Mansoura Engineering Journal

Volume 41 | Issue 4 Article 2

12-1-2016

A New Multi-layer Perceptron Trainer based on Dragonfly Optimization Algorithm.

Mohy Eldin Abo-Elsoud
Prof.,Communication department., Faculty of Eng., Mansoura University

Mohamed Morsy
Doctor.,Communication department., Faculty of Eng., Mansoura University.

Shaima Elnazer
, lecture assistant at Nile academy, Mansoura University, shima_elnazer@yahoo.com

Follow this and additional works at: https://mej.researchcommons.org/home

Recommended Citation

Abo-Elsoud, Mohy Eldin; Morsy, Mohamed; and Elnazer, Shaima (2016) "A New Multi-layer Perceptron Trainer based on Dragonfly Optimization Algorithm.," *Mansoura Engineering Journal*: Vol. 41 : Iss. 4 , Article 2.
Available at: https://doi.org/10.21608/bfemu.2020.103962

This Original Study is brought to you for free and open access by Mansoura Engineering Journal. It has been accepted for inclusion in Mansoura Engineering Journal by an authorized editor of Mansoura Engineering Journal. For more information, please contact mej@mans.edu.eg.

Mansoura University
Faculty of Engineering
Mansoura Engineering Journal

A New Multi-layer Perceptron Trainer based on Dragonfly Optimization Algorithm

تدريب جديد للنوع المتعدد من الشبكات العصبيه الاصطناعيه يعتمد علي خوازمية اليعسوب

Mohy Eldin A. Abo-Elsoud, *Mohamed Morsy* and Shaima Elnazer

KEYWORDS:
Dragonfly Optimizer, Neural Network and Multi-Layer Perceptron

الملخص العربي:- **في هذا البحث تم استخدام خوارزميه اليعسوب لتدريب النوع المتعدد الطبقات من الشبكات العصبيه الاصطناعيه. و استخدمت خوارزميه اليعسوب لايجاد الاوزان والانحيازات للنوع المتعدد الطبقات من الشبكات العصبيه الاصطناعيه لتحقق الحد الادني من الخطأ واعلي معدل من التصنيف. ولقياس قوه الطريقه المقترحه تم استخدام اربع مجموعات من البيانات بالاضافه لذلك تمت مقارنه اداء الطريقه المقترحه مع اربع خوارزميات معروفه للتحسين،وهي خوارزميات الجنينيه (GA)،سرب الجسيمات (PSO)، مستعمرة النمل (ACO)، الذئب الرمادي(GWO) التي تستخدم لايجاد الاوزان و الانحيازات للنوع المتعدد الطبقات من الشبكات العصبيه الاصطناعيه .و أظهرت النتائج ان الخوارزميه اليعسوب(DO) مع النوع المتعدد الطبقات من الشبكات العصبيه الاصطناعيه كانت تنافسيه جدا لانها تحل مشكله الاوبتما المحليه وحقق معدل دقه عالي.**

***Abstract*— In this paper, Dragonfly Optimizer (DO) was used to train Multi-Layer Perceptron (MLP). DO was used to find the weights and biases of the MLP to achieve a minimum error and a high classification accuracy. Four standard classification datasets were used to benchmark the performance of the proposed method. In addition, the performance of the proposed method were compared with three well-known optimization algorithms, namely, Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Grey Wolf Optimizer (GWO) which were used to train MLP also. The experimental results showed that the DO algorithm with the MLP was very competitive as it solved the local optima problem and achieved high accuracy.**

Received: 11 May, 2016 - revised: 15 October, 2016 - accepted: 13 December, 2016
Mohy Eldin A. Abo-Elsoud, Prof.,Communication department., Faculty of Eng., Mansoura University.
Mohamed Morsy, Doctor.,Communication department., Faculty of Eng., Mansoura University.
Shaima Elnazer, lecture assistant at Nile academy, Mansoura University.

I. INTRODUCTION

ARTIFICIAL Neural Networks (ANNs)-based techniques are widely used in the domain of the computational Intelligence.

They are bio-inspired by the neurons of the human brain to generally solve classification problems. The ANNs were introduced in 1943 [1] and since then, there exist various kinds of ANNs: Radial basis function (RBF) neural network [2], Kohonen self-organizing (KSO) neural network [3], Spiking neural networks [4], recurrent neural network [5], and Feed forward Neural Network (FNN) [6]. Here are some examples of how the information is processed in each NNs type.

The information in FNN is passed in one direction throughout the networks. On the other side, the information in the recurrent neural network is shared among the neurons in two directions, whereas in the spiking neural networks, neurons are activated by spikes [4].

Regardless the different types of NNs, they have a common point, they are using one learning approach. Similar to biological neurons, the Artificial Neural Network (ANN) have been equipped with mechanisms to adjust themselves to a set of provided inputs. There exist two common kinds of learning techniques: unsupervised [7], [8] and supervised [9].

In the unsupervised techniques, the NNs adjusts itself to the inputs without the need for any extra external feedbacks, whereas in the supervised ones, the NNs receives feedbacks from an external source .In general, the technique providing the learning mechanism to the neural networks is called a trainer. Such trainer is responsible for adapting the NNs technique to give the maximal accuracy for new sets of given inputs. Hence it can be considered as the most significant element of any NNs techniques.

