A hybrid constriction coefficient-based particle swarm optimization and gravitational search algorithm for training multi-layer perceptron

Sajad Ahmad Rather and P. Shanthi Bala
Department of Computer Science, School of Engineering and Technology, Pondicherry University, Puducherry, India

Received 30 September 2019
Revised 13 January 2020
Accepted 4 February 2020

Abstract

Purpose – In this paper, a newly proposed hybridization algorithm namely constriction coefficient-based particle swarm optimization and gravitational search algorithm (CPSOGSA) has been employed for training MLP to overcome sensitivity to initialization, premature convergence, and stagnation in local optima problems of MLP.

Design/methodology/approach – In this study, the exploration of the search space is carried out by gravitational search algorithm (GSA) and optimization of candidate solutions, i.e. exploitation is performed by particle swarm optimization (PSO). For training the multi-layer perceptron (MLP), CPSOGSA uses sigmoid fitness function for finding the proper combination of connection weights and neural biases to minimize the error. Secondly, a matrix encoding strategy is utilized for providing one to one correspondence between weights and biases of MLP and agents of CPSOGSA.

Findings – The experimental findings convey that CPSOGSA is a better MLP trainer as compared to other stochastic algorithms because it provides superior results in terms of resolving stagnation in local optima and convergence speed problems. Besides, it gives the best results for breast cancer, heart, sine function and sigmoid function datasets as compared to other participating algorithms. Moreover, CPSOGSA also provides very competitive results for other datasets.

Originality/value – The CPSOGSA performed effectively in overcoming stagnation in local optima problem and increasing the overall convergence speed of MLP. Basically, CPSOGSA is a hybrid optimization algorithm which has powerful characteristics of global exploration capability and high local exploitation power. In the research literature, a little work is available where CPSO and GSA have been utilized for training MLP. The only related research paper was given by Mirjalili *et al.*, in 2012. They have used standard PSO and GSA for training simple FNNs. However, the work employed only three datasets and used the MSE performance metric for evaluating the efficiency of the algorithms. In this paper, eight different standard datasets and five performance metrics have been utilized for investigating the efficiency of CPSOGSA in training MLPs. In addition, a non-parametric pair-wise statistical test namely the Wilcoxon rank-sum test has been carried out at a 5% significance level to statistically validate the simulation results. Besides, eight state-of-the-art meta-heuristic algorithms were employed for comparative analysis of the experimental results to further raise the authenticity of the experimental setup.

Keywords Neural network, Feedforward neural network (FNN), Gravitational search algorithm (GSA), Particle swarm optimization (PSO), Hybridization, CPSOGSA, Multi-layer perceptron (MLP)

Paper type Research paper

1. Introduction

The neural network (NN) is one of the best computational tools used in the field of soft computing. McCulloch and Pitts (1943) were the first who introduced a computational model for neural networks. Currently, neural networks are quite popular among researchers. Besides, NNs have the properties of mathematical simplicity, high efficiency and low computational cost. There are different types of neural networks including recurrent neural network (RNN) (Dorffuer, 1996), Kohonen self-organizing network (KSON) (Kohonen, 1990),

The authors would like to thank reviewers for their valuable feedback and extremely useful comments. Actually incorporation of referee(s) suggestions have only increased the quality of the work.

International Journal of Intelligent Computing and Cybernetics
Vol. 13 No. 2, 2020
pp. 129-165

1756-378X
DOI 10.1108/IJICC-09-2019-0105

feedforward neural network (FNN) (Bebis and Georgiopoulos, 1994), and radial basis function (RBF) (Park and Sandberg, 1993). FNN is the simplest neural network in which information moves in a unidirectional manner from input to output nodes.

FNN is commonly classified into single-layer perceptron (SLP) and multi-layer perceptron (MLP). In SLP, there is an input layer, only one hidden layer (if any), and output layer. At each node, the product of input and weights is calculated. If the value is greater than the threshold, then some action is taken. Moreover, SLP is preferred for finding linearly separable patterns in data. On the other hand, MLP consists of an input layer, one or more hidden layer(s) and an output layer. In fact, the sigmoid function is generally used as a fitness function in MLP because of its continuous nature. Besides, MLP is utilized for the classification of data in nonlinear systems.

There may be structural and modeling differences among different neural networks but all of them have the common feature of learning. It is defined as the property of NNs to perform a particular task by considering sample observations. The accuracy of results in NNs is directly related to its learning capability. The learning is categorized into supervised and unsupervised. In supervised learning, there is a one to one correspondence between inputs and outputs. In fact, NN gets feedback from the fitness function regarding the quality of the solutions. In contrast, unsupervised learning consists of input data, fitness function, and output. Besides, NN adapts itself without a need for any supervisory feedback.

Previously, backpropagation (BP) and gradient descent methods have been readily employed for training MLPs. It has been observed that BP has the advantages of mathematical simplicity and high speed. But it suffers from the shortcomings of stagnation in local minima (Gori and Tesi, 1992; Lee *et al.*, 1993; Mangasarian and Wolberg, 1990) and premature convergence (Fahlman, 1988; Ng *et al.*, 2003; Vogal *et al.*, 1988). Due to the above drawbacks, BP has some issues in solving practical problems.

During the learning process, the weights and biases of MLP are modified to reduce the error in training and testing samples. But it has been reported that MLP spends a large time in sub-optimal regions of search space. In simpler terms, the training algorithm takes MLP towards the local minimum rather than the global minimum. BP and other gradient descent algorithms suffer from this problem. Researchers (Jacobs, 1988; Ooyen *et al.*, 1992; Weir, 1991) have tried to overcome stagnation in local minima, but the only small improvement has been carried out. In contrast, heuristic algorithms have efficient local minima avoidance capability (Mirjalili *et al.*, 2013) and high convergence speed (Gudise and Venayagamoorthy, 2003) as compared to most of the gradient descent algorithms including BP. Hence, heuristic techniques are a suitable alternative for gradient descent algorithms in training MLPs (Branke, 1995; Yao, 1993).

In literature, there are many heuristic algorithms utilized for training MLPs such as differential evolution (DE) (Llonen *et al.*, 2003; Slowik and Bialko, 2008), ant colony optimization (ACO) (Blum and Socha, 2005; Socha and Blum, 2007), genetic algorithm (GA) (Whitney *et al.*, 1990; Mirjalili *et al.*, 2012), artificial bee colony (ABC) (Karaboga *et al.*, 2007; Ozturk and Karaboga, 2011) and particle swarm optimization (PSO) (Mendes *et al.*, 2002; Gudise and Venayagamoorthy, 2003). The recent additions in the list of stochastic training algorithms include social spider optimization algorithm (SSO) (Pereira, 2014), teaching-learning based optimization (TLBO) (Uzlu *et al.*, 2014), biogeography based optimization (BBO) (Mirjalili *et al.*, 2014), symbiotic organisms search algorithm (SOS) (Wu *et al.*, 2016), glowworm swarm optimization (GSO) (Alboaneen *et al.*, 2017) and improved PSO (Li, 2018).

In this paper, constriction coefficient based particle swarm optimization and gravitational search algorithm (CPSOGSA) has been used for training MLP. It gets diversification power from GSA and high exploitation capability from PSO. The CPSOGSA will be tested for efficiency in avoiding stagnation in local minima and convergence speed. In this work, 9 standard datasets will be employed to benchmark the efficiency of CPSOGSA in training MLP.

The rest of the paper is structured as follows: Motivation regarding utilization of CPSOGSA for FNN training is described in Section 2 while Section 3 provides a literature survey of previous works in connection with MLP training using gradient descent and heuristic algorithms. Moreover, Section 4 gives brief description of the FNN and Multi-layer Perceptron (MLP). The CPSO, GSA, and CPSOGSA are explained in Sections 5, 6, and 7, respectively. Section 8 introduces the CPSOGSA algorithm for MLP training. The experimental analysis is covered in Section 9. Finally, Section 10 provides a conclusion and future direction.

2. Motivation

Hybridization is the process of combining two or more techniques to resolve the issues of participating algorithms and solve complex problems. The hybridization results in the increase of performance and accuracy of the algorithms. The limitation of an algorithm is overcome by the ability of others. The CPSOGSA is a hybrid optimization algorithm which has powerful characteristics of global exploration capability and high local exploitation power. The CPSOGSA gets diversification power from GSA which helps to search the whole problem space. In simpler terms, more exploration power means minimum chances of getting stuck in local minima. So, CPSOGSA has the intrinsic property of getting away from local minima.

In addition, CPSOGSA gets local exploitation power from the gbest operator of PSO which helps to attract the optimal solutions towards the global minima. In other words, more exploitation power means a high convergence rate. So, CPSOGSA has the capability of finding the global minimum in fewer iterations and hence, providing high accuracy of results.

It has been reported that if an optimization algorithm has good exploration capability, then it will be lacking in good exploitation power and vice versa (Eiben and Schippers, 1998). In fact, they are inversely proportional to each other. Every optimization algorithm faces this problem, i.e. a particular technique may be having good randomization power but performs poorly in finding the global minimum. This issue may be termed as "optimization balance glitch (OBG)." However, CPSOGSA does not have an OBG problem because of its hybrid nature.

Furthermore, a little work is available in research literature where CPSO and GSA have been utilized for training MLP. The only related research paper was given by Mijalili *et al.* in 2012. They have used standard PSO and GSA for training simple FNNs. However, the work employed only three datasets and used MSE performance metric for evaluating the efficiency of the algorithms. In this paper, eight different standard datasets and five performance metrics including a statistical test have been utilized for investigating the efficiency of CPSOGSA in training MLPs.

3. Literature survey

The backpropagation (BP) algorithm is one of the first gradient descent techniques utilized for training ANN. BP is a mathematical optimization technique that modifies the connection weights of NN by calculating the differential coefficient of the activation function. Adaptive CGA is one of the techniques in the BP family of algorithms. It has been used for resolving the learning rate and momentum ratio problem of BP by introducing step length search during the learning process. The efficiency of adaptive CGA has been tested by applying it to the engineering domain and image classification problem (Adeli and Hung, 1994).

Marquardt algorithm (MA) is a least-square optimization technique that has been employed for increasing the convergence speed of BP. The MA is benchmarked by using a 4-D function, 3-D function, 3-D Sinc function, square and sine wave datasets. The results of MA are compared with the variable learning rate method and conjugate gradient algorithm (Hugan and Menhaj, 1994).

It has been seen that BP has the advantages of simplicity and high speed. However, it has the shortcomings of stagnation in local minima and slow exploitation rate. To resolve the aforementioned problems, researchers have introduced modified versions of BP. Magnified gradient function (MGF) algorithm, basically a modified BP technique, has been employed for overcoming the convergence speed issue of BP. The MGF makes adjustments in the gradient of a cost function which in turn enhances the global exploration of the NN. The proposed algorithm was tested for efficiency on four nonlinear benchmarks including XOR problem, 5-bit counting problem, 3-bit parity and regression problems (Ng *et al.*, 2003).

There have been attempts made by prominent researchers to modify the mathematical formulation of BP to observe its effects on the performance of the algorithm. The experiments have demonstrated that small modifications in the total error function of BP resulted in the accelerated convergence speed of the NN (Ooyen *et al.*, 1992).

The performance of BP is directly related to the architecture of the NN. Gori and Tesi (1992) have proposed two mathematical theorems that have a direct impact on the network structure and learning environment of BP. The simulation results have confirmed that modified BP works comfortably with linear and non-linear separable problems. However, the performance enhancement achieved is small as compared to standard BP.

Heuristic algorithms (HA) as compared to gradient descent techniques have the property of randomness meaning they can initialize the search space with random candidate solutions. Moreover, the fitness function is used to train the NN iteratively until a stopping condition is met. The randomized iterative process makes HA less prone to entrapment in local minima and convergence speed issues. Genetic Algorithm (GA) is the first evolutionary technique to be used to train the NN. The various steps in GA include selection, reproduction, recombination, and crossover. It has been reported that GA is less prone to stagnation in local optima. This property of GA was employed to adjust the connection weights and biases of NNs to find the new patterns in data (Whitley *et al.*, 1990). Besides, Researchers like Itano *et al.* (2018) have applied GA based MLP for the optimization of topology, bias, and weights of ANN.

Most of the HAs are inspired by nature like PSO which is based on the group dynamics of fishes and birds. It has three important variables including inertia factor (for exploration), pbest (personal component) and gbest (global component) for exploitation. The PSO was always popular among researchers due to its mathematical simplicity, easy implementation, and usage. NNs were trained using PSO in which logistic cost function was used to adjust the weights and biases for minimizing the threshold error. Classification and regression datasets were employed to investigate the efficiency of PSO as compared to BP (Mendes *et al.*, 2002).

It has been verified by a number of studies that NNs have local minima and convergence speed issues. However, HAs, on the other hand, have simple implementation and can easily resolve the aforementioned issues of NNs. To support the above argument, levy flight distribution based PSO combined with neighborhood search has been successfully employed for MLP training to overcome exploitation and diversification shortcomings of FNNs. The hybrid model was benchmarked using UCI classification datasets. However, the study lacks a proper statistical analysis of the proposed hybrid model (Tarkhaneh *et al.*, 2019).

Moreover, PSO is employed for training FNN to approximate the nonlinear function. The results are compared with BP. The experimental results convey that PSO takes six times a lesser number of iterations to find the global minimum (Gudise and Venayagamoorthy, 2003).

The researchers have also employed hybrid approaches for training NNs. Chen *et al.* (2015) have embedded PSO and CS (Cuckoo Search) algorithms as a learning mechanism for MLP training. In the hybrid algorithm, PSO is used for exploitation whereas CS performs global search of the solution space. Moreover, function approximation and classification datasets were considered for benchmarking the hybrid algorithm.

The randomization and intensification capabilities of GSA and PSO respectively were applied for training FNN. In fact, GSA is efficient in searching for the problem space globally and is less prone to local minima issues. Furthermore, PSO is known for its high convergence power due to its gbest and pbest operators. Hybrid GSA and PSO have been employed for training FNN by employing the multiple number of hidden layers. The performance of the hybrid algorithm was investigated using classification and function approximation datasets (Mirjalili *et al.*, 2012).

Similarly, Zhang *et al.* (2007) have framed a hybrid model consisting of PSO and BP algorithms to train FNN. The PSO carries out a global search while BP helps in the convergence of agents towards the global optimum. The hybrid PSOBP approach was applied to three bits parity, classification and function approximation benchmarks. The simulation results indicate the efficient performance of the PSOBP algorithm as compared to adaptive PSO and BP classifiers. However, the limitation of the study is that they have used only two algorithms for comparative analysis and no statistical test was carried out to statistically validate the results.

ACO is another specialized HA inspired from the group behavior of real ant colonies. It was invented to solve discrete and combinatorial optimization problems. In addition, ACO is also efficient in working out solutions for continuous computational problems. ACO is efficiently employed to train FNN for solving pattern recognition problems. Different classification datasets including heart, diabetes and cancer from UCI repository were used as benchmarks to test the efficiency of ACO against GA, BP and LM (Socha and Blum, 2007).

From the perspective of stochastic algorithms, ABC is an intelligent technique inspired by the swarm behavior of honey bee colonies. It was proposed by Karaboga (2005) to solve numerical optimization problems. It has the properties of simplicity, robustness and stochasticity. Moreover, FNN has been trained using ABC in which different nonlinear datasets including XOR, 3-bit parity and encoder-decoder problems were utilized to test the performance of ABC. The experimental results were compared with GA, BP and LM algorithms (Karaboga *et al.*, 2007).

DE is another evolutionary technique that finds the global minimum of the problem by iteratively improving the solution quality of searching agents. It was proposed by Storn and Price (Llonen *et al.*, 2003) for solving real-valued numerical optimization problems. In the case of training FNN, DE has been utilized for searching the problem space to find the optimal candidate solutions. It has been reported that DE has not any appreciable edge in terms of performance over gradient descent algorithms. The experimental results on various classification datasets supported the above statement (Llonen *et al.*, 2003).

Furthermore, an adaptive selection strategy has been introduced in DE to resolve the issue of stagnation in local minima of error BP. In the modified DE, the focus was on the proper utilization of searching agents to adjust the FNN weights. The results were compared with LM, error BP and evolutionary algorithm ANN (Slowik and Bialko, 2008).

