Optimizing Connection Weights in Neural Networks Using Hybrid Metaheuristics Algorithms

Rabab Bousmaha, GeCoDe Laboratory, Department of Computer Science, University of Saida, Algeria
Reda Mohamed Hamou, GeCoDe Laboratory, Department of Computer Science, University of Saida, Algeria
Abdelmalek Amine, GeCoDe Laboratory, Department of Computer Science, University of Saida, Algeria

https://orcid.org/0000-0002-0388-1275
https://orcid.org/0000-0001-9327-7903

ABSTRACT

The learning process of artificial neural networks is an important and complex task in the supervised learning field. The main difficulty of training a neural network is the process of fine-tuning the best set of control parameters in terms of weight and bias. This paper presents a new training method based on hybrid particle swarm optimization with multi-verse optimization (PMVO) to train the feedforward neural networks. The hybrid algorithm is utilized to search better in solution space which proves its efficiency in reducing the problems of trapping in local minima. The performance of the proposed approach was compared with five evolutionary techniques and the standard momentum backpropagation and adaptive learning rate. The comparison was benchmarked and evaluated using six bio-medical datasets. The results of the comparative study show that PMVO outperformed other training methods in most datasets and can be an alternative to other training methods.

KEYWORDS

Multi-Verse Optimization, Particle Swarm Optimization, Real-World Datasets, Training Feedforward Neural Networks

INTRODUCTION

Artificial neural network (ANN) is one of the most important data mining techniques. It has been successfully applied to many domains. The feedforward multilayer perceptron (MLP) is one of the best-known neural networks. The multilayer perceptron (MLP) consists of three layers composed of neurons organized into input, output and hidden layers. The first layer receives the input, the second layer is the hidden layer and the third layer produces the output. The success of an MLP generally depends on the training process that is determined by training algorithms. The objective of the training algorithms is to find the best connection between weights and biases that minimize the classification error.

Training algorithms can be classified into two classes: gradient-based and stochastic search methods. Backpropagation (BP) and its variants are gradient-based methods and considered as one of the most popular techniques used to train the MLP neural network. Gradient-based methods have many drawbacks, such as the slow convergence, the high dependency on the initial value of weights and...
biases and the tendency to be trapped in local minima (Zhang, Zhang, Lok, & Lyu, 2007). To address these problems, stochastic search methods, such as metaheuristics have been proposed as alternative methods for training feedforward neural network. Metaheuristics have many advantages: they apply to any type of NN with any activation function (Kiranyaz, Ince, Yildirim, & Gabbouj, 2009), provide acceptable solutions within a reasonable time to solve complex and difficult problems (Raidl, 2006), and are particularly useful for dealing with large complex problems that generate many local optima (Kenter et al., 2018; Wang, Li, Huang, & Lazebnik, 2019).

Metaheuristics can be divided into single and population solution-based algorithms. The population-based algorithms can be divided into groups: swarm intelligence and evolutionary algorithms. For swarm intelligence algorithms, several authors proposed Particle Swarm Optimization as training methods. The biggest challenges at PSO are a poor compromise between exploration and exploitation and limiting the diversity of the population. Some works tried to address them in terms of learning approaches, parameters setting and hybridized methods. For example, several works attempted to fine-tune and modify parameters through Gaussian adaptation, Memory adaptation or Fuzzy-based methods, while others work sought to avoid premature convergence through the use of hybrid methods such as LFPSO (Haklı, & Üguz, 2014) and (LPSONS) (Tarkhaneh, & Shen, 2019).

The particle swarm optimization is a technique inspired by birds flocking or fish schooling. In PSO each individual is bird or fish in the search space that has position and velocity. Particles try to keep up with their local best positions before they search for the best global position.

In this paper, we propose a new training algorithm based on hybrid Particle Swarm Optimization (PSO) with Multi-Verse Optimization (MVO) to train MLP neural networks.

Though a wide variety of swarm-based and evolutionary algorithms are investigated and deployed in the MLP training literature. There is a question here as if new training algorithms still need to be developed. The answer is yes, local minimum issues remain available. The no-free-lunch theorem (NFL) states that there is no superior optimization algorithm to solve all optimization problems.

Training MLP is also an optimization problem that varies for each dataset (Faris et al., 2016). Based on those reasons, this paper presents a new training approach based on particle swarm optimization (PSO) with Multi-Verse Optimization (MVO) to train MLP neural networks.

Moreover, the application of the trainer was investigated in bio-medical. The performance of PMVO was compared with five well-known trainer metaheuristics algorithms in the literature: PSO (Mendes, Cortez, Rocha, & Neves, 2002), MFO (Yamany, Fawzy, Tharwat, & Hassanien, 2015), MVO (Faris et al., 2016), WOA (Aljerah et al., 2018), HACPSO (Khan et al., 2019) based on Accuracy, Mean Square Error (MSE), F-measure, Specificity, Sensitivity, and Precision. Friedman statistical test shows that the proposed training algorithm outperforms other training algorithms.

RELATED WORK

In the literature, the evolutionary and swarm-intelligence based approaches are the most investigated methods to optimize the training of feed-forward neural networks (Dai, Yuan, & Zhang, 2015) (Tarkhaneh, & al, 2019) (Melin, & al, 2013).

Metaheuristics such as particle swarm optimization (Zhang et al., 2000) and genetic algorithm are the earliest works on training feedforward neural networks. The Firework algorithm is proposed by Bolaji et al. to train the MLP neural network and it compared with other algorithms using different datasets (Bolaji, Ahmad & shoal, 2016). Faris et al. suggested the lightning search algorithm (LSA) and tested it with different measurements to achieve optimal results (Faris, Aljarah, Al-Madi & Mirjalili, 2016). Faris et al. proposed Multi-verse optimization for training MLP and, it evaluated using nine bio-medical datasets and compared to five metaheuristics trainer algorithms. The results show that
MVO outperforms other algorithms (Faris et al., 2016). The Whale Optimization algorithm developed by Aljarah et al. to find optimal connexion weights (Aljarah, Faris, & Mirjalili, 2018). The algorithm is compared using a set of twenty datasets for the first time in the literature. The results prove that WOA performs better than the other current algorithms. Green II et al. suggested a Central Force Optimization (CFO) approach to train feedforward neural networks (Green, Wang, & Alam, 2012). The results show that the CFO algorithm performs better than PSO in terms of computational complexity, algorithm design, and a natural basis. Faris et al. proposed Grey Wolf Optimizer to optimize connexion weight and the number of hidden neurons in single hidden feedforward neural networks (FFNN) (Faris, Mirjalili, & Aljarah, 2019). The results show the superiority of GWO compared to other algorithms.

Several other metaheuristic algorithms used to train MLP neural networks such as artificial bee colony (ABC) (Nandy, 2012) and BAT (BA) (Nawi, Rehman, & Khan, 2014), the modified BAT (Jaddi, Abdullah, & Hamdan, 2015), Biogeography Based on Optimizer (BBO) (Aljarah, Faris, Mirjalili, & Al-Madi, 2018), Moth-Flame Optimization (MFO) (Faris, Aljarah, & Mirjalili, 2017) and Improved Monarch Butterfly Optimization (IMBO) (Faris, Aljarah, & Mirjalili, 2017).