There exist two kinds of learning/training techniques in the literature: stochastic and deterministic. In the deterministic techniques, e.g., Back Propagation [10] and gradient-based [11], the training stage results in the same accuracy if the training samples stay compatible. The trainers, in these techniques, are mostly mathematical optimization techniques which are aiming to achieve a high performance (i.e. minimum error).On the other hand, the stochastic trainers, e.g., Particle Swarm Optimization (PSO) [12] and Grey Wolf Optimization [13], employ stochastic optimization methods to increase the performance of neural networks.

Both of the stochastic and deterministic trainers have advantages and disadvantages. The deterministic trainers achieve the convergence quickly but the quality of the obtained solution is mainly based on the beginning solution [14]. In addition, these trainers are highly subject to the local optima trap [15].On the other hand, the stochastic trainers can highly avoid the local optima trap, but they are slower than deterministic trainers [16].As the avoidance of the local optima problem is crucial to the NNs applications, there is high focus in the literature about the stochastic training methods [17].

As explained in [13], the bio-inspired techniques, e.g., Genetic Algorithm (GA), (PSO), and Ant Colony Optimization (ACO), etc, have shown a high performance for approximating the global optimum as training algorithms. This motivates us to study and investigate the possibility of using the recently proposed Dragonfly Optimizer (DO) [18] as an effective trainer for Feed forward Neural Networks (FNNs). The DO was chosen as it shows a high exploration and exploitation which could lead to a significant improvement over the other related trainers.

The rest of the paper is organized as follows: Section 2 presents the fundamentals of an MLP and DO algorithm. The proposed DO-based trainer is described in Section 3.Experimental results with discussions are presented in Section 4. Conclusions and future work are introduced in Section 5.

II. PRELIMINARIES

2.1Multi-Layer Perceptron (MLP)

The Multi-Layer Perceptron (MLP) is a special type of the Feed-forward Neural Networks (FNNs) in which in the information is passed in one direction throughout the NNs and its neurons are arranged in various parallel layers [2] where the first one is known as the input layer and the last one is called the output layer. The layers, between these two layers, are named hidden layers. When the FNNs has only one hidden layer, it is known as a Multi-Layer Perceptron (MLP).Figure 1, illustrate accuracy an example of the MLP.

According to the inputs, weights, and biases, the outputs of MLP are calculated as in the following steps [10]:

1) The weighted totals of inputs are initially computed as follows[10],

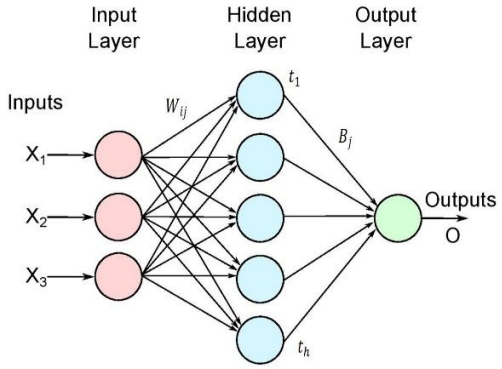

Fig.1: An Example of MLP with three inputs

$$t_j = \sum_{i=1}^{n}(W_{ij}, Y_i) - B_j \quad j = 1,2,\dots\dots h \tag{1}$$

Where n is the number of the input nodes, W_{ij}demonstrate accuracy the association weight from the i^{th}nodein the input layer to the j^{th} node in the hidden layer, $Y_i$indicates the i^{th}input, $h$number of hidden nodes,and B_j is the bias (threshold) of the j^{th} hidden node.

2) The output of each hidden node is computed as [19]:

$$T_j = sigmoid(t_j) = 1/(1+\exp(-t_j))\, j = 1,2,\dots\dots h \tag{2}$$

3) The final outputs are characterized depend on the computed outputs of the hidden nodes[10]:

$$f_k = \sum_{j=1}^{k}(W_{jk}, T_j) - B'_k \quad k = 1,2,\dots\dots m \tag{3}$$

$$F_k = sigmoid(f_k) = 1/(1+\exp(-f_k))\, k = 1,2,\dots\dots m \tag{4}$$

Where$W_{ij}$is the connection weight from the j^{th}hidden node to the k^{th}output node, m is the number of outputs, and $B'_k$is the threshold of the k^{th}output node. From these three steps,it is clear that the output of MLPs is determined through theweights and biases. Thus, in this paper, the DO algorithm was utilized as a trainer for MLP's parameters.

2.2. Dragonfly Optimizer (DO):

The Dragonfly optimizer (DO) is one of the most recent meta-heuristic optimization techniques [11]. The main inspiration of the (DO) algorithm originates from static and dynamic swarming behaviors. These two behaviors of swarming are very comparable to the essential two phases of optimization utilizing meta-heuristics: diversification and intensification. As indicated by Reynolds, the conduct of swarms takes after three primitive standards:

- The separation, which alludes to the static impact shirking of the individuals from other individuals in neighborhood.
- The alignment, which demonstrate accuracy speed coordinating of individuals to that of other individuals in the neighborhood.
- The Cohesion, which alludes to the propensity of individuals towards the focal point of the mass of the neighborhood.

