A Modified Invasive Weed Optimization Algorithm for Training of Feed-Forward Neural Networks

Ritwik Giri[1], Aritra Chowdhury[1], Arnob Ghosh[1], Swagatam Das[1], Ajith Abraham[2] and Vaclav Snasel[3]
[1]Department of Electronics and Telecommunication Engineering, Jadavpur University, India
[2] Machine Intelligence Resarch Labs (MIR Labs), USA. Email: ajith.abraham@ieee.org
[3]VSB Technical University Ostrava, Czech Republic. Email: vaclav.snasel@vsb.cz

Abstract— **Invasive Weed Optimization Algorithm IWO) is an ecologically inspired metaheuristic that mimics the process of weeds colonization and distribution and is capable of solving multi-dimensional, linear and nonlinear optimization problems with appreciable efficiency. In this article a modified version of IWO has been used for training the feed-forward Artificial Neural Networks (ANNs) by adjusting the weights and biases of the neural network. It has been found that modified IWO performs better than another very competitive real parameter optimizer called Differential Evolution (DE) and a few classical gradient-based optimization algorithms in context to the weight training of feed-forward ANNs in terms of learning rate and solution quality. Moreover, IWO can also be used in validation of reached optima and in the development of regularization terms and non-conventional transfer functions that do not necessarily provide gradient information**

Keywords: *metaheuristics, invasive weed optimization, differential evolution, feed-forward neural networks, classification, back-propagation.*

I. Introduction

Since the advent of the Artificial Neural Network (ANN) it is severely used in the field of pattern recognition and function approximation. Among various kinds of ANNs, Feed-Forward Artificial Neural Networks (FFANN) are considered to be powerful tools in the area of pattern classification [1] where universal FFANN approximators, for arbitrary finite-input environment measures, can be constituted by using only a single hidden layer [2]. The technique involves training of the FFANN with the dataset to be recognized. The process of training an ANN is concerned with adjusting the weights between each pair of the individual neurons and corresponding biases until a close approximation of the desired output is achieved. Usually ANN, unless specified, uses Back Propagation (BP) algorithm for training purposes [3, 4]. The BP algorithm is a trajectory driven technique, which are analogous to an error minimizing process. BP learning requires the neuron transfer function to be differentiable and it also suffers from the possibility of falling into the local optima. BP is also known to be sensitive to the initial weight settings and many weight initialization techniques have been proposed to lessen such a possibility [5, 6, and 7]. So, BP is considered to be inefficient in searching for global minimum of the search space [8].

The computational drawbacks of existing derivative-based numerical methods have forced the researchers all over the world to rely on metaheuristic algorithms founded on simulations to solve engineering optimization problems. A common factor shared by the metaheuristics is that they combine rules and randomness to imitate some natural phenomena. Two closely related families of algorithms that primarily constitute this field today are the Evolutionary Algorithms (EAs) [9 – 11] and the Swarm Intelligence (SI) algorithms [12 – 14]. While the EAs emulate the processes of Darwinian evolution and natural genetics, the SI algorithms draw inspiration from the collective intelligence emerging from the behavior of a group of social insects (like bees, termites and wasps) and also from the socio-cognition theory of human beings.

To overcome the shortcomings of BP in training the ANN, metaheuristic ANN training models i.e. the combination of stochastic optimization algorithms like Genetic Algorithms (GAs) [9, 10], Particle Swarm Optimization (PSO) [11, 12], and Differential Evolution (DE) [13] with the ANN learning process has been proposed. A survey and overview of the evolutionary techniques in evolving ANN can be found in [10]. This kind of evolutionary ANN models do not exhibit the inefficiencies of BP algorithms like need of the differentiability of the neuron transfer function, possibility of getting trapped in a local optima etc. Further the search techniques of the evolutionary models are population driven instead of the trajectory driven techniques of the BP.

The common evolutionary techniques are biologically inspired stochastic global optimization methods. They have one common underlying idea behind them, which is based on a population of individuals [11]. Environmental pressure causes natural selection that in turn causes a rise in the fitness of the population. An objective (fitness) function represents a heuristic estimation of solution quality and the variation and selection operators drive the search process. Such process is iterated until convergence is reached. The best population member is expected to be a near-optimum solution [12].

Using a suitable ANN representation, the process of supervised ANN training using an evolutionary method involves performing several iterations in order to minimize or maximize a certain fitness function [8, 13, and 14]. Such optimization process would usually stochastically generate vectors representing the network's weight values, including biases, calculate the fitness for the generated vectors and tries to keep those vectors that give better fitness values. It is possible also to include the ANN structure in such representation where the structure can also evolve [15]. The cycle is repeated to generate new offspring and eventually after several iterations the training process is halted based on some criteria.

