Int. J. Advance Soft Compu. Appl, Vol. 6, No. 1, March 2014
ISSN 2074-8523; Copyright © SCRG Publication, 2014

A Novel hybrid algorithm of Differential evolution with Evolving Spiking Neural Network for pre-synaptic neurons Optimization

Abdulrazak Yahya Saleh[1], Haza Nuzly Bin Abdull Hameed[1] and Mohd Najib Mohd Salleh[2]

[1]Soft Computing Research Group, Faculty of Computing, Universiti Teknologi Malaysia (UTM), Skudai, 81310 Johor, Malaysia
email: Abdulrazakalhababi@gmail.com ; mariyam@utm.my ; haza@utm.my
[2]Software and Multimedia Center, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia
email: najib@uthm.edu.my

Abstract

Spiking neural network (SNN) is considered as the third generation of artificial neural networks. Although there are many models of SNN, Evolving Spiking Neural Network (ESNN) is widely used in many recent research works. Evolutionary algorithms (EAs) have been used to optimize the ESNN, which has been used to solve the learning problems. In this paper, the differential evolution (DE) - as one of EAs- is used to determine the optimal number of pre-synaptic neurons for a given dataset which is required before the ESNN structure. This number of pre-synaptic neurons is deemed the complexity of ESNN. Five standard datasets from the UCI machine learning are used for evaluating the performance of this hybrid model. The experimental results have proved that the hybrid of Differential Evolution with Evolving Spiking Neural Network (DE-ESNN) gives promising results in terms of accuracy and complexity.

Keywords *Spiking neural network, Evolving spiking neural networks ,pre-synaptic neurons, Differential Evolution, Evolutionary algorithms.*

1 Introduction

Artificial neural network (ANN) is one of the most common artificial intelligent techniques in machine learning for solving classification problem[1]. There are

three generations in ANN: binary networks, real-valued networks and spiking neural network (SNN). In spite of being computationally stronger than classic neural, SNN still suffer from the lack of research. There are many models for SNN depending on their abstraction level. The two popular types of models are the conductance and threshold models[2]. The first type "conductance" is invented by Hodgkin and Huxley, the two winners of Nobel Prize. This model was reduced by Izhikevich who created two equations. The second type "threshold fire" models which represent the high abstraction level are leaky integrate and fire model (LIF) and spike response model (SRM). In addition, there are new models of SNN like Evolving spiking neural network (ESNN) which is used in this paper. ESNN architecture was proposed by Wysoski et al. [3] based on evolving model described by Kasabov [4].

Among the many real issues that needs to be explored in ESNN, determining the optimal number of pre-synaptic neurons for a given data set is the most important one [5, 6]. Numbers of pre-synaptic neurons which are required before the ESNN structure can be constructed. This problem is similar to identifying the number of hidden nodes in Multilayer Perceptron (MLP). Lower number of pre-synaptic neurons causes less input spikes generated and subsequently may affect learning accuracy, while higher number increases computational time. Another real issue of the ESNN is the achieving optimization balance between the accuracy and the complexity.

The major advantage of using EAs is its ability to adapt itself to a varying environment [7]. It is very common to use DE optimizer in many classification models like artificial neural network[8, 9], Wavelet Neural Network[10] and support vector machine (SVM) [11]. However, DE has not applied yet for ESNN to optimize pre-synaptic neurons.

The remaining parts of this paper are organized as follows: Evolving Spiking Neural Network is presented in section 2, Differential Evolution is discussed in section 3, section 4 explain in details the methodology used in this paper, section 5 elucidates the experimental results, and finally, section 6 concludes the paper with future works.

2 Evolving Spiking Neural Network

Nowadays, there are many attempts to improve new models of SNN. Wysoski improved one of these new models was called evolving Spiking neural network[12]. It is the model that has been used in this paper. This model has been created based on two principles: possibility of creation new classes and merging for the same similarities. The encoding method which is used for SNN is the population as explained in [13].The firing times can be calculated which represent the input neuron e using the intersection of Gaussian function. The centre is calculated using Equation (1) and the width is computed using Equation (2) with

the variable interval of [Emin, Emax]. The parameter β controls the width of each Gaussian receptive field.