The primary target of any swarm is survival, so all of the people ought to be pulled in towards nourishment sources also, occupied outward adversaries. Considering these two practices, there are five principle components in position upgrading of people in swarms as appeared in Fig.2.
Each of these practices is mathematically modeled as takes after:
The separation is computed as takes after:

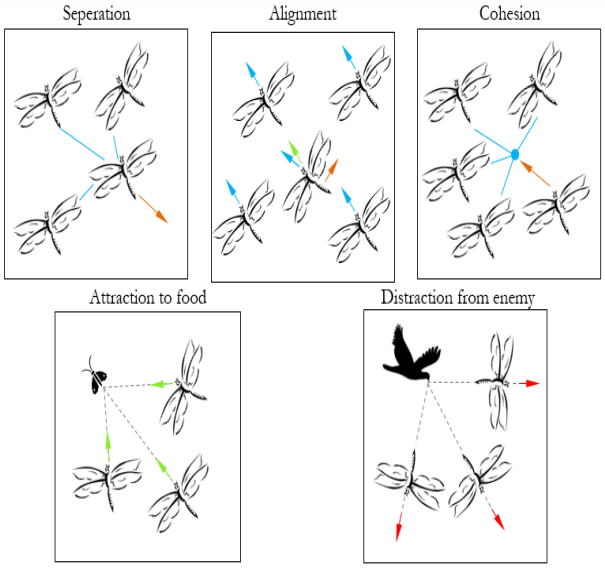

Fig. 2: Primitive corrective examples between individuals in a swarm

$$S_i = -\sum_{i=1}^{n} Y - Y_i \quad (5)$$

Where Y is the position of the present individual, Y_i is the position of i^{th} individual neighborhood.
The Alignment is computed as takes after [20]:

$$A_i = \frac{\sum_{i=1}^{n} V_i}{n} \quad (6)$$

WhereV_i present the i^{th}neighborhood individual velocity and *N*is the number of neighbouring individuals.
The Cohesion is computed as takes after:

$$C_i = \frac{\sum_{i=1}^{n} Y_i}{n} - Y \quad (7)$$

Attraction towards a sustenance source is computed as takes after:

$$F_i = Y^+ - Y \quad (8)$$

Where $Y^+$is the food source position and Y is the current individual position.

Distraction from an enemy is computed as takes after:

$$E_i = Y^- + Y \quad (9)$$

Where Y^- is the enemy position and Y is the current individual position.

For the process of position update of artificial dragonflies in the search space mimic their developments, two vectors are considered: **Position** (Y) and **Step** (ΔY). The vector of step is present the movement direction of dragonflies and computed as take after:

$$\Delta Y_{t+1} = (sS_i + aA_i + cC_i + eE_i \ fF_i) + w\Delta Y_t \quad (10)$$

Where s, a, c are the weight of separation, alignment and cohesion respectively and f, e are the food and enemy factor and t is the number of iterations.

Algorithm: Dragonfly Optimizer [18]

1- Initialize the population of dragonflies populationY_i
2- Initialize the step vector ΔY_i
3- **While** the end criterion is not satisfied.
4- Compute the fitness values of all dragonflies
5- Update the enemy and food source
6- Update e, s, a, c, f, and w
7- Compute E, A, C, F, and S

If a dragonfly has at least one neighboring dragonfly
Update the vector of velocity and the vector of position
Else
Update the vector of position
End if
Check and correct the new positions based on the boundaries of variables
End While

The vector of position is computed as take after:

$$Y_{t+1} = Y_t + \Delta Y_{t+1} \quad (11)$$

To enhance the randomness and diversification of the artificial dragonflies, they are required to fly around the search space utilizing an arbitrary walk (Le´vy flight) when there is no neighboring solutions. To update the position of dragonflies in this case using this equation [21]:

$$Y_{t+1} = Y_t + Levy(d) \times Y_t \quad (12)$$

Where *t* is the number of current iteration and *d* is the position vector dimension.
The levy flight is computed as the following equation:

$$levy(Y) = 0.01 \times \frac{r_1 \times \alpha}{|r_2|^{\beta}} \quad (13)$$

Where r_1, r_2two random numbers are belong to the interval [0, 1] and β is constant value equal 1.5.

III. PROPOSED MODEL

As explained above, the variables, weights and biases, affects the output of the MLP and the aim of any optimizer is to search for values for these variable such that they give the highest classification accuracy and the lowest error accuracy.

To achieve this, in the proposed model (see Fig.3), the DO algorithm was used to optimize the weights and biases which represent the input to the DO algorithm as a vector as follows:

$$\vec{V}=\{\vec{W},\vec{\theta}\}=\{W_{1,1},W_{1,2},\dots,W_{n,n},h,\theta_1,\theta_2,\dots,\theta_h\} \quad (14)$$

Where n represents the number of inputs, $W_{i,j}$is the weight of the connectionbetween the i^{th}node to the j^{th}node, and $\theta_k$represents thebias of the k^{th}hidden node.