In recent past Mehrabian and Lucas proposed the Invasive Weed Optimization (IWO) [16], a derivative-free, metaheuristic algorithm, mimicking the ecological behavior of colonizing weeds. Since its inception, IWO has found successful applications in many practical optimization problems like optimization and tuning of a robust controller [16], optimal positioning of piezoelectric actuators [17], developing a recommender system [18], design of E-shaped MIMO Antenna [19], and design of encoding sequences for DNA computing [20]. In this article IWO, with a modification from it's original self has been used as an evolutionary optimization technique to train artificial neural network for the purpose of pattern recognition and function approximation. A single case for function approximation and three instances for pattern recognition have been used to illustrate the application of the proposed algorithm. Comparison with results obtained by another very common and largely used evolutionary algorithm DE [21], and three common back propagation algorithms namely gradient descent BP, resilient BP, one step secant BP establishes the superiority of the proposed method.

The rest of the paper is organized in the following way. Section II outlines the method to construct the FFANN structure and it's details, Section III gives a short description of the IWO algorithm along with it's modification, Section IV describes the performance index, Section V represents the results on various datasets by IWO and comparison with the competiting algorithms, Section V finally concludes the paper and unfold some future research works.

II. Proposed Methodology

A. FFANN Model

Artificial Neural networks are highly interconnected simple processing units designed in a way to model how the human brain performs a particular task. Each of those units, also called neurons, forms a weighted sum of its inputs, to which a constant term called bias is added. This sum is then passed through a transfer function: linear, sigmoid or hyperbolic tangent. Figure1 shows the internal structure of a neuron.

Multi-Layer Perceptrons (MLP) are the best known and most widely used kind of neural network. Networks with interconnections that do not form any loops are called Feed Forward Artifical Neural Network (FFANN). The units are organized in a way that defines the network architecture. In feed forward networks, units are often arranged in layers: an input layer, one or more hidden layers and an output layer. The units in each layer may share the same inputs, but are not connected to each other. Typically, the units in the input layer serve only for transferring the input pattern to the rest of the network, without any processing. The information is processed by the units in the hidden and output layers. Figure 2 depicts the architecture of a generic three-layered FFANN model.

The neural network considered is fully connected in the sense that every unit belonging to each layer is connected to every unit belonging to the adjacent layer. In order to find the optimal network architecture, several combinations were evaluated. These combinations included networks with different number of hidden layers, different number of units in each layer and different types of transfer functions. We converged to a configuration consisting of a one hidden layer, one input layer and one output layer.

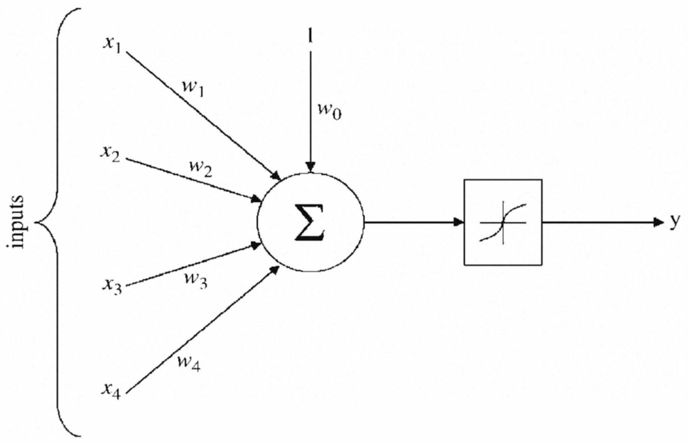


Figure 1: Structure of a neuron

However the no. of neurons of each layer and the transfer function of each layer varies upon different problems of function approximation and pattern recognition. The structure of the neural network for each case has been described later in different cases.

This configuration has been proven to be a universal mapper, provided that the hidden layer has enough units. On the one hand, if there are too few units, the network will not be flexible enough to model the data well and, on the other hand, if there are too many units, the network may overfit the data. Typically, the number of units in the hidden layer is chosen by trial and error, selecting a few alternatives and then running simulations to find out the one with the best results. Training of feed forward networks is normally performed in a supervised manner. One assumes that a training set is available, given by the dataset, containing both inputs and the corresponding desired outputs, which is presented to the network. Evolutionary Algorithm has been used in this training by choosing the appropriate values of weights and biases of the ANN to minimize the training error of the corresponding problem. The error minimization process is repeated until an acceptable criterion for convergence is reached. The knowledge acquired by the neural network through the learning process is tested by applying new data that it has never seen before, called the testing set. The network should be able to generalize and have an accurate output for this unseen data. It is undesirable to overtrain the neural network, meaning that the network would only work well on the training set, and would not generalize well to new data outside the training set. In case of ANN the most common learning algorithm is the back propagation algorithm. However, the standard back propagation learning algorithm is not efficient numerically and tends to converge slowly. To improve the results in case of training we have used an ecologically inspired algorithm IWO rather than the BP algorithms and has found that proposed algorithm can outperform the others i. e. DE, traingd, trainoss, trainrp etc.