Recently, CFO has been introduced into the family of HAs. It is a physics-based intelligent technique that supports deterministic behavior. It uses probes as agents to explore the search space. In the case of training NN, CFO shows promising results by solving the XOR problem and Iris classification dataset with high performance. The simulation results indicate CFO provides competitive values for fitness function in reducing threshold error as that of PSO (Green *et al.*, 2011).

From the perspective of HAs, the SSO algorithm is an intelligent technique inspired by the swarm behavior of spiders. The search space consists of candidate solutions in the form of spiders moving in the communal web. The interesting feature of SSO is that the female population (70–90%) is more than the male population (20–30%). The global minimum of the agent system is the position of the spider in the problem space. Pereira *et al.* (2014) have recently employed SSO for training MLP and solved Parkinson's disease identification

dataset with high performance. Moreover, Mirjalili *et al.* (2015) have also used the SSO algorithm to train FNN for pattern recognition. They utilized five classification datasets such as breast cancer, iris, balloon, XOR and heart for testing the efficiency of the SSO algorithm. Besides, the statistical results of SSO were compared with PSO, GA, ACO, PBIL and ACO algorithms.

It has been observed that classification datasets specifically heart, breast cancer, Iris and function approximation benchmarks such as sine, cosine, and sphere functions were utilized for investigating the performance of deterministic gradient descent algorithms like BP and stochastic intelligent techniques such as GA and PSO. In a similar pattern, CRO – a new meta-heuristic method based on the interaction and transition of molecules in a reaction – has been benchmarked for training MLP using classification datasets. The simulation results show that CSO gives efficient results for iris and Wisconsin breast cancer datasets while providing competitive values for Pima Indian diabetes dataset (Yu *et al.*, 2011).

In the field of meta-heuristics, it can be seen that the majority of the algorithms are nature or evolution inspired and only a few are motivated from the physical and chemical sciences. In the former category, CSS is a physics-based optimization technique mimicking the interaction model of the charged particles in the electrical field. The searcher agents are electrostatic charges obeying coulombs, gauss and Newton's laws. Recently, CSS was applied to non-technical losses problem in power systems by training MLP. The accuracy rates of CSS were more as compared to BP (Perira *et al.*, 2013). Likewise, GSA is another physics-based stochastic algorithm inspired by Newton's law of universal gravitation. Adaptive best mass GSA has been combined with BP to classify iris, lenses, and sonar datasets. The hybrid approach uses convergence speed, fail probability, and recognition rate as performance measures. The simulation results are compared with traditional GSA, BP, GA and PSOGSA (Mosavi *et al.*, 2019).

The stochastic techniques have been applied to almost all areas of computer science. However, computer vision and data mining were attractive areas for meta-heuristics. In the field of computer vision, image segmentation is an active area of research. IWO, is a HA based on the colonizing aspect of agricultural weeds have been applied to find defects in potato color images by training MLP. In fact, in the hybrid approach, IWO is used for global search and MLP is employed for handling constraints of the problem. The experimental results indicate the efficient performance of IWO as compared to traditional BP (Moallem and Razmjooy, 2012).

Researchers namely Uzlu *et al.* (2014) applied the TLBO (Teaching Learning-based optimization) algorithm, a global searching method based on the teacher-student model of instruction; to forecast the energy consumption in Turkey. They used an import, export and population data as input for the study. The MLP trained by TLBO gave energy predictions that were close to the projections given by the Turkish ministry of development.

Magnetic optimization algorithm (MOA), a stochastic method based on the law of magnetic field has been applied to train MLP for calculation of XOR bit values and approximation of sine wave curve (Mirjalili and Sadiq, 2011). Likewise, moth-flame optimization (MFO) algorithm is another HA inspired by the traversal and spiral path orientation of moths. RBF neural networks were trained using MFO for pattern recognition in binary classification datasets (Faris *et al.*, 2017a). Recently, Faris *et al.* (2016a) trained NN by using lightning search algorithm (LSA). It has been seen that LSA has good exploration and exploitation capabilities. These merits of LSA were utilized to overcome entrapment in local minima and convergence issues of NNs. Moreover, 16 classification datasets were used for performance analysis and the Wilcoxon rank-sum test was employed to statistically validate the experimental results. In the line of stochastic algorithms, monarch butterfly optimization (MBO) algorithm is based on the population behavior of monarch butterflies while migrating from Canada to the USA and Mexico. The researchers have

applied Improved MBO trained MLP to 12 UCI datasets and compared simulation results with other HAs. The statistical results depict the optimal performance of IMBO (Faris *et al.*, 2017b).

Similarly, Krill Herd (KH) algorithm has also been effectively exploited for MLP training for finding useful patterns from the classification datasets. Different performance metrics were used for checking the efficiency of the algorithms like classification error, the sum of square errors, and computational time. The simulation results of KH were compared with Heuristic Search, BP and GA (Kowalski *et al.*, 2016).

BBO is inspired by the biological distribution of organisms in a particular habitat. It consists of emigration, immigration, and mutation operators for exploration and exploitation, respectively. Mijalili *et al.* (2014) have employed it to train MLP to overcome the shortcoming of local minima entrapment and slow convergence speed of MLPs. The various classification and function approximation datasets were utilized for performance evaluation. It is a fact that radial basis function (RBF) networks are readily used NNs having simple structure, high learning speed, significant noise tolerance level and very good approximation power. RBF mainly uses the Gaussian activation function for training. Alijarah *et al.* (2018) for the first time used one of the highly regarded HA, that is, BBO for training RBF neural network. They utilized 12 datasets for performance evaluation while 11 different optimization algorithms were employed for comparative analysis. In another study, a chaotic version of BBO has also been used as a learning method to train MLP to optimize NN weights and neural biases. Classification datasets like balloon, iris, heart, and vehicle were utilized for performance evaluation. The comparative analysis of the experimental results of Chaotic BBO was done with standard BBO, GSA, and PSO (Pu *et al.*, 2018).

GWO is a new addition in the list of stochastic algorithms motivated by the group conduct and hunting skills of forest wolves. MLP was trained using GWO for solving pattern recognition and function approximation problems. The results were compared with PSO and GA for efficiency and classification accuracy (Mijalili, 2015). Besides, Amirsadri *et al.* (2017) have embedded GWO with levy flight distribution and BP for training MLP in order to resolve entrapment in local minima issue of NNs. In the hybrid GWOBP approach, GWO carries out a global search while BP provides local searching support. The performance evaluation includes comparative analysis with other stochastic algorithms, benchmarking with classification and function approximation datasets. In another work, researchers have hybridized GSA, GWO and BBO for MLP training. The embedded approach was used to predict forest fires in Vietnam (Bui *et al.*, 2018).

In related work, the SOS algorithm was applied to pattern recognition task by training MLP. Different UCI repository classification datasets were consulted to benchmark the efficiency of SOS (Wu *et al.*, 2016). Similarly, Whale Optimization Algorithm (WOA) is a recent HA inspired by the hunting tactics and feeding behavior of humpback whales. WOA has been utilized for FNN training due to its advantages of high convergence speed and local minima avoidance capability. Moreover, 20 different datasets were used to benchmark the efficiency of WOA. Also, six optimization algorithms like BP, GA, PSO, DE, ACO and PBIL (population based incremental learning) were considered for comparative analysis (Alijarah *et al.*, 2016).

Multi-verse optimizer (MVO) is a Cosmology inspired algorithm. It utilizes three concepts such as wormhole, whitehole and blackhole for local search, diversification, and intensification of the solution space. It has been quite successfully employed for training FNNs in order to accurately classify 9 UCI medical datasets. Moreover, the experimental results of MVO were classified with DE, GA, PSO, CS and FF (FireFly) algorithms. Also, the simulation results convey the optimal performance of MVO as compared to other HAs (Faris *et al.*, 2016b).

Also, glowworm swarm optimization (GSO) algorithm is an HA inspired by the swarm behavior of glowworms. The glowworms act as agents in the solution space moving from one place to another to find the best solution. GSO acts as a learning algorithm to train the MLP for the classification of feature vectors of datasets including XOR, breast cancer, liver disorder and vertebral column. Moreover, GSO has been compared with BBO, GA, and MVO algorithms for classification accuracy (Alboaneen *et al.*, 2017). Furthermore, Li (2018) has applied an improved version of PSO on the CEC 2014 test benchmarks to find the global optimum. Besides, MLP was trained to investigate the efficiency of the proposed algorithm in handling complex search spaces.

In very recent work, Li *et al.* (2018) have used spotted hyena optimization algorithm (SHOA) to train FNN. Basically, they have used it to overcome the local minima problem of FNNs. Moreover, SHOA has been applied to the heart classification dataset consisting of SPECT images. The statistical results depict SHOA is efficient as compared to GWO, HS GGSA, and DE. Likewise, grasshopper optimization algorithm (GOA) is another HA inspired from the group behavior of the nature grasshoppers. The GOA is known for its high global searching capability. Heidari *et al.* (2018) have applied GOAMLP for pattern recognition. The experimental results depict the efficient performance of the proposed method.

Similarly, bird swarm optimization (BSO) is a new stochastic algorithm inspired by the foraging and vigilance behavior of birds. The BSOMLP training model was benchmarked using 3 function approximation and 13 pattern recognition datasets. Besides, the reactor dataset was employed to test the practical applicability of the hybrid framework (Alijarah *et al.*, 2019). Also, Zhao *et al.* (2019) trained MLP using Modified Selfish Herd Optimization (MSHO) algorithm. It has been seen that MSHO has good exploration and convergence properties. MSHOMLP was applied to UCI classification datasets and compared with six swarm optimization algorithms.

Table 1 depicts the summary of the literature survey regarding the utilization of gradient descent (GD) and evolutionary algorithms in training MLP. From the literature survey, it is clear that HAs are efficient learning algorithms in training MLPs while gradient descent algorithms such as BP and LM also have appreciable convergence speed in finding the global optimum. However, gradient descent algorithms (GD) face the entrapment in local minima issue which makes them unsuitable for training FNNs. However, it is recommended to use GD algorithms with HA in which exploitation will be done by GD and exploration by HA. Also, HAs have the properties of avoiding entrapment in local minima, fast convergence speed, and less sensitivity to initialization which makes them ideal and preferred for MLP training. Besides, when applied to classification datasets such as breast cancer, heart, iris and XOR problem and function approximation datasets such as sine wave, square function and cosine function provide high recognition rate and accuracy of results. In this work, we have used a hybrid CPSOGSA stochastic algorithm as a learning mechanism to train MLP. The CPSOGSA has powerful global exploration capability due to GSA which helps to search the whole solution space in less computational time and hence, minimum chances of getting stuck in the local minima. Also, it has appreciable exploitation power due to CPSO which enhances its convergence speed and hence, high probability of finding the global optimum. The aforementioned properties of CPSOGSA makes it capable to handle easily entrapment in local minima and resolve convergence issues of the MLPs. Moreover, CPSOGSA will be tested for its efficiency in optimizing connection weights and neural biases of MLP and also minimization of the threshold error of MLP activation function.

4. FNN and multi-layer perceptron (MLP)

FNN are those NNs in which computational information is unidirectional, i.e. from input to output. There are three layers present in a basic model of FNN which are the input layer,

Literature Reference	Year	Training Algorithm	Nature of Proposed Method	Dataset Employed
Whitley *et al.*	1990	GA	Stochastic	Weight and Bias Adjustment
Ooyen *et al.*	1992	Error BP	Gradient descent	Convergence Speed Analysis
Gori and Tesi	1992	Modified BP	Gradient Descent	Linear and Non-Linear Dataset
Adeli and Hung	1994	Conjugate gradient Algorithm	Gradient Descent	Engineering and Image Classification Benchmarks
Hugan and Menhaj	1994	MA	Gradient Descent	Function Approximation
Mendes *et al.*	2002	PSO	Stochastic	Regression problem
Llonen *et al.*	2003	DE	Stochastic	Classification problem
Gudise *et al.*	2003	PSO	Stochastic	Convergence Analysis
Ng *et al.*	2003	BP with modified gradient function	Gradient descent	XOR, 3- bit counting Parity bit and Regression
Blum *et al.*	2005	ACO	Stochastic	UCI ML Datasets
Zhang *et al.*	2007	PSOBP	Gradient Descent and Stochastic	Function Approximation and Classification Benchmarks
Karaboga *et al.*	2007	ABC	Stochastic	XOR, parity and Enc-Dec
Slowik *et al.*	2008	DE	Stochastic	Weight and Bias Adjustment
Ozturk *et al.*	2011	ABC-LM	Stochastic and Gradient Descent	XOR, 3-parity and 4-Enc-Dec
Green *et al.*	2011	CFO	Stochastic	XOR, Iris
Yu *et al.*	2011	CRO	Stochastic	Iris, Breast Cancer, Diabetes
Mirjalili *et al.*	2011	MOA	Stochastic	XOR, Sine function
Moallem *et al.*	2012	IWO	Stochastic	Image Segmentation Problem
Mirjalili *et al.*	2012	PSOGSA	Stochastic	Classification and Function Approximation
Pereira *et al.*	2013	CSS	Stochastic	Non-Technical Loss Problem
Pereira *et al.*	2014	SSO	Stochastic	Parkinson Disease Identification
Uzlu *et al.*	2014	TLBO	Stochastic	Import, Export, and Population
Mirjalili *et al.*	2014	BBO	Stochastic	Classification and Function Approximation
Mirjalili *et al.*	2015	SSO	Stochastic	5 Classification Datasets
Chen *et al.*	2015	PSOCS	Stochastic	Function Approximation and Classification Datasets
Mirjalili	2015	GWO	Stochastic	Classification and Function Approximation
Faris *et al.*	2016	LSA	Stochastic	16 Classification Datasets
Wu *et al.*	2016	SOS	Stochastic	Pattern Recognition Problem
Faris *et al.*	2016	MVO	Stochastic	9 UCI Medical Datasets
Alijarah *et al.*	2016	WOA	Stochastic	20 Classification Datasets
Kowalski *et al.*	2016	KH	Stochastic	Classification Datasets
Faris *et al.*	2017	MFO	Stochastic	Pattern Recognition
Alboaneen *et al.*	2017	GSO	Stochastic	XOR, Breast Cancer, Liver Disorder, Vertebral Column
Faris *et al.*	2017	MBO	Stochastic	12 UCI Datasets
Amirsadri *et al.*	2017	GWOBP	Gradient Descent and Stochastic	Classification and Function Approximation Datasets
Li	2018	Improved PSO	Stochastic	Classification Datasets
Li *et al.*	2018	SHOA	Stochastic	SPECT Heart Dataset
Alijarah *et al.*	2018	BBO	Stochastic	12 Classification Datasets
Bui *et al.*	2018	GSA, GWO, and BBO	Stochastic	Forest Fire prediction Problem
Pu *et al.*	2018	CBBO	Stochastic	4 Pattern Recognition Datasets
Heidari *et al.*	2018	GOA	Stochastic	Pattern Recognition
Itano *et al.*	2018	GA	Stochastic	Classification Datasets
Mosavi et al.	2019	GSABP	Gradient Descent and Stochastic	Function Approximation and Classification Benchmarks
Tarkhaneh *et al.*	2019	PSO	Stochastic	Pattern Recognition Datasets
Alijarah *et al.*	2019	BSO	Stochastic	3 Function Approximation and 13 Classification Datasets
Zhao *et al.*	2019	MSHO	Stochastic	Classification Problem

Table I.
Summary of the Literature related to the training of MLP by gradient descent and HAs

hidden layer, and an output layer. In fact, MLP is the most commonly used and famous type of MLP. It consists of at least one input layer, one or more hidden layer(s) and an output layer. A simple model of MLP is shown in Figure 1.

In Figure 1, it can be noted that "n" is the number of input vertices, "h" denotes hidden vertices, and "m" represents the number of output vertices. Generally, the fitness function is used to calculate the error in the actual and desired output of MLP. Here, the fitness function is mathematically calculated as follows:

The first two layers of MLP consist of weights and biases. The weighted (w) sum of inputs (x) is mathematically represented in Equation (1).

$$s_p = \sum_{i=1}^{n} w_{ip}.x_i - \phi_p,\ p = 1, 2, \ldots h \tag{1}$$

In Equation (1), ϕ_p is the bias of the hidden vertex of the MLP.

