models are depicted in Table 7.

Table 7. The best results derived by the EGOA and 7 algorithms on the test optimization functions in (10, 30, 60 and 90) dimensions

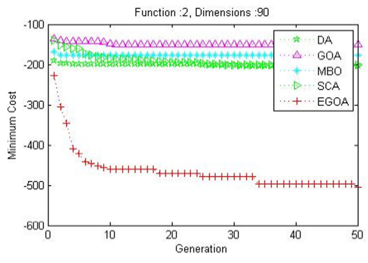
F	D	ABC	CS	PSO	HS	DA	MBO	SCA	EGOA
F1	10	**−2.00E + 00**	**−2.00E + 00**	**−2.00E + 00**	**−2.00E + 00**	**−2.00E + 00**	**−2.00E + 00**	**−2.00E + 00**	**−2.00E + 00**
	30	−5.87E + 00	−5.88E + 00	−5.96E + 00	−6.00E + 00	−5.30E + 00	−6.00E + 00	**−5.20E + 00**	−6.00E + 00
	60	**−8.95E + 00**	−1.14E + 01	−1.14E + 01	−1.20E + 01	−1.01E + 01	−1.20E + 01	−9.79E + 00	−1.20E + 01
	90	**−1.21E + 01**	−1.66E + 01	−1.69E + 01	−1.79E + 01	−1.38E + 01	−1.80E + 01	−1.37E + 01	−1.80E + 01
F2	10	−3.50E + 01	−3.50E + 01	−3.50E + 01	−3.40E + 01	**−2.50E + 01**	−3.50E + 01	−E + 01	−3.50E + 01
	30	−1.40E + 02	−1.07E + 02	−1.07E + 02	−1.48E + 02	−9.30E + 01	**−1.55E + 02**	−1.00E + 02	**−1.55E + 02**
	60	−1.81E + 02	−1.67E + 02	−2.03E + 02	−3.12E + 02	−1.38E + 02	−1.46E + 02	−1.68E + 02	**−3.35E + 02**
	90	−2.10E + 02	−2.08E + 02	−2.60E + 02	−4.59E + 02	−2.16E + 02	−4.77E + 02	−2.20E + 02	**−5.15E + 02**
F3	10	**0.00E + 00**	**0.00E + 00**	3.16E−30	8.86E−10	3.16E−30	3.16E−30	1.96E−04	**0.00E + 00**
	30	4.04E−04	1.79E−04	6.35E−10	1.23E−06	5.90E−11	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
	60	1.46E−01	9.93E−02	3.54E−09	1.65E−03	2.74E−04	1.31E−04	**0.00E + 00**	**0.00E + 00**
	90	7.76E + 00	8.48E−02	9.44E−08	1.52E−01	3.42E−08	2.91E + 00	6.24E−09	**0.00E + 00**

(*continued*)

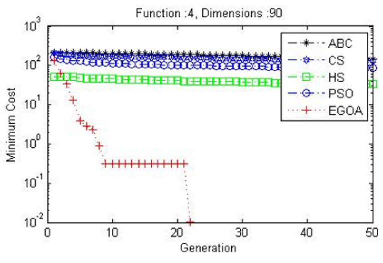
Table 7. (*continued*)

F	D	ABC	CS	PSO	HS	DA	MBO	SCA	EGOA
F4	10	4.89E−02	1.15E−01	1.08E−03	2.80E−02	2.64E−08	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
	30	8.04E + 00	2.09E + 01	5.44E + 00	1.75E−01	1.43E + 01	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
	60	5.48E + 01	6.51E + 01	2.84E + 01	4.65E + 00	6.63E + 01	1.39E−01	**0.00E + 00**	**0.00E + 00**
	90	1.22E + 02	1.15E + 02	4.00E + 01	1.48E + 01	1.06E + 02	1.42E + 02	4.98E−05	**0.00E + 00**
Rank	10	2	2	1	1	2	2	2	3
	30	0	0	0	0	0	3	3	3
	60	1	0	0	0	0	0	2	3
	90	1	0	0	0	0	0	0	3
Total wins		4	2	1	1	2	5	7	12

Table 7 revealed that the EGOA show much a better statistical results compared to the other seven algorithms on almost all the dimensional optimization functions. In contrast, It can be seen in this table that CS, PSO, HS, DA, MFO, and ABC introduced worse results compared to the others in terms of best result (optimum solution), with the outlier of the SCA algorithm that outperformed the proposed algorithm EGOA on 7 dimensions. Due to the limited space, shown two figures to accentuate the majority of the convergent curves of the proposed EGOA against the 7 models. Figure 3 (a-b) show the results for F1 Cosine Mixture, F4 Alpine No.1. From Fig. 3 (a and b), it is indicated that EGOA outperforms all other methods in test optimization functions.