III. CLASSICAL IWO AND ITS MODIFICATION

Invasive Weed Optimization (IWO) is a metaheuristic algorithm that mimics the colonizing behavior of weeds. IWO can be summarized as follows.

A. Initialization

A finite number of weeds are initialized randomly in the entire search space. For the training purpose of the FFANN, each weed consists of a string of network weights followed by network biases. So the i'th weed $\vec{w_i}$ can be represented as

$$\overrightarrow{W_i} = [w_{i,1}, w_{i,2}, \ldots, w_{i,n}, b_{i,1}, b_{i,2}, \ldots, b_{i,m}]$$

$w_{i,p}$ = p'th weight term of the network

$b_{i,q}$ = q'th bias term of the network and n & m being the total number of weights and biases respectively.

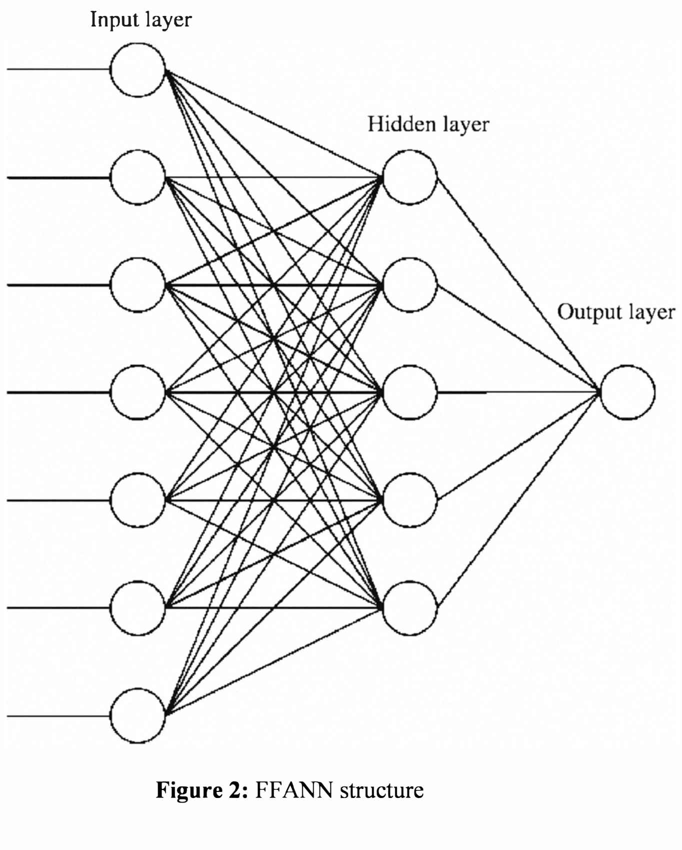


Figure 2: FFANN structure

B. Reproduction

Each weed $\overrightarrow{W_{i,G}}$ of the population G is allowed to produce seeds depending on its own, as well as the highest and lowest fitness of the colony, such that the number of seeds produced by a weed increases linearly from lowest possible for a weed with worst fitness to the maximum number of seeds for a weed with best fitness.

C. Spatial Dispersal

The generated seeds are then randomly distributed in the entire search space by normally distributed random numbers with zero mean but varying variance. This means that the seeds will be randomly distributed at the neighborhood of the parent weed. Here the standard deviation (σ) of the random function will be reduced from a previously defined initial value $\sigma_{initial}$ to a final value σ_{final} in every iteration of the algorithm following eq.1.

$$\sigma_{iter} = \left(\frac{iter_{\max} - iter}{iter_{\max}} \right)^{pow} (\sigma_{final} - \sigma_{initial}) + \sigma_{initial} \quad (1)$$

$iter_{\max}$ is the maximum number of iteration, σ_{iter} is the standard deviation at the present iteration and *pow* is the non-linear modulation index.

This step ensures that the probability of dropping a seed in the distant area decreases nonlinearly at each iteration which results in grouping fitter plants and elimination of inappropriate plants.