It has been reported that hidden vertices are important for increasing the result accuracy in MLP. Normally, sigmoid function acts as activation function in MLP and is calculated as:

$$S_p = \text{sigmoid}(s_p) = \frac{1}{1 + \exp(-s_p)},\ p = 1, 2, \ldots, h \tag{2}$$

Now, the actual output of MLP is mathematically shown in Equation (4).

$$o_l = \sum_{t=1}^{h} w_{pl} \cdot s_p - \phi_l, 1 = 1, 2, \ldots, m \tag{3}$$

$$O_l = \frac{1}{1 + \exp(-o_l)},\ 1 = 1, 2, \ldots, m \tag{4}$$

In Equation (4), ϕ_l is the threshold of the output layer.

It is obvious from Equations (1) and (3) that the weights and biases are pivotal components of MLP. In fact, the sole goal of MLP training is to find the best results for connection weights and biases. In this paper, CPSOGSA has been utilized to achieve the optimal values for weights, biases and activation function by training MLP.

5. Constriction coefficient based PSO

In PSO, the candidate solutions are in the form of particles. It consists of three important operators which are inertia weight (w), pbest (personal particle best) and gbest (global particle best). The inertia weight provides diversification (exploration) and (pbest and gbest) gives intensification (exploitation) power to PSO, respectively. The mathematical equations of position and velocity of the swarm particles are given in Equations (5) and (6).

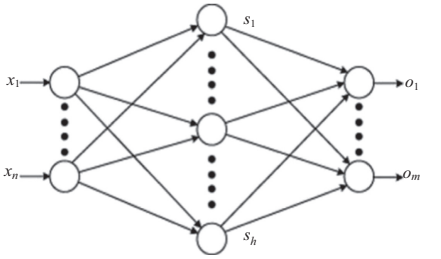

Figure 1.
Basic 2 layer model of MLP

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{5}$$

$$v_i^d(t+1) = w(t)\cdot v_i^d(t) + c_1 r_{i1}\left(\text{pbest}_i - x_i^d(t)\right) + c_2 r_{i2}\left(\text{gbest} - x_i^d(t)\right) \tag{6}$$

where (c_1, c_2) are cognitive and social constants and (r_{i1}, r_{i2}) are the probabilistic numbers in closed interval [0,1].

The constriction coefficient based PSO was mathematically formulated by Clerc and Kennedy (2002) after doing rigorous algebraic and Eigenvalue (search procedure) analysis of the classical PSO. The main goal of CPSO is to control the particles from going outside the search space and thus, increasing the convergence of the particles towards pbest and gbest. The various CPSO parameters and their values are given as:

$$\varphi_1 = 2.05,\ \varphi_2 = 2.05,\ \varphi = \varphi_1 + \varphi_2$$

$$K = 2/\left(\varphi - 2 + \text{sqrt}\left(\varphi^2 - 4\right)\right) \tag{7}$$

where $(\varphi, \varphi_1, \varphi_2)$ are control parameters and their values are important for controlling the particle trajectory. Moreover, "K" is the constriction coefficient which is pivotal for particle convergence towards the global optimum.

Also, when $w(t) = K$, $c_1 = K\ \varphi_1$ (personal learning coefficient) and $c_2 = K\ \varphi_2$ (global learning coefficient); the velocity Equation (6) in modified form is given in equation (8).

$$v_i^d(t+1) = \left(2/\left(\varphi - 2 + \text{sqrt}(\varphi^2 - 4\varphi)\right)v_i^d(t) + K\varphi_1 r_{i1}\left(\text{pbest}_i - x_i^d(t)\right) + K\varphi_2 r_{i2}\left(\text{gbest} - x_i^d(t)\right)\right) \tag{8}$$

Equation (8) shows the velocity of the particles is inversely proportional to the control parameter, φ. Also, the value of φ must be greater than 4 ($\varphi > 4$) for maintaining the stability in the particle system.

6. Gravitational search algorithm (GSA)

According to modern physics, nature is composed of four forces such as the gravitational force, the strong nuclear force, the electromagnetic force and the weak nuclear force. As per classical Newtonian mechanics, gravitational law is stated as "the gravitational force between two masses is directly proportional to the product of their masses and inversely proportional to the square of the distance between them" (Halliday *et al.*, 2000; Rashedi *et al.*, 2009; Rather *et al.*, 2019a, 2019b).

Let $X = \{x_1, x_2, x_3 \ldots, x_n\}$ be a system with "n" agents, such that $x_i \in R^n$. The force between mass i and j is shown in Equation (9).

$$F_{ij} = G(t)\frac{m_{pi}(t)m_{aj}(t)}{R_{ij}(t) + \in}\left(x_j^d(t) + x_i^d(t)\right) \tag{9}$$

Here, $m_{pi}(t)$ is passive gravitational mass and $m_{aj}(t)$ is active gravitational mass. Moreover, $R_{ij}(t)$is the Euclidian distance and $\in$ is a positive constant in d^{th} dimensional space.

The gravitational constant "G" is important for adjusting the accuracy of the search and is given by Equation (10).

$$G(t) = G(t_0)\, e^{\left(-\alpha \frac{CI}{MI}\right)} \tag{10}$$

where $G(t)$ is the gravitational constant with respect to time interval t, α is a coefficient which decreases with time, CI is the current iteration, MI is the maximum number of iterations, and $G(t_0)$ indicates the initial value of the gravitational constant.

Moreover, the calculation of total force exerted on the system due to masses can be calculated using Equation (11).

$$F_i^d(t) = \sum_{j=1, j \neq i}^{m} \gamma_j F_{ij} \tag{11}$$

where $\gamma_j$belongstto [0 1].

According to the second law of motion, "The acceleration ($a_i^d(t)$) of the agent is directly proportional to the force ($F_i^d(t)$) applied by the agent and inversely proportional to the mass (m_i(t)) of the agent." It is calculated using Equation (12).

$$a_i^d(t) = \frac{F_i^d(t)}{m_i(t)} \tag{12}$$

To find the global optimum, it is important to calculate the position and velocity of the heavy mass. These can be represented mathematically as in Equations (13) and (14), respectively.

$$v_i^d(t+1) = \gamma_j v_i^d(t) + a_i^d(t) \tag{13}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{14}$$

7. CPSOGSA: hybrid heuristic algorithm

The main purpose of hybrid CPSOGSA is to use the exploration capabilities of the gravitational search algorithm (GSA) and convergence power of constriction coefficient based particle swarm optimization (CPSO) (Rather *et al.*, 2019c). Moreover, randomization is necessary for preventing CPSOGSA from getting stuck in local optima and intensification increases possibilities in attracting optimal candidate particles towards the global minimum. Hence, CPSOGSA supports both randomization and intensification which are necessary to solve an optimization problem. The merging equation that combines both the algorithms is given in Equation (15).

$$\begin{aligned} V_i^d(t+1) = \Big(2 \Big/ \Big(\varphi - 2 + \mathrm{sqrt}(\varphi^2 - 4)\Big) V_i^d(t) + K\varphi_1 r_{i1}\big(a_i^d(t) - x_i^d(t)\big) \\ + K\varphi_2 r_{i2}\big(\mathrm{gbest} - x_i^d(t)\big)\Big) \end{aligned} \tag{15}$$

where V_i^d is the velocity of the swarm particles, $<\varphi, \varphi_1, \varphi_2>$ are control parameters, K is the constriction coefficient, a_i^d is the acceleration of the particles and gbest represents the social capability component of the particle system.

The position of the particles is given by Equation (16).

$$X_i^d(t+1) = X_i^d(t) + V_i^d(t+1) \tag{16}$$

The pseudo-code of CPSOGSA is as follows:

Algorithm 1. CPSOGSA Algorithm

Begin
Initialize the search space randomly in an 'n' particle system as:
$X = \{x_1, x_2, x_3 \ldots, x_n\}$
$V = \{v_1, v_2, v_3 \ldots, v_n\}$
While (all particles update position) **do**
Update the Gravitational Constant, G (t) using Equation (10).
Find the Gravitational force, $F_i^d(t)$ by utilizing Equation (11).
Calculate the Acceleration, $a_i^d(t)$ with the help of Equation (12).
Update the Velocity: $V_i^d(t+1)$ by making use of Equation (15).
Update agent Positions: $X_i^d(t+1)$ using Equation (16).
Until the completion of the maximum number of iterations
End While
Return the best candidate solution
End

The flowchart of CPSOGSA hybrid algorithm is shown in Figure 2.

8. CPSOGSA for training MLP

In MLP, weights and biases are pivotal variables. A learning algorithm should provide optimal values for these variables because their values are directly related to the accuracy of the MLP. During the training process of MLP, weights and biases are depicted in the form of candidate solutions (searcher agents), that is, encoding. In literature, there are three encoding strategies mentioned which are vector, binary, and matrix. When agents are encoded as vectors it is called vector encoding. Moreover, bit encoding deals with representing agents with 0 and 1-bit values. Furthermore, matrix encoding depicts agents in the form of rows and columns of a linear matrix.

There are three methods of using stochastic algorithms as trainers for MLP. The first method involves finding proper values for weights and biases to reduce threshold error. Here, the structure of the MLP is fixed. Moreover, HAs are also used to find the optimal structure for MLP. In addition, stochastic algorithms are utilized for parameter tuning such as momentum and training speed of MLPs.

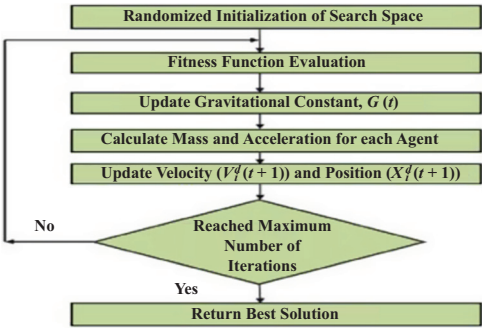

Figure 2.
Hybrid CPSOGSA algorithm

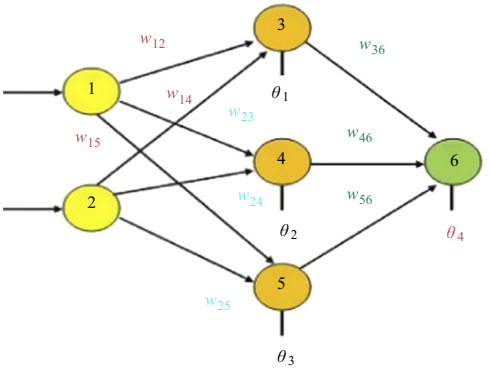

Figure 3.
MLP with 2 Inputs, 3 Hidden Layers, and 1 output Layer (2-3-1)

In this paper, matrix encoding has been employed to encapsulate weights and biases of MLP to CSPOGSA agents. The matrix encoding scheme can be explained by considering the MLP as shown in Figure 3.

$$\text{Particle}\,(:,:,\mathrm{i}) = \left[w_1, b_1, w_2^{'}, b_2\right] \tag{17}$$

$$w_1 = \begin{bmatrix} w_{13} & w_{23} \\ w_{14} & w_{24} \\ w_{15} & w_{25} \end{bmatrix},\ b_1 = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix},\ w_2^{'} = \begin{bmatrix} w_{36} \\ w_{46} \\ w_{56} \end{bmatrix},\ b_2 = [\theta_4] \tag{18}$$

where w_1 and w_2 are weight matrices of hidden and output layers, respectively. Moreover, $w_2^{'}$ is the transpose of w_2. Also, b_1 and b_2 are bias matrices of hidden and output layers.

The chief aim of MLP is to find the highest prediction accuracy. The standard performance metric for evaluating the efficiency of MLP is Mean Square Error (MSE). The MSE calculates the difference between the actual and desired output of MLP as shown in Equation (19).

$$MSE = \sum_{i=1}^{m} \left(o_i^l - D_i^l\right)^2 \tag{19}$$

where o is the actual output and D the desired output of the MLP for input i and lth training sample.

Besides, the efficiency of the MLP is related to its modeling of the entire set of training specimens. Therefore, the mean of MSE is calculated to cover all training feature vectors. It is mathematically represented as shown in Equation (20).

$$\text{Average(MSE)} = \sum_{l=1}^{R} \frac{\sum_{i=1}^{m}\left(o_i^l - D_i^l\right)^2}{R} \tag{20}$$

Here, "R" is the number of training feature vectors.

Hence, the problem of training MLP by CPSOGSA can be mathematically modeled as shown in Equation (21).

$$\text{Minimize}: F(\text{Particle}) = \text{Average(MSE)} \tag{21}$$

The whole process of MLP training using CPSOGSA is diagrammatically depicted in Figure 4.

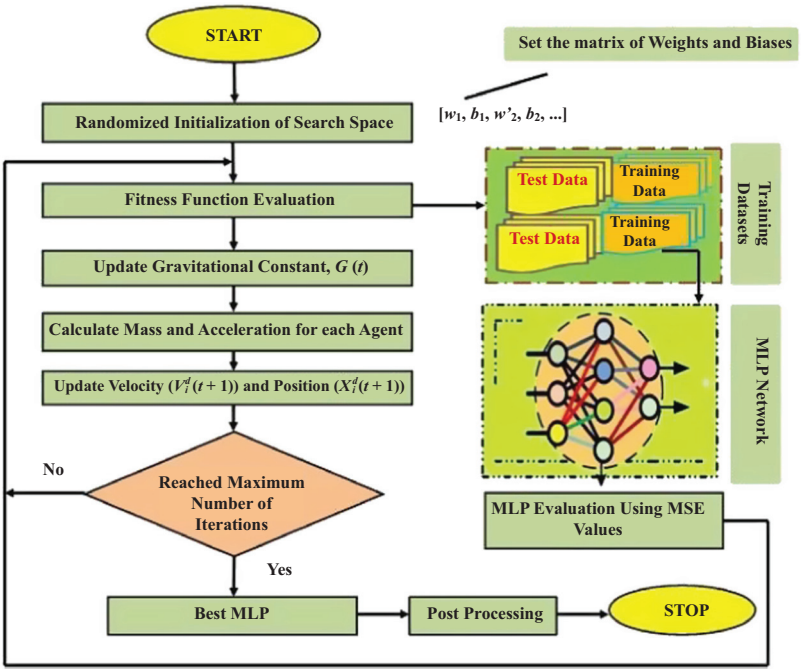

Figure 4.
Hybridization of MLP and CPSOGSA algorithm

The CPSOGSA has been applied to different classification and function benchmarks. The next section will present the experimental results of CPSOGSA and other HAs.

9. Results and discussion

The efficiency of the CPSOGSA algorithm has been investigated by utilizing classification and function approximation datasets. The UCI classification datasets (Blake *et al.*, 1998) including XOR, Iris, Heart, Balloon and Breast Cancer were used to test the recognition capability of CPSOGSA. Moreover, Sine, Sigmoid and Cosine function datasets will test the approximation power of CPSOGSA.

9.1 Experimental setup and data sets used

The CPSOGSA has been compared with eight state-of-the-art heuristic algorithms including Standard GSA (Rashedi *et al.*, 2009), PSO (Clerc and Kennedy, 2002), BBO (Mirjalili *et al.*, 2014), DE Llonen *et al.*, 2003, ACO (Socha and Blum, 2007), DA (Mirjalili, 2015), SCA (Mijalili, 2016) and SSA (Mijalili *et al.*, 2017) for overcoming stagnation in local minima problem, classification accuracy and error rate analysis. For a fair comparison, all the algorithms were initialized with the same number of searcher agents.

The experiment is conducted on a system having 64-bit operating system, 2.2 GHz Intel @ core processor and 4GB RAM. Further, the implementation of the algorithms and datasets has been done using MATLAB R2013a. The source codes are publicly available on Github, i.e. https//:github.com/SAJADAHMAD1.

The particles are initialized in a randomized way with a range of [−10, 10]. Besides, all algorithms are having the same population size of 20. The maximum iteration(s) is selected as a stopping criterion for getting a global minimum. Moreover, the maximum iterations are 100

for XOR, Sigmoid, Iris, and cosine datasets; 60 for Balloon while 50 for Breast Cancer, Heart and Sine datasets.

The specification of classification and function approximation datasets are provided in Table 2 and Table 3, respectively. In Table 2, it can be observed that the XOR dataset has the minimum number of training (8) and testing instances (8). On the other hand, the Heart dataset has the maximum number of attributes (22) and test instances (187). Moreover, Table 2 indicates that the sigmoid function has 61 instances for training MLP and 121 for testing MLP. The sigmoid dataset is a simple approximation dataset in terms of sample size while the Sine function dataset is difficult having 126 training and 252 testing instances.

