IASJ المجلات الاكاديمية العراقية

Tikrit Journal of Pure Science

ISSN: 1813 – 1662 (Print) --- E-ISSN: 2415 – 1726 (Online)

Journal Homepage: http://tjps.tu.edu.iq/index.php/j

A New Hybrid Grasshopper Optimization - Backpropagation for Feedforward Neural Network Training

Samer Alsammarraie , Nazar K. Hussein

Department of Mathematics, College of Computer Science and Mathematics, University of Tikrit, Tikrit, Iraq

https://doi.org/10.25130/tjps.v25i1.221

ARTICLE INFO.

Article history:

-Received: 29 / 8 / 2019

-Accepted: 30 / 9 / 2019

-Available online: / / 2019

Keywords: artificial neural network, Grasshopper optimization algorithm, backpropagation algorithm, optimization.

Corresponding Author:

Name: Nazar K. Hussein

E-mail: nazar.dikhil@tu.edu.iq

Tel:

ABSTRACT

The Grasshopper optimization algorithm showed a rapid converge in the initial phases of the global search, however while being around the global optimum, the searching process became so slow. On the contrary, the gradient descending method around achieved faster convergent speed global optimum, and the convergent accuracy was showed to be higher at the same time. As a result, the proposed hybrid algorithm combined Grasshopper optimization algorithm (GOA) along with the back-propagation (BP) algorithm, also referred to as GOA–BP algorithm, was introduced to provide training to the weights of the feed forward neural network (FNN), the proposed hybrid algorithm can utilize the strong global searching ability of the GOA, and the intense local searching ability of the Back-Propagation algorithm. The results of experiments showed that the proposed hybrid GOA–BP algorithm was better and faster in convergent speed and accuracy than the Grasshopper optimization algorithm (GOA) and BP algorithm.

Introduction

Artificial neural networks (ANNs)[1] are able to recognize, learn, and address many sophisticated and intricate tasks in science and engineering [2,3,4,5] .ANNs also can be considered as one of the influential learning models that has the ability to show the required results while dealing with multiple supervised and unsupervised challenges of machine learning [6]. ANNs are designed in a good way to handle duties of machine perception, where the primary convenient features can't be individually interpreted [7]. therefore, ANNs have been broadly utilized and scrutinized to determine the recognition of pattern, classification, prediction tasks, and clustering [3;8;9;10;11;12]. For example, researchers effectively utilized different types of neural networks in order to address hard diagnostic tasks in medical applications, [13]. Also they use neural networks to arrange biomedical information in classes such as diabetes and heart diseases. These methods success is due to the abilities of ANNs to process big volume of information in the training phase and minimize the wanted diagnosis time [14].

Single-layer perceptron (SLP) has one input layer and one output layer only, it is considered as the most basic and purest form of ANNs [15]. It has been proved that SLPs don't have the ability to effectively deal with the separable nonlinear patterns [16]. As a result, multilayer perceptron (MLP) ANNs are proposed, and they are not suffering from the disadvantages of SLP by using one or more than one hidden layers in the Artificial neural networks. As a result, the most used type of ANNs is the Multilayer type [17]. Few of the main powerful features of the MLP are its fault tolerance, non-linearity, robustness to noise, learning capacity, parallelism, and its great ability to generalize [14]. The effectiveness of Artificial Neural Networks can be strongly enhanced based on the learning method applied to provide training for the system [16]. As proposed in Kolmogorov's theorem, Multiple Layer Perceptron with one hidden layer is able to approximate multiple continuous functions [18].

In the recent times, learning models grab the attention in the community of machine learning. For example, [19] proposed a belief-based chaos-based technique to provide a support for the description of vector data. An automatic description for the support vector data is developed by [20,21] Introduced a chaotic algorithm of bat to effectively handle the weighted description of support vector data. Also there are several other techniques that are devoted to the purpose of development of linked concepts and

learning models [22, 23, 24, 25; 26,27; 28]. For the Multilayer perceptron, either unsupervised or supervised training methods may be used to train ANNs [3]. Also we can see the main supervised trainers in two different categories: stochastic-based approaches and the gradient-based approaches [16]. The different types of techniques for back-propagation may be considered as one of the most celebrated types of approaches that are based on gradient-descent [29]. We can make use of These methods as an algorithm for local search, due to their tendency to exploitation [30]. To reach the global optimum, every optimizer should be able to make a good balance between two of the main capacities: exploration and exploitation. While the exploration is needed to explore new and unknown regions on the fitness space, the exploitation is vital for the use of the previously explored positions [31;32]. However, lazy convergence, and the intense reliance on parameters are some of many limitations of the gradient-descent strategies [33; 34]. Regarding this, the necessity of multiple stochastic optimizers to train Multilayer perceptron networks is identified in literature [16].

When the goal is to optimize the weights and the structure simultaneously, then the MLP trainer should address an extensive issue [16]. A lot of researchers frequently make use of the swarm-based and evolutionary (MHAs) or meta-heuristic algorithms to address MLP networks. The meta-trainers may deploy them to optimize not exclusively the parameters but also the weights of the connection and the structure of the MLP network.