D. Competitive Exclusion

If a plant leaves no offspring then it would go extinct otherwise it would take over the world. Thus there is need of some kind of competition between plants for limiting maximum number of plants in a colony. Initially the plants will reproduce fast and all the produced plants will be included in the colony, until the number of plants in the colony reaches a maximum, pop_{max}. However it is expected the by this time the fitter plants have reproduced more than the undesirable plants. From there on, only the fittest plants among the existing ones and the reproduced ones are taken in the colony and then steps IIb to IId are repeated until the maximum number of iterations are reached i.e the colony size is fixed from there on at pop_{max}. This step is known as the Competitive Exclusion and it is the selection procedure of IWO.

E. Modification of IWO

Here we aim at reducing the standard deviation σ for a weed when the objective function value of a particular weed nears the minimum objective function value of the current population, so that the weed disperses it's seeds within a small neighborhood of the suspected optima. Eqn (2) describes the scheme by which the standard deviation σ_i of the i'th weed is varied .

$$\sigma_i = \sigma_{final} + \left(1 - e^{-\Delta f_i}\right)\left(\sigma_{initial} - \sigma_{final}\right) \quad (2)$$

where, $\Delta f_i = \left| f(\overrightarrow{W_i}) - f(\overrightarrow{W}_{best}) \right|$ (3)

so when $\Delta f_i \to 0$ then $\sigma_i \to \sigma_{final}$. As $\sigma_{final} << \sigma_{initial}$, so when $\Delta f_i \to 0$ i.e. the i'th weed is in close proximity of the optima ,then the standard deviation of the weed becomes very small resulting in dispersal of the corresponding seeds within a small neighborhood around the optima. Thus in this scheme, instead of using a fixed σ for all weeds in a particular iteration we are varying the standard deviation for each weed depending on it's objective function value. So this scheme in one hand increases the explorative power of the weeds and on the other creates some probability for the seeds dispersed by the undesirable weeds (the weeds with higher objective function value) to be a fitter plant. These features were absent in the classical IWO algorithm. Figure 3 shows the variation of σ vs Δf_i and Figure 4 represents the flowchart of the modified IWO algorithm.

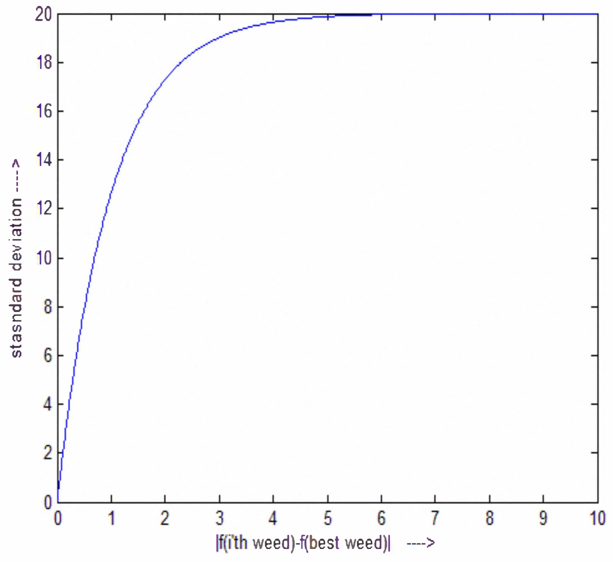


Figure 3: Variation of σ_i with Δf_i for σ_{final}=0.001 and $\sigma_{initial}$=20

IV. Index of Performance Evaluation

In this paper, we have used two different classes of problem set: Function Approximation & Pattern Recognition. The different Indices of Performance Evaluation for these two classes have elaborated as follows:

A. Index for Function Approximation Problem Set

In case of function Approximation Problem, *Measurement of Mean Square Error (MSEREG)* i.e. the square of the difference between the actual and obtained outputs is used as the index of the performance evaluation. As the neural network may have a large number of solutions of network weights and biases having the the same *MSEREG,* the network parameters may grow explosively. So to allow only network parameters with lowest numerical value to be selected, a penalty term consisting of the square of the weights and biases has been added with *MSEREG* to form the complete fitness function of the evolutionary optimization algorithms like IWO and DE for training the neural network. Hence for well trained networks, *MSEREG* should me as minimum as possible.

B. Index for Pattern Recognition Problem Set

In case of pattern recognition problem, *Classification Error Percentage (CEP)* as defined in (4) is used as the fitness function of the IWO and DE algorithms.

$$CEP = \frac{E_P}{P} * 100 \quad (4)$$

E_p = Total Number of incorrectly recognized training or testing patterns.
P= Total number of training or testing patterns.
Hence for well trained networks CEP should be as minimum as possible.

V. Experimental Results and Discussion

A. Algorithms Compared

The performance of IWO in training the FFANN has been compared with the FFANN trained by the following algorithms.