The algorithms were repeated over 10 independent runs. The mean (or, average), standard deviation (SD), and median of 10 independent trials are reported. Further, the minimum value of mean and SD shows that the algorithm has more capability as compared to others in resolving stagnation in the local minima problem. It is not enough to calculate the mean and SD of algorithms over 10 independent runs to declare a particular algorithm as efficient (Derrac *et al.*, 2011). Rather, a statistical test should be conducted for a fair comparative analysis of HAs. In this paper, the non-parametric statistical test namely the Wilcoxon Rank sum test has been carried out at a 5% significance level. In this test, p-values are calculated for every algorithm. Generally, p-values less than 0.05 are enough to disprove the null hypothesis. When the particular algorithm has minimum mean and SD values, it shows it will not be compared with itself and is represented in tables by N/A, i.e. "Not Applicable." In addition, the best MSE values are highlighted in a italic font and p-values greater than 0.05 are underlined to indicate the competitiveness of the algorithm with the best one. In short, the null hypothesis consists of an algorithm having minimum values for mean and standard deviation i.e. p-values less than 0.05. On the other hand, the alternate hypothesis consists of the algorithm(s) having p-values greater than 0.05. Simply mean and standard deviation values are close to the best performing algorithm.

The other performance metrics reported in the simulation results are average classification and test error rates. Besides, the datasets having different ranges for attributes are normalized using min-max normalization. It is mathematically formulated as shown in Equation (22).

$$Y' = \frac{(y-p)\times(s-r)}{(q-p)} + r \tag{22}$$

such that y belongs to the closed interval from $[a,b]$ to $[c,d]$.

Table 2. UCI repository datasets for classification

Dataset	Attribute	Training sample	Test sample	Class	MLP structure
XOR	3	8	8	2	3-7-1
Baloon	4	16	16	2	4-9-1
Iris	4	150	150	3	4-9-3
Breast Cancer	9	599	100	2	9-19-1
Heart	22	80	187	2	22-45-1

Table 3. Description of function approximation datasets

Dataset	Training sample	Test sample	MLP structure
Sigmoid : $q = \frac{1}{(1+e^{-p})}$	61: $p \in [-3 : 01 : 3]$	121: $p \in [-3{:}0.05{:}3]$	1-15-1
Cosine : $q = \left(\cos\left(\frac{p\pi}{2}\right)\right)^7$	31: $p \in [1.25{:}0.05{:}2.75]$	38: $p \in [1.25{:}0.04{:}2.75]$	1-15-1
Sine: $q = \sin(2p)$	126: $p \in [-2\pi : 0.1 : 2\pi]$	252: $p \in [-2\pi : 0.05 : 2\pi]$	1-15-1

Another important element for getting proper simulation results is the structure of MLP. The number of hidden nodes (h) is calculated by using Equation (23).

$$h = 2 \times n + 1 \tag{23}$$

where n is the number of MLP input(s).

All optimization algorithms are first initialized before going through the various phases of iterations. They have different parameters that should be first initialized. Table 4 shows the initial values of 9 participating algorithms in this study. The initial values of all algorithms are according to their traditional default versions, i.e. values used by the authors who have proposed the algorithms. Let us take the example of PSO. Table 4 depicts C1, C2, Wmax, and Wmin parameters of PSO having values of 2, 2, 0.9, and 0.2, respectively. Every researcher in the swarm optimization field knows that personal and social constants of PSO have a value of 2 because it shows more suitable results on these values. Similarly, parameters for other algorithms were selected considering the stability and fairness of the experimental setup.

The remaining part of the paper deals with statistical and simulation analysis of the UCI datasets and function approximation benchmarks by employing nine different HAs including CPSOGSA.

9.2 Classification problems

The classification benchmarks were selected in increasing levels of complexity meaning if a benchmark is having more number of features then it is difficult for an algorithm to classify the attributes of a dataset. In simpler terms, the size of MLP is directly related to the number of inputs (weights and biases). Besides, for a larger MLP, it is quite difficult for an algorithm to find correct values for weights and biases. So, the XOR dataset is the simplest recognition benchmark; while the Heart dataset is the strenuous classification benchmark. The simulation results of various UCI classification datasets are presented and discussed in the following sub-sections.

9.2.1 XOR dataset. It is a non-linear separable problem which makes it impossible to solve without using a hidden layer(s). The main goal of the XOR problem is to find the number of 1's in the input. In a simple XOR truth table, if the number of 1's is odd, then the output is 1 else 0 as shown in Table 5.

Algorithm	Name of the parameter	Value of the parameter
GSA	Elitist Check (Number of Fittest Agents after Stopping Criterion)	1
	Rpower (Exponent of Distance between Agents)	1
	Min_flag (1: Minimum; 0: Maximum)	1
PSO	C_1, C_2 (Personal and Social Constants)	2
	W_{max} (Maximum Inertia Weight)	0.9
	W_{min} (Minimum Inertia Weight)	0.2
CPSOGSA	φ_1, φ_2 (Control Parameters)	2.05
ACO	Pheromone Update Constant	1
	Initial Pheromone	10
	Pheromone Sensitivity	0.3
	Visibility Sensitivity	0.1
DE	Lower bound of Scaling factor	0.2
	Upper Bound of Scaling factor	0.8
	PCR (Crossover Probability)	0.8
BBO	nKeep (Number of Habitats retained after every Generation)	0.2
	Pmutation (Mutation Probability)	0.9
DA	β (Constant)	1.5
SCA	a (Constant)	2
SSA	c_2, c_3 (Random Numbers)	[0,1]

Table 4. Initialization parameters of algorithms

The MLP structure for the XOR problem is 3-7-1 i.e. 3 inputs, 7 hidden layers, and 1 output. The statistical results are reported in Table 6. It can be seen that PSO provides minimum values for average and SD. Moreover, DE, SSA and CPSOGSA have also competitive values for MSE. It indicates that BBO, DE, SSA and CPSOGSA have the capability to resolve stagnation in local minima issue of MLP. In addition, average classification rates of DE and CPSOGSA are very close while BBO has the highest classification rate (CR) of 76.25%. Besides, the execution time of participating algorithms is reported in which SCA takes less time for getting optimal values while DA takes maximum time. However, the execution time of CPSOGSA, PSO, GSA, SCA, and SSA is also close to each other indicating good exploitation power.

Further, the convergence curves of participating algorithms are shown in Figure 5. It can be clearly observed that SSA and BBO have higher convergence speed than PSO while DE and CPSOGSA curves overlap with each other indicating close values for MSE. The *p*-values also confirm the efficient performance of SSA and BBO over PSO. To further validate the results, box plots were taken for all participating algorithms. Figure 6 box plots depict that PSO has minimum values for lower quartile, median and upper quartile while ACO has large values for MSE indicating its sub-optimal performance.

9.2.2 Balloon dataset. This is one of the simplest datasets available in the UCI repository having four features including act, size, color, and age. The output of the dataset involves balloon inflating. The output is 1 if balloon inflates otherwise 0. Moreover, the MLP structure for the balloon dataset is 4-9-1 meaning 4 inputs, 9 hidden nodes, and 1 output layer.

Table 7 shows the statistical results of the balloon dataset. It may be observed that PSO provides minimum values for MSE. Besides, SSA, DE, and CPSOGSA also show impressive values for average and SD. Further, CPSOGSA gives a high classification rate of 78.5% only behind BBO. It indicates that CPSOGSA has the capability to resolve local minima problems.

The convergence curves are depicted in Figure 7. It is clearly evident that PSO has the steepest curve showing the high speed of convergence while CPSOGSA also indicates

Input	Output
0 0 0	0
0 0 1	1
0 1 0	1
0 1 1	0
1 0 0	1
1 0 1	0
1 1 0	0
1 1 1	1

Table 5.
3-bit XOR truth table

Algorithm	Best	Worst	Average	SD	Median	CR	*p*-values	Exec. Time
GSA	0.13075	0.24259	0.19875	0.037184	0.2044	3.75%	0.001953	6.1299
PSO	2.76e-09	0.0627	*0.0063*	*0.0198*	1.69e-05	*0%*	N/A	7.3583
CPSOGSA	0.00434	0.18781	*0.09400*	*0.066473*	0.07638	*31.25%*	0.001953	7.682
ACO	0.5	0.5	0.5	0	0.5	50%	0.001953	19.5911
BBO	3.411e-06	0.18766	0.03993	0.06649	0.00134	76.25%	0.19335	11.7511
DE	0.011614	0.08079	0.044064	0.022422	0.04194	40%	0.037109	5.5833
DA	0.25048	0.36012	0.30768	0.04111	0.30798	0%	0.001953	21.2589
SCA	0.0075	0.16565	0.11168	0.0525	0.13044	0%	0.001953	5.1481
SSA	7.15e-08	0.1916	0.04517	0.0612	0.02066	0%	0.130859	5.1932

Table 6.
Simulation Results of XOR classification Dataset

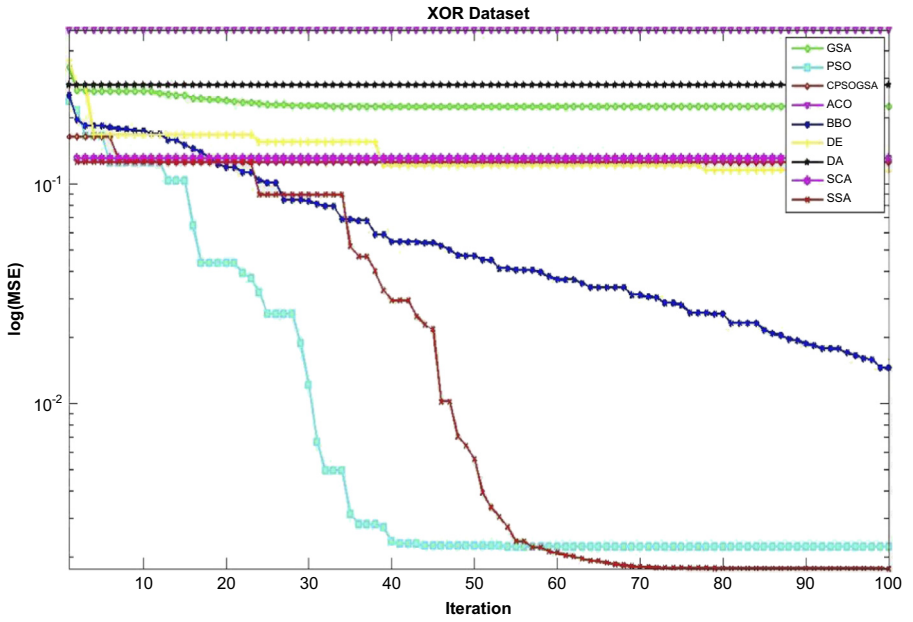

Figure 5.
Convergence curve for XOR problem

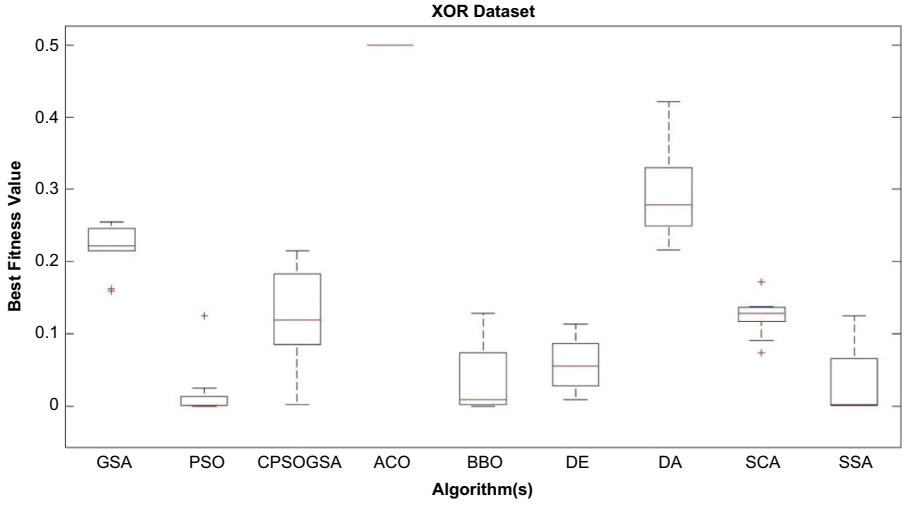

Figure 6.
Box plot of XOR classification problem

appreciable performance in finding a global minimum. Besides, Figure 8 shows box plots of all participating HAs. In fact, CPSOGSA, BBO, and DE are at the same level, i.e. their MSE values are approaching zero while CPSOGSA, SCA and SSA showing some outliers. Moreover, GSA has somewhat bigger values for both quartiles and median. On the other hand, ACO, DA and GSA provide large values for MSE and show sub-optimal performance as compared to other algorithms.

9.2.3 Iris dataset. This dataset is one of the popular and challenging classification benchmarks. It consists of 150 instances and three classes, i.e. setosa, versicolor and virginica. The MLP structure used for iris dataset is 4-9-3.

Table 8 provides simulation results of HAs in which BBO, CPSOGSA, SCA, SSA and GSA have close values for average, SD and median. Moreover, the classification rate of all algorithms is zero because the calculations were taken on a computer having less processing power. In fact, it is recommended to increase the number of searcher agents and the maximum number of iterations while using a high-speed computer. Clearly, it will increase the accuracy of MSE values of existing optimal algorithms like PSO, BBO and CPSOGSA. On the other side; it will provide an improvement in the recognition rates of HAs. Besides, *p*-values indicate that MSE values of CPSOGSA, DE, SCA, SSA, and GSA are statistically significant as compared to PSO because they have *p*-values greater than 0.05 i.e. 0.845703, 0.322265, 0.625000, 0.130859 and 0.375000, respectively.

The convergence curves of HAs are shown in Figure 9 indicating CPSOGSA having a high exploitation rate as compared to ACO, DA, and PSO while GSA, BBO, SCA, and SSA also have appreciable convergence speed. Furthermore, box plots are shown in Figure 10

Algorithm	Best	Worst	Average	SD	Median	CR	*p*-values	Exec. Time
GSA	0.07287	0.24712	0.13706	0.053347	0.11945	16%	0.001953	14.9058
PSO	1.77e-14	2.35e-07	*2.414e-8*	*7.44e-08*	3.62e −10	*0%*	N/A	14.8567
CPSOGSA	1.878e-06	0.10844	*0.015461*	*0.033976*	0.00143	*78.5%*	0.001953	16.4368
ACO	0.6	0.6	0.6	1.170e-16	0.6	40%	0.001953	29.7848
BBO	1.46e-10	2.27e-6	2.426e-7	7.152e-7	4.47e-9	100%	0.921875	22.9508
DE	1.803e-6	0.00483	7.963e-4	0.001520	0.00197	73%	0.001953	14.5402
DA	0.0473	0.308	0.1792	0.0789	0.1724	0%	0.001953	27.2529
SCA	3.63e-07	0.047	0.0128	0.0161	0.0058	0%	0.001953	13.9575
SSA	2.78e-08	0.003	0.0004	0.0009	1.81e-05	0%	0.001953	13.8013

Table 7.
Simulation Results of Balloon classification Dataset

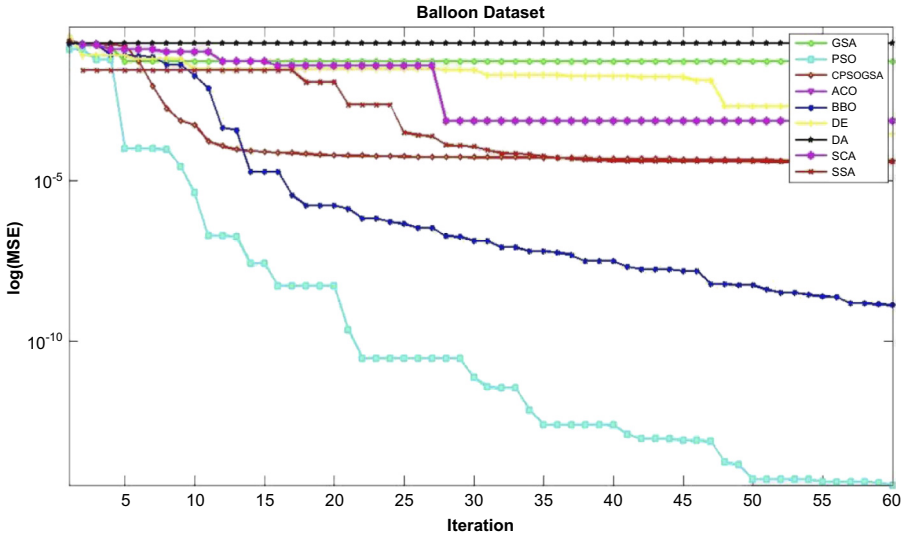

Figure 7.
Convergence curve for balloon problem

depicting the sub-optimal performance of ACO, DA, and SCA while CPSOGSA, SSA, and BBO show efficient results for average MSE having minimum median values and no outliers.