Furthermore, several hybrid algorithms have been proposed to train a neural network. Zade et al., Tarkhanne et al. suggested a hybrid approach to neural network training by combining PSO, Mantegna Lévy flight and neighbor search (LPSONS) (Tarkhanne, & Shen, 2019). The comparison experiments showed that the proposed algorithm can find optimal results. Khan (Khan et al., 2019) introduced a new method based on two algorithms, accelerated particle swarm optimization (APSO) and cuckoo search (CS), named HACPSO. The comparison results demonstrated that the proposed algorithm outperforms other algorithms in term of accuracy, MSE and standard deviation.

In summary, metaheuristics are proved useful in training feed-forward neural network (FFNN) and a lot of work has been done to improve their efficiency from different perspectives, using various measures such as classification accuracy and training error.

ARTIFICIAL NEURAL NETWORKS (ANNs)

An artificial neuron network (ANN) is a computational model based on the structure and functions of the biological brain and nervous system. The feedforward neural network (FFNN) is one of the most popular types of artificial neural network (Faris et al., 2016). FFNN has three interconnected layers. The first layer consists of input neurons. These neurons send the data to the second layer, called the hidden layer, which sends the output neurons to the third layer. In FFNN, the information travels in one direction, from the input layer to the output layer. The node or the artificial neuron multiplies each of these inputs by weight, as shown in (1):

\[ S_j = \sum_{i=1}^{n} w_{ij} I_i + \beta_j \]  

(1)

Where, \( n \) is the total number of neuron inputs, \( w_{ij} \) is the connection weight connecting \( I_i \) to neuron \( j \) and \( \beta_j \) is a bias weight (Faris et al., 2016).

Then, the node or the artificial neuron adds the multiplications and sends the sum to a transfer function, for example, Sigmoid function presented in (2):

\[ \frac{1}{1 + e^{-x}} \]  

(2)

The output of the neuron \( j \) can be described as follows (3):
After building the neural network, the set of network weights are adjusted to approximate the desired results. This process is carried out by applying a training algorithm to adapt the weights until error criteria are met (Faris et al., 2016).

PARTICLE SWARM OPTIMIZATION (PSO)

In 1995 Russell Eberhart and James Kennedy have invented the particle swarm optimization which is a population-based stochastic optimization technique inspired by birds flocking around food sources. Like each other evolutionary computational algorithms. In PSO, each individual is a bird in the search space. We call it a particle. All of the particles have fitness values which are evaluated by the fitness function to be optimized and flies in the space with a velocity which is dynamically adjusted according to its own flying experience. (Kennedy, & Eberhart, 1995)

\[ V_{i,j}^{t+1} = V_{i,j}^t + C_1 R_1 (P_{best}^t - X^t) + C_2 R_2 (G_{best}^t - X^t) \]  

\[ X_{i,j}^{t+1} = X^t + V_{i,j}^{t+1} \]  

Where, \( P_{best}^t \) and \( G_{best}^t \) denote the best particle position and best group position and \( W \) is inertia weight \( W = \frac{w_{max} - w_{min}}{\text{max iteration}} \). \( C_1, C_2 \) two positive constants, \( R_1, R_2 \) are random numbers in the interval of \([0, 1]\), \( V_{i,j}^{t+1} \) is the velocity of \( j \text{th} \) member of \( i \text{th} \) particle at iteration number \((t)\) and \((t+1)\).

The new position values \( X_{i,j}^{t+1} \) are obtained by adding the velocity updates determined by the formula given in Equation (5).

The steps of the PSO are given in Algorithm 1.

Algorithm 1. PSO algorithm

**Initialize the population**

**Repeat**

**For** \( I = 1 \) to Population Size **do**

Calculate the objective function value.

If the fitness value is better than the best one (Pbest).

Then set the current value as the new PBest.

Choose the particle with the best fitness value of all the particle as Gbest.

Calculate a new velocity in accordance with Equation (4)

Update the particle position in accordance with Equation (5)

**End For**

**Until** termination criterion is met.
MULTI-VERSE OPTIMIZATION (MVO)

Multi-verse optimization proposed by Syed Ali Mirjalili in 2015 (Mirjalili, Mirjalili, & Hatamlou, 2015) as inspired by the concepts of white holes, black holes, and wormholes in the multi-verse theory and big bang theory. In this algorithm, the models of these three concepts are developed to perform exploration and exploitation and local search. The fitness function for each search agent is indicated by the inflation rate, and each object and each universe in the search agent represent a candidate solution and a variable in the candidate solution.

In this algorithm, the larger universes tend to send objectives to smaller universes. A large universe is defined based on inflation rate in the multi-verse theory. The following rules are applied to the universes of the MVO:

- If the inflation rate is higher, the probability of having a white hole is higher.
- If the inflation rate is higher, the probability of having black holes is lower.
- Universes having higher inflation rate send the objects through white holes.
- Universes having lower inflation rate tend to receive more objects through black holes.
- The objects of all universes may be replaced by the objects of the universe with the greater inflation rate.

The mathematical model of this algorithm is as follows:

\[
x_i^j = \begin{cases} 
X_j + TDR * ((ub_j - lb_j) * r_4 + lb_j) & r_3 < 0.5 \\
X_j - TDR * ((ub_j - lb_j) * r_4 + lb_j) & r_3 \geq 0.5 \\
x_i^j & r_2 \geq WEP \\
\end{cases} 
\]  

Where \(X_j\) indicates the \(j\)th variable in the bests universe, \(lb_j\) indicates the lower bound in \(j\)th variable, \(ub_j\) shows the upper bound in \(j\)th variable, \(r_2, r_3, r_4\) are random numbers in the interval of \([0,1]\), \(TDR \div WEP\) are coefficients, and \(x_i^j\) indicates the \(j\)th parameter in \(i\)th universe.

Algorithm 2 . MVO algorithm

SU=Stored universes

NI=Normalize inflation rate (fitness) of Each universe indexed by \(i\)

\[
Black\_hole\_index = i; \\
For\ each\ object\ indexed\ by\ \(j\) \\
\quad r_1 = \text{random}([0,1]); \\
\quad IF\ r_1 < NI(U_i) \\
\quad\quad White\_hole\_index = \text{RouletteWheelSelection}(-NI); \\
\quad\quad U(Black\_hole\_index,j) = SU(white\_hole\_index,j); \\
\quad\quad END\ IF \\
\quad END\ For \\
\quad END\ For \\
\quad For\ each\ universe\ indexed\ by\ \(i\) 
\]
For each object indexed by \( j \)

\[
r_2 = \text{random}([0,1]);
\]

\[
IF \ r_2 < \text{Wormhole existence probability}
\]

\[
r_3 = \text{random}([0,1]);
\]

\[
r_4 = \text{random}([0,1]);
\]

\[
IF \ r_3 < 0.5
\]

\[
U(i, j) = \text{Best universe}(j) + \text{Travelling distance rate} \ast ((ub_j - lb_j)) \ast r_4 + lb_j
\]

ELSE

\[
U(i, j) = \text{Best universe}(j) - \text{Travelling distance rate} \ast ((ub_j - lb_j)) \ast r_4 + lb_j
\]

END IF

END IF

END For

END For

HYBRID PSO-MVO (PMVO)

Hybrid PSO-MVO (PMVO) is sequential combination of PSO and MVO. The algorithm merges the best strength of both PSO in exploitation and MVO in exploration towards the optimum solution when the universe value of MVO replaces the \( P_{best} \) value of PSO (Sagarika, & Jyothsna, 2015) (Karthikeyan, & Dhal, 2015).