(a)



(b)

Fig. 3. Convergence curves of EGOA, and seven algorithms against function (F2, F4) with 90 dimensions over 50 generations

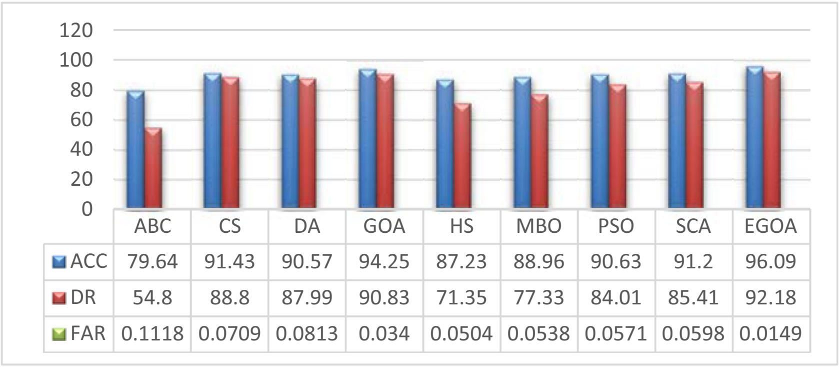
5.3 Experiment 3: Performance of EGOA with Eight Optimizations of the SpamAssassin Dataset

To investigate the usefulness of EGOA, a number of experimental analysis are conducted to evaluate its success on the global optimization challenge. The investigation compared the success of the EGOA with seven optimization methods, which are ABC, CS, DA, GOA, HS, MBO, PSO, and SCA.

Table 8. The estimations of performance of eight algorithms were utilized to train the MLP to detect spam email in Spam Assassin dataset

Alg	ACC	DR	FAR
ABC	79.64	54.8	0.1118
CS	91.43	88.8	0.0709
DE	90.57	87.99	0.0813
GOA	94.25	90.83	0.034
HS	87.23	71.35	0.0504
MBO	88.96	77.33	0.0538
PSO	90.63	84.01	0.0571
SCA	91.2	85.41	0.0598
EGOA	**96.09**	**92.18**	**0.0149**

Tables 8 highlighted performance of the 9 algorithms evaluated. The scores of the new algorithm are marked in bold, and of each algorithm concerning the three major performance indicators such as ACC, DR, and FAR. The results Performance of EGOA with 8 optimizations of the spam email dataset. Our proposal outperformed other algorithms that are utilized to train the MLP. Figure 4 is the result of measurements of the 9 algorithms evaluated. The figure, indicated that EGOA is significantly superior to GOA and 7 algorithms in term of optimization regarding convergence speed. EGOA-MLP accomplished the top performance when compared to the other 8 methods according to the attained results with respect to ACC, DR, and FAR at 96.09%, 92.18, and 0.019, respectively. The GOA-MLP ranked 2nd to ACC of 94.25%, and the false alarm rate of 0.034. Accompanied by CS -MLP with an ACC of 91.43%, but 7th to FAR with values of 0.0709. The SCA-MLP was ranked 4th to ACC, 5th as regards DR, and 6th to FAR with values of 91.2%, 85.41, and 0.0598, respectively. The PSO-MLP model was ranked 5th to ACC as well FAR, and DR at 90.63%, 84.01, and 0.0571 respectively. The HS-MLP and MBO-MLP were closely antecedent to PSO-MLP in terms of FAR with values of 0.0504 and 0.0538. The DE-MLP was positioned 6th as regards ACC, 4th regarding DR, and 8th with reference to FAR with estimations of 85.28%, 78.03, and 0.112, respectively. However, the ABC-MLP model performed poorly with an inferior ACC of 79.64% and DR of 64.8%.