9.2.4 Breast cancer dataset. The data for this classification benchmark was collected at the University of Wisconsin, USA. It consists of 599 instances, 9 attributes, and 2 classes. In this dataset, cancers are categorized into benign and malignant. When the MLP gives 2 as output, it represents benign cancer and 4 indicates malignant cancer. The architecture of MLP employed for breast cancer dataset is 4-9-3. In fact, this dataset has more complexity and difficulty level as compared to previous datasets. So, if an algorithm provides good results for this dataset; it depicts a high level of diversification power.

Moreover, Table 9 shows the convergence curves of algorithms. It is clearly evident that CPSOGSA has efficient results for all statistical measures, i.e. average, SD, and median while GSA provides second-best MSE results for breast cancer dataset. Further, CPSOGSA gives an absolute recognition rate of 100% and GSA shows 76.2%. Besides, the Wilcoxon rank-sum test (p-values) also confirms the best performance of the CPSOGSA algorithm as all p-values are less than 0.05. However, CPSOGSA takes somewhat more execution time than other HAs.

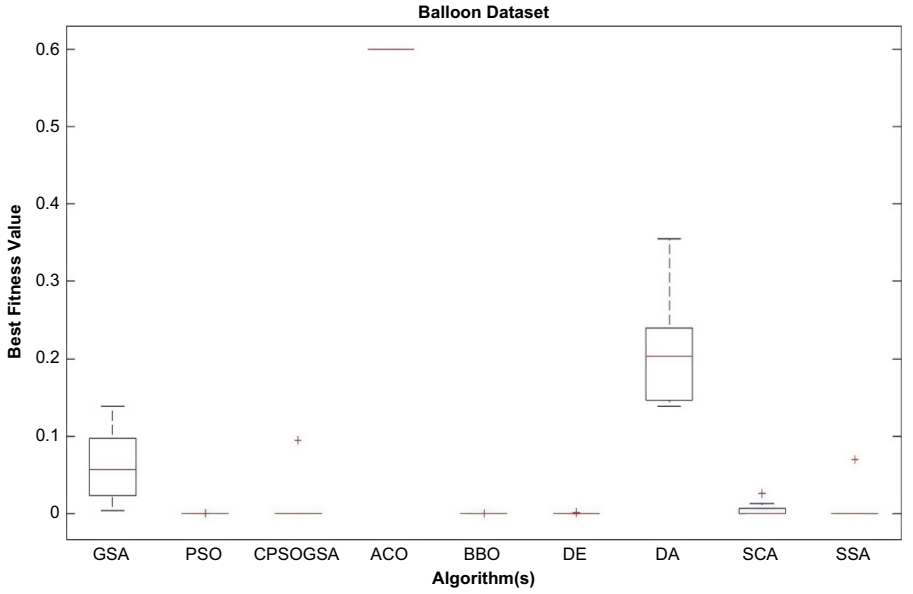

Figure 8.
Box plot of balloon classification problem

Algorithm	Best	Worst	Average	SD	Median	CR	p-values	Exec. Time
GSA	0.68791	0.77607	0.71715	0.02673	0.7147	0%	0.375000	258.5619
PSO	0.6671	0.7540	*0.7015*	*0.0291*	0.6989	0%	N/A	261.5995
CPSOGSA	0.68083	0.85908	*0.77038*	*0.06155*	0.77482	0%	0.845703	246.5869
ACO	1.7094	1.7725	1.7436	0.01763	1.7414	0%	0.001953	281.8333
BBO	0.67162	0.79227	0.71403	0.03862	0.7063	0%	0.027343	252.5695
DE	0.74389	0.83701	0.77415	0.02792	0.7695	0%	0.322265	242.0731
DA	0.8757	0.9775	0.9138	0.03596	0.8955	0%	0.001953	251.6013
SCA	0.6785	0.8263	0.7438	0.0483	0.7353	0%	0.625000	253.4301
SSA	0.6692	0.8889	0.7015	0.0673	0.6778	0%	0.130859	230.9610

Table 8.
Simulation Results of Iris classification Dataset

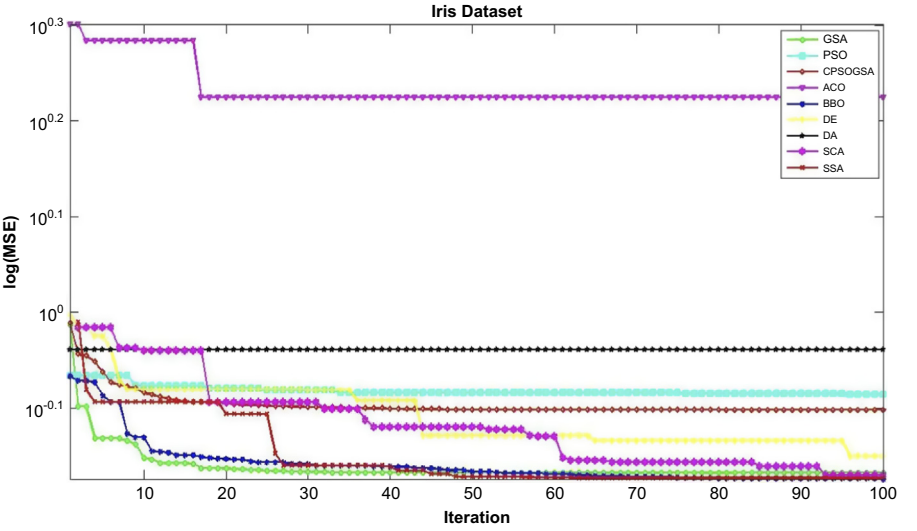

Figure 9.
Convergence curve for iris problem

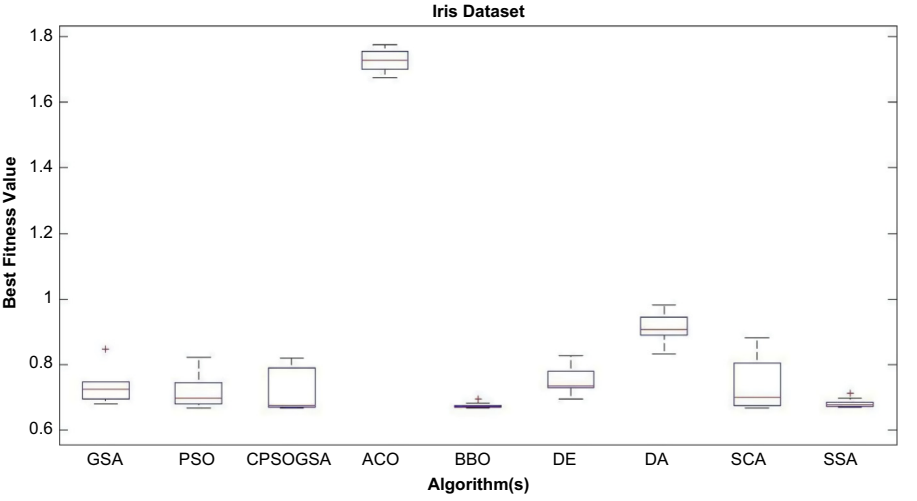

Figure 10.
Box plot of iris classification problem

Algorithm	Best	Worst	Average	SD	Median	CR	*p*-values	Exec. Time
GSA	0	0.01509	*0.001509*	*0.0047749*	0	*76.2%*	0.001953	375.3096
PSO	0.01902	0.0426	0.0284	0.0084	0.0288	0%	0.0019531	364.503
CPSOGSA	0	0.00518	*0.000518*	*0.0016406*	0	*100%*	N/A	395.1375
ACO	0.66316	0.66316	0.66316	1.1703e-16	0.66316	0%	0.001953	505.1518
BBO	0.01005	0.02865	0.01878	0.0059462	0.01748	16.2%	0.005859	394.1052
DE	0.03558	0.04330	0.03852	0.0023677	0.03778	4%	0.001953	387.7554
DA	0.04185	0.04849	0.04644	0.00230	0.04741	0%	0.001953	364.5036
SCA	0.033689	0.04424	0.03835	0.0038225	0.038322	0%	0.001953	348.9934
SSA	0.0070087	0.03786	0.01776	0.010077	0.015105	0%	0.001953	342.778

Table 9.
Simulation Results of Breast Cancer classification Dataset

Figure 11 shows the convergence curves of HAs. It can be seen that CPSOGSA has the highest convergence speed showing its capability of providing optimal values for connection weights and biases. Furthermore, box plots are presented in Figure 12 indicating minimum values of lower and upper quartiles for CPSOGSA while ACO, DA and DE have large values for both quartiles.

9.2.5 Heart dataset. The main purpose of designing this dataset was to correctly interpret STEM images for diagnosing heart disease patients. It consists of 22 attributes, 80 instances and 2 classes. The attributes are encoded in binary format in which 1 represents the patient is normal while 0 portrays the patient is abnormal. The architecture of MLP considered for the heart dataset is 22-45-1.

The simulation results are reported in Table 10. Clearly, BBO and CPSOGSA have the best values for average MSE. Moreover, BBO, DE, and CPSOGSA have high recognition rates

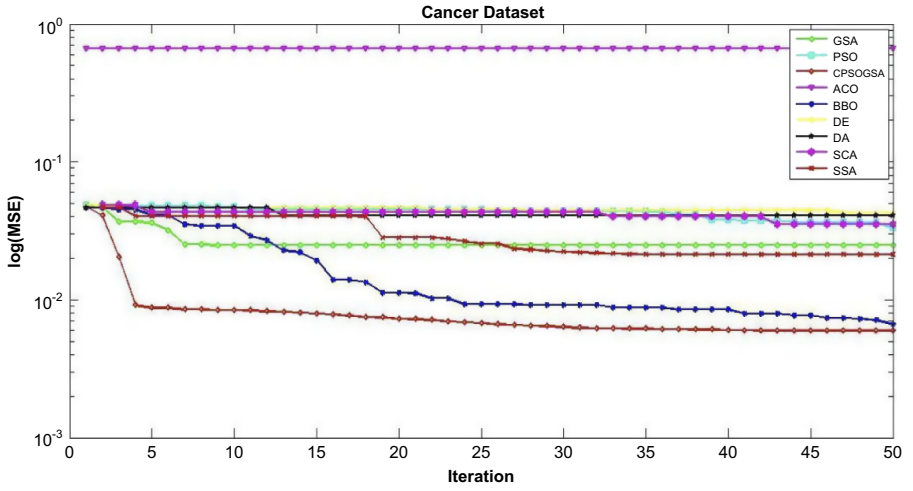

Figure 11.
Convergence curve for breast cancer problem

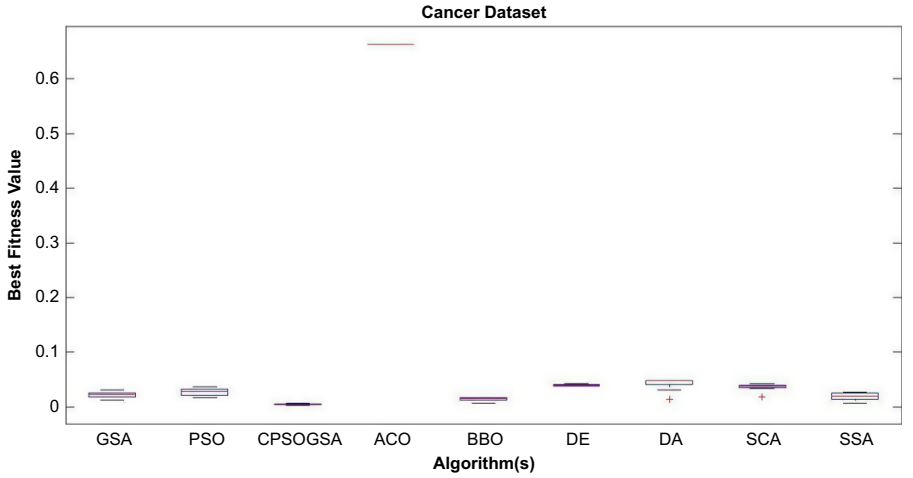

Figure 12.
Box plot of breast cancer classification problem

while DA, SCA and SSA have the same average classification rate of 50%. However, *p*-values indicate the statistically significant performance of CPSOGSA and PSO as compared to BBO.

Further, Figure 13 shows the high convergence speed of PSO, BBO, and CPSOGSA in finding a global minimum. Furthermore, box plots in Figure 14 convey that CPSOGSA and BBO have close MSE values while GSA, ACO, DA, SCA, and DE have large values for the median, lower and upper quartiles.

It is evident from the summarization of classification benchmark results that CPSOGSA shows the best results for breast cancer and heart datasets. Moreover, the average recognition rate of CPSOGSA is best for breast cancer and the second most for iris datasets. Besides, the convergence speed attained by CPSOGSA is more in the iris, breast cancer, and heart classification benchmarks. Additionally, it is evident that PSO, BBO, and SSA also provide good simulation results for classification datasets.

In the next section, HAs will be applied to function approximation datasets for handling complex search spaces.

9.3 Function approximation problems

MLPs are trained by evolutionary algorithms to generate the approximate curves for three 1- dimensional functions (sigmoid, cosine and sine). The architecture of MLPs employed for estimating the values of functions is 1-15-1 meaning 1 input, 15 hidden nodes, and 1 output. Moreover, the difficulty level of a dataset is directly related to the training data of the benchmark. In fact, function approximation datasets were selected based on the complexity

Table 10.
Simulation Results of Heart classification Dataset

Algorithm	Best	Worst	Average	SD	Median	CR	*p*-values	Exec. Time
GSA	0.22991	0.32575	0.28066	0.03377	0.28248	46.25%	0.001953	133.9812
PSO	0.08814	0.17631	0.12005	0.02693	0.11902	0	0.1933593	101.5754
CPSOGSA	0.12639	0.28295	*0.1764*	*0.0534*	0.15289	*62.5%*	0.845703	147.681
ACO	0.5	0.5	0.5	0	0.5	50%	0.001953	2193.9785
BBO	0.088232	0.15915	*0.11964*	*0.024412*	0.11607	*8.5%*	N/A	205.6027
DE	0.17413	0.22725	0.21137	0.016534	0.21934	64.5%	0.001953	101.8787
DA	0.22626	0.38126	0.33525	0.048395	0.33909	50%	0.0019531	282.5446
SCA	0.20872	0.25399	0.2365	0.017038	0.24395	50%	0.0019531	106.993
SSA	0.11083	0.23781	0.17414	0.040455	0.17278	50%	0.019531	110.8564

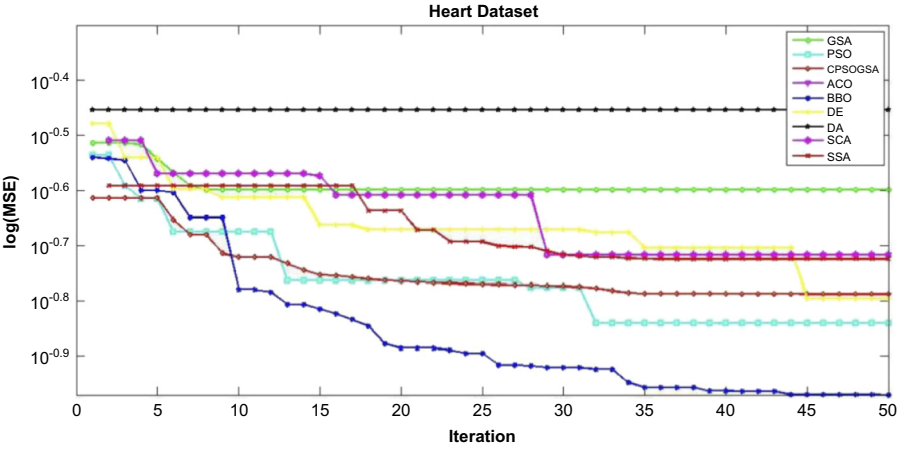

Figure 13.
Convergence curve for heart classification problem

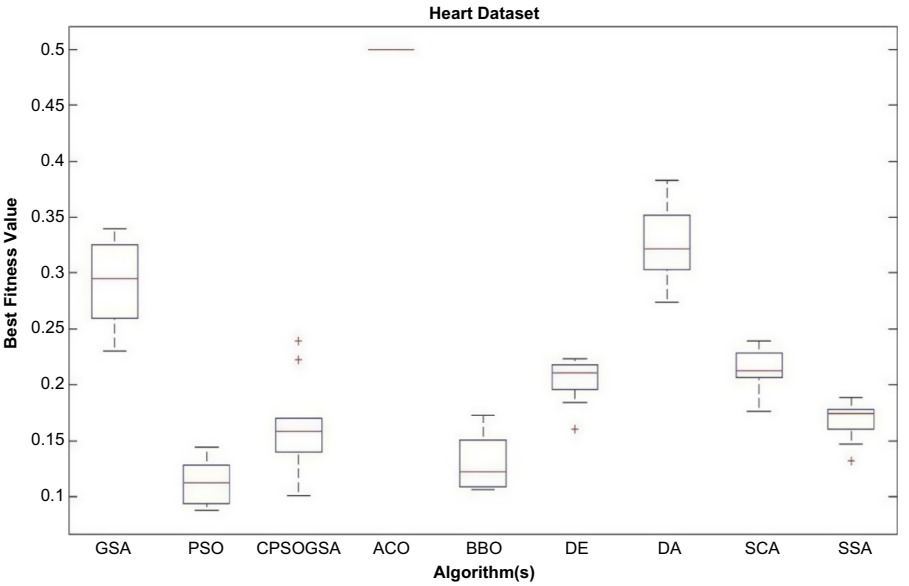

Figure 14.
Box plot of heart classification problem

in which sigmoid function is simple to approximate having only 61 instances while sine function is the most strenuous to approximate as it consists of 126 training samples. So, it will be quite interesting to see how HAs will train MLP to handle instance complexity of function approximation benchmarks.