In this paper we propose a novel training algorithm based on this algorithm for the first time in the following section. The equation can be written as follows:

\[
V_{i,j}^{t+1} = V_{i,j}^t \ast W + C_1 R_1 (Universes^t - X^t) + C_2 R_2 (G_{best}^t - X^t)
\]  

The workflow of the PMVO can be demonstrated as follow:

**Step 1**: initialize the MVO values

**Step 2**: evaluate the inflation rate of the universe (fitness function)

**Step 3**: update the position of the universes

**Step 4**: if the convergence criterion is reached; get the results

**Step 5**: if the convergence criterion is not reached; continue the process from step 2-5.

**Step 6**: use the optimal solutions of MVO as boundary to PSO algorithm

**Step 7**: initialize the PSO values

**Step 8**: evaluate the fitness function of each particle

**Step 9**: determine \( G_{best} \) from the \( P_{best} \) value
Step 10: updated velocity and position values of each particle when the universe value of MVO replace the $P_{best}$ value of PSO.

Step 11: verify the solution whether it is feasible or not

Step 12: steps 8 to 12 were repeated until the maximum number of iterations was reached.

PMVO FOR TRAINING MLP

This section presents the proposed approach based on the PMVO to train the MLP network named PMVO-MLP. Two important points are taken into consideration: the fitness function and the representation of the PMVO solutions.

In this work, the PMVO algorithm was applied for training MLP network with a single hidden layer and each PMVO solution (weights and biases) was formed by three parts: the connection weights between the input layer and the hidden layer, the weights between the hidden layer and the output layer, and the bias weights. The length of each solution vector is given by equation (8), where $n$ is the number of input features and $m$ is the number of neurons in the hidden layer (Faris et al., 2016), as follows:

$$ \text{IndividualLength} = (n \times m) + (2 \times m) + 1 $$

(8)

PMVO solutions are implemented as real number vectors when each vector belongs to the interval [-1, 1], as illustrated in Figure 1. The mean square error (MSE) was used to measure the fitness value of PMVO solutions. MSE was calculated based on the difference between the estimated and actual values of the neural network using the training datasets, as shown in equation (9), where $n$ is the number of samples in the training dataset and $y, \hat{y}$ are respectively the actual and predicted values:

$$ \text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y - \hat{y})^2 $$

(9)

Figure 1. Representation of PMVO solution structure

| $w_{11}$ | $w_{12}$ | ... | $w_{nm}$ | $\varphi_1$ | ... | $\varphi_m$ | $\beta_1$ | ... | $\beta_{m+1}$ |
|---|---|---|---|---|---|---|---|---|---|

Weights between inputs and hidden

Weights between hidden layer and output layer

Bias Weights

The workflow of the PMVO for training MLP network can be demonstrated as follow:

Step 1: initialize the MVO values
Step 2: evaluate the inflation rate of the universe (fitness function): assignment of the initial vectors solutions (weights and biases) to MLP network and each network is evaluated by MSE fitness function;
Step 3: update the position of the universes
Step 4: if the convergence criterion is reached; get the results
Step 5: if the convergence criterion is not reached; continue the process from step 2-5.
Step 6: use the optimal solutions of MVO as boundary to PSO algorithm.
Step 7: initialize the PSO values.
Step 8: evaluate the fitness function of each particle: evaluation of the MLP network using MSE.
Step 9: determine $G_{best}$ from the $P_{best}$ value
Step 10: updated velocity and position values of each particle when the universe value of MVO replace the $P_{best}$ value of PSO.
Step 11: verify the solution whether it is feasible or not
Step 12: steps 8 to 12 were repeated until the maximum number of iterations was reached.

Figure 3 represents the steps of the PMVO-MLP approach.

EXPERIMENTS AND RESULTS

This section presents the evaluation of the proposed PMVO for training MLP networks on nine well-known datasets, which were selected from (UCI) and Kaggle dataset repositories. Table 1 shows the classification of these datasets in terms of features number, classes, training and testing simples.
The comparison of PMVO was carried out with five approaches used to train feedforward neural network in the literature: PSO (Mendes et al., 2002), MFO (Yamany, Fawzy, Tharwat, & Hassanien, 2015), MVO (Faris et al., 2016), WOA (Aljarah et al., 2018) and HACPSO (Khan et al., 2019).

In addition, the proposed algorithm was compared with standard momentum Back-Propagation and adaptive learning rate and (BP), which are gradient-based algorithms.

**Experimental Setup**

The proposed trainer and other algorithms were implemented with Python language and a personal computer with Intel(R) Core(TM) CPU 1.60 GHz 2.30 GHz, 64 Bits Windows 7 operating system and 4 GB (RAM).

The metaheuristics are sensitive to the value of their parameters, which requires a careful initialization. Therefore, the control parameters recommended in the literature were used (Mendes et al., 2002; Nawi et al., 2013; Faris et al., 2016; Aljarah et al., 2019) and summarized in Table 2. All datasets were divided into 66% for training and 34% for testing. Moreover, all features were mapped to the interval [0, 1] to eliminate the effect of features that have different scales.

**RESULTS**

Each dataset was divided into two parts to evaluate MLP, 66% for training and 34% for testing. All algorithms were tested ten times for every dataset and the MLP performance was evaluated according
to best, worst, average and standard deviation values of the classification accuracy and MSE. The population size and the maximum number of generations were set to 50 and 200, respectively.

Table 3 shows the statistical results: average, best, worst and standard deviation of classification accuracy. The results of PMVO outperformed other approaches in Breast cancer, Blood, Liver, Vertebral, Parkinson, Hepatitis and Heart with an average accuracy of 0.961, 0.765, 0.737, 0.837, 0.902, 0.973, and 0.755. In addition, PMVO gave the same results as MVO in diabetes and BreastEW datasets with an average accuracy of 0.788, 1.00 respectively. In terms of best, the proposed algorithm outperforms other algorithms in Parkinson, Hepatitis, Heart, and Breast EW with best values of 0.921, 0.989, 0.866, and 1.00 respectively. For worst classification accuracy the PMVO performs better in 7/9 datasets: Breast cancer, Blood, Diabetes, Liver, Parkinson, Hepatitis, and BreastEW with values of 0.958, 0.762, 0.782, 0.726, 0.882, 0.947, and 1.00 respectively. Moreover, it can also be seen that the PMVO has a smaller STD for 7/9 datasets: Breast cancer, Blood, Diabetes, Liver, Vertebral, Hepatitis, and BreastEW which indicates that PMVO is robust and stable.