- Differential Evolution (DE)—It is a novel evolutionary algorithm first introduced by Storn and Price [21], which is inspired from the theory of evolution. It is successfully used in many artificial and real optimization problems and applications [22] including training a neural network [23]. In this paper DE/rand/1/bin variant is used for comparison.
- Back propagation algorithm with an adaptive learning rate (TRAINGDX).
- One step secant learning method (TRAINOSS).
- Resilient back propagation algorithm (TRAINRP)

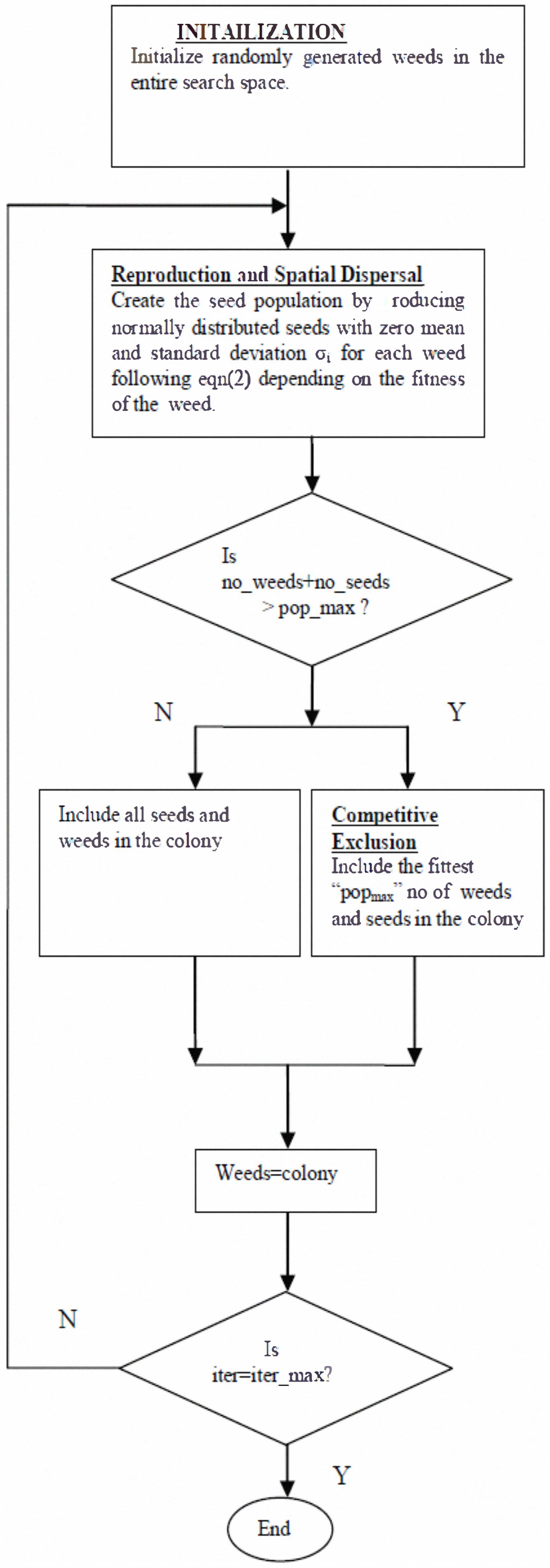
B. Experimental Results

In this section performance of IWO is evaluated by experiments. The experiments were conducted by various configurations of FFANN and two commonly used problem domains: Function Approximation and Pattern Recognition.

.

1) Function Approximation

Here IWO trained FFANN has been used to approximate a very simple and conventional function *SIN(X).* The network structure of the selected FFANN consists of one input,one hidden and one output layer each containing 1,5,1 number of neurons respectively. The transfer of the networks are *tansig-tansig-tansig* (*tansig*: Hyperbolic Tangent Sigmoid) respectively. Such kind of networks has been selected after much experimentation. This same network architecture is used for other competiting algorithms.


INITAILIZATION
Initialize randomly generated weeds in the entire search space.
Reproduction and Spatial Dispersal
Create the seed population by roducing normally distributed seeds with zero mean and standard deviation σi for each weed following eqn(2) depending on the fitness of the weed.
Is no_weeds+no_seeds > pop_max ?
N
Y
Include all seeds and weeds in the colony
Competitive Exclusion
Include the fittest "popmax" no of weeds and seeds in the colony
Weeds=colony
N
Is iter=iter_max?
Y
End

IWO		DE/rand/1/bin	
Paremeter	Value	Parameter	Value
Search range	[-500,500]	Search range	[-500,500]
Maximum Population Size	200	Population Size	200
Initial Population Size	50	Scale Factor, *F*	0.8
Max seed	8	Crossover Probability, *Cr*	0.9
Min seed	1		
$\sigma_{initial}$	10% of the entire search range		
σ_{final}	0.01% of the entire search range		