9.3.1 Sigmoid function (SF) dataset. The sigmoid function is basically an exponential function which lies in the closed interval of $[-3, +3]$ having 61 data instances. The MLP structure considered for the SF benchmark is 1-15-1. Besides, Figure 15 depicts the sigmoid function graphically.

Table 11 presents statistical results of HAs for the SF dataset in which average MSE values of CPSOGSA, PSO, and BBO are close to each other while SD values of CPSOGSA are lower than BBO. It indicates that CPSOGSA and PSO are the best MLP trainer(s) for SF benchmark. Moreover, *p*-values also indicate that CPSOGSA is statistically significant than BBO. Besides, CPSOGSA also has minimum average test error (ATE) portraying capability in handling local minima entrapment problem.

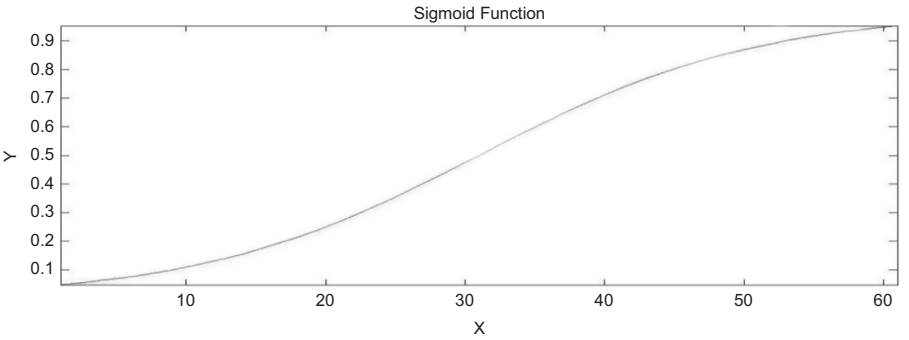

Figure 15.
Sigmoid function

In addition, Figures 16 and 17 depict approximate curves of CPSOGSA and BBO. It can be seen that CPSOGSA approximate curve is having fewer deflections and intercepts as compared to BBO approximate curve. Moreover, the convergence curves are shown in Figure 18. Basically, the MSE values of CPSOGSA, BBO, GSA, and DE are close to each other, that is why convergence curves are overlapping. In fact, CPSOGSA is having a steep slope showing the high speed of convergence towards the global minimum. Furthermore, box plots are reported in Figure 19. It also indicates the best MSE results for CPSOGSA while ACO and DA again show unsatisfactory values for statistical measures.

9.3.2 Cosine function (CF) dataset. The cosine function benchmark is more complex than the SF dataset. It consists of 31 training and 38 testing instances. Moreover, it lies in the closed interval of [1.25, 2.75]. The graph of CF used for simulation analysis is shown in Figure 20. It may be observed that CF has one lower minimum.

The statistical results are reported in Table 12. It is evident that BBO, CPSOGSA, PSO, and SSA have optimal values for average and SD. Also, they are close to each other indicating the same level of competitiveness in handling stagnation in local minima issues. Besides, CPSOGSA shows a second-best average test error of 0.48036 only behind BBO (0.4678). It

Table 11.
Simulation results of sigmoid function approximation dataset

Algorithm	Best	Worst	Average	SD	Median	ATE	*p*-values	Exec. Time
GSA	0.24748	0.26024	0.25253	0.0040427	0.25195	5.4761	0.001953	97.4195
PSO	0.24637	0.24772	0.24688	0.00047	0.2466	5.046	0.10546	102.6512
CPSOGSA	0.24641	0.24872	*0.24687*	*0.0006934*	0.24666	*1.8032*	0.625000	98.3518
ACO	1.4925	1.4925	1.4925	1.3512e-11	1.4925	5.4761	0.001953	116.9261
BBO	0.24639	0.24671	*0.24671*	*9.4725e-5*	0.24653	*3.5529*	N/A	105.9214
DE	0.24658	0.25043	0.24803	0.0011518	0.24772	1.931	0.001953	103.6034
DA	0.25397	0.45032	0.32027	0.68524	0.29963	5.1311	0.001953	113.8915
SCA	0.24718	0.25167	0.2494	0.001568	0.24885	5.0417	0.0019531	94.1501
SSA	0.2465	0.24727	0.2468	0.000254	0.24671	5.0562	0.005859	102.7184

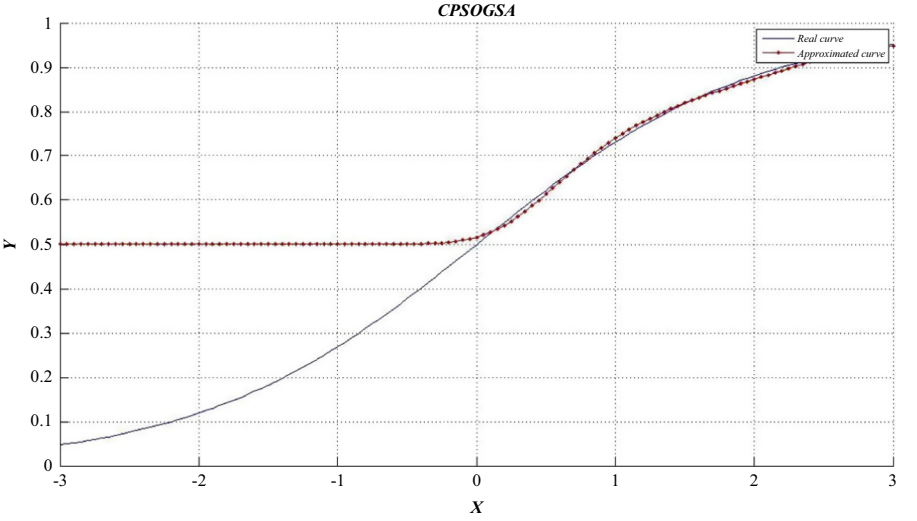

Figure 16.
Approximate curve of CPSOGSA for SF benchmark

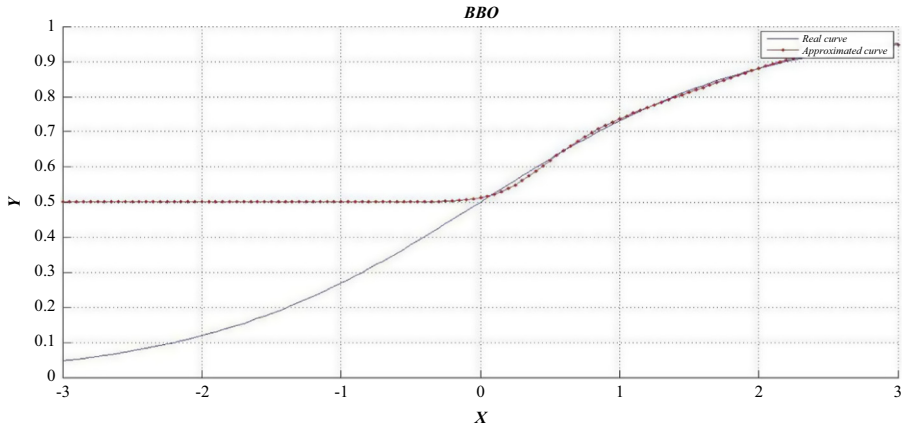

Figure 17.
Approximate curve of BBO for SF benchmark

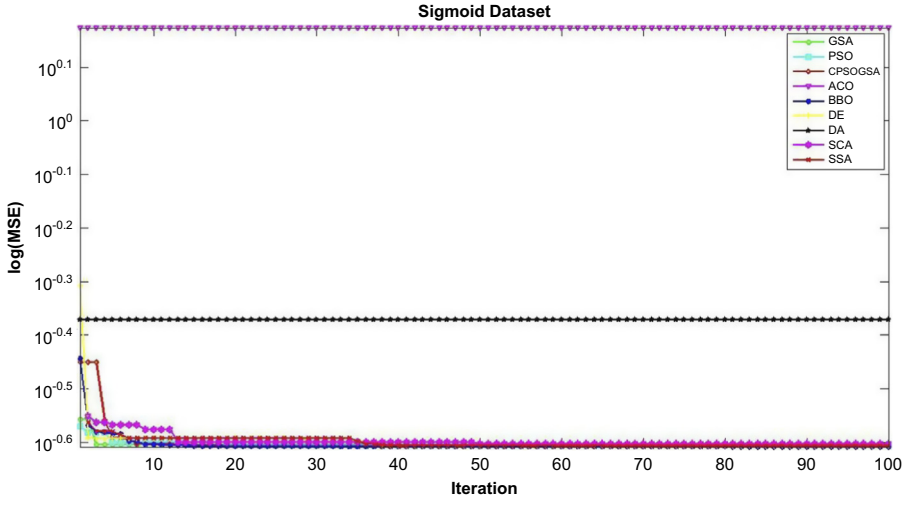

Figure 18.
Convergence curve for SF benchmark

shows the ability of CPSOGSA in providing correct values for MLP inputs (weights, biases, and hidden nodes). In addition, *p*-values prove that CPSOGSA, PSO, and DE statistical results are more significant and impressive than BBO.

Furthermore, Figures 21 and 22 present approximate curves of CPSOGSA and BBO, respectively. It can be observed that both curves show identical behavior. However, CPSOGSA curve deflects more in the beginning while the BBO curve maintains its trajectory throughout the process of iterations.

Moreover, the convergence graphs are depicted in Figure 23 indicating BBO has a high speed of convergence while CPSOGSA, DE, SCA, SSA, and GSA show appreciable intensification power. Also, the box plots are shown in Figure 24. It indicates that the median, lower and upper quartile values of CPSOGSA, BBO, and DE are spread in adjacent numerical range of MSE while GSA, DA and ACO have large values for median and MSE depicting ordinary performance in handling complex and non-linear search spaces.

9.3.3 Sine function (SFN) dataset. It is the most complex and strenuous function approximation dataset having 126 instances for training algorithms and 252 test instances for investigating the performance of HAs in handling entrapment in local minima problems.

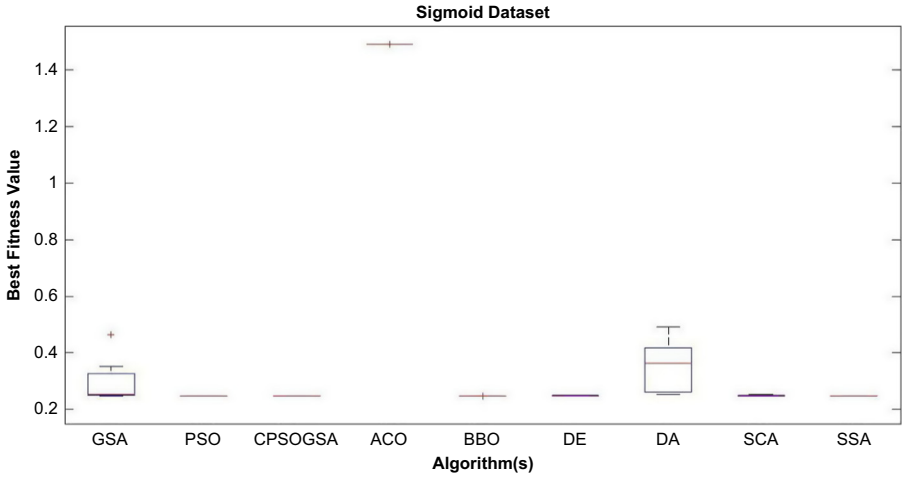

Figure 19.
Box plot of SF benchmark

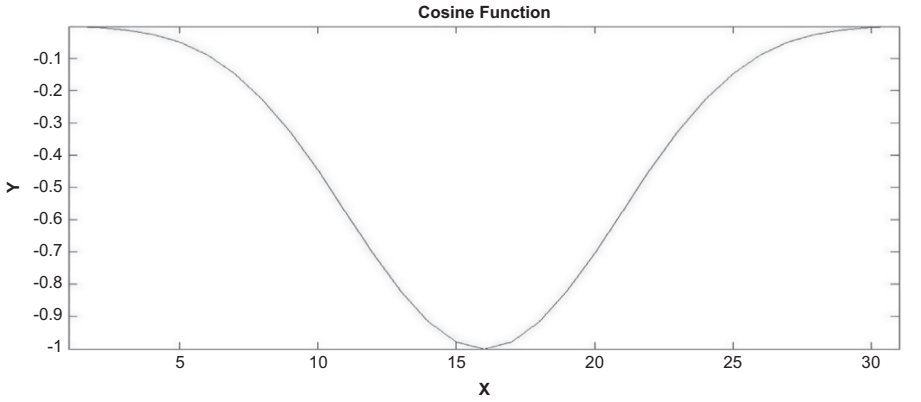

Figure 20.
Cosine function

Algorithm	Best	Worst	Average	SD	Median	ATE	*p*-values	Exec. Time
GSA	0.29298	0.38412	0.34744	0.032926	0.3549	0.97471	0.003906	79.9584
PSO	0.17581	0.1811	0.17767	0.001843	0.17671	1.3017	0.322265	76.4363
CPSOGSA	0.17652	0.37893	*0.25797*	*0.10395*	0.17814	*0.48036*	0.431640	77.2955
ACO	1.0801	1.0801	1.0801	1.0961e-13	1.0801	1.4501	0.001953	99.6037
BBO	0.17608	0.17842	*0.17719*	*0.0007064*	0.17718	*0.4678*	N/A	82.5839
DE	0.17844	0.18999	0.18331	0.0036538	0.18303	0.7295	0.083984	85.6778
DA	0.21288	0.49761	0.39592	0.073706	0.39789	1.426	0.0019531	93.678
SCA	0.18067	0.20902	0.19256	0.008623	0.19202	1.3444	0.0019531	89.7449
SSA	0.17623	0.17999	0.17789	0.001154	0.17754	1.3164	0.625000	83.8211

Table 12.
Simulation results of cosine function approximation dataset

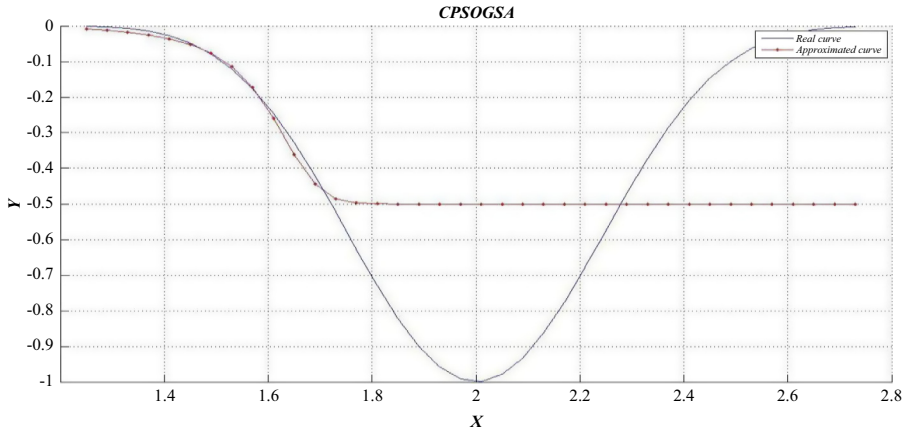

Figure 21.
Approximate curve of CPSOGSA for CF benchmark

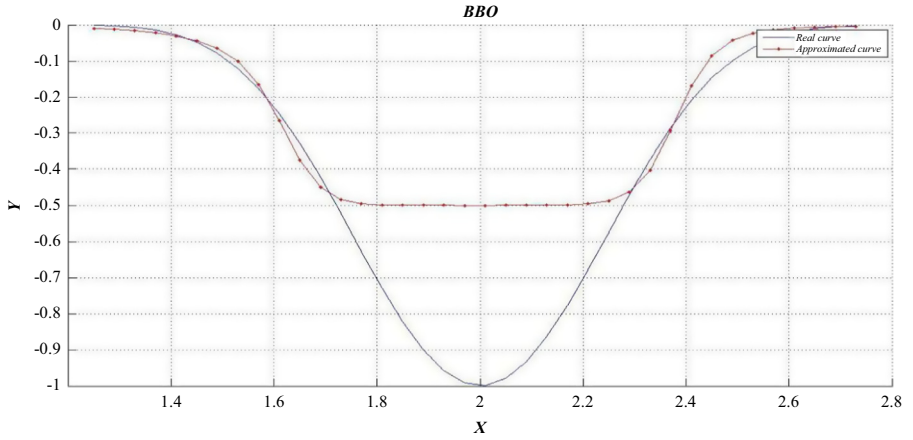

Figure 22.
Approximate curve of BBO for CF benchmark

The function numerical values belong to the closed interval $[-2\pi, +2\pi]$ with an increment of 0.1 for training instances and 0.05 for testing instances. Also, the SFN graph is having four peaks as shown in Figure 25.