Table 4 shows the average, best and worst MSE with standard deviation, obtained for each algorithm. As a result, it can be noted that PMVO outperforms other techniques in: breast cancer, blood, liver Vertebral, Parkinson, Heart and hepatitis with an average MSE of 0.032, 0.169, 0.186, 0.127, 0.080, 0.170, and 0.043 respectively. In terms of Best, it yields better results for 8/9 datasets: Breast cancer, Blood, Parkinson, Liver, Vertebral, Hepatitis, Heart and, Breast EW with best values of 0.030, 0.160, 0.175, 0.125, 0.071, 0.027, 0.113 and, and 0.002 respectively. For worst MSE it performs better than other approaches for 5/9 datasets: breast cancer, Blood, Liver, Heart, and Parkinson with worst values of 0.036, 0.174, 0.187, 0.194, and 0.089 respectively. In addition, it can be also noticed that PMVO has small standard deviation for all datasets which proves the robustness and efficiency of this algorithm.

Table 5 shows the average and standard deviation of sensitivity, specificity, precision, and F-Measure obtained for each algorithm. In terms of sensitivity, it can be seen that PMVO outperforms other approaches in five datasets: Diabetes, Breast cancer, Heart, Hepatitis, and breast.

In terms of precision, our algorithm performs better in five datasets: Parkinson, Breast cancer, Heart, Hepatitis, Breast EW. In specificity, the results of PMVO outperform other approaches in Diabetes, Blood, Parkinson, Breast cancer, Vertebral, Hepatitis, and Breast EW.

In terms of F-measure, it yields better results for eight datasets: Diabetes, Blood, Parkinson, Breast cancer, Heart, Vertebral, hepatitis, and Breast EW with average values of 0.779, 0.679, 0.893, 0.960, 0.742, 0.842, 0.963 and 1.00. Also, it can be noticed that PMVO has a small standard deviation value for all datasets which proves the efficiency of this algorithm.

Table 1. Summary of the classification datasets

| Data Sets   | Features | Training samples | Testing samples |
|-------------|----------|------------------|-----------------|
| Blood       | 4        | 493              | 255             |
| Breast cancer | 8       | 461              | 238             |
| Diabetes    | 8        | 506              | 262             |
| Vertebral   | 6        | 204              | 106             |
| Liver       | 6        | 79               | 41              |
| Parkinson   | 22       | 128              | 67              |
| Hepatitis   | 10       | 102              | 53              |
| Heart (Statlog) | 13   | 179              | 91              |
| BreastEW    | 30       | 376              | 193             |
Figures 4, 5, 6, 7, 8 show the convergence curves of all metaheuristic training algorithms based on the average values of MSE. The convergence curves show that PMVO has the lowest value of MSE for seven datasets: Breast cancer, Blood, Liver, Vertebral, Parkinson, Hepatitis and Heart.

For Diabetes and BreastEW datasets, PMVO provides the same performance as MVO algorithm. Moreover, PMVO has the fastest convergence speed in liver, vertebral. These results show that PMVO has a faster convergence and a better optimization than other metaheuristic algorithms.

Table 9 shows the average ranks obtained by each optimization technique in the Friedman test (the lower is better). The comparative study shows that the proposed training algorithm outperforms other algorithms.

The swarm intelligence algorithms such as PSO, ABC usually are unable to perform better than the evolutionary algorithms in exploration because these algorithms need to travel in search space and there is no abrupt change to move from one side to another side.

The performance of these algorithms benefits from good convergence rate and exploitation but it can be trapped in a local optimum. The combination of PSO and MVO gave better results than other trainers algorithms such as PSO and MVO for most datasets because the PMVO can achieve a good balance between exploration and exploitation.

After the experiments, it clear that the hybridization between PSO and MVO algorithms is having a major impact on a global search. In some cases, the original PSO can be stuck in a local optimum which leads to lower Precision, Accuracy, sensitivity, and Specificity. Moreover, combine PSO in exploitation with MVO in exploration give better results due to the superior exploration of the MVO algorithm; the black and white holes allow the objects to travel between the universes which promote exploration and avoid local optima. Abrupt universes changes are useful in the stagnation of local optima. We can conclude that MVO can contribute to exploration and PSO to exploitation. Good exploitation implies a convergence speed and a good exploration prevents the stagnation of local optima (Faris et al., 2016).

The hybrid algorithm PMVO helps to create a good trade-off between exploration and exploitation.

**Overview**

To evaluate the performance of the proposed PMVO-MLP in the field of information retrieval, more precisely, in twitter sentiment analysis we reproduce the paper results of (Despotovic, Tanikic, 2017) when the authors proposed sentiment analysis technique for microblogs based on feed-forward neural network. The experiments were tested on two datasets: the Stanford Twitter Sentiment and SemEval.
2014 Task 9. The results were compared to standard machine learning techniques, such as Support Vector Machines (SVM), Maximum Entropy (MaxEnt) and Multinomial Naive Bayes (MNB).

The authors used unigrams and bigrams as features and they applied feature redaction such as removing: infrequent words, stop words, usernames, hashtags, repeated letters, hyperlinks. Despotovic et al. replaced emoticons when the common positive word replaces the positive emotions and the negative word replaces the negatives ones. Despotovic et al. used only the feature that influences sentiment classification according to the mutual information.

In this paper, we compared our algorithm PMVO-MLP with results obtained in (Despotovic et al., 2017) and we used the same methods and datasets.

Firstly, we present the results obtained on the Stanford Twitter Sentiment. We selected a subset of features as training and testing data as mentioned in Table 6.

We used the unigram and a combination of unigram and bigram as features (Despotovic et al., 2017).

Table 7 indicates that our approach performs better than MLP, MaxEnt, SVM, and MNB with F-Measure equals to 0.8294. With unigram + bigram our approach outperforms MLP with F-Measure equals to 0.7988. We used PMVO-MLP and the same machine learning algorithms SVM, MNB and MaxEnt for SemEval 2014 Task 9 corpus, we used unigram and unigram +bigram as features. We compared our algorithm with MLP with one hidden layer.

Table 8 shows the F-Measure results of classification. The results of PMVO-MLP outperformed MLP with an improvement of 1%.

Compared to the other standard machine learning our algorithm gave the best results with unigram.

From the results, it can be noted that PMVO-MLP performs better than MLP in all cases and give superior or at least comparable results to standard machine learning techniques and can be applied in twitter sentiment analysis.

CONCLUSION

Inspired by no-free-lunch and motivated by local optima stagnation and the determination gabs of premature convergence, in this paper, we have proposed a new training approach based on Particle swarm optimization, Multi-verse optimization to train the feedforward neural network (FFNN). The training method took into account the capabilities of the PMVO in terms of high exploration and exploitation to locate the optimal values for weights and biases of FFNN. The approach is proposed to minimize the training error and to increase accuracy.