In other words, the objective function of the proposed algorithm is to achieve the highest classification accuracy at both training and testing samples. To evaluate the MLP output, the Mean Square Error (MSE) was used where the MSE calculates the difference between the desired output and the actual output of the MLP. In other words, MSE is used to measure how the value of desired output is deviated from the value of the actual output as follows,

$$MSE=\sum_{i=1}^{m}(o_i^k-d_i^k)^2 \quad (15)$$

Where m represents the number of outputs, $d_i^k$and $o_i^k$are the desiredand actual outputs, respectively, of the i^{th}input unit whenthe k^{th}training sample is used.

Thus, the average of MSE is calculated for all training samples as follows:

$$\overline{MSE}=\sum_{k=1}^{N}\frac{\sum_{i=1}^{m}(o_i^k-d_i^k)^2}{N} \quad (16)$$

Where N is the total number of training samples. The objective function of the DO algorithm aims to minimize the average MSE as follows,

$$Min: F(\vec{V})=\overline{MSE} \quad (17)$$

Thus, the weights and biases of the MLP move to minimize average MSE in each iteration. Hence, DO iteratively converge to a global solution that is better than random initial solutions.

Classification accuracy of models has been calculated in terms of classified pattern if CM is confusion matrix of order mxn, the accuracy of classification is computed as follows:

$$\text{Classification accuracy}=\frac{\sum_{I=1}^{N}\sum_{J=1}^{M}CM_{IJ}}{\sum_{I=1}^{N}\sum_{J=1}^{M}CM_{IJ}}\ X100 \quad (18)$$

Test error has been calculated as follow:

$$\text{Test Error}=\frac{\sum_{k=0}^{y}(t_k s_k)^2}{y} \quad (19)$$

Where *y* represents number of outputs, *t* represent set of model output values and *s* represent set of calculated output values.

IV. Experimental Results and Discussion

The aim of all experiments is to search for the weights and biases to train the MLP to reduce the MSE and test error and increase the accuracy of Classification.

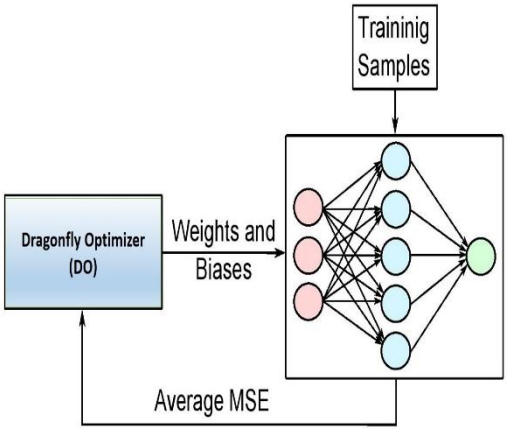

Fig. 3: DO algorithm searches for the weights and biases to train the MLP with the training samples and calculate the average MSE.

4.1. Data Sets

The aim of all experiments is to optimize the weights and biases of MLPs to reduce the MSE and test error and increase the classification accuracy. In this section, four different standard datasets, namely, XOR, heart, iris, and breast cancer dataset are used to evaluate the proposed Dragonfly Optimizer (DO) trainer. The datasets are obtained from University of California at Irvine (UCI) Machine Learning Repository and the description of the datasets are summarized in Table (1)[19].

The XOR dataset consists of three attributes, eight training samples, eight testing samples, two classes, and one output while the IRIS dataset, which is one of the most common standard datasets, composes of four attributes, 150 training samples, 150 testing samples, three classes, and three outputs .Moreover, the Heart dataset includes 22 attributes, 80 training samples, 187 testing samples, two classes, and one output. The last but not the least, the Breast Cancer dataset consists of nine attributes, 599 training samples, 100 testing samples, two classes, and one output.

These classification datasets were deliberately chosen with different training/test samples and levels of difficulty to test the performance of the proposed DO-based MLP trainer effectively.

A similar problem representation and objective function is utilized to train MLPs with the algorithms in Table 1. The initial parameter for every algorithm in Table 2.These values subject to trial, practice and pervious work. The datasets are then solved 10 times using each algorithm to generate the results. The statistical results that are presented are average of the obtained MSEs in the last iteration by the algorithms. Obviously, lower average and standard deviation of MSE in the last iteration indicates the better performance. Please note that the best classification accuracy or test errors obtained by each of the algorithms during 10 runs are reported as another metrics of comparison.

Normalization is an essential step for MLP when solving datasets with attributes in different ranges. The normalization used in this work is called min-max normalization, which is formulated as follows:

$$X' = \frac{(x-a)\times(d-c)}{(b-a)} + c \quad (20)$$

This formula maps x in the interval of [a, b] to[c, d].Another key factor in the experimental setup is the structure of MLPs. This work does not concentrate on finding the optimal number of hidden nodes and consider them equal to (2×N+1) where N is the number of features (inputs) of the datasets. The structure of each MLP that employed for each data set is presented in Table 1.

As the size of the neural network becomes larger, obviously, the more weights and biases would be involved in the system. Consequently, the training process also becomes more challenging

4.2. Experimental Setup:

The description of the used datasets is shown in Table (I). It is worth noting that the XOR dataset is the smallest dataset which means that it may less training iteration while the breast cancer dataset is the largest dataset which could be more complicated for the training phase as it has nine attributes, 599 training samples, 100 testing samples, and two classes. The training and testing samples are chosen from each dataset to evaluate the performance of the proposed model.