Differential evolution (DE) [35] and evolutionary tactic (ES) [36] are well-known, popular evolutionary optimizers. Several researchers have used this MHAs for training the MLP network [36; 37; 38; 39]. The results show that the MHA can provide preferred solutions to implement Multilayer perceptron networks. (SMHAs) or the swarm based MHAs - is a different type of MHA. SMHA try to use the idealistic and self-organized cooperative movement of the swarm, like birds, wolves, grasshoppers and whales in nature [31]. (PSO) or the Particle Swarm Optimization is one of the first revolutionary SMHAs that uses the social life of birds as an inspiration [41]. The optimization of the colony of ants (ACO) [43] and Artificial bee colony (ABC) [42] might be considered as an another well-established SMHA. The efficiency of ACO, ABC, PSO based trainers accompanied by their modified variants for Multilayer perceptron networks has been intensively assessed in many previous researchers. Study of these contributions can be found in [44, 45, 46, 47, 42, 16]. The results confirm that those trainers can appear to have a high tendency of avoiding LO while dealing with the MLP networks. The (WOA) or the Whale Optimization algorithm [3], (LSA) or the Lightning Search Algorithm [14] (GWO) or the Gray Wave Optimization [33], and the (SSO) or the Social Spider Optimizer [33] can be considered as one of the most recent applied training methods this problem.

Although many MHAs have been discussed and analyzed in the previous works, searching for global results remains available [14, 34]. (NFL) or No Free Lunch theorem states that there is no MHA that outperform every other MHAs for all the kinds of problems [48]. As a result, by Referring to NFL theorem, new MHAs can still be developed to train Multilayer perceptron networks. Inspired by this, this research's main motivation is to design a new MHA-based training algorithm for MLP networks. One of the most observable MHAs is the (GOA) [49]. Also GOA is a recently discovered nature-inspired optimization algorithm proposed by [50]. The GOA is highly efficient SMHA that uses the behaviors of grasshoppers in team hunting in the nature as an inspiration. GOA showed extensively promising results specially in optimizing difficult benchmark functions and complex problems [50, 51]. The main contributions of this paper may be summarized as following:

- A brand new hybrid training algorithm has been developed by using the Grasshopper Optimization Algorithm for Multilayer Perceptron neural networks.
- The exploitative and exploratory capabilities of GOA has been used to determine the optimal biases and the optimal weights of the Multilayer perceptron simultaneously.
- In this paper, the proposed GOA-BP algorithm is utilized to deeply study medical problems related to Leukemia and problems related to chemistry.

The acquired results of all the problems that have been investigated show high stability and a strong promising performance of the proposed GOA-BP model in solving highly sophisticated problems.

The structure of the paper is as follows: the 2nd section illustrates the GOA optimizer. The 3rd section introduces the Generalized delta rule and BP. The 4th section is devoted to explain and illustrate perceptron neural model while the 5th Section illustrates the newly proposed hybrid GOA-BP. the 6th section is devoted to experimental results and the conclusion.

2- Grasshopper optimization algorithm (GOA)

The GOA or Grasshopper optimization algorithm is an optimizer that uses the social life of grasshoppers in nature as an inspiration [50]. It has been confirmed by the researchers who discovered the GOA optimizer that it has the ability to outperform multiple strong optimizers like the GSA, FA, PSO, GA, and BA algorithms in the subject of dealing with realistic tasks and artificial tasks of optimization [50].

The most important behaviors of grasshoppers are foraging, team behaviors, and target pursuing in both the adulthood phase and nymph phase. While in the larval level, grasshoppers usually exhibit a small-length jumps with slow motion. On the contrary in adulthood, they exhibit long-range with quick movements to reach the targeted food sources from

various farming regions. A model was designed to simulate these facts as follows [50]:

$X\,i = Si + G\,i + A\,i$ ….(1)

Xi stands for the location of i[th] grasshopper, *Si* demonstrates the social communications, *Gi* denotes the strength of gravity on ith grasshopper, and *Ai* is used to denote the advection of wind. Note that we can write Eq. (1) as:

$X_i = r_1S_i + r_2G_i + r_3A_i$ …..(2)

Where *r*1, *r*2, *r*3 are used to denote random numbers inside the interval [0, 1]. We can achieve the components of Eq. [1] by:

$S_i = \sum_{j=1,j\neq i}^{N} s[d_{ij}]\hat{d}_{ij}, d_{ij} = |x_j - x_i|, \hat{d}_{ij} = \frac{[x_j - x_i]}{d_{ij}}$ ….(3)

$G_i = -g\hat{e}_g$…..(4)

$A_i = -u\hat{e}_w$ ….(5)

where d_{ij} denotes the distance between any two grasshoppers, $\hat{d}_{ij}$ is used to denote the unit vector, g stands for the constant of gravitation, $\hat{e}_g$ is used to represent the unity vector of gravity, while the variable u represents a constant drift , also $\hat{e}_w$ stands for the wind's unity vector. While The (s) function in the Eq. [3] calculates either the social attraction or repulsion forces and can be obtained by

$s(r) = fe^{-r/l} - e^{-r}$ ……(6)

In this equation f denotes the attraction's amplitude and l demonstrates the scale of length. Noticing that the comfort zone is a condition that the tendency of s(r) function is neither repulsive function nor attractive function. As a result, the f parameter and the r parameter can affect not only the comfort zone significantly, but as well as the repulsion and attraction regions.

The fact that the behavior of s-function can affect the social interaction of grasshoppers as it is demonstrated in Fig. 1.