The approach is benchmarked and evaluated using nine standard bio-medical datasets. The comparison between the proposed algorithm and PSO, MFO, MVO, WOA, HACPSO and standard BP with momentum term and adaptive learning rate shows the superiority of the PMVO algorithm in term of measurement metrics of Accuracy, F-measure, Sensitivity, Precision and Specificity, and MSE in most of the datasets compared with other training algorithms. Moreover, the small value of the standard deviation shows that our trainer is robust and stable.

Finally, from the experiment, we can note that PMVO can give good results, and we can conclude that the PMVO algorithm is a good trainer for a feed-forward neural network with a good balance between exploration and exploitation, and can avoid local optima.

To evaluate the performance of proposed PMVO-MLP in Twitter sentiment analysis we compared our algorithm to standard machine learning techniques (MaxEnt, SVM, MNB, and FFNN). The experiments were tested on two datasets: the Stanford Twitter Sentiment and SemEval 2014 Task 9.

From the results, it can be noted that PMVO-MLP performs better than MLP in all cases and give superior or at least comparable results to standard machine learning techniques and can be applied in twitter sentiment analysis.
In future works, we focus on how to extend this work to solve function approximation and more real-world problems and we test the performance of PMVO to train other types of neural networks. As well as, improve the proposed algorithm to solve the big data optimization problems.

Table 3. Accuracy results

| DataSets / Algorithm | Planar | PSO | MFO | MVO | WOA | HAC-PSO | BP |
|----------------------|--------|-----|-----|-----|-----|---------|----|
| Breast cancer        | Avg    | 0.961 | 0.959 | 0.958 | 0.958 | 0.956 | 0.959 | 0.744 |
|                      | Std    | 0.001 | 0.003 | 0.002 | 0.002 | 0.006 | 0.004 | 0.254 |
|                      | Best   | 0.963 | 0.960 | 0.963 | 0.960 | 0.963 | 0.965 | 0.945 |
|                      | Worst  | 0.958 | 0.953 | 0.954 | 0.954 | 0.947 | 0.952 | 0.680 |
| Blood                | Avg    | 0.765 | 0.762 | 0.763 | 0.764 | 0.762 | 0.760 | 0.744 |
|                      | Std    | 0.002 | 0.002 | 0.005 | 0.009 | 0.004 | 0.003 | 0.254 |
|                      | Best   | 0.767 | 0.765 | 0.765 | 0.784 | 0.774 | 0.760 | 0.945 |
|                      | Worst  | 0.762 | 0.760 | 0.760 | 0.760 | 0.758 | 0.758 | 0.680 |
| Diabetes             | Avg    | 0.788 | 0.780 | 0.724 | 0.788 | 0.720 | 0.768 | 0.619 |
|                      | Std    | 0.004 | 0.011 | 0.008 | 0.006 | 0.028 | 0.006 | 0.080 |
|                      | Best   | 0.794 | 0.796 | 0.735 | 0.802 | 0.750 | 0.774 | 0.690 |
|                      | Worst  | 0.782 | 0.758 | 0.709 | 0.782 | 0.657 | 0.764 | 0.601 |
| Liver                | Avg    | 0.737 | 0.722 | 0.722 | 0.725 | 0.665 | 0.679 | 0.519 |
|                      | Std    | 0.006 | 0.017 | 0.010 | 0.008 | 0.030 | 0.020 | 0.055 |
|                      | Best   | 0.744 | 0.748 | 0.744 | 0.732 | 0.700 | 0.709 | 0.586 |
|                      | Worst  | 0.726 | 0.700 | 0.709 | 0.709 | 0.603 | 0.669 | 0.495 |
| Vertebral            | Avg    | 0.837 | 0.836 | 0.836 | 0.836 | 0.774 | 0.817 | 0.651 |
|                      | Std    | 0.004 | 0.012 | 0.007 | 0.005 | 0.052 | 0.015 | 0.170 |
|                      | Best   | 0.843 | 0.848 | 0.843 | 0.838 | 0.862 | 0.833 | 0.866 |
|                      | Worst  | 0.823 | 0.808 | 0.820 | 0.828 | 0.676 | 0.794 | 0.627 |
| Parkinson            | Avg    | 0.902 | 0.836 | 0.802 | 0.890 | 0.824 | 0.876 | 0.750 |
|                      | Std    | 0.013 | 0.040 | 0.022 | 0.008 | 0.016 | 0.013 | 0.199 |
|                      | Best   | 0.921 | 0.898 | 0.828 | 0.906 | 0.852 | 0.905 | 0.849 |
|                      | Worst  | 0.882 | 0.773 | 0.773 | 0.835 | 0.800 | 0.863 | 0.623 |
| Hepatitis            | Avg    | 0.973 | 0.914 | 0.866 | 0.867 | 0.791 | 0.854 | 0.758 |
|                      | Std    | 0.013 | 0.017 | 0.016 | 0.014 | 0.063 | 0.015 | 0.155 |
|                      | Best   | 0.989 | 0.947 | 0.894 | 0.894 | 0.867 | 0.875 | 0.865 |
|                      | Worst  | 0.947 | 0.884 | 0.842 | 0.842 | 0.695 | 0.828 | 0.669 |
| Heart                | Avg    | 0.755 | 0.704 | 0.733 | 0.741 | 0.703 | 0.738 | 0.651 |
|                      | Std    | 0.063 | 0.019 | 0.044 | 0.039 | 0.064 | 0.020 | 0.020 |
|                      | Best   | 0.866 | 0.722 | 0.816 | 0.827 | 0.822 | 0.761 | 0.704 |
|                      | Worst  | 0.661 | 0.665 | 0.700 | 0.700 | 0.577 | 0.705 | 0.577 |
| BreastEW             | Avg    | 1.000 | 0.966 | 0.993 | 1.000 | 0.973 | 0.994 | 0.865 |
|                      | Std    | 0.000 | 0.014 | 0.006 | 0.000 | 0.026 | 0.006 | 0.080 |
|                      | Best   | 1.000 | 0.994 | 1.000 | 1.000 | 0.994 | 1.000 | 0.890 |
|                      | Worst  | 1.000 | 0.955 | 0.978 | 1.000 | 0.907 | 0.981 | 0.884 |
Table 4. MSE Results