In all experiments, the weights and biases are randomly initialized in ranges [-10, 10] for all datasets. In addition, the population size of all algorithms was 50 for XOR dataset and 200 for the rest of datasets and the maximum number of iterations was 250 iterations. Also, the initial parameters of the algorithms (GA, PSO, and ACO) which were used in all experiments are summarized in Table (2). Furthermore, the structure of the MLPs for each dataset is presented in Table (1).

In this research, the hidden nodes of MLPs are assumed to be equal to (2 N+1), where N represents the number of features or attributes (inputs) of the datasets.

Each algorithm was run 10 times on each dataset and the average (AVG) and standard deviation (STD) of the best Mean Square Errors (MSEs) in the last iteration in each algorithm were calculated. Moreover, the best classification accuracy or test errors of each algorithm were calculated.

4.3. Experimental Scenarios

To evaluate the proposed MLP trainer, four experimental scenarios were performed. In each scenario, four optimization algorithms (i.e. DO (the proposed one), PSO, GA, ACO, and GWO) were applied on the same dataset to evaluate DO and compare it with the other three algorithms.

In the first scenario, the XOR dataset, described in Tables (1) and with MLP structure 3-7-1, was used (see Fig.4) while in the second scenario, the Iris dataset, described in Table (1) with MLP structure 4-9-3, was used to evaluate the DO(see Fig.5) against GWO, PSO, GA, and ACO). In the third and the fourth scenarios, the Heart dataset and with MLP structure 22-45-1(see Fig.6) and the Breast cancer dataset (see Fig.7) and with MLP structure 9-19-1 (see Table (1)) were used, respectively. The results of these four scenarios are summarized in Table (3).

4.4 Discussion

From Table (3), the following remarks can be noticed.

Firstly, using the XOR dataset, DO as the trainers for MLP achieved the best average for MSE (0.00016), while ACO achieved the worst average MSE (0.183328). This means that DO can solve the local optimum problem better than all other algorithms listed in Table (3).

Secondly, the classification accuracy of the DO and GA algorithms reached to 100%, while the accuracy of the PSO algorithm reached to 37.5%.Two findings show that the DO gave the best results when using the XOR dataset. Secondly, the DO algorithm achieved results better than all other algorithms (i.e. minimum MSE = 0.0260 and maximum classification accuracy =90.0%) when the IRIS dataset was used whereas the ACO algorithm gave the worst results (i.e. minimum MSE =0.405979 and maximum classification accuracy =33.5%).

Thirdly, when the heart dataset was used, the MSE of the GA algorithm was the lowest (0.0956) (i.e. better than the MSE of the DO algorithm (0.142)). However, the classification accuracy of the DO algorithm was the best by classification accuracy at 2.5 %while the classification accuracy of GA was at 58.75%. Surprisingly, the ACO algorithm achieved 0% classification accuracy and high MSE.

Fourthly, when using the breast cancer dataset, DO algorithm achieved MSE at $(6.3607e-011)$ which was much lower than the other algorithms. At the same time, the DO achieved the highest classification accuracy at 99.33% while GA algorithm was the second in terms of the classification accuracy and MSE. On the other hand, the classification accuracy is decreased dramatically when GWO, PSO and ACO algorithms are used.

4.5 Comparative Analysis

Statistically speaking, the DO-MLP algorithm provides superior local optima avoidance in almost of the datasets and the best classification accuracy in all of the datasets. The reason for improved MSE is the high local optima avoidance of this algorithm. According to the mathematical formulation of the DO algorithm, half of the iterations are devoted to exploration of the search space. This promotes exploration of the search space that leads to finding diverse MLP structures

during optimization. This mechanism is very helpful for resolving local optima stagnation even when the DO algorithm is in the exploitation phase. The results of this work show that although evolutionary algorithms have high exploration, the problem of training an MLP needs high local optima avoidance during the whole optimization process. This is because the search space is changed for every dataset in training MLPs. The results prove that the DO is very effective in this regard.

Another finding in the results is the weak performance of PSO-MLP, GWO-MLP and ACO-MLP. These two algorithms belong to the class of swarm-based algorithms. In contrary to evolutionary algorithms, there is no mechanism for significant abrupt movements in the search space and this is likely to be the reason for the weak performance of PSO-MLP, GWO-MLP and ACO-MLP. Although DO is also a swarm-based algorithm, its mechanisms described in the preceding paragraph are the reasons why it is advantageous in training MLPs.

Generally speaking, the GA algorithm has been designed based on various mutation mechanisms. Mutation in evolutionary algorithms maintains the diversity of population and promotes exploitation, which is one of the main reasons for the weak performance of GA.

In addition, selection of individuals in this algorithm is done by a deterministic approach. Consequently, the randomness in selecting an individual is less and therefore local optima avoidance is less as well. This is another reason why Do have good results compare with GA.