Algorithm 1. ESNN Training Algorithm[14]

***step 1**: Encode input sample into firing time of pre-synaptic neuron f*
***step 2**: Initialize neuron repository R*
***step 3**: Set ESNN parameters Mod = [0,1], C = [0,1] and Sim = [0,1]*
***step 4**: for all input sample e belong to the same output class do*
***step 5**: Set weight for all pre-synaptic neuron where:* $wf = Mod_e^{order(f)}$
***step 6**: Calculate*

$$PSPmax(e) = \sum wfe * Mod_e^{order(f)}$$

***step 7**: Get PSP threshold value* $\theta = PSPmax(e) * C$
***step 8**: if the trained weight vector <= Sim of trained weight in R then*
***step** 9: Merge weight and threshold value with most similar neuron*
***step 10**:* $w = \frac{w(new)+w*N}{N+1}$
***step 11**:* $\theta = \frac{\theta(new)+\theta*N}{N+1}$
where N is number of merge before
***step 12**: else*
***step 13**: Add new neuron to output neuron repository R*
***step 14**: end if*
***step 15**: end for (Repeat to all input samples for other output class)*

$$R = E_{\min} + (2*e-3)/2*(E_{\max} - E_{\min})/(M-2) \quad (1)$$

$$\sigma = 1/\beta\,(E_{\max} - E_{\min})/(M-2) where 1 \le \beta \le 2 \quad (2)$$

Thorpe model [15] is similar to the Fast Integrate and Fire Model used in this paper. Thorpe's model shows that the earliest spikes received by a neuron will get a better weight depending on the later spikes. When the Post-Synaptic Potential (PSP) exceeds the threshold value, it will fire and becomes disabled. The computation of PSP of neuron e is presented in Equation 3,

$$\mu_e = \begin{cases} 0 & \text{if fired} \\ \sum wfe * Mod_e^{order(f)} & \text{else} \end{cases} \quad (3)$$

Where W_{fe} is the weight of pre-synaptic neuron f; $Mod_e^{order(f)}$ is a parameter called modulation factor with an interval of [0,1] and order (f) represents the rank of spike emitted by the neuron. The order (f) starts with 0 if it spikes first among all pre-synaptic neuron and increases according to the firing time. In the One-pass Learning algorithm, each training sample creates a new output neuron. ESNN training steps can be understood from algorithm 1 as illustrated below. More details about the model ESNN are in [16].

3 Differential Evolution

Differential Evolution (DE) is one of the most powerful tools of global optimization in evolutionary algorithms[17]. Compared to some other competitive optimizers, DE has several strong advantages : much more simple to implement , much better performance, very few control parameters and low space complexity [18] . Besides that, DE is good at exploring the search Space and locating the region of the global minimum [19]. Another advantage, DE won a higher position on a real-valued function test suite due its manner in using the distance and direction information from the current population to guide the further search[20]. Many practical problems have objective functions that are non differentiable, non-continuous, non-linear, noisy or have many local minima, constraints or stochastic. Such problems are difficult, if not impossible to solve analytically. As a good point, DE can be used to find approximate solutions to such problems. According to Ilonen, Kamarainen et al. [21], DE has been introduced and applied successfully in many artificial and real optimization problems and also for feed forward neural network training. DE which belongs to the evolutionary algorithms uses three steps: mutation, crossover and selection. In the initialization, a population of individual candidates, each of dimension Nvariables (number of decision variable in the optimization problem), is randomly generated over the feasible region. A typical value of an individual is about 5-10 times Nvariables to ensure DE has enough in to work with. The fitness of each individual candidate is evaluated. Out of these individual candidates, one of them is randomly selected as the target candidate. Differential evolution algorithm contains three parameters: mutation constant F, crossover constant CR, and size of population Nindividuals. The rest of the parameters are.