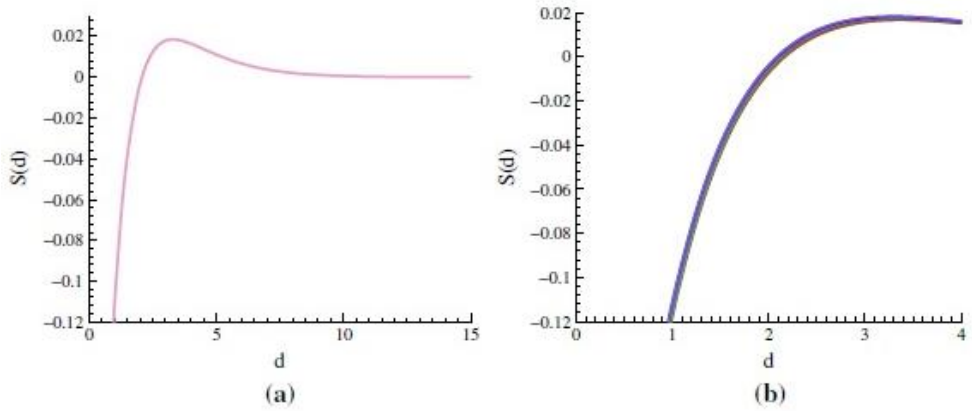


Fig.1: exhibition of *s* when *f* = 0.5 and *l* = 1.5 when *d* lies inside the interval [0, 15] and a window when *d* lies inside the interval [0, 4]. a *l* = 1.5 & *f* = 0.5. b *d* lies inside the interval [0, 4]

The grasshoppers in GOA have the ability to show different social activities depending on both of the parameters in s-function the l parameter and the f parameter as it appears in Fig. 2.

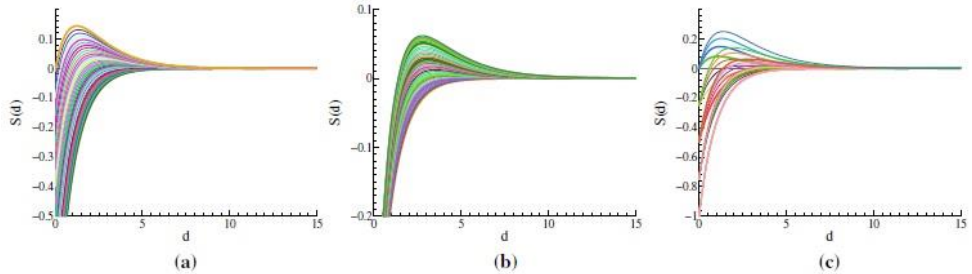


Fig.2: The various Dynamics of the s-function results by changing the parameters *l* and *f*. a *f* in [0, 1] & *l* = 1.5 b *l* in [1, 2] & *f* = 0.5. c *l* in [1, 2] , while *f* is in [0, 1]

According to [50], the value of f can be set to 0.5 and l can be set as 1.5. In addition to that, the distances between the insects can be mapped into the interval [1,4]. Fig 3 demonstrates the grasshoppers' interactions between themselves with respect to comfort zone.

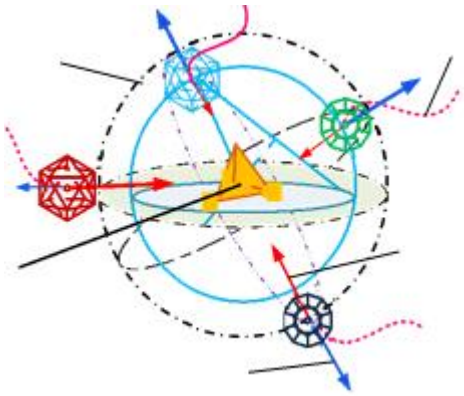


Fig.3: Corrective patterns among grasshoppers

According to the former Eqs. (3)–(5), we can reformulate the essential rule in Eq. [1] as:

$$X_i = \sum_{j=1, j\neq i}^{N} s[|x_j - x_i|] \frac{[x_j - x_i]}{d_{ij}} - g\hat{e}_g - u\hat{e}_w \quad \ldots.(7)$$

Where X_i is the position of the i[th] grasshopper, N is used to denote the swarm's size. The population can't converge to the specific target according to Eq. (7), because the grasshoppers will attain the comfort zone quickly [50]. As a result, a customized rule can be used as follows :

$$X_i^d = c[\sum_{j=1, j\neq i}^{N} \quad c\frac{UB_d - LB_d}{2} s[|x_j^d - x_i^d|] \frac{[x_j^d - x_i^d]}{d_{ij}}] + \hat{T}_d \quad \ldots.(8)$$

where d demonstrates dimension, LBd and UBd are the lower and upper boundaries, Td is used to denote the best target (solution) obtained so far, while the parameter c is a factor of decreasing. Also we can note that the internal parameter c is used to help GOA decrease the attraction / repulsion powers between grasshoppers, on the other side the external parameter c is used to decrease the tendency of search near the best position by more iteration.

The factor c can be obtained as follows:

$$c = c_{max} - l\frac{c_{max} - c_{min}}{L} \ldots.(9)$$

where c_{min} is the minimum value and c_{max} is the maximum values, l is used to denote the iteration of progress, while L represents the iterations' upper bound.

In Grasshoppers optimizer, the new grasshopper's position is obtained depending on its current position, also depending on the specific target's location, and by depending on the situation of the all the population. In Grasshopper optimization algorithm, the best-found solution has to be considered as the required target that must be discovered and enhanced by the other grasshoppers. Also to note that the decreasing c factor to helps the GOA to shrink the comfort zone gradually. Therefore, it is able to make a very smooth transition between the exploration phase to the exploitation phase inside the fitness landscape. Considering that the repulsion forces has the ability to help population for a wider exploration of the topography of fitness, on the other hand the forces of attraction have the ability to stimulate grasshoppers to exploit the promising regions to obtain a better quality solution.