The reason for the high classification rate provided by the DO-MLP algorithm is that this algorithm is equipped with adaptive parameters to smoothly balance exploration and exploitation. Half of the iteration is devoted to exploration and the rest to exploitation. In addition, the DO algorithm always saves the best obtained solution at any stage of optimization. Consequently, there are always guiding search agents for exploitation of the most promising regions of the search space. In other words, DO-MLP benefits from intrinsic exploitation guides, which also assist this algorithm to provide remarkable results.

According to this comprehensive study, the DO algorithm is highly recommended to be used in hybrid intelligent optimization schemes such as training MLPs. Firstly, this recommendation is made because of its high exploratory behavior, which results in high local optima avoidance when training MLPs. The high exploitative behavior is another reason why a DO-based trainer is able to converge rapidly towards the global optimum for different datasets. However, it should be noted here that DO is highly recommended only when the dataset and the number of features are very large. Obviously, small datasets with very few features can be solved by gradient-based training algorithms much faster and without extra computational cost. In contrast, the DO algorithm is useful for large datasets due to the extreme number of local optima that makes the conventional training algorithm almost in effective.

To conclude, the DO algorithm as a trainer for MLP achieved superior results than the other three algorithms, i.e.it can avoid the local minimum problem. Thus, the DO algorithm is recommended to optimize the training process in MLPs. This is because as reported in [18], the DO has high exploratory behavior over GA and PSO, which could help in the local optima avoidance. Moreover, it has high exploitation behavior [18], thus it converges rapidly towards the global optimum.

TABLE 1
DATASETS DESCRIPTION

Dataset	# Attributes	# Training Samples	# Testing Sample	# Classes	MLP Structure
3-bits XOR	3	8	8	2	3-7-1
IRIS	4	150	150	3	4-9-3
Heart	22	80	187	2	22-45-1
Breast Cancer	9	599	100	2	9-19-1

TABLE 2
INITIAL PARAMETERS OF THE OPTIMIZATION ALGORITHMS.

Optimization Algorithm	Parameter	Value
GA	Crossover	Single point (probability=1)
	Mutation	Uniform (probability=0.01)
	Type	Real Code
PSO	Topology	Fully Connected
	Social constant (C2)	1
	Cognitive constant (C1)	1
	Inertia constant (ω)	0.3
ACO	Initial pheromone (τ)	1e − 06
	Pheromone update constant (Q)	20
	Pheromone constant (q)	1
	Global pheromone decay accuracy (pg)	0.9
	Local pheromone decay accuracy (pt)	0.5
	Pheromone sensitivity (α)	1
DO	The weight of separation (s)	0.1
	The weight of alignment (a)	0.1
	The weight of cohesion (c)	0.7
	The food factor (f)	1
	The enemy factor (e)	1

TABLE 3
CLASSIFICATION ACCURACY FOR THE XOR, IRIS, HEART, AND BREAST CANCER DATASETS.

Dataset	Algorithm	MSE	Classification Accuracy (%)
XOR	GWO-MLP	0.009410	100.00
	PSO-MLP	0.087050	37.50
	GA-MLP	0.000192	100.00
	ACO-MLP	0.183328	62.50
	DO-MLP	1.6098e − 004	100.00
Heart	GWO-MLP	0.122600	75.00
	PSO-MLP	0.1892	68.75
	GA-MLP	0.0956	58.75
	ACO-MLP	0.2283	00.00
	DO-MLP	0.14200	92.50
Breast Cancer	GWO-MLP	0.0012	98.5
	PSO-MLP	0.04211	11.50
	GA-MLP	0.003026	98
	ACO-MLP	0.014800	40
	DO-MLP	6.3607e − 011	99.33
Iris	GWO-MLP	0.0269	91.33
	PSO-MLP	0.238680	37.50
	GA-MLP	0.092	89.33
	ACO-MLP	0.405979	33.5
	DO-MLP	0.0220	92.50

TABLE 4
EXPERIMENTAL RESULTS FOR THE SINE DATASET

Data Set (Function)	Algorithm	Test Error
Sine	GWO-MLP	43.754
	GA-MLP	111.25
	PSO-MLP	124.89
	ACO-MLP	117.71
	DA-MLP	1.5354

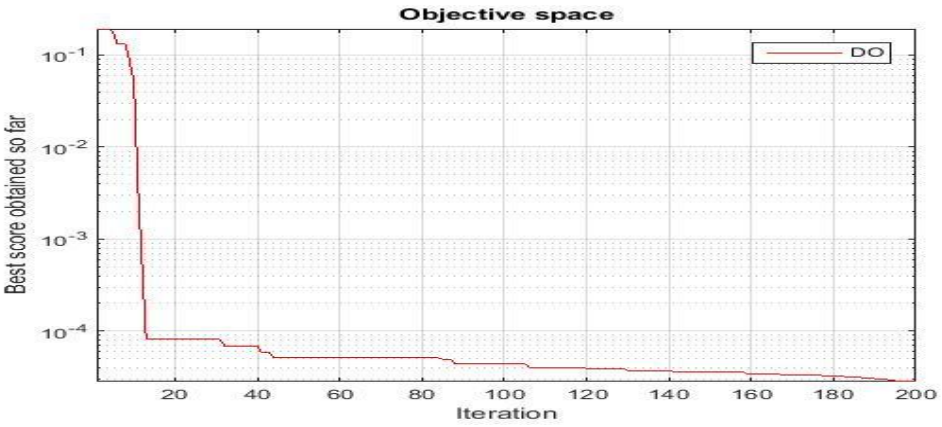

Fig 4:Relation between no. of iteration and best score for Dragonfly Algorithm for XOR Data Set