1. Dimension of problem Nvariables that scales the difficulty of the optimization task.
2. Maximum number of iterations T_MAX, which serves as a stopping condition.
3. Low and high boundary constraints, LOWER and UPPER, respectively that limit the feasible area.

Fig. 1 illustrates the flowchart of DE training. More details can be found in [22].

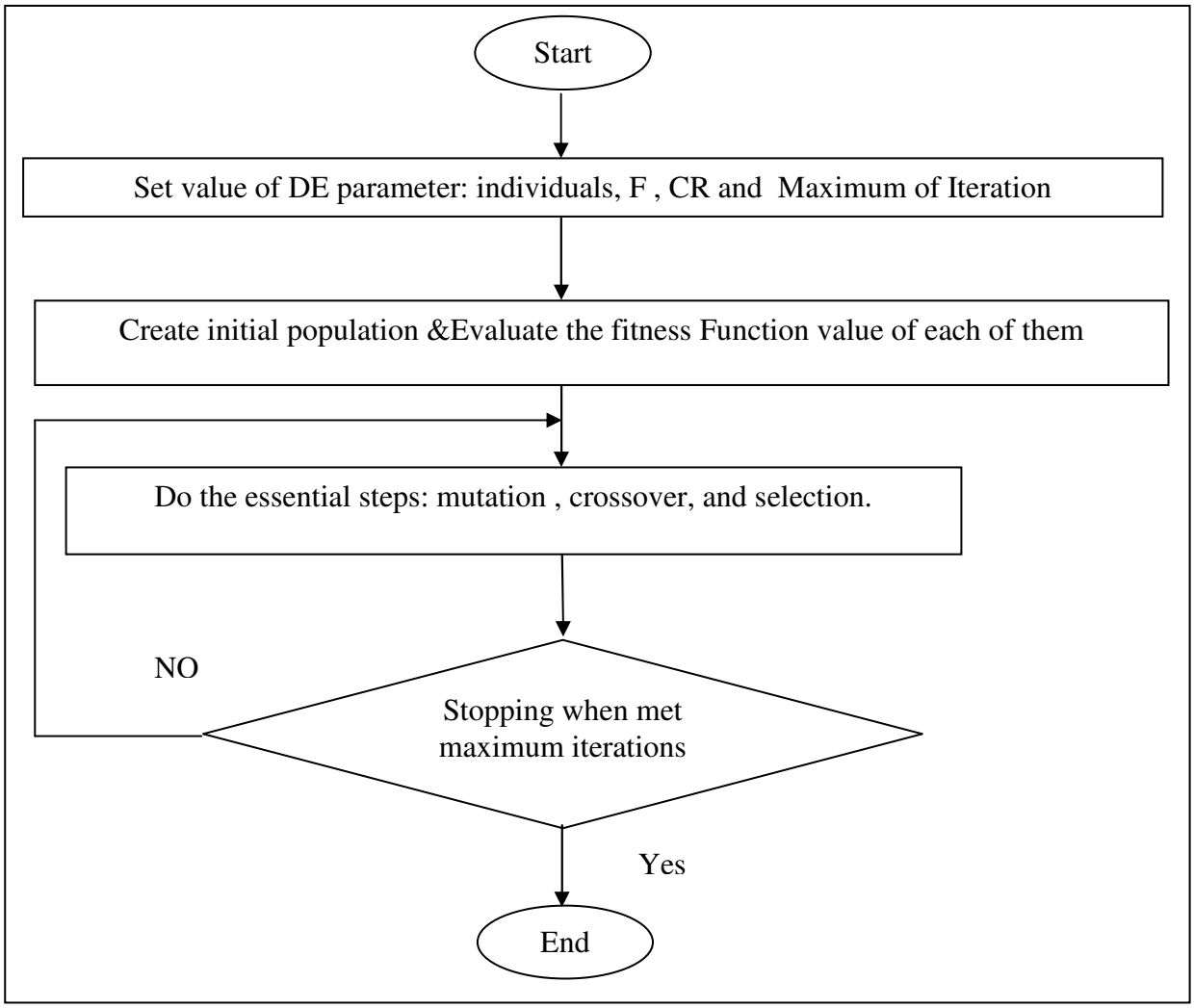

Fig.1 The flowchart for training Differential Evolution (DE)