Figure 4: Flowchart of IWO Algorithm along with its modification

ALGORITHMS	Mean MSEREG (Std_dev)	Minimum	Maximum
IWO	0.1128 (0.0031)	0.1042	0.1146
DE	0.1285 (0.0196)	0.1051	0.3265
TRAINGDX	0.1044 (0.0026)	0.1040	0.1069
TRAINRP	NA	NA	NA
TRAINOSS	0.1045 (0.0046)	0.1040	0.1047

Table 1: Parametric Set-up of IWO and DE algorithms

Table 2: Comparison of MSEREG for SIN(X) Function

The detailed parametric set-up for the two evolutionary technique IWO & DE is shown in Table 1. Table 2 summarizes the MSEREG obtained by FFANN, trained by IWO and other comparative algorithms. In the table mean, maximum and minimum values of MSEREG obtained from 50 statistical runs are reported. IWO clearly performs better than the other evolutionary technique DE/rand/1/bin as reflected by the lower value of *MSEREG* obtained by IWO. But TRAINGDX performs even better than IWO. This is indeed the case for function approximation problems, where the sole objective is to train the network not to test it with some new data. BP algorithms like TRAINGDX in many cases "overtrain" the network i.e trains the network exceedingly well for only the training dataset thus producing lesser performance index. Its limitation gets exposed when it is tested with some new data points as evident by the pattern recognition problems to be discussed next. Approximated Sine curve obtained by IWO along with the original one shown in Figure 5.

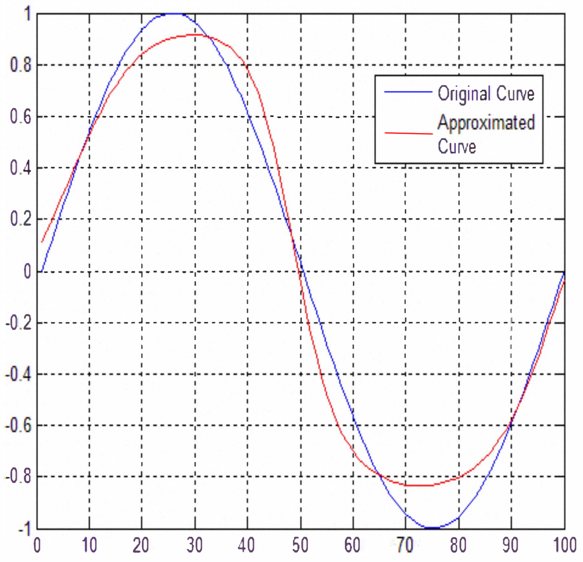


Figure 5: Approximated Sine Curve along with the original one obtained by IWO

2) Pattern Recognition Problem

In this paper three datasets Diabetes, Cancer, Glass Dataset have been used available online available from [25].

The properties of the data sets are summarized in Table 3. The last row of Table 3 shows the percentage of various classes in each dataset. There are two classes in each of Cancer and Diabetes dataset and six classes in Glass dataset. A three layer FFANN was used for each problem to work as a pattern classifier. For all the three problems the transfer function of layers has been chosen as *purelin-tansig-tansig* respectively. Such a selection has been done after much experimentation to obtain the best possible results. Network configurations used for each dataset are summarized in Table 3. Number of neurons in each class is equal to the number of classes in each dataset. Parametric set-up for IWO & DE/rand/1/bin algorithms is same as shown in Table 1. 80% of the entire datasets has been used for training purpose and rest 20% has been used for testing purpose. Both training and testing *CEP*s obtained by IWO and the other competiting algorithms are shown in Tables 4, 5 and 6 respectively.

Various classes are numbered in the Tables according to Table 3. We have run 50 independent training session of IWO,DE & BP algorithms for each of the selected datasets and reporting the mean of these runs along with the standard deviation, best and worst *CEP*. It is evident that at some instances, the training *CEP* obtained by BP algorithms are better than that obtained by IWO. It is again occurring due to the classical problem of "overtraining"[24]. The overall testing *CEP* i.e. the *CEP* for unknown 20% datapoints obtained by IWO is much better than those obtained by DE & BP algorithms in case of each dataset as can be verified from Table 4,5,6.This fact establishes the claim of "overtraining" of the network only for the training dataset by BP algorithms. Thus whether the network is properly trained or not can be understood only by comparing the testing *CEP*, not the training one. Now, as IWO comfortably beats the other competitors when testing *CEP* is concerned, these experiments establish the superiority of IWO in training FFANN to use it as a pattern classifier.