The proceeding note can help the readers recognize the reason that grasshoppers' optimization algorithm has the ability to realize a highly promising solutions for many variant optimization cases:

– Grasshoppers have the ability to perform multiple abrupt, big jumps in the beginning stages of the search, and that enables them to search the unexplored areas globally.

– Grasshoppers have a tendency to search in the local areas in the final stages of the optimization, which enhances the exploitation capacities of them.

– The reduction factor of the comfort zone forces the grasshoppers to progressively make a fine balance between the exploration phase and the exploitation phase proneness, which helps the GOA to stay away from early unready convergence and find out a likely targeted global peak.

– The GOA has the ability to improve all the grasshoppers' average fitness, which helps Grasshoppers Optimization Algorithm to effectually enhance the initial solutions which are generated in a random way.

– The target location's fitness can be enhanced while in the progression of search, and this reveals that the approximation of the global optima can be enhanced after an additional iteration.

3-Generalized Delta Rule [Back-Propagation]

One of many supervised learning algorithms is the algorithm of generalized delta rule [52]. It is used to train a MLP network that demonstrates the connection between the target output and the actual output. While in the period of the training, the input pattern passes through the network with the biases and weights of the transfer or activation functions of the network connection. In the beginning, the biases and values are allocated by a small random numbers. In this rule, presenting repetitive pairs of input-output and then adjust the weights; the process of weight adjustment reduces the error of the network. The process of training the network goes as follows:

- The First step is to appoint the input pattern to input neurons; at the same time send to the hidden neurons in the hidden layers with weights and compute its activation; and after that it sends this to the output neurons with weights and computed its activation, this presents the response of the network's output of the input pattern.
- The Second step, a comparison takes place between the two categories: the desired output and the output responses also on the same time an error term is computed.

- The Third step is to use the information of the error to amend and update the network biases and weights. The weight's adjustment is computed through using four various parameters: a learning rate, the error term, the prevailing activity, and the derivative of activation function in the input layer.
- The Fourth step is that each one of the hidden units computes its own error. This usually gets done with an error of an output unit's, and this sends it as a sign in a backwards way to an invisible unit.
- The Fifth step takes place after computing the hidden unit's error, the updated of the weights of the input-to-hidden units happens by the use of the same equation that has been used in the output layer.

assuming there are more than only one layer of hidden units, this procedure can be repeated iteratively. This means that each error of the hidden unit takes place in one layer, as an error signal, this can get inseminated in a backward way to an adjacent layer once the modification of hidden unit weights takes place. After that the proceeding pattern for the training may be presented to the units of the input, and the learning process takes place all over again [53].

4-Perceptron neural networks

(FFNNs) of the Feed forward neural networks are one the most famous forms of ANN models that has the ability to approximate as well as perceive computational models by the help of their advance parallel layered structure [16]. The (FFNNs) consist of a group of neurons, those neurons works as an elements of processing and they are distributed over a series of the fully connected stacked layers. Multilayer perceptron is one of the special classes of Feed forward neural networks.

In Multilayer perceptron, the suitable way to coordinate the neurons is to be in a mode of one direction. Also the transition of the data in the MLP takes place between the three various types of parallel layers: the input layer, the output layer, and the hidden layers. Figure 4 represents a Multilayer network that has only one single hidden layer.

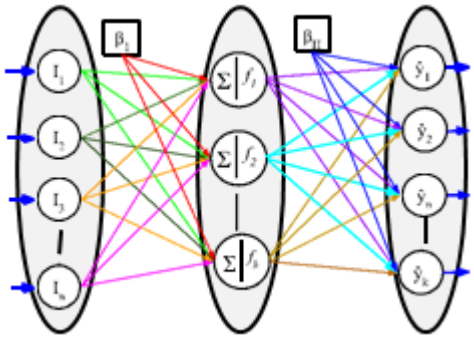


Fig. 4: MLP neural network

The connections between each layer must be described by weights that lay within the interval [−1, 1]. Every node in the Multilayer perceptron can perform as two various functions: the activation function and the summation function. The summation function in Eq. (9) is used to sum the result of the inputs, the weights, and the biases.

$$S_j = \sum_{i=1}^{n} \omega_{ij} I_i + \beta_j \quad \ldots..(10)$$

where I_i represents the input variable I, n stands for the number of inputs, βj is used to present a bias term, and the parameter ω_{ij} is used to represent the weight of the connection.

As a second step, an instigation for the activation function must take place by using the result that comes from Eq. (9). Many different forms of activation functions perchance utilized in the multilayer perceptron. The S-shaped curved sigmoid function is the most applied function in the previous works [3].

And Eq. (10). describes it.

$$f_j[x] = \frac{1}{1+e^{-S_j}} \quad \ldots.(11)$$

Therefore, the final output of the neuron j can be attained from the Eq. (12):

$$y_i = f_j[\sum_{i=1}^{n} \quad \omega_{ij} I_i + \beta_j] \quad \ldots.(12)$$

When the design of the structure of ANN gets started, after that the learning step takes place in order to fine-tune and on the same time update the weights of the network. The following is the weights get rationalized for the reason of estimating the outcomes and in order to minimize the results' error. The procedure of learning (training) of the neural networks is a difficult task that may challenge the capacity of the multilayer perceptron to tackle various types of problems.