DE operations: Mutation, crossover and selection can be explained as below:

3.1. Mutation

For each target individual X_{ij}^{t} , a mutant vector v is generated according to

$$V_i^{t+1} = X_{r1}^{t} + F.(X_{r2}^{t} - X_{r3}^{t}) \quad (4)$$

The r_1,r_2 and r_3 are randomly chosen indexes and r_1 ,r_2,r_3 [1,2,….., Nindividuals]. F is a real number to control the amplification of the difference vector $(X_{r2}^{t} - X_{r3}^{t})$ Range of F is in [0, 1] 0<F<=1

3.2. Crossover

The target individual candidate is mixed with the mutated vector, using the following scheme, to yield the trial vector u.

$$u_{ij}^{t+1} = \begin{cases} v_{ij}^{t+1}, rand(j) \le CR \quad or \quad j = randn(i) \\ x_{ij}^{t+1}, rand(j) \le CR \quad or \quad j \ne randn(i) \end{cases} \quad (5)$$

where j=1,2,….., Nvariables, rand (j) ∈ [0,1] is the j_{th} evolution of a uniform random generator number∈ r. CR∈ [0,1] is the crossover probability constant ,which has to be determined previously by the user rand (i) v_i^{t+1} (1, 2,…. Nvariables) is a randomly chosen index which ensures that u_i^{t+1} Gets at least one element from u_i^{t+1} .Otherwise , no new parent candidate would be produced and the population would not alter

3.3. Selection

DE adopts greedy selection strategy. If and only if, the trial candidate u_i^{t+1} yields a better fitness function value than x_i^t then u_i^{t+1} is set to .Otherwise, the old value is x_i^t retained.

4. The Proposed Hybridization of DE-ESNN

Hybridization of EAs is being trendy due to their abilities in handling several real world problems[23] . The purpose of hybridization is to leverage the best function from each component of the hybrid[24]. In this work, we propose a hybrid DE approach, called DE-ESNN, which combines DE with ESNN. The integrated

structure of DE and ESNN is created to optimize the pre-synaptic parameter which affects the accuracy of ESNN model. All candidates are initialized with random value and consequently act together based on classification accuracy. In other words, All input samples will be encoded into spikes and pass through ESNN model to find the current fitness explained in algorithm1.This process of searching the best pre-synaptic neurons has been conducted by updating DE candidates until meeting the maximum number of iterations. The methodology for training of hybrid of (DE-ESNN) is presented (see Fig. 2).