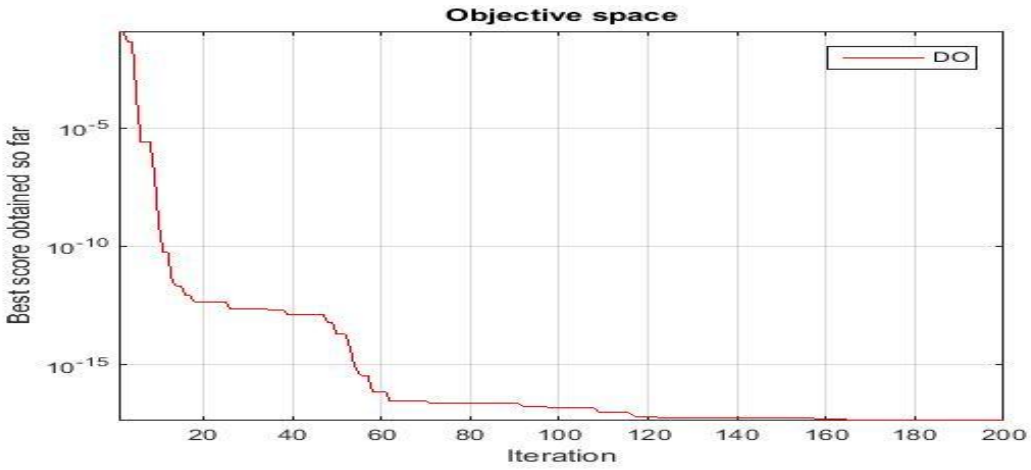

Fig 5: Relation between no.of iteration and best score for Dragonfly Algorithm for IRIS Data Set

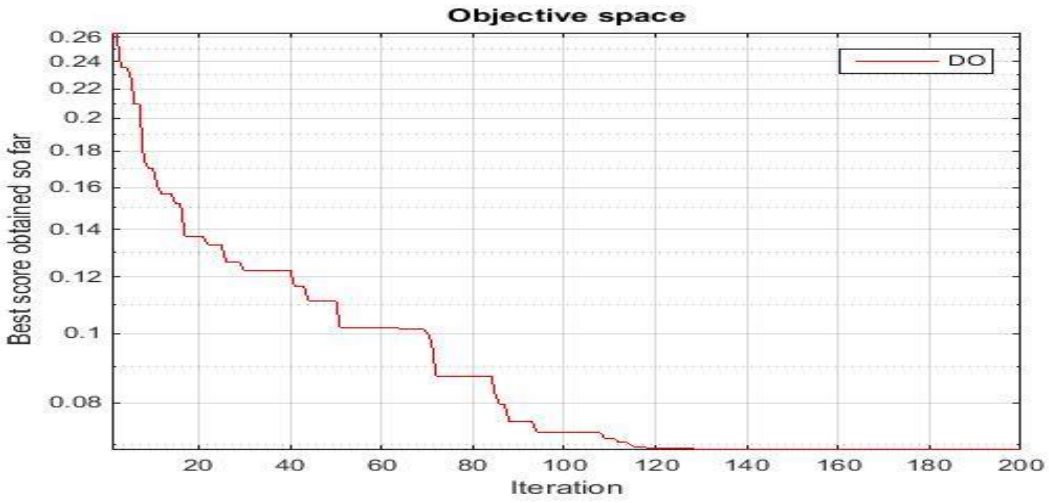

Fig 6: Relation between no. of iteration and best score for Dragonfly Algorithm for Breast Cancer Data Set

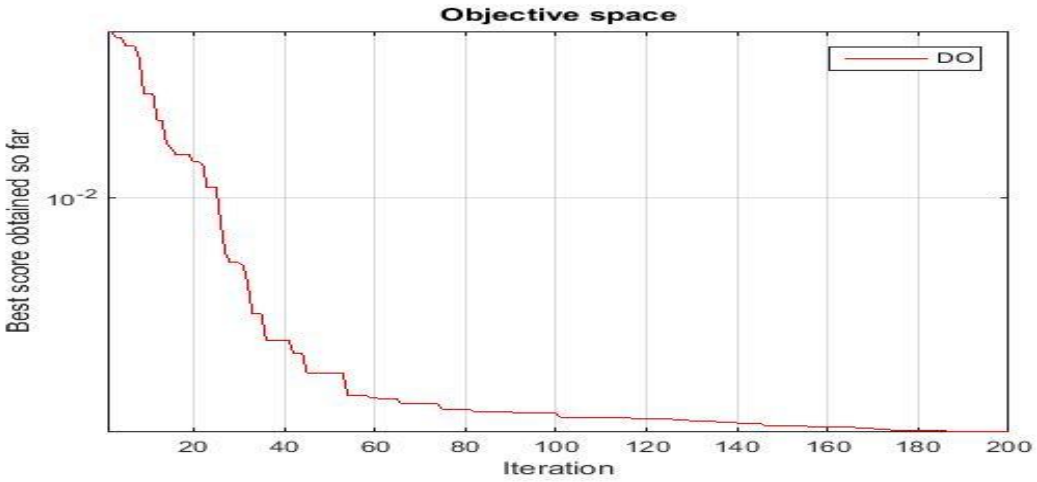

Fig 7 Relation between no. of iteration and best score for Dragonfly Algorithm for Heart Data Set