5- The Hybrid Grasshoppers–Back-Propagation algorithm

The proposed hybrid GOA–BP is an optimization algorithm that associated the Grasshopper optimization with the Back-Propagation algorithm. The GOA algorithm is also a global algorithm just like the GA, which has an intense capability in finding optimistic global result, the GOA algorithm, however, has a negative character of a very slow search when being close to the global optimum. In contrast, the Back-Propagation algorithm, has an intense capability of finding local optimistic result, but it has a disadvantage of weak ability to find the global optimistic result. In this paper we combine the Grasshoppers optimizer with the Back-Propagation, this generates a new algorithm assigned as the hybrid GOA–BP algorithm. The essential principle for the proposed hybrid algorithm is that in the initial phases of the search for the optimum, the Grasshopper optimization algorithm runs to train speed to go faster. There are two altering conditions the first one is that when the fitness function gets fixed for multiple generations, or the change in the value is smaller than an already predefined number, if one of the satisfied conditions of the searching algorithm alters to gradient descending BP algorithm searching as reported by this heuristic knowledge. The process of searching of the GOA–BP algorithm's is started by

initializing a group of random grasshoppers. The first step is according to the Eq. (8) that all the grasshoppers gets updated, until it generates a new generation of grasshoppers, and after that these new grasshoppers are used to search in the solution area for the best global position. Finally, the Back-Propagation algorithm is utilized for the reason of searching near the global optimum. Following this pattern, new proposed hybrid algorithm can reach an optimum solution more rapidly.

The process of this hybrid GOA–BP algorithm can be illustrated as proceeding:

Step 1: set cMax , cMin, I,f and maxiteration

Step 2: generate initial population of n grasshoppers Xi=[1,2,3,……, n] randomly and each solution is evaluated by calculating its value using objective function.

Step 2: Evaluate the grasshoppers' positions' fitness for each one of them, and then determine and fix the best solution as the target position.

Step 3: updating c at each iteration according to eq. (9)

Step 4: Normalize the distance between the current grasshopper and other grasshoppers into the interval [1,4]

Step 5: Amend the location of the current solution according to eq. [8]

Step 6: reset the current solutions if it violates the boundaries of search space, if all the solutions of the populations visited then go to the [0] otherwise go to 4^{th} step.

Step 7: update overall best solution in the population X[t] and set t=t+1, go to step 8 if the termination criteria satisfied; otherwise go back to step 3.

Step 8: Start Using the Back-Propagation algorithm to search near the best position for some epochs, output the current result if the search result is better than target position; otherwise, output the target position.

Step 9: Output the global optimum target position.

6. The results of experiments and discussion

We compared the performance between the proposed GOA-BP algorithm with the BP algorithm in developing and updating the weights of the feedforward neural network by using the following two experiments. Suppose that the FNN consists of a couple of hidden layers only and assume the number of nodes are S1, S2 respectively, and also to note that the number of nodes in the input layer and the output layer depends on the number of patterns in the problem.

Experiment 6.1 Leukemia data:

In this problem, there are 72 patterns where each pattern consists of 3571 inputs and one output, we divided the dataset 72 units into two subsets, the first set to train our model consists 60 units and the second set consists of 12 units to test our model, so for the training phase the structure of our model or FFNN is 3571–S1–S2-1, where S1, S2 stands for the number of nodes in the 1st and 2nd hidden layers where S1=3,3,5 and S2=1,3,3. in the proceeding example, a comparison between the performance of S1 and S2 shall be made. For the Back-Propagation algorithm, the study used a learning rate as 0.01, and a maximum iteration as 15000 and the RMSE goal is 0.001. For the GOA–BP algorithm, the GOA is 10 generations, and 10 iterations and 5000 iterations for the BP and the RMSE goal is 0.001.

It can be noticed in the experiment that attaining the best position for the GOA if the grasshoppers' position didn't change for ten generations in the searching history, the study traces the global optimum in the GOA.

After reaching the best position for the GOA the GOA–BP algorithm starts using the Back-Propagation algorithm to search near the best position.

And the results are revealed as follows:

Table 1 : Results of Leukemia dataset training

NO.	S1	S2	Average RMSE (GOA_BP)	Average Time (GOA_BP)	Average RMSE (BP)	Average Time (BP)
1	3	1	0.1253	124.9969	0.46517619	82.1156249
2	3	3	0.2112	119.8969	0.46517619	78.9906249
3	5	3	0.0895	183.7562	0.46517619	95.0812499

After testing the training of our model with different numbers of the two hidden layers we noticed that the best one is S1=5, S2=3, so we will use the structure 3571-5-3-1 to train our model for five performances and we will take the best performance, the best performance is showed in the following table.

Table 2: RMSE, Time and Coefficient of correlation R of best performance in leukemia dataset training.

Algorithm	Average Training (RMSE)	Average Times	Coefficient of correlation R
GOA-BP	0.064674	7.738437 e+02	0.9935
BP	0.465176	1.03500e+02	0.0000

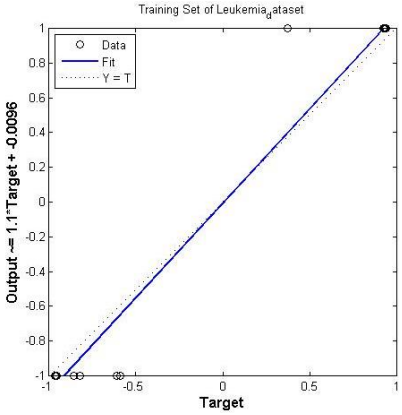


Fig. 5: training set for Leukemia dataset

Table 3 : Comparison between GOA-BP and BP on RMSE and R in Leukemia dataset

Algorithm	Testing (RMSE)	Coefficient of correlation R
GOA-BP	0.40059	0.67300
BP	0.532551	0.0000