Moreover, simulation results are reported in Table 13. It may be observed that SSA is having minimum average MSE values indicating the efficient performance of the algorithm. Besides, CPSOGSA also provides efficient results for MSE and average test error measures. Moreover, the Wilcoxon rank-sum test reveals that CPSOGSA, PSO, BBO, SCA, and DE are having p-values greater than 0.05 indicating statistical edge over SSA.

Also, Figures 26 and 27 present approximate curves for CPSOGSA and SSA in which the CPSOGSA curve is showing symmetrical behavior at the beginning but afterward, it remains flat. On the other hand, the SSA curve indicates sine-like behavior on initial iterations; however, afterward, it depicts a linear straight line pattern. It also conveys that CPSOGSA has more approximation power than SSA.

Furthermore Figure 28 shows convergence curves of HAs in which CPSOGSA has the highest and ACO lowest speed of convergence in selecting optimal candidate solutions for

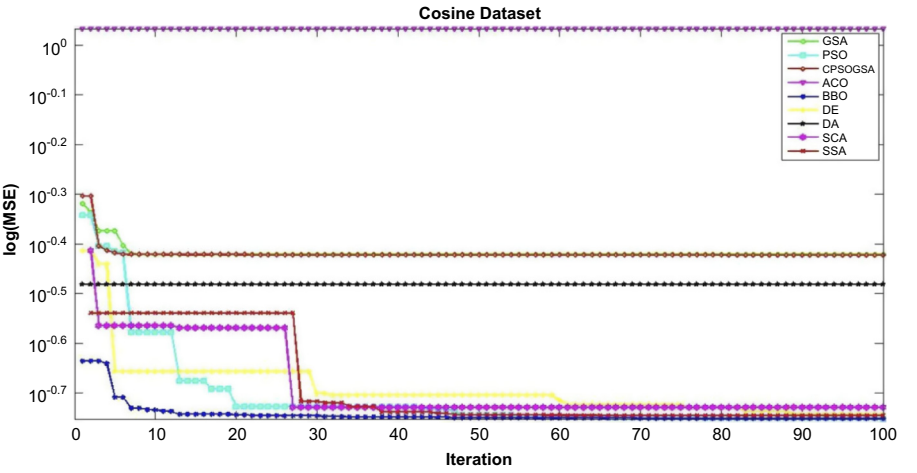

Figure 23.
Convergence curve for CF benchmark

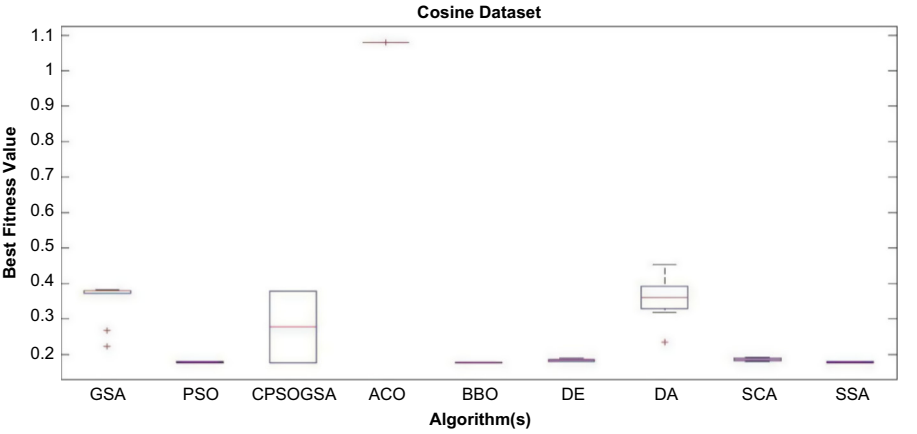

Figure 24.
Box plot of CF benchmark

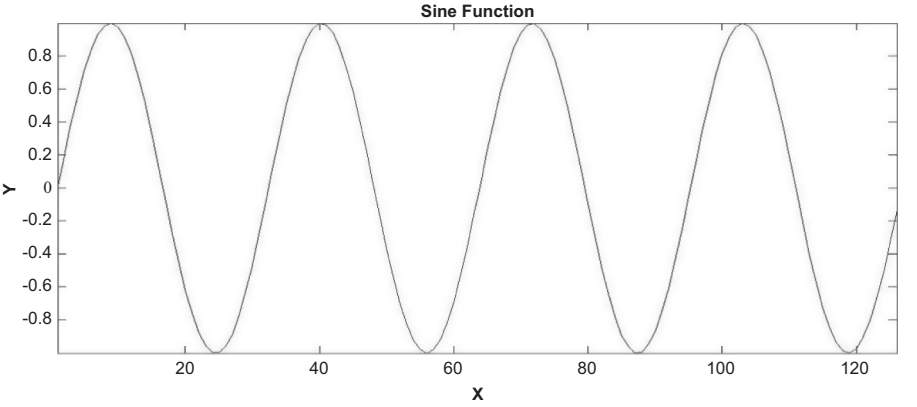

Figure 25.
Sine function

finding the global minimum meaning enhanced intensification power. In addition, Figure 29 shows box plots of HAs in which the median and inter-quartile range values of CPSOGSA, BBO, SCA, SSA, and DE are at the same level. Moreover, ACO maintains its sub-standard and ordinary performance.

Algorithm	Best	Worst	Average	SD	Median	ATE	*p*-values	Exec. Time
GSA	0.46709	0.49328	0.48162	0.008625	0.48312	15.6411	0.0019531	85.6322
PSO	0.41501	0.45016	0.43962	0.00943	0.44188	24.9082	0.625000	78.7682
CPSOGSA	0.42092	0.45764	*0.45113*	*0.010952*	0.45521	*14.9418*	0.193359	115.016
ACO	1.4989	1.4989	1.4989	1.789e-10	1.4989	25.1959	0.0019531	96.3006
BBO	0.44896	0.4589	0.45517	0.0026836	0.45564	14.7107	0.322265	88.15
DE	0.45351	0.45677	0.45576	0.0096901	0.45592	14.7188	0.431640	73.4876
DA	0.21288	0.49761	0.39592	0.07370	0.39789	1.426	0.0019531	93.678
SCA	0.18067	0.20902	0.19256	0.008623	0.19202	1.3444	0.083984	89.7449
SSA	0.17623	0.17999	*0.17789*	*0.001154*	0.17754	*1.3164*	N/A	83.8211

Table 13.
Simulation results of sine function approximation dataset

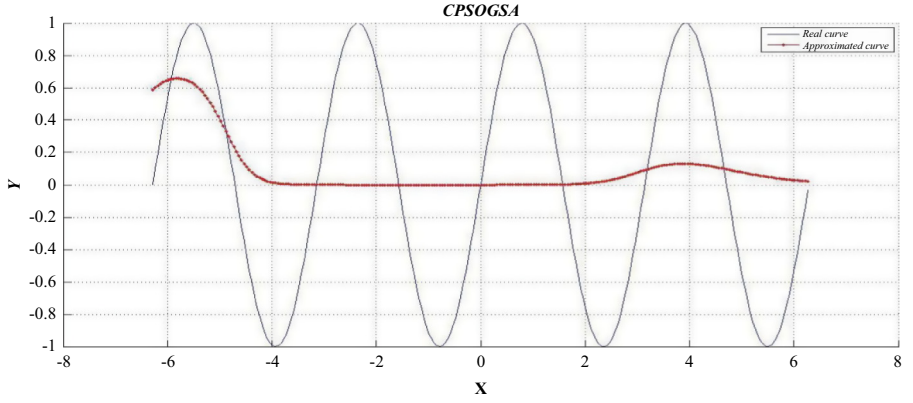

Figure 26.
Approximate curve of CPSOGSA for SFN benchmark

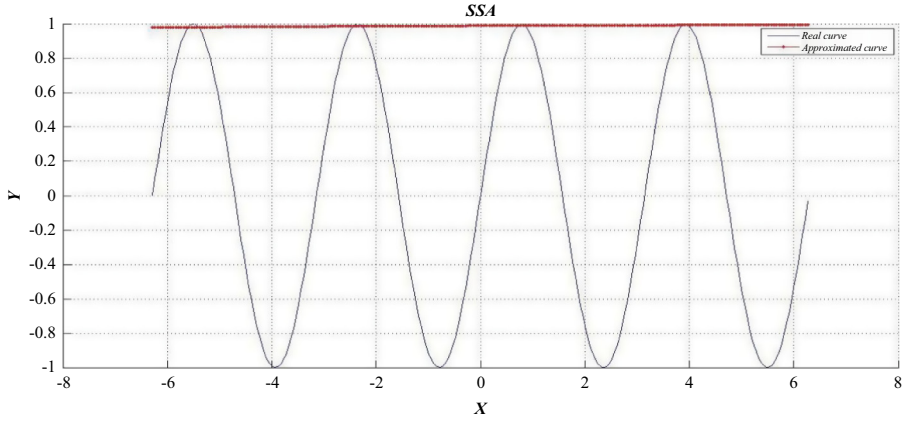

Figure 27.
Approximate curve of SSA for SFN benchmark

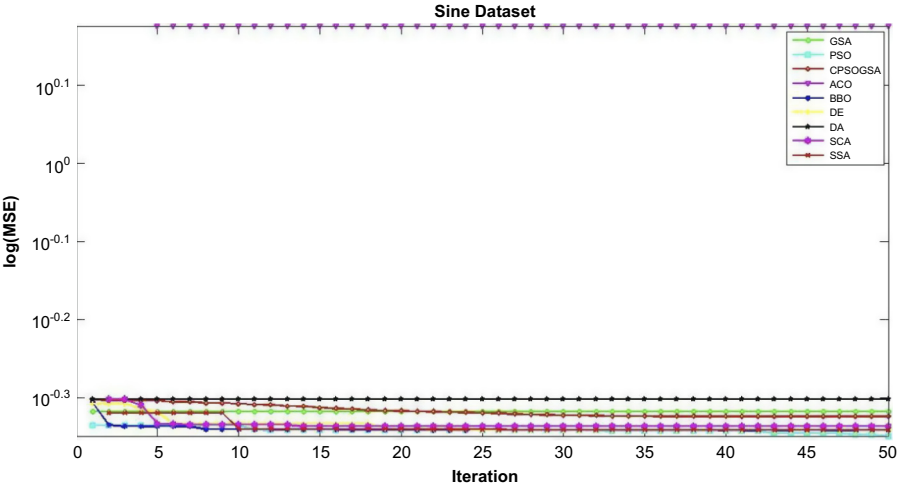

Figure 28.
Convergence curve for SFN benchmark

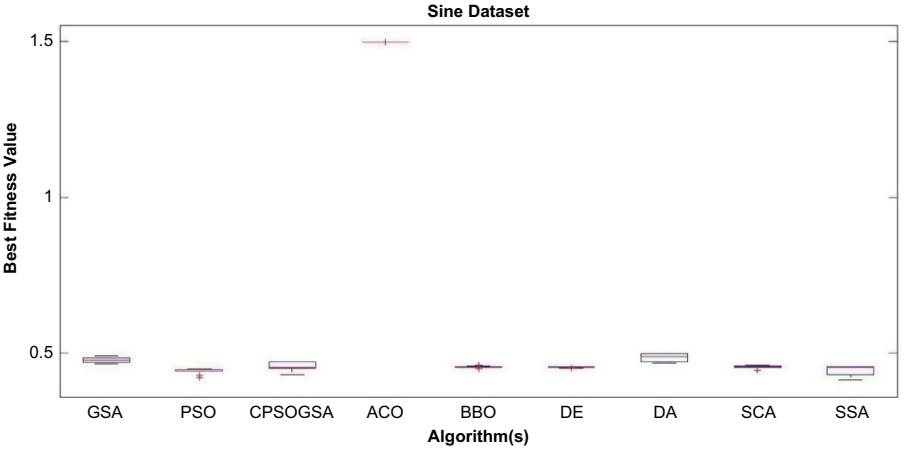

Figure 29.
Box plot of SFN benchmark

To sum up the results of HAs for function approximation datasets; it is clear that CPSOGSA shows efficient performance in handling local minima problems and optimization of input values for MLP. Moreover, CPSOGSA has the best MSE values for sigmoid and sine function datasets and also impressive average test error (ATE) results for the cosine function benchmark. However, BBO, SSA, PSO and DE also gave fine results for MSE and test errors while ACO performs poorly in all three function approximation benchmarks.

10. Conclusion and future directions

In this paper, a recently proposed hybrid CPSOGSA algorithm has been employed for training MLP. Eight benchmark datasets including five classification datasets (XOR, Balloon, Iris, Breast Cancer and Heart) and three function-approximation datasets (Sigmoid, Sine, and Cosine) were used for performance evaluation of CPSOGSA in training MLP.

The statistical results indicate that CPSOGSA gives the best results for two classification benchmarks, that is, breast cancer and heart while also providing efficient MSE values for

two function approximation benchmarks including sine and sigmoid function as compared to GSA, BBO, ACO, SCA, SSA, PSO and DE. Moreover, CPSOGSA also provides very competitive results for other datasets only behind to BBO. In addition, CPSOGSA also shows the best average recognition rates for the iris dataset and impressive average test error rates for all function benchmarks. Besides, CPSOGSA has effectively resolved stagnation in local optima problem and increasing the overall convergence speed of MLP.

For future studies, it would be interesting to apply CPSOGSA for training other types of neural networks such as convolution neural network (CNN), modular neural network, and long/short term memory (LSTM) networks. Moreover, CPSOGSA can be applied to find the structural parameters of the MLP. Besides, the fine-tuning of CPSOGSA can also be investigated.

Abbreviations

NN	Neural Network
ANN	Artificial Neural Network
FNN	Feedforward Neural Network
MLP	Multi-Layer Perceptron Neural Network
BP	Back Propagation Algorithm
CGA	Conjugate Gradient Algorithm
MA	Marguardt Algorithm
LM	Levenberg-Marguardt Algorithm
HA	Heuristic Algorithm
GSA	Gravitational Search Algorithm
PSO	Particle Swarm Optimization
GA	Genetic Algorithm
DE	Differential Evolution
ACO	Ant Colony Optimization
BBO	Biogeography Based Optimization
CPSOGSA	Constriction Coefficient Based PSO and GSA
ABC	Artificial Bee Colony Algorithm
CFO	Central Force Optimization
SSO	Social Spider Optimization
CRO	Chemical Reaction Optimization
CSS	Charged System Search
IWO	Invasive Weed Optimization
MOA	Magnetic Optimization Algorithm
GWO	Grey Wolf Optimizer
GSO	Glowworm Swarm Optimization
SOS	Symbiotic Organisms Search Algorithm
DA	Dragonfly Algorithm
SCA	Sine-Cosine Algorithm
SSA	Salp Swarm Algorithm

References

Adeli, H. and Hung, S. (1994), "An adaptive conjugate gradient learning algorithm for efficient training of neural networks", *Applied Mathematics and Computation*, Vol. 62 No. 1, pp. 81-102.