V. CONCLUSION

In this paper, the Dragonfly Optimizer (DO) was proposed as a stochastic trainer to the MLP. The problem of training the MLP was formulated for the DO algorithm to find the optimal values for the weights and biases. The proposed model based on DO was then evaluated by four standard classification datasets. The results of the proposed model were compared with four optimization trainers namely, GWO, PSO, ACO, and GA each of them used to train MLP. The results showed that the proposed model as a trainer for the MLPs can efficiently solve the local minimum problem, which helped to find the optimal values for the weights and biases parameters of MLP. Moreover, the proposed model achieved low MSE and high classification accuracy due to a high exploitation of the dragonfly trainer, while the other algorithms (e.g. GWO) suffer from low exploration. In the future work, different types the data sets will be used to evaluate that the DO-based trainer for MLP can efficiently find the optimal values for the weights and biases in these data sets.

REFERENCES

[1] W. S. McCulloch and W. Pitts. (1943), "A logical calculus of the ideas immanent in nervous activity."The bulletin of mathematical biophysics (1943) 115–133.
[2] David Lowe (2015), "Radial Basis Function Networks – Revisited" Mathematics Today (2015) 124-126
[3] Le Yang. (2014),''A Modified Self-Organizing Maps and Its Applications". International Conference on Computational Science (2014)1-9
[4] S. Ghosh- Dastidar and H. Adeli. (2009), "Spiking neural networks," International journal of neural systems 19 (2009) 295–308.
[5] J. S. Zhang.(2014), "A survey of neural network algorithms," Chinese Journal of Computers, vol. 34, no. 8, pp. 1399–1410, 2014
[6] Zhen-Guo Che.(2011) ," Feed forward neural networks training a comparative between genetic algorithm and back propagation learning algorithm "International Journal of Innovative Computing, Information and Control Volume 7, Number 10, October 2011.
[7] G. E. Hinton and T. J. Sejnowski .(1999), "Unsupervised learning," foundations of neural computation. MIT press (1999).
[8] D. Wang. (2001), "Unsupervised learning: foundations of neural computation," AI Magazine 22 (2001) 101.
[9] R. Caruana and A. Niculescu-Mizil. (2006), "An empirical comparison of supervised learning algorithms," in Proceedings of the 23rd international conference on Machine learning. ACM (2006) 161–168.
[10] Ndiaye (2014), "Development of a multilayer perceptron (MLP) based neural network controller for grid connected photovoltaic system,". Int. J. Phys. Sci. 9(3), (2014)41–47.
[11] G.-B. Huang, Q.-Y. Zhu and C.-K. Siew. (2004), "Extreme learning machine: a new learning scheme of feed forward neural networks," in Proceedings IEEE International Joint Conference on Neural Networks IEEE 2 (2004) 985–990.
[12] Himansu Das. (2015), "A novel PSO based back propagation learning-MLP (PSO-BP-MLP) for Classification,"science direct (2015)132-139.
[13] S. Mirjalili. (2015), "How effective is the grey wolf optimizer in training multilayer perceptrons," Applied Intelligence43 (2015) 150–161.
[14] M. Soleymanpour, P. Vrat, and R. Shankar. (2002), "A transiently chaotic neural network approach to the design of cellular manufacturing," International Journal of Production Research 40 (2002) 2225–2244.
[15] C. Chua and A. Goh. (2003), "A hybrid bayesian back-propagation neural network approach to multivariate modeling," International Journal for numerical and analytical methods in geomechanics. 27 (2003) 651–667.
[16] Y. Tang and R. R. Salakhutdinov. (2013), "Learning stochastic feed forward neural networks," in Advances in Neural Information Processing Systems (2013) 530–538.
[17] -G.-G. Wang, A. H. Gandomi, A. H. Alavi, and G.-S. Hao. (2014), "Hybrid krill herd algorithm with differential evolution for global numerical optimization," Neural Computing and Applications 25 (2014) 297–308.
[18] S. Mirjalili. (2015), "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems." Neural Computing and Applications (2015) 1-21.
[19] S. Mirjalili, Sadiq AS (2011), "Magnetic optimization algorithm for training multi-layer perceptron. In: Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference, IEEE, (2011) 42–46.
[20] -Linga Venkatesh .(2011),"Flocks, Herds, and Schools: A Distributed Behavioral Model " CS775 Paper Abstract(2011)1-3
[21] Yang X-S (2010) "Nature-inspired metheuristic algorithms", 2ndedn. Luniver Press.
[22] Bache, K., Lichman M. (2013), UCI machine learning repository. University of California, Irvine. CA School of Information and Computer Science.(2013)[http://archive.ics.uci.edu/ml]