	ABC	CS	DA	GOA	HS	MBO	PSO	SCA	EGOA
🟦 ACC	79.64	91.43	90.57	94.25	87.23	88.96	90.63	91.2	96.09
🟥 DR	54.8	88.8	87.99	90.83	71.35	77.33	84.01	85.41	92.18
🟩 FAR	0.1118	0.0709	0.0813	0.034	0.0504	0.0538	0.0571	0.0598	0.0149

Fig. 4. Convergence curves of EGOA-MLP, nine optimizations of the Assassin Spam dataset

6 Conclusion

This work introduced a new method for spam detection, that is, the EGOA trained MLP. It centres around the applicability of a new algorithm referred to as EGOA for training MLP. The confusion matrix is the fundamental determinant of TP, FN, TN and FP of the new model based on the Assassin spam dataset. Numerous strategies of established and well-known spam detection were relatively compared with the new EGOA model. This study employed eight optimization models to train the MLP such as ABC, HS, CS, DA, GOA, MBO, PSO, and SCA. The EGOA-MLP trained with Assassin Spam dataset had the detection rates of 96.09%, with the false alarm rate value of 0.0149. The results revealed the potential applicability a model for developing practical Email detection. Nonetheless, this study only assessed models with feature spam detection datasets where more selection of the feature has not been considered. Hence, the future research plans will be to verify the success of the EGOA with other NN and investigate its effectiveness with spam email datasets.

References

1. Naem, A.A., Ghali, N.I., Saleh, A.A.: Antlion optimization and boosting classifier for spam email detection. Futur. Comput. Inf. J. **3**(2), 436–442 (2018)
2. ZhiWei, M., Singh, M.M., Zaaba, Z.F.: Email spam detection: a method of meta-classifiers stacking. In: The 6th International Conference on Computing and Informatics, pp. 750–757 (2017)
3. Yang, L., Dumais, S.T., Bennett, P.N., Awadallah, A.H.: Characterizing and predicting enterprise email reply behavior. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 235–244, August 2017
4. Douzi, S., AlShahwan, F., Lemoudden, M., Ouahidi, B.: Hybrid email spam detection model using artificial intelligence. Int. J. Mach. Learn. Comput. **10**(2), 316–322 (2020)
5. Yasin, A., AbuAlrub, F.: Enhance RFID security against Brute force attack based on password strength and Markov model. Int. J. Netw. Secur. Appl **8**(5), 19–38 (2016)
6. Temitayo, F., Stephen, O., Abimbola, A.: Hybrid GA-SVM for efficient feature selection in e-mail classification. Comput. Eng. Intell. Syst. **3**(3), 17–28 (2012)