| DataSets/Algorithm | PMVO | PSO  | MFO  | MVO  | WOA  | HAC-PSO | BP   |
|--------------------|------|------|------|------|------|---------|------|
| Breast cancer      | 0.032| 0.032| 0.033| 0.032| 0.047| 0.040   | 0.049|
| Std                | 0.001| 0.001| 0.001| 0.002| 0.004| 0.001   | 0.015|
| Best               | 0.030| 0.030| 0.031| 0.030| 0.043| 0.038   | 0.030|
| Worst              | 0.036| 0.036| 0.036| 0.032| 0.058| 0.041   | 0.050|
| Blood              | 0.169| 0.178| 0.176| 0.173| 0.180| 0.169   | 0.174|
| Avg                | 0.005| 0.003| 0.005| 0.008| 0.004| 0.003   | 0.009|
| Std                | 0.160| 0.172| 0.174| 0.170| 0.174| 0.162   | 0.172|
| Best               | 0.174| 0.182| 0.177| 0.181| 0.187| 0.175   | 0.175|
| Parkinson          | 0.149| 0.155| 0.171| 0.149| 0.186| 0.163   | 0.179|
| Avg                | 0.001| 0.004| 0.005| 0.001| 0.013| 0.002   | 0.066|
| Std                | 0.147| 0.151| 0.165| 0.145| 0.168| 0.160   | 0.168|
| Best               | 0.151| 0.157| 0.175| 0.151| 0.213| 0.166   | 0.180|
| Worst              | 0.186| 0.191| 0.193| 0.188| 0.220| 0.212   | 0.210|
| Std                | 0.004| 0.003| 0.002| 0.004| 0.007| 0.002   | 0.003|
| Best               | 0.175| 0.185| 0.189| 0.179| 0.210| 0.208   | 0.190|
| Blood              | 0.127| 0.134| 0.136| 0.130| 0.163| 0.146   | 0.168|
| Avg                | 0.002| 0.002| 0.002| 0.002| 0.018| 0.002   | 0.015|
| Std                | 0.125| 0.132| 0.135| 0.129| 0.137| 0.142   | 0.160|
| Best               | 0.130| 0.135| 0.137| 0.131| 0.202| 0.147   | 0.172|
| Parkinson          | 0.080| 0.141| 0.158| 0.089| 0.165| 0.119   | 0.158|
| Avg                | 0.006| 0.023| 0.019| 0.020| 0.030| 0.005   | 0.018|
| Std                | 0.071| 0.099| 0.135| 0.080| 0.127| 0.112   | 0.137|
| Best               | 0.089| 0.182| 0.203| 0.125| 0.228| 0.130   | 0.203|
| Worst              | 0.043| 0.089| 0.119| 0.116| 0.143| 0.090   | 0.168|
| Hepatitis          | 0.015| 0.016| 0.011| 0.010| 0.020| 0.003   | 0.015|
| Std                | 0.027| 0.056| 0.100| 0.102| 0.109| 0.084   | 0.159|
| Best               | 0.056| 0.117| 0.138| 0.130| 0.176| 0.094   | 0.173|
| Parkinson          | 0.070| 0.178| 0.178| 0.203| 0.191| 0.205   | 0.173|
| Avg                | 0.034| 0.022| 0.021| 0.033| 0.041| 0.017   | 0.060|
| Std                | 0.113| 0.189| 0.133| 0.153| 0.182| 0.194   | 0.243|
| Best               | 0.194| 0.208| 0.196| 0.198| 0.243| 0.199   | 0.243|
| BreastEW           | 0.004| 0.040| 0.017| 0.005| 0.041| 0.024   | 0.040|
| Avg                | 0.001| 0.009| 0.005| 0.001| 0.024| 0.004   | 0.015|
| Std                | 0.002| 0.054| 0.008| 0.004| 0.010| 0.009   | 0.040|
| Best               | 0.006| 0.022| 0.023| 0.005| 0.070| 0.024   | 0.075|

*Avg=average; Std= standard deviation
Table 5. Training specificity, sensitivity, precision, and f-measure

|                  | F-Measure | Sensitivity | Precision | Specificity |
|------------------|-----------|-------------|-----------|-------------|
| Diabetes         | 0.779+/-.007 | 0.789+/-.005 | 0.779+/-.006 | 0.668+/-.007 |
|                  | 0.776+/-.011 | 0.779+/-.007 | 0.779+/-.005 | 0.651+/-.007 |
|                  | 0.776+/-.007 | 0.785+/-.005 | 0.780+/-.007 | 0.668+/-.007 |
|                  | 0.771+/-.013 | 0.782+/-.011 | 0.779+/-.011 | 0.660+/-.009 |
|                  | 0.668+/-.005 | 0.721+/-.028 | 0.709+/-.038 | 0.540+/-.064 |
|                  | 0.765+/-.006 | 0.763+/-.006 | 0.767+/-.008 | 0.642+/-.015 |
| Liver            | 0.717+/-.012 | 0.724+/-.016 | 0.721+/-.008 | 0.668+/-.007 |
|                  | 0.722+/-.015 | 0.729+/-.015 | 0.729+/-.009 | 0.682+/-.009 |
|                  | 0.721+/-.011 | 0.724+/-.011 | 0.724+/-.011 | 0.674+/-.010 |
|                  | 0.666+/-.003 | 0.666+/-.008 | 0.666+/-.008 | 0.595+/-.043 |
|                  | 0.675+/-.007 | 0.675+/-.021 | 0.675+/-.021 | 0.251+/-.027 |
| Blood            | 0.679+/-.040 | 0.667+/-.018 | 0.697+/-.023 | 0.666+/-.005 |
|                  | 0.674+/-.009 | 0.676+/-0.006 | 0.767+/-.008 | 0.666+/-.013 |
|                  | 0.674+/-.009 | 0.676+/-0.006 | 0.767+/-.008 | 0.666+/-.013 |
|                  | 0.664+/-.015 | 0.670+/-0.111 | 0.792+/-.008 | 0.666+/-.013 |
|                  | 0.677+/-.013 | 0.683+/-0.017 | 0.682+/-0.009 | 0.674+/-0.010 |
| Parkinson        | 0.893+/-.009 | 0.823+/-0.033 | 0.875+/-0.010 | 0.815+/-0.033 |
|                  | 0.901+/-.008 | 0.821+/-0.037 | 0.890+/-0.010 | 0.815+/-0.033 |
|                  | 0.902+/-.010 | 0.822+/-0.029 | 0.891+/-0.010 | 0.815+/-0.033 |
|                  | 0.751+/-.055 | 0.724+/-0.39  | 0.725+/-0.25  | 0.666+/-.038 |
|                  | 0.829+/-.101 | 0.254+/-0.029 | 0.264+/-0.069 | 0.205+/-0.03  |
|                  | 0.829+/-.101 | 0.254+/-0.029 | 0.264+/-0.069 | 0.205+/-0.03  |
| Breast cancer    | 0.960+/-.000 | 0.959+/-0.003 | 0.959+/-0.003 | 0.959+/-0.005 |
|                  | 0.960+/-.000 | 0.959+/-0.003 | 0.959+/-0.003 | 0.959+/-0.005 |
|                  | 0.960+/-.000 | 0.959+/-0.003 | 0.959+/-0.003 | 0.959+/-0.005 |
|                  | 0.960+/-.000 | 0.960+/-0.010 | 0.960+/-0.006 | 0.956+/-0.005 |
|                  | 0.961+/-.000 | 0.954+/-0.005 | 0.961+/-0.006 | 0.966+/-0.006 |
| Heart            | 0.742+/-.057 | 0.702+/-0.024 | 0.734+/-0.043 | 0.714+/-0.040 |
|                  | 0.747+/-.053 | 0.711+/-0.018 | 0.725+/-0.025 | 0.720+/-0.040 |
|                  | 0.754+/-.050 | 0.720+/-0.015 | 0.732+/-0.009 | 0.728+/-0.036 |
|                  | 0.707+/-.073 | 0.656+/-0.044 | 0.682+/-0.064 | 0.690+/-0.043 |
|                  | 0.842+/-0.006 | 0.833+/-0.013 | 0.833+/-0.007 | 0.831+/-0.005 |
|                  | 0.842+/-0.006 | 0.833+/-0.013 | 0.901+/-0.008 | 0.831+/-0.005 |
|                  | 0.842+/-0.006 | 0.836+/-0.013 | 0.900+/-0.009 | 0.831+/-0.005 |
|                  | 0.842+/-0.006 | 0.836+/-0.013 | 0.900+/-0.009 | 0.831+/-0.005 |
|                  | 0.842+/-0.006 | 0.836+/-0.013 | 0.900+/-0.009 | 0.831+/-0.005 |
| Vertebral        | 0.809+/-.010 | 0.797+/-0.021 | 0.735+/-0.021 | 0.793+/-0.008 |
|                  | 0.963+/-.021 | 0.895+/-0.041 | 0.957+/-0.019 | 0.926+/-0.019 |
|                  | 0.966+/-.017 | 0.907+/-0.037 | 0.962+/-0.015 | 0.932+/-0.019 |
|                  | 0.966+/-.017 | 0.919+/-0.028 | 0.956+/-0.022 | 0.935+/-0.014 |
|                  | 0.886+/-0.065 | 0.674+/-0.012 | 0.869+/-0.069 | 0.758+/-0.070 |
| Hepatitis        | 1.00+/-.0000 | 0.975+/-0.009 | 1.00+/-.0000 | 0.994+/-0.007 |
|                  | 1.00+/-.0000 | 0.975+/-0.009 | 1.00+/-.0000 | 0.994+/-0.007 |
|                  | 1.00+/-.0000 | 0.975+/-0.009 | 1.00+/-.0000 | 0.994+/-0.007 |
|                  | 1.00+/-.0000 | 0.960+/-0.014 | 1.00+/-.0000 | 0.988+/-0.012 |
|                  | 1.00+/-.0000 | 0.960+/-0.014 | 1.00+/-.0000 | 0.988+/-0.012 |
|                  | 1.00+/-.0000 | 0.960+/-0.014 | 1.00+/-.0000 | 0.988+/-0.012 |
Table 6. Twitter sentiment analysis datasets