Alboaneen, D.A., Tianfield, H. and Zhang, Y. (2017), "Glowworm swarm optimization for training multi-layer perceptrons", *Proceeding of the 4th IEEE/ACM International Conference on Big Data Computing, Applications, and Technologies*, pp. 131-138.

Alijarah, I., Faris, H. and Mirjalili, S. (2016), "Optimizing connection weights in neural networks using whale optimization algorithm", *Soft Computing*, Vol. 22, pp. 1-15.

Alijarah, I., Faris, H., Mirjalili, S. and Al-Madi, N. (2018), "Training radial basis function networks using biogeography based optimizer", *Neural Computing*, Vol. 29, pp. 529-553.

Alijarah, I., Faris, H., Mirjalili, S., Madi, N.A., Sheta, A. and Mafarja, M. (2019), "Evolving neural networks using bird swarm algorithm for data classification and regression applications", *Cluster Computing*, Vol. 22 No. 22, pp. 1317-1345.

Amirsadri, S., Mousaviradi, S.J. and Komleh, H.E. (2017), "A levy flight grey wolf optimizer combined with backpropagation for neural network training", *Neural Computing and Applications*, Vol. 3 No. 12, pp. 3707-3720.

Bebis, G. and Georgiopoulos, M. (1994), "Feed-forward neural networks", *IEEE Potentials,* Vol. 13 No. 4, pp. 27-31.

Blake, C. and Merz, C.J. (1998), "UCI: repository of machine learning databases", available at: http://www.ics.uci.edu/~mlearn/MLRepository.html.

Blum, C. and Socha, K. (2005), "Training feed-forward neural networks with ant colony optimization: an application to pattern classification", *5th International Conference on Hybrid Intelligent Systems*, p. 6.

Branke, J. (1995), "Evolutionary algorithms for neural network design and training", *Proceedings of the 1st Nordic Workshop on Genetic Algorithms and its Applications*, pp. 1-21.

Bui, Q.T. (2018), "Meta-heuristic algorithms in optimizing neural network a comparative study for forest fire susceptibility in Dak Nong, Vietnam", *Geometrics and Hazards and Risk*, Vol. 10 No. 1, pp. 136-150.

Chen, J.F., Do, Q.H. and Hsieh, H.N. (2015), "Training artificial neural networks by a hybrid pso-cs algorithm", *Algorithms*, Vol. 8 No. 2, pp. 292-308.

Clerc, M. and Kennedy, J. (2002), "The particle swarm—explosion, stability, and convergence in a multidimensional complex space", *IEEE Transactions on Evolutionary Computation*, Vol. 6 No. 1, pp. 58-73.

Derrac, J., García, S., Molina, D. and Herrera, F. (2011), "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms", *Swarm and Evolutionary Computation*, Vol. 1 No. 1, pp. 3-18.

Dorffner, G. (1996), "Neural networks for time series processing", *Neural Network World*, Vol. 6, pp. 447-468.

Eiben, A.E. and Schippers, C.A. (1998), "On evolutionary diversification and intensification", *Fundamenta Informaticae*, Vol. 35 Nos 1-4, pp. 35-50.

Fahlman, S.E. (1988), "An empirical study of learning speed in backpropagation networks", Technical report.

Faris, H., Aliharah, I., Al-Madi, N. and Mirjalili, S. (2016a), "Optimizing the learning process of Feedforward neural networks using lightning search algorithm", *International Journal of Artificial Intelligence Tools*, Vol. 25 No. 6, pp. 1-32.

Faris, H., Aljarah, I. and Mirjalili, S. (2016b), "Training feedforward neural networks using multi-verse optimizer for binary classification problems", *Applied Intelligence*, Vol. 45 No. 2, pp. 322-332.

Faris, H., Aliharah, I. and Mirjalili, S. (2017a), "Evolving radial basis function networks using moth-flame optimizer", in Samui, P., Roy, S.S. and Balas, V.E. (Eds), *Handbook of Neural Computation*, Academic Press, New York NY, pp. 537-550.

Faris, H., Aljarah, I. and Mirjalili, S. (2017b), "Improved monarch butterfly optimization for unconstrained global search and neural network training", *Applied Intelligence*, Vol. 48 No. 2, pp. 445-464.

Gori, M. and Tesi, A. (1992), "On the problem of local minima in backpropagation", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 14, pp. 76-86.

Green II, R.C., Wang, L. and Alam, M. (2011), "Training neural networks using central force optimization and particle swarm optimization: insights and comparisons", *Expert Systems with Applications,* Vol. 39 No. 1, pp. 555-563.

Gudise, V.G. and Venayagamoorthy, G.K. (2003), "Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks", *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, pp. 110-117.

Hagan, M.T. and Menhaj, M.B. (1994), "Training feedforward networks with the Marquardt algorithm", *IEEE Transactions on Neural Networks*, Vol. 5 No. 6, pp. 989-993.

Halliday, D., Resnick, R. and Walker, J. (2000), *Fundamentals of Physics*, 6th ed., Wiley, Delhi.

Heidari, A.S., Faris, H., Alijarah, I. and Mirjalili, S. (2018), "An efficient hybrid multi-layer perceptron neural network with grasshopper optimization", *Soft Computing*, Vol. 23 No. 17, pp. 7941-7958.

Itano, F., DeSousa, M.A.D.A. and Hernandez, E.D.M. (2018), "Extending MLP ANN hyperparameters optimization by using genetic algorithm", *Proceedings of the IEEE 2018 International Joint Conference on Neural Networks (IJCNN)*, Rio de Janeiro, Brazil, pp. 1-8.

Jacobs, R.A. (1988), "Increased rates of convergence through learning rate adaptation", *Neural Networks*, Vol. 1 No. 4, pp. 295-307.

Karaboga, D. (2005), "An idea based on honey bee swarm for numerical optimization", Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, Vol. 200, pp. 1-10.

Karaboga, D., Akay, B. and Ozturk, C. (2007), "Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks", *Springer International Conference on Modeling Decisions for Artificial Intelligence*, pp. 318-329.

Kohonen, T. (1990), "The self-organizing map", *Proceedings of the IEEE*, Vol. 78 No. 9, pp. 1464-1480.

Kowalski, P.A. and Lukasik, S. (2016), "Training neural networks with krill herd algorithm", *Neural Processing Letters*, Vol. 44 No. 1, pp. 5-17.

Lee, Y., Oh, S.H. and Kim, M.W. (1993), "An analysis of premature saturation in backpropagation learning", *Neural Networks,* Vol. 6 No. 5, pp. 719-728.

Li, J., Luo, Q., Lio, L. and Zhou, Y. (2018), "Using spotted hyena optimization for training Feedforward neural network", in Huang, D., Gromiha, M.M. and Hussian, A. (Eds), *Intelligent Computing Methodologies*, Springer, Cham, pp. 828-833.

Li, W. (2018), "Improving particle swarm optimization based on neighborhood and historical memory for training multi-layer perceptron", *Information*, Vol. 9 No. 1, pp. 1-20.

Llonen, J., Kamarainen, J.K. and Lampinen, J. (2003), "Differential evolution training algorithm for feed-forward neural networks", *Neural Processing Letters*, Vol. 17 No. 1, pp. 93-105.

Mangasarian, O.L. and Wolberg, W.H. (1990), *Cancer Diagnosis via Linear Programming*, University of Wisconsin-Madison, Computer Science Department, New York.

McCulloch, W.S. and Pitts, W. (1943), "A logical calculus of the ideas immanent in nervous activity", *Bull Math Biophysics,* Vol. 5 No. 4, pp. 115-133.

Mendes, R., Cortez, P., Rocha, M. and Neves, J. (2002), "Particle swarms for feed-forward neural network training", *Proceedings of the IEEE International Joint Conference on Neural Networks*, Vol. 6, pp. 1895-1899.

Mirjalili, S. and Sadiq, A.F. (2011), "Magnetic optimization algorithm for training multilayer perceptron", *IEEE 3rd International Conference on Communication, Software, and Networks (ICCSN)*, pp. 42-46.

Mirjalili, S., Mohd Hashim, S.Z. and Moradian Sardroudi, H. (2012), "Training Feed-forward neural networks using hybrid particle swarm optimization and gravitational search algorithm", *Applied Mathematics and Computation*, Vol. 218 No. 22, pp. 11125-11137.

Mirjalili, S.M., Abedi, K. and Mirjalili, S. (2013), "Optical buffer performance enhancement using particle swarm optimization in ring-shape-hole photonic crystal waveguide", *Optic,* Vol. 124 No. 23, pp. 5989-5993.

Mirjalili, S., Mirjalili, M.S. and Lewis, A. (2014), "Let a biogeography-based optimizer train your multi-layer perceptron", *Information Sciences*, Vol. 269, pp. 88-209.

Mirjalili, S.Z., Saremi, S. and Mirjalili, S.M. (2015), "Designing evolutionary feedforward neural networks using social spider optimization algorithm", *Neural Computing and Applications*, Vol. 26 No. 8, pp. 1919-1928.

Mirjalili, S., Gandomi, A.H., Mirjalili, S.Z., Saremi, S., Faris, H. and Mirjalili, S.M. (2017), "Salp Swarm Algorithm: a bio-inspired optimizer for engineering design problems", *Advances in Engineering Software*, Vol. 0, pp. 1-29.

Mirjalili, S. (2015), "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems", *Neural Computing and Applications*, Vol. 27, pp. 1053-1073, doi: 10.1007/s00521-015-1920-1.

Mirjalili, S. (2015), "How effective is the Grey Wolf optimizer in training multi-layer perceptrons", *Applied Intelligence*, Vol. 43 No. 1, pp. 150-161.

Mirjalili, S. (2016), "SCA: a Sine Cosine algorithm for solving optimization problems", *Knowledge-Based Systems*, Vol. 96, pp. 120-133.

Moallem, P. and Razmjooy, N. (2012), "A multi-layer perceptron neural network trained by invasive weed optimization for potato color image segmentation", *Trends in Applied Sciences Research*, Vol. 7 No. 6, pp. 445-455.

Mosavi, M.R., Khishe, M., Naseri, M.J., Parvizi, G.R. and Ayat, M. (2019), "Multi-layer perceptron neural network utilizing adaptive best mass gravitational search algorithm to classify sonar dataset", *Archives of Acoustics*, Vol. 44 No. 1, pp. 137-151.

Ng, S., Cheung, C., Leung, S. and Luk, A. (2003), "Fast convergence for backpropagation network with magnified gradient function", *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, Vol. 3, pp. 1903-1908.

Ooyen, A. and Nienhuis, B. (1992), "Improving the convergence of the backpropagation algorithm", *Neural Networks*, Vol. 5 No. 3, pp. 465-471.

Ozturk, C. and Karaboga, D. (2011), "Hybrid artificial bee algorithm for neural network training", *IEEE Congress on Evolutionary Computation (CEC)*, pp. 84-88.

Park, J. and Sandberg, I.W. (1993), "Approximation and radial-basis-function networks", *Neural Computation,* Vol. 5 No. 2, pp. 305-316.

Pereira, L.A., Afonso, L.C., Papa, J.P., Vale, Z.A., Ramos, C.C., Gastaldello, D.S. and Souza, A.N. (2013), "Multi-layer perceptron neural networks training through charged system search and its application for non-technical losses detection", *IEEE PES Conference on Innovative Smart Grid Technologies*, pp. 1-6.

Pereira, L., Rodrigues, D., Ribeiro, P., Papa, J. and Weber, S.A. (2014), "Social-spider optimization-based artificial neural networks training and its applications for Parkinson's disease identification", *IEEE 27th International Symposium on Computer-based Medical Systems (CBMS)*, pp. 14-17.

Pu, X., Chen, S., Yu, X. and Zhang, L. (2018), *Developing a Novel Hybrid Biogeography Based Optimization Algorithm for Multi-Layer Perceptron Training under Big Data Challenge*, Scientific Programming, pp. 1-7, doi: 10.1155/2018/2943290.

Rashedi, E., Nezamabadi-pour, H. and Saryazdi, S. (2009), "GSA: a gravitational search algorithm", *Information Sciences*, Vol. 179 No. 13, pp. 2232-2248.

Rather, S.A. and Bala, P.S. (2019a), "A holistic review on gravitational search algorithm and its hybridization with other algorithms", in *IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, Coimbatore, Tamil Nadu, India, 2019, pp. 1-6. doi: 10.1109/ICECCT.2019.8869279.

Rather, S.A. and Bala, P.S. (2019b), "Analysis of gravitation based optimization algorithms for clustering and classification", in Pedro, G.M. (Ed.), *Handbook of Research on Big Data Clustering and Machine Learning*, IGI Global, New York, NY, pp. 77-99. doi: 10.4018/978-1-7998-0106-1.ch005.

Rather, S.A. and Bala, P.S. (2019c), "Hybridization of constriction coefficient based particle swarm optimization and gravitational search algorithm for function optimization", *Paper Presented at the 2019 Elsevier International Conference in Electronics, Electrical, and Computational Intelligence (ICAEEC-2019)*, (paper accepted for publication).

Slowik, A. and Bialko, M. (2008), "Training of artificial neural networks using differential evolution algorithm", *IEEE Conference on Human System Interactions*, pp. 60-65.

Socha, K. and Blum, C. (2007), "An ant colony optimization algorithm for continuous optimization: application to feed-forward neural network training", *Neural Computing Applications*, Vol. 16 No. 3, pp. 235-247.

Tarkhaneh, O. and Shen, H. (2019), "Training of feedforward neural networks for data classification using hybrid particle swarm optimization, mantega levy flight, and neighborhood search", Vol. 5 No. 4, doi: 10.1016/j.heliyon.2019.e01275.

Uzlu, E., Kankal, M., Akpinar, A. and Dede, T. (2014), "Estimates of energy consumption in Turkey using neural networks with the teaching-learning-based optimization algorithm", *Energy*, Vol. 75, pp. 295-303.

Vogal, T.P., Mangis, J., Rigler, A., Zink, W. and Alkon, D. (1988), "Accelerating the convergence of the backpropagation method", *Biological Cybernetics*, Vol. 59 Nos 4-5, pp. 257-263.

Weir, M.K. (1991), "A method doe self-determination of adaptive learning rates in backpropagation", *Neural Networks,* Vol. 4 No. 3, pp. 371-379.

Whitley, D., Starkweather, T. and Bogart, C. (1990), "Genetic algorithms and neural networks: optimizing connections and connectivity", *Parallel Computing*, Vol. 14 No. 3, pp. 347-361.

Wu, H., Zhou, Y., Luo, Q. and Baseet, M.A. (2016), "Training Feedforward neural networks using symbiotic organisms search algorithm", *Computational Intelligence and Neuroscience*, Vol. 2016, doi: 10.1155/2016/9063065.

Yao, X. (1993), "Evolutionary artificial neural networks", *International Journal of Neural Systems*, Vol. 4 No. 3, pp. 203-222.

Yu, J.J., Lam, A.Y. and Li, V.O. (2011), "Evolutionary artificial neural network based on chemical reaction optimization", *IEEE Congress on Evolutionary Computation (CEC)*, pp. 2083-2090.

Zhang, J.R., Zhang, J., Lok, T.M. and Lyu, M.R. (2007), "A hybrid particle swarm optimization-back propagation algorithm for Feedforward neural network training", *Applied Mathematics and Computation*, Vol. 185 No. 2, pp. 1026-1037.

Zhao, R., Wang, Y., Hu, P., Jelodar, H., Yuan, C., Li, Y., Masood, I. and Rabbani, M. (2019), "Selfish herds optimization algorithm with orthogonal design and information update for training multi-layer perceptron neural networks", *Applied Intelligence*, Vol. 49 No. 6, pp. 2339-2381.

Corresponding author
Sajad Ahmad Rather can be contacted at: sajad.win8@gmail.com