7. Rawashdeh, G., Bin Mamat, R., Bakar, Z.B.A., Rahim, N.H.A.: Comparative between optimization feature selection by using classifiers algorithms on spam email. Int. J. Electr. Comput. Eng. **2088–8708**, 9 (2019)
8. Statista. https://www.statista.com/statistics/255080/number-of-e-mail-users-world-wide/. Accessed 29 Nov 2020
9. Renuka, D.K., Visalakshi, P., Sankar, T.: Improving E-mail spam classification using ant colony optimization algorithm. Int. J. Comput. Appl. **ICICT 2015**, 22–26 (2015)
10. Dada, E.G., Bassi, J.S., Chiroma, H., Abdulhamid, S.M., Adetunmbi, A.O., Aji-buwa, O.E.: Machine learning for email spam filtering: review, approaches and open research problems. Heliyon **5**(6), e01802 (2019)
11. Bibi, A., Latif, R., Khalid, S., Ahmed, W., Shabir, R.A., Shahryar, T.: Spam mail scanning using machine learning algorithm. JCP **15**(2), 73–84 (2020)
12. Ebadati, O.M.E., Ahmadzadeh, F.: Classification spam email with elimination of unsuitable features with hybrid of GA-naive Bayes. J. Inf. Knowl. Manage. **18**(01), 1950008 (2019)
13. Idris, I.: E-mail spam classification with artificial neural network and negative selection algorithm. Int. J. Comput. Sci. Commun. Netw. **1**(3), 227–231 (2011)
14. Faris, H., Aljarah, I., Alqatawna, J.F.: Optimizing feedforward neural networks using krill herd algorithm for e-mail spam detection. In: 2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), pp. 1–5. IEEE, November 2015
15. Rodan, A., Faris, H., Alqatawna, J.F.: Optimizing feedforward neural networks using Biogeography based optimization for e-mail spam identification. Int. J. Commun. Netw. Syst. Sci. **9**(01), 19 (2016)
16. Jantan, A., Ghanem, W.A., Ghaleb, S.A.: Using modified bat algorithm to train neural networks for spam detection. J. Theoret. Appl. Inf. Technol. **95**(24), 6788–6799 (2017)
17. Singh, S., Chand, A., Lal, S.P.: Improving spam detection using neural networks trained by memetic algorithm. In: 2013 Fifth International Conference on Computational Intelligence, Modelling and Simulation, pp. 55–60. IEEE, September 2013
18. Manjusha, K., Kumar, R.: Spam mail classification using combined approach of Bayesian and neural network. In: 2010 International Conference on Computational Intelligence and Communication Networks, pp. 145–149. IEEE, November 2010
19. Mohammad, A.H., Zitar, R.A.: Application of genetic optimized artificial immune system and neural networks in spam detection. Appl. Soft Comput. **11**(4), 3827–3845 (2011)
20. Saremi, S., Mirjalili, S., Lewis, A.: Grasshopper optimisation algorithm: theory and application. Adv. Eng. Softw. **105**, 30–47 (2017)
21. Heidari, A.A., Faris, H., Aljarah, I., Mirjalili, S.: An efficient hybrid multilayer perceptron neural network with grasshopper optimization. Soft. Comput. **23**(17), 7941–7958 (2019). https://doi.org/10.1007/s00500-018-3424-2
22. Ghanem, W.A., Jantan, A.: Hybridizing artificial bee colony with monarch butterfly optimization for numerical optimization problems. Neural Comput. Appl. **30**(1), 163–181 (2018). https://doi.org/10.1007/s00521-016-2665-1
23. Ghanem, W.A.H.M., Jantan, A.: A novel hybrid artificial bee colony with monarch butterfly optimization for global optimization problems. In: Vasant, P., Litvinchev, I., Marmolejo-Saucedo, J. (eds.) Modeling, Simulation, and Optimization. EAI/Springer Innovations in Communication and Computing, pp. 27–38. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-70542-2_3
24. Ghanem, W.A., Jantan, A.: An enhanced Bat algorithm with mutation operator for numerical optimization problems. Neural Comput. Appl. **31**(1), 617–651 (2019). https://doi.org/10.1007/s00521-017-3021-9

25. Ghanem, W.A.H., Jantan, A., Ghaleb, S.A.A., Nasser, A.B.: An efficient intrusion detection model based on hybridization of artificial bee colony and dragonfly algorithms for training multilayer perceptrons. IEEE Access **8**, 130452–130475 (2020)
26. Ghanem, W.A., Jantan, A.: Training a neural network for cyberattack classification applications using hybridization of an artificial bee colony and monarch butterfly optimization. Neural Process. Lett. **51**(1), 905–946 (2020). https://doi.org/10.1007/s11063-019-10120-x
27. Ghanem, W.A., Jantan, A.: A cognitively inspired hybridization of artificial bee colony and dragonfly algorithms for training multi-layer perceptrons. Cogn. Comput. **10**(6), 1096–1134 (2018). https://doi.org/10.1007/s12559-018-9588-3
28. Ghanem, W.A., Jantan, A.: A new approach for intrusion detection system based on training multilayer perceptron by using enhanced Bat algorithm. Neural Comput. Appl., 1–34 (2019). https://doi.org/10.1007/s00521-019-04655-2
29. Ghanem, W.A.H., Jantan, A.: Using hybrid artificial bee colony algorithm and particle swarm optimization for training feed-forward neural networks. J. Theoret. Appl. Inf. Technol. **67**(3), 664-674 (2014)
30. Ghanem, W.A.H. Jantan, A.: Swarm intelligence and neural network for data classification. In 2014 IEEE International Conference on Control System, Computing and Engineering (ICCSCE 2014), pp. 196–201. IEEE, November 2014
31. Hopkins, M., et al.: UCI Machine Learning Repository: SpamAssassin Data Set. https://www.kaggle.com/beatoa/spamassassin-public-corpus