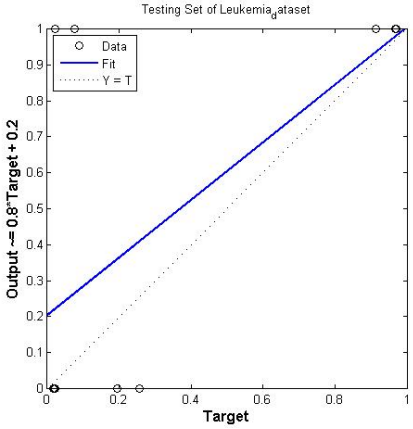


Fig. 6: Testing set of Leukemia dataset

Experiment 6.2 Chemical dataset:

In this problem, there are 498 pattern where each pattern consists of 8 inputs and one output, we divided the dataset into subsets, the first one consists of 400 patterns and the second set consists of 98 pattern to test our model, so for the training phase the structure of our FFNN is 8-S1-S2-1, where S1 and S2 denotes the number of the nodes in hidden layers.

For the Back-Propagation algorithm, we used a learning rate as 0.01, and we used a maximum iteration as 15000 and the RMSE goal is 0.001. For the GOA–BP algorithm, the GOA is 10 generations, and 10 iterations and 5000 iterations for the BP and the RMSE goal is 0.001.

It can be noticed in the experiment that attaining the best position for the GOA if the grasshoppers' position didn't change for ten generations in the searching history, the study traces the global optimum in the GOA.

After reaching the best position for the GOA the GOA–BP algorithm starts using the Back-Propagation algorithm to search near the best position.

And the results are revealed as follows:

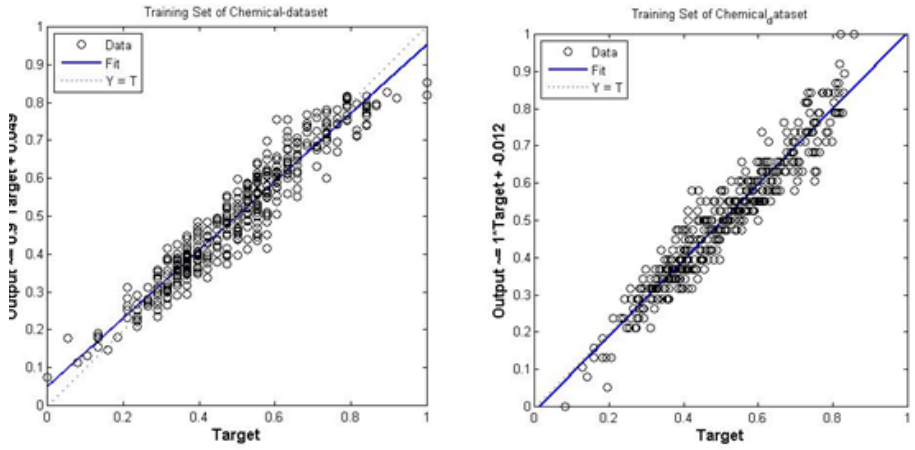
Table 1 : Results of chemical dataset training

NO.	S1	S2	RMSE (GOA_BP)	Average Time (GOA_BP)	RMSE (BP)	Average Time(BP)
1	3	2	0.05216391	8.21250	0.0554879	18.578125
2	5	4	0.05106652	11.612500	0.0537524	24.659375
3	7	8	0.05199512	22.449999	0.0519434	42.853124

The following table shows that the RMSE, times and Coefficient of correlation R of the best performance. The best performance after five performances

Table 2: RMSE, Time and Coefficient of correlation R of best performance in chemical dataset training.

Algorithm	Training (RMSE)	Average Times	Coefficient of correlation R
GOA-BP	0.0506	24.6250	0.9559
BP	0.050747	46.640625	0.9561

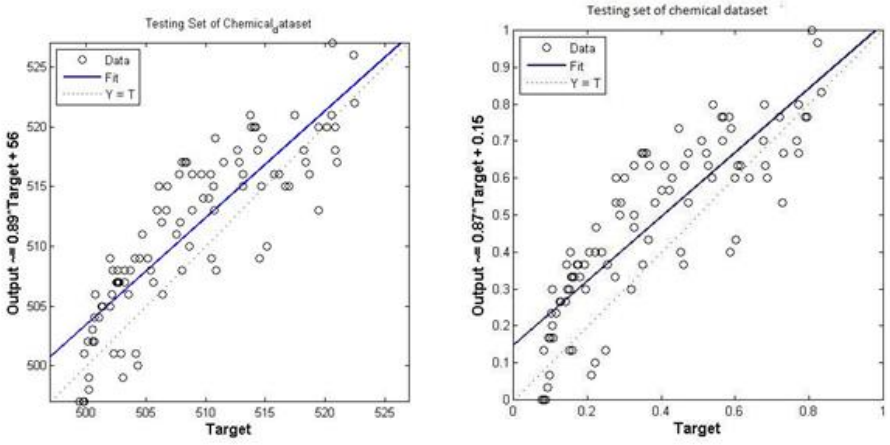


(a) GOA-BP (b) BP

Fig. 7: (a)Training set of chemical dataset for GOA-BP (b) Training set of chemical dataset for BP

Table 3 : Comparison between GOA-BP and BP on RMSE and R in chemical dataset

Algorithm	Testing (RMSE)	Coefficient of correlation R
GOA-BP	0.1481	0.8548
BP	0.16096	0.8466



(a) (b)

Fig. 8: (a)Testing set of chemical dataset for GOA-BP (b) Testing set of chemical dataset for BP

7. The Conclusion

This paper introduces a new hybrid GOA algorithm that combines the GOA algorithm with the BP algorithm, and this means to mix the strong search capability of GOA to find global optimum and the intense ability of the back-propagation algorithm in local search. The hybrid GOA-BP algorithm starts by distributing grasshoppers' positions randomly and after the distribution the evaluation of the fitness of every position takes place to determine the best position, after meeting fitness condition when approaching the global optima, the BP starts by initiating search around the optima which provides a more accurate results than the BP or GOA alone due to the weakness of GOA in local search and the weakness of BP in global search.