Table 3: Properties of the three datasets and the configuration of the neural networks used

Properties	CANCER	DIABETES	GLASS
FFANN structures	9-8-2	8-7-2	9-12-6
Weights	169	134	261
Biases	19	17	27
Classes and their percentages	I.Benign (65.14%) II Malign (34.86%)	I.Diabetes (33.07%) II.No Diabetes (66.92%)	I. Building float processed (40.19%) II. Building non-float processed (27.10%) III. Vehicle float processed (6.54%) IV. Containers (8.41%) V. Tableware (5.61%) VI. Headlamps (12.15%)

Table 4: Comparison of the CEP for the DIABETES dataset among IWO and other algorithms

Algorithms	Training			Testing			Classes	
	Mean CEP (std_dev)	Best CEP	Worst CEP	Mean CEP (std_dev)	Best CEP	Worst CEP	I	II
IWO	1.8229 (0.1015)	1.7719	2.0000	**5.2083 (0.2376)**	5.0019	5.6290	**33.33**	**66.67**
DE/rand/1/bin	2.8571 (0.2887)	2.6719	3.1190	9.3750 (0.4527)	8.7562	10.0018	40.62	59.38
TRAINGDX	2.3437 (0.2003)	2.0019	3.3310	8.3333 (0.1129)	7.8990	8.5326	38.54	61.46
TRAINOSS	2.0833 (0.3018)	2.0000	2.3019	4.6875 (0.2454)	4.1098	5.3390	36.97	63.03
TRAINRP	1.8229 (0.2998)	1.7244	2.0000	11.9791 (0.3675)	11.1106	12.1897	34.89	65.11

Table 5: Comparison of the CEP for the DIABETES dataset among IWO and other algorithms

Algorithms	Training			Testing			Classes (Testing)	
	Mean CEP (std _dev)	Best CEP	Worst CEP	Mean CEP (std _dev)	Best CEP	Worst CEP	I	II
IWO	17.4285 (0.6347)	16.9945	18.1362	**24.7126 (0.2876)**	**20.1625**	**28.1967**	**62.07**	**37.93**
DE/rand/1/bin	24.5714 (0.3923)	24.1744	25.5261	37.9310 (0.6495)	35.8744	39.1108	62.64	37.36
TRAINGDX	22.2857 (0.8235)	21.8534	23.9908	32.1839 (0.9732)	30.7367	33.0023	64.94	35.06
TRAINOSS	16.3128 (0.2163)	16.1109	16.8874	29.3103 (0.0957)	28.7656	29.8069	71.26	28.74
TRAINRP	19.1428 (0.2195)	19.0053	19.8763	27.0114 (0.8365)	26.8721	27.5300	70.11	29.89

Table 6: Comparison of the CEP for the DIABETES dataset among IWO and other algorithms

Algorithms	Training			Testing			Classes					
	Mean CEP (std_dev)	Best CEP	Worst CEP	Mean CEP (std_dev)	Best CEP	Worst CEP	I	II	III	IV	V	VI
IWO	30.8411 (0.2356)	30.1656	31.5434	**39.6226 (0.4532)**	**39.2354**	**40.1154**	**41.50**	**18.86**	**15.09**	**9.43**	**3.77**	**11.35**
DE/rand/1/bin	41.1215 (0.2276)	40.5435	41.2565	47.1634 (0.7356)	46.7752	47.9178	43.39	22.64	13.20	7.54	1.88	11.35
TRAINGDX	35.5140 (0.1189)	35.0017	35.9800	41.568 (0.6190)	40.0000	41.8142	39.61	18.86	16.97	11.31	1.88	11.37
TRAINOSS	33.6449 (0.3100)	32.9178	33.8716	43.147 (0,3254)	43.0018	43.9913	37.72	20.74	18.86	13.19	3.77	5.72
TRAINRP	30.8411 (0.2781)	30.1798	31.6724	40.746 (0.3855)	40.0011	40.9278	41.50	16.97	16.97	11.31	3.77	9.48

VI. Conclusions

In this study, a modified Invasive Weed Optimization (IWO) training algorithm was used to train feed- forward multilayer perceptron neural networks. From the reported results it is evident that our proposed algorithm has advantage over gradient based methods in case of training the FFANN. In some cases if the error surface is very rough and the gradient information frequently leads to local optimums, then the proposed algorithm performs much better than the other gradient based methods. Other than this advantages training by this proposed algorithms has an disadvantage, i.e. the time required for obtaining the convergence in case of this algorithm sometimes become intolerable in case of larger datasets. But as the performance is much better we have to go for trade off between time and performance, thus the future work should consider this trade off. In case of larger FFANN with larger datasets the intrinsic parallel nature of FFANN feed-forward calculations would invite the use of a parallel implementation to speedup the fitness function calculations resulting in a reduction in the overall training time required

by our proposed algorithm. Future research may focus into recognition of more complex and useful applications like speech, character etc.