| Data Sets            | Training Samples | Selected Training samples | Testing samples | Number of classes |
|----------------------|------------------|---------------------------|-----------------|------------------|
| Stanford Twitter Sentiment | 1.6 million of tweets | 50000 | 359 | 2 |
| SemEval 2014 task 9   | 9353             | 9353                      | 8055            | 3 |

Table 7. Performance of Stanford Twitter Sentiment corpus

| Stanford Twitter Sentiment corpus | MNB  | SVM    | MaxEnt | MLP   | MVO-MLP |
|----------------------------------|------|--------|--------|-------|---------|
| Unigram F-Measure                | 0.7911 | 0.7994 | 0.7688 | 0.8106 | 0.8294  |
| Uni+Bigram F-Measure             | 0.7604 | 0.7883 | 0.7939 | 0.7855 | 0.7988  |

Table 8. Performance of SemEval 2014 task 9 corpus

| SemEval 2014 task 9 corpus | MNB  | SVM    | MaxEnt | MLP   | MVO-MLP |
|----------------------------|------|--------|--------|-------|---------|
| Unigram F-Measure          | 0.6007 | 0.6493 | 0.6493 | 0.6461 | 0.6507  |
| Uni+Bigram F-Measure       | 0.6123 | 0.6540 | 0.6579 | 0.6479 | 0.6561  |

Table 9. Average Rankings of the techniques (Friedman)

| Algorithms | Ranking |
|------------|---------|
| PMVO       | 1.11    |
| MVO        | 2.38    |
| PSO        | 3.72    |
| HACPSO     | 4.05    |
| MFO        | 4.11    |
| WOA        | 5.61    |
| BP         | 7.00    |
Figure 4. MSE convergence curve for Breast cancer and Blood, respectively

Figure 5. MSE convergence curve for Diabetes and Liver, respectively

Figure 6. MSE convergence curve for Vertebral and Parkinson, respectively
Figure 7. MSE convergence curve for Hepatitis and Heart, respectively

Figure 8. MSE convergence curve for BreastEW
REFERENCES

Aljarah, I., Faris, H., & Mirjalili, S. (2018). Optimizing connection weights in neural networks using the whale optimization algorithm. *Soft Computing, 22*(1), 1–15. doi:10.1007/s00500-016-2442-1

Aljarah, I., Faris, H., Mirjalili, S., & Al-Madi, N. (2018). Training radial basis function networks using biogeography-based optimizer. *Neural Computing & Applications, 29*(7), 529–553. doi:10.1007/s00521-016-2559-2

Bolaji, A., Ahmad, A., & Shola, P. (2016). Training of neural network for pattern classification using fireworks algorithm. *International Journal of System Assurance Engineering and Management, 9*(1), 208–215.

Dai, X., Yuan, X., & Zhang, Z. (2015). A self-adaptive multi-objective harmony search algorithm based on harmony memory variance. *Applied Soft Computing, 35*, 541–557. doi:10.1016/j.asoc.2015.06.027

Despotovic, V., & Tanikic, D. (2017). Sentiment Analysis of Microblogs Using Multilayer Feed-Forward Artificial Neural Networks. *Computer Information, 36*(5), 1127–1142. doi:10.4149/cai_2017_5_1127

Faris, H., Aljarah, I., Al-Madi, N., & Mirjalili, S. (2016). Optimizing the Learning Process of Feedforward Neural Networks Using Lightning Search Algorithm. *International Journal of Artificial Intelligence Tools, 25*(06), 1650033. doi:10.1142/S0218213016500330

Faris, H., Aljarah, I., & Mirjalili, S. (2016). Training feedforward neural networks using multi-verse optimizer for binary classification problems. *Applied Intelligence, 45*(2), 322–332. doi:10.1007/s10489-016-0767-1

Faris, H., Aljarah, I., & Mirjalili, S. (2017). Evolving Radial Basis Function Networks Using Moth–Flame Optimizer. *Handbook of Neural Computation, 537–550*. doi:doi:10.1016/B978-0-12-811318-9.00028-4

Faris, H., Aljarah, I., & Mirjalili, S. (2017). Improved monarch butterfly optimization for unconstrained global search and neural network training. *Applied Intelligence, 48*(2), 445–464. doi:10.1007/s10489-017-0967-3

Faris, H., Mirjalili, S., & Aljarah, I. (2019). Automatic selection of hidden neurons and weights in neural networks using grey wolf optimizer based on a hybrid encoding scheme. *International Journal of Machine Learning and Cybernetics, 10*(10), 2901–2920. Advance online publication. doi:10.1007/s13042-018-00913-2

Green, R. I. I., Wang, L., & Alam, M. (2012). Training neural networks using Central Force Optimization and Particle Swarm Optimization: Insights and comparisons. *Expert Systems with Applications, 39*(1), 555–563. doi:10.1016/j.eswa.2011.07.046

Hakli, H., & Uğuz, H. (2014). A novel particle swarm optimization algorithm with Levy flight. *Applied Soft Computing, 23*, 333–345. doi:10.1016/j.asoc.2014.06.034