From the conducted experiments, we can conclude that the proposed hybrid GOA–BP algorithm uses less time than the BP in order to reach the same target, in other words, the GOA–BP algorithm uses less time to attain a higher training accuracy than the BP algorithm.

References

[1] McC ulloch WS, Pitts W(1943). A logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 5[4]:115–133

[2] Heidari AA, Abbaspour RA (2018). Enhanced chaotic grey wolf opti-mizer for real-world optimization problems: a comparative study. In: Handbook of research on emergent applications of optimization algorithms. IGI Global, pp 693–727

[3] Aljarah I, Faris H, Mirjalili S (2016). Optimizing connection weights in neural networks using the whale optimization algorithm. Soft Comput 22:1–15

[4] Faris H, Aljarah I, Mirjalili S (2016b). Training feedforward neural networks using multi-verse optimizer for binary classification problems. Appl Intell 45(2):322–332

[5] Trujillo MCR, Alarcón TE, Dalmau OS, Ojeda AZ (2017). Segmentation of carbon nanotube images through an artificial neural network. Soft Comput 21(3):611–625

[6] Krogh A (2008). What are artificial neural networks? Nat Biotechnol 26(2):195–197

[7] Esteva A, Kuprel B, Novoa RA, Ko J, Swetter SM, Blau HM, Thrun S (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature 542(7639):115–118

[8] Almonacid F, Fernandez EF, Mellit A, Kalogirou S (2017). Review of techniques based on artificial neural networks for the electrical characterization of concentrator photovoltaic technology. Renew Sustain Energy Rev 75:938–953

[9] Ata R (2015). Artificial neural networks applications in wind energy systems: a review. Renew Sustain Energy Rev 49:534–562

10- Ding S, Li H, Su C, Yu J, Jin F (2013). Evolutionary artificial neural networks: a review. Artif Intell Rev 39(3):251–260

[11] Lee S, Choeh JY (2014). Predicting the helpfulness of online reviews using multilayer perceptron neural networks. Expert Syst Appl 41(6):3041–3046

[12] Chaudhuri BB, Bhattacharya U (2000). Efficient training and improved performance of multilayer perceptron in pattern classification. Neurocomputing 34(1):11–27

[13] Esteva A, Kuprel B, Novoa RA, Ko J, Swetter SM, Blau HM, Thrun S (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature 542(7639):115–118

[14] Faris H, Aljarah I, Al-Madi N, Mirjalili S (2016a). Optimizing the learning process of feedforward neural networks using lightning search algorithm. Int J Artif Intell Tools 25(06):1650033

[15] Yi-Chung H (2014). Nonadditive similarity-based single-layer perceptron for multi-criteria collaborative filtering. Neurocomputing 129:306–314

[16] Ojha VK, Abraham A, Snášel V (2017). Metaheuristic design of feed-forward neural networks: a review of two decades of research. Eng Appl Artif Intell 60:97–116

[17] Chen J-F, DoQH, Hsieh H-N (2015). Training artificial neural networks by a hybrid PSO-CS algorithm. Algorithms 8(2):292–308

[18] Cybenko G (1989). Approximation by super positions of a sigmoidal function. Math Control Signals Syst (MCSS) 2(4):303–314

[19] Hamidzadeh J, Namaei N (2018). Belief-based chaotic algorithm for support vector data description. Soft Comput 1–26

[20] Sadeghi R, Hamidzadeh J (2018). Automatic support vector data description. Soft Comput 22[1]:147–158

[21] Hamidzadeh J, Sadeghi R, Namaei N (2017). Weighted support vector data description based on chaotic bat algorithm Appl. Soft comput. 60:540-551

[22] Hamidzadeh J, Monsefi R, Yazdi HS (2012) DDC: distance based decision classifier. Neural Comput Appl 21(7):1697–1707

[23] Hamidzadeh J, Monsefi R, Yazdi HS (2015). IRAHC: instance reduction algorithm using hyperrectangle clustering. Pattern Recognit 48[5]:1878–1889

[24] Hamidzadeh J, Monsefi R, Yazdi HS (2016). Large symmetric margin instance selection algorithm. Int J Mach Learn Cybern 7[1]:25–45

[25] Hamidzadeh J, Zabihimayvan M, Sadeghi R (2018). Detection of web site visitors based on fuzzy rough sets. Soft Comput 22(7):2175– 2188

[26] Moghaddam VH, Hamidzadeh J (2016). New hermite orthogonal poly-nomial kernel and combined kernels in support vector machine classifier. Pattern Recognit 60:921–935

[27] Hamidzadeh J, Moradi M (2018). Improved one-class classification using filled function. Appl Intell 1–17

[28] Hamidzadeh J, Monsefi R, Yazdi HS (2014). LMIRA: large margin instance reduction algorithm. Neurocomputing 145:477–487

[29] Wang L, Zeng Y, Chen T (2015). Back propagation neural network with adaptive differential evolution algorithm for time series forecast-ing. Expert Syst Appl 42(2):855–863