ACKNOWLEDGEMENT

This work was supported by the Czech Science Foundation under the grant no.102/09/1494.

REFERENCES

[1] U. Seiffert, Training of Large-Scale Feed Forward Neural Networks. International Joint Conference on Neura Networks(2006),5324-5329

[2] X.Jiang, A.H.K.S. Wah, Constructing and Training feed-forward neural networks for Pattern classification, Pattern recognition 36 (2003) 853-867

[3] A.T. Chronopoulos, J. Sarangapani, A distributed discrete-time neural network architecture for pattern allocation and control. Proceedings of the International Parrelel and Distributed Processing Symposium (IPDPS'02),(2002) 204-211.

[4] K. M. Lane, R.D. Neidinger, Neural networks from idea to implementation, ACM Sigapl APL Quote Quad 25(3) (1995) 27-37

[5] L. Fausett , Fundamentals of Neural Networks Architecture, Algorithms, and Applications, Prentice Hall, New Jersey, 1994.

[6] L.G.C. Hamey , XOR has No Local Minima: A case study in neural network error surface Analysis, Neural Networks 11(1998) 669-681

[7] G.Wei, Study of Evolutinary Neural Network based on Ant Colony Optimization, International Conference on Computational Intelligence and Security Workshops (2007) 3-6

[8] D. Kim, H. Kim, D. Chung, A Modified Genetic Algorithm for Fast Training Neural Networks, Advances in Neural Networks-ISNN (2005)(Springer Berlin / Heidelberg) volume 3496/2005,660-665

[9] W. Gao, Evolutionary Neural Network based on New Ant Colony Algorithm. international Symposium on Computational Intelligence and Design (ISCID)(2008) 318-321

[10] X.Yao, Evolving Artificial Neural Networks, Proceedings of the IEEE, (1999) 87(9) 1423-1447

[11] H. Pierreval , C. Caux, J.L. Paris, F. Viguier, Evolutionary approaches to design and organization of the manufacturing systems, Computer and Industrial Engineering, 44 (2003) 339-364

[12] M.G.H. Omran, M. Mahadavi, Global Best Harmony Search, Applied Mathematics and Computation 198 (2008) 643-656

[13] E. Alba, J.F.Chicano, Training Artificial Nural Networks with GA Hybrid Algorithm,Genetic and Evolutionary Computation (GECCO) 2004

[14] R.E.Dorsey ,J.D.Johnson, W.J.Mayer, A Genetic Algorithm for the Training of Feedforward Neural Networks, Advances in A.pngical Intelligence in Economics, Finance and Management (1994) 93-111

[15] J. Yu, S. Wang, L. Xi, Evolving Artificial Neural Networks using an Improved PSO and DPSO, Neurocomputing 71 (2008) 1054-1060

[16] A.R. Mehrabian, C. Lucas, A novel Numerical Optimization Algorithm Inspired from Weed Colonization, Ecological Informatics, 2006,vol 1.pp-355-366

[17] A.R. Mehrabian, A. Yousefi-Koma, Optimal Positioning of Piezoelectric Actuators on a Smart Fin using Bio-inspired Algorithms,Aerospace Science and Technology,2007,vol 11, pp 174-182

[18] H. Sepehri Rad , C. Lucas, " A Recommender System based on Inavasive Weed Optimization Algorithm", IEEE Congress on Evolutionary Computation, CEC 2007,pp 4297-4304

[19] A. R. Mallahzadeh ,S. Es'haghi, A Alipour, " Design of an E shaped MIMO Antenna using IWO Algorithm for Wireless Application at 5.8 Ghz", Progress in Electromagnetic Research,PIER 90, 2009,187-203

[20] X. Zhang, Y. Wang, G. Cui, Y. Niu, J. Xu, Applicationof a novel IWO to the design of encoding sequence for DNA computing,Comput. Math. Appl. 57, pp. 2001-2008,Jun,2009

[21] R. Storn and K. V. Price, "Differential Evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces", Technical Report TR-95-012,ICSI,

[22] Lampinen J, A Bibliography of Differential Evolution Algorithm,http://www.lut.fi

[23] Masters T, Land .W , A New training algorithm for the general regression neural network, IEEE International Conference on System, Man and Cybernetics, Computational Cybernatics and Simulations, 3 (1997),1990-1994

[24] Y.Liu, J. A Starzyk, Z. Zhu , Optimized Approximation Algorithm in Neural Networks without Overfitting, IEEE Transactions on Neural Networks,19(6) (2008) 983-995

[25] ftp://ftp.ira.uka.de/pub/neuron/proben1.tar.gz.