Hassanin, M. F., Shoeb, A. M., & Hassanien, A. E. (2016). Grey wolf optimizer-based back-propagation neural network algorithm. *2016 12th International Computer Engineering Conference (ICENCO)*. doi:doi:10.1109/ICENCO.2016.7856471

Jaddi, N. S., Abdullah, S., & Hamdan, A. R. (2015). Optimization of neural network model using modified bat-inspired algorithm. *Applied Soft Computing, 37*, 71–86. doi:10.1016/j.asoc.2015.08.002

Jangir, P., Parmar, S., Trivedi, I., & Bhesdadiya, R. (2017). A novel hybrid Particle Swarm Optimizer with multi verse optimizer for global numerical optimization and Optimal Reactive Power Dispatch problem. *Engineering Science And Technology, International Journal (Toronto, Ont.), 20*(2), 570–586. doi:10.1016/j.jestch.2016.10.007

Karthikeyan, K., & Dhal, P. (2015). Transient Stability Enhancement by Optimal Location and Tuning of STATCOM Using PSO. *Procedia Technology, 21*, 345–351. doi:10.1016/j.protcy.2015.10.046

Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. *Proceedings of IEEE International Conference on Neural Networks. ICNN.1995.488968 IV, 1942–1948*. doi:10.1109/ICNN.1995.488968

Kenter, T., Borisov, A., Gyssel, C. V., Dehghani, M., Rijke, M. D., & Mitra, B. (2018). Neural Networks for Information Retrieval. *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining - WSDM 18*. doi:10.1145/3159652.3162009
Khan, A., Shah, R., Imran, M., Khan, A., Bangash, J. I., & Shah, K. (2019). An alternative approach to neural network training based on hybrid bio meta-heuristic algorithm. *Journal of Ambient Intelligence and Humanized Computing, 10*(10), 3821–3830. Advance online publication. doi:10.1007/s12652-019-01373-4

Kiranyaz, S., Ince, T., Yildirim, A., & Gabbouj, M. (2009). Evolutionary artificial neural networks by multidimensional particle swarm optimization. [PubMed]. *Neural Networks, 22*(10), 1448–1462. doi:10.1016/j.neunet.2009.05.013

Melin, P., Olivas, F., Castillo, O., Valdez, F., Soria, J., & Valdez, M. (2013). Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic. *Expert Systems with Applications, 40*(8), 3196–3206. doi:10.1016/j.eswa.2012.12.033

Mendes, R., Cortez, P., Rocha, M., & Neves, J. (2002). Particle swarms for feedforward neural network training. *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN02* (Cat. No.02CH37290). doi:10.1109/IJCNN.2002.1007808

Mirjalili, S. (2015). How effective is the Grey Wolf optimizer in training multi-layer perceptrons. *Applied Intelligence, 43*(1), 150–161. doi:10.1007/s10489-014-0645-7

Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2015). Multi-Verse Optimizer: A nature-inspired algorithm for global optimization. *Neural Computing & Applications, 27*(2), 495–513. doi:10.1007/s00521-015-1870-7

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Let a biogeography-based optimizer train your Multi-Layer Perceptron. *Information Sciences, 269*, 188–209. doi:10.1016/j.ins.2014.01.038

Nandy, S. (2012). Training a Feed-Forward Neural Network with Artificial Bee Colony based Backpropagation Method. *International Journal of Computer Science and Information Technologies, 4*(4), 33–46. doi:10.5121/ijcst.2012.4404

Nawi, N. M., Rehman, M. Z., & Khan, A. (2014). A New Bat Based Back-Propagation (BAT-BP) Algorithm. *Advances in Intelligent Systems and Computing, 288*, 319–329. doi:10.1007/978-3-319-01857-7_38

Raidl, G. R. (2006). A Unified View on Hybrid Metaheuristics. Hybrid Metaheuristics Lecture Notes in Computer Science, 1–12. doi:10.1007/11890584_1

Sagarika & Jyothsna, T. (2015). Tuning of PSO algorithm for single machine and multi machine power system using STATCOM controller. *IACSIT International Journal of Engineering and Technology, 2*(4), 175–182.

Tarkhan, O., & Shen, H. (2019). Training of feedforward neural networks for data classification using hybrid particle swarm optimization, Mantegna Lévy flight and neighborhood search. [PubMed]. *Heliyon (London), 5*(4), e01275. Advance online publication. doi:10.1016/j.heliyon.2019.e01275

Valian, E., Mohanna, S., & Tavakoli, S. (2011). Improved Cuckoo Search Algorithm for Feed forward Neural Network Training. *International Journal of Artificial Intelligence & Applications, 2*(3), 36–43. doi:10.5121/ijaia.2011.2304

Wang, L., Li, Y., Huang, J., & Lazebnik, S. (2019). Learning Two-Branch Neural Networks for Image-Text Matching Tasks. [PubMed]. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 41*(2), 394–407. doi:10.1109/TPAMI.2018.2797921

Yamany, W., Fawzy, M., Tharwat, A., & Hassanien, A. E. (2015). Moth-flame optimization for training Multi-Layer Perceptrons. *2015 11th International Computer Engineering Conference (ICENCO)*. doi:10.1109/ICENCO.2015.7416360

Yang, X.-S. (2010). A New Metaheuristic Bat-Inspired Algorithm. *Nature Inspired Cooperative Strategies for Optimization (NICOSEO) 2010* Studies in Computational Intelligence, 65–74. doi:10.1007/978-3-642-12538-6_6

Zhang, J.-R., Zhang, J., Lok, T.-M., & Lyu, M. R. (2007). A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training. *Applied Mathematics and Computation, 185*(2), 1026–1037. doi:10.1016/j.amc.2006.07.025
Reda Mohamed Hamou received an engineering degree in computer Science from the Computer Science department of Djillali Liabes University of Sidi-Belabbes-Algeria and PhD (Artificial intelligence) from the same University. He has several publications in the field of BioInspired and Metaheuristic in many journals as IJAMC, IJIRR, IJAEC, IJALR, IJISP, IJIIT, JITR, IJCINI, IJOCI, IJSIR, IJSI, IJAIS, IJDSST, IJBR, Applied Intelligence. His research interests include Data Mining, Text Mining, Classification, Clustering, computational intelligence, neural networks, evolutionary computation and Biomimetic optimization method. He is a head of research team in GecoDe laboratory. Dr. Hamou is an associate professor in technology faculty in UTMS University of Saida-Algeria.

Abdelmalek Amine received an engineering degree in Computer Science, a Magister diploma in Computational Science and PhD from Djillali Liabes University in collaboration with Joseph Fourier University of Grenoble. His research interests include IoT, big data, data mining, text mining, ontology, classification, clustering, neural networks, and biomimetic optimization methods. He participates in the program committees of several international conferences and on the editorial boards of international journals. Prof. Amine is the head of GeCoDe-knowledge management and complex data-laboratory at UTM University of Saida, Algeria; he also collaborates with the “knowledge base and database” team of TIMC laboratory at Joseph Fourier University of Grenoble.