[30] Zhang J-R, Zhang J, Lok T-M, Lyu MR (2007). A hybrid particle swarm optimization–back-propagation algorithm for feed forward neural network training. Appl Math Comput 185(2):1026–1037

[31] Heidari AA, Pahlavani P (2017). An efficient modified grey wolf optimizer with lévy flight for optimization tasks. Appl Soft Comput 60:115–134

[32] Faris H, Mafarja MM, Heidari AA, Aljarah I, Al-Zoubi AM, Mirjalili S, Fujita H (2018a). An efficient binary salp swarm algorithm with crossover scheme for feature selection problems. Knowl Based Syst 154:43–67

[33] Mirjalili S (2015). How effective is the grey wolf optimizer in training multi-layer perceptrons. Appl Intell 43[1]:150–161

[34] Faris H, Aljarah I, Mirjalili S (2016b). Training feedforward neu-ral networks using multi-verse optimizer for binary classification problems. Appl Intell 45(2):322–332
[35] Mallipeddi R, Suganthan PN, Pan QK, Tasgetiren MF (2011). Differential evolution algorithm with ensemble of parameters and mutation strategies. Appl Soft Comput 11(2):1679–1696
[36] Hansen N, Müller SD, Koumoutsakos P (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol Comput 11(1):1–18
[37] Ilonen J, Kamarainen J-K, Lampinen J (2003). Differential evolution training algorithm for feed-forward neural networks. Neural Pro-cess Lett 17[1]:93–105
[38] Slowik A, Bialko M (2008). Training of artificial neural networks using differential evolution algorithm. In: 2008 Conference on human system interactions. IEEE, pp 60–65
[39] Wienholt W (1993). Minimizing the system error in feedforward neural networks with evolution strategy.In:ICANN93.Springer,pp490– 493
[40] Wdaa ASI [2008] Differential evolution for neural networks learning enhancement. PhD thesis, Universiti Teknologi Malaysia
[41] Jordehi AR, Jasni J (2013). Parameter selection in particle swarm optimization: a survey. J Exp Theor Artif Intell 25(4):527–542
[42] Karaboga D, Basturk B (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony [ABC]algorithm. J Glob Optim 39(3):459–471
[43] Dorigo M, Birattari M, Stutzle T (2006). Ant colony optimization. IEEE Comput Intell Mag 1(4):28–39
[44] Jianbo Y, Wang S, XiL (2008). Evolving artificial neural networks using an improved PSO and DPSO. Neurocomputing 71(4):1054–1060
[45] Braik M, Sheta A, Arieqat A [2008] A comparison between GAs and PSO in training ANN to model the TE chemical process reactor. In: AISB 2008 convention communication, interaction and social intelligence, vol 1, p 24.
[46] Blum C, Socha K. (2005). Training feed-forward neural networks with ant colony optimization: an application to pattern classification. In: Fifth international conference on hybrid intelligent systems, 2005. HIS'05. IEEE, p 6
[47] Socha K, Blum C (2007). An antcolony optimization algorithm for con-tinuous optimization: application to feed-forward neural network training. Neural Comput Appl 16(3):235–247
[48] Wolpert DH, Macready WG [1997] No free lunch theorems for opti-mization. IEEE Trans Evol Comput 1(1):67–82
[49] Mafarja M, Aljarah I, Heidari AA, Hammouri AI, Faris H, Al-Zoubi AM, Mirjalili S (2018). Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems. Knowl Based Syst 145:25–45
[50] Saremi S, Mirjalili S, Lewis A (2017). Grasshopper optimisation algorithm: theory and application. Adv Eng Softw 105:30–47
[51] Aljarah I, Al-Zoubi AM, Faris H, Hassonah MA, Mirjalili S, Saadeh H (2018a) Simultaneous feature selection and support vector machine optimization using the grasshopper optimization algorithm. Cognit Comput 10:1–18
[52] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "Learning representations by back- propagating Errors Nature", 323, 533-536, 1986.
[53] http://www.bcp.psych.ualberta.ca/~mike/Pearl_Street /Dictionary/contents/G/generalizeddr.html.

خوارزمية هجينة جديدة لتدريب الشبكات العصبية ذات التغذية الأمامية بالإعتماد على خوارزمية أمثلية الجراد والإنحدار العكسي

سامر عبد القادر صالح ، نزار خلف حسين

قسم الرياضيات ، كلية علوم الحاسوب والرياضيات ، جامعة تكريت ، تكريت ، العراق

الملخص

خوارزمية أمثلية الجراد أظهرت تقارب سريع في الأطوار الأولى من البحث عن النهاية العظمى العالمية بينما تتباطئ عملية البحث بصورة كبيرة حول النهايات المحلية بينما خوارزمية الإنحدار العكسي تحقق تقارب أسرع حول النهايات المحلية ودقة النتائج أعلى في نفس الوقت. ونتيجة لذلك اقترح هذا البحث خوارزمية هجينة تجمع بين خوارزمية أمثلية الجراد مع الإنحدار العكسي لتوفير تدريب لأوزان الشبكات العصبية ذات التغذية الأمامية. الخوارزمية المقترحة تستطيع الجمع بين قدرة خوارزمية الجراد على البحث في النهايات العالمية مع القدرة القوية لخوارزمية الإنحدار العكسي في البحث حول النهايات المحلية. نتائج التجارب أظهرت ان الخوارزمية المقترحة أفضل وأسرع في التقارب وأدق من خوارزمية أمثلية الجراد والإنحدار العكسي.