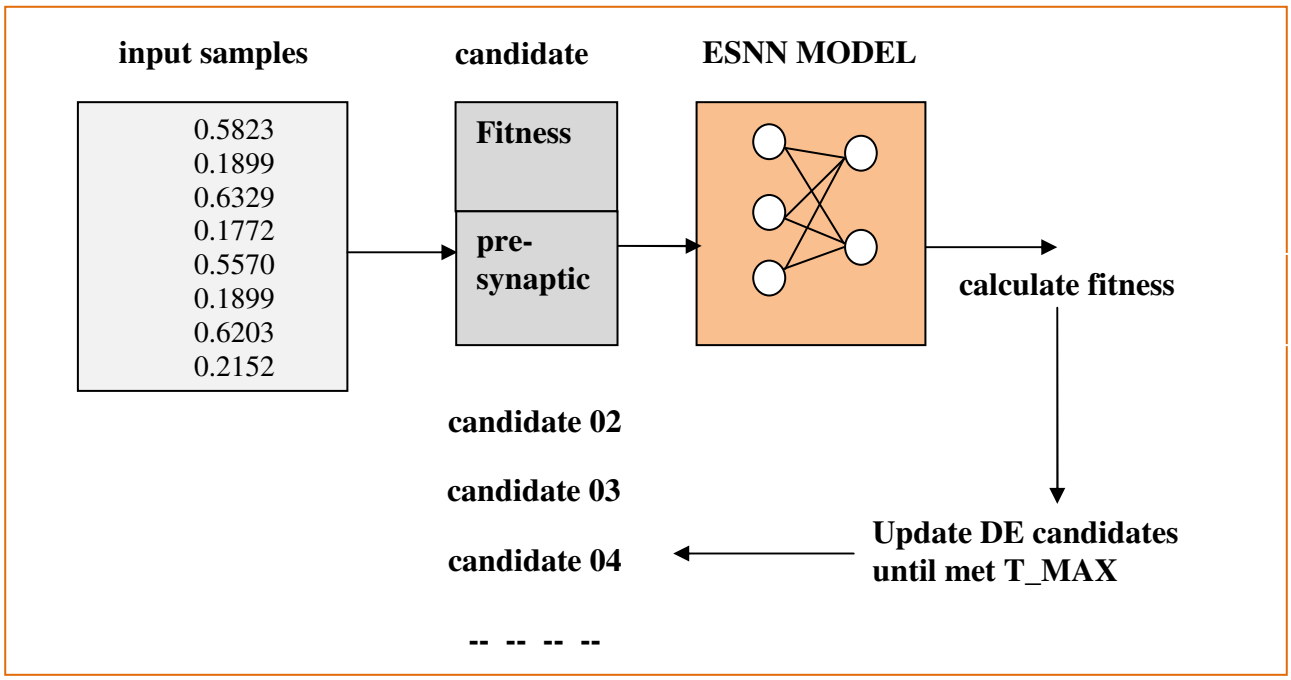

Fig .2 The Process of hybrid DE-ESNN Learning

The methodology is created through three phases: data preparation, normalization of data, learning phase.

4.1 Data Preparation

Five standard datasets were downloaded from the machine learning benchmark repository (http://www.ics.uci.edu/~mlearn/MLRepository.html).

The first dataset is the Iris dataset .This dataset is a multi-class problem and has been one of the most widely used datasets in the literature of pattern recognition [25]. There are 150 patterns with four attributes/inputs and three output classes.
The second dataset is the Breast Cancer Dataset. The cancer problem aims to diagnose breast cancer in patients by classifying a tumor as benign or

malignant[26]. It contains 699 instances with nine attributes/inputs and two output classes (benign or malignant).

The third data set is the Hepatitis Dataset. The problem of hepatitis is a complex and noisy data as it contains a large number of missing data. The learning task is to predict whether a patient with hepatitis will live or die. It contains 155 examples. There are nineteen attributes/inputs and two output classes (live or die). The fourth dataset used is the ionosphere dataset and it also has been taken from the UCI repository[27] . The dataset comprises of 351 instances having 34 attributes each, where every instance shows radar returns classified as good and bad.

The fifth dataset is heart which forecast the presence or absence of heart disease given the results of various medical tests carried out on a patient. It contains 303 patterns of which 139 are positive instances and 164 are negative instances.

4.2 Data Normalization

The data should be undergone some pre-processing to become suitable for training. In this paper, both training and testing datasets are normalized into the range [0, 1] by applying Equation (6).

$$\overline{m} = \frac{m - \min_m}{\max_m - \min_m} \tag{6}$$

Where m is the original value of attribute, $\overline{m}$ is the normalized value of attribute, and $\max_m$ and $\min_m$ is the maximum and minimum values of attribute m.

4.3 The Learning Phase of the Proposed DE-ESNN

Initially, the prepared datasets are divided randomly into training data and testing. Subsequently, once the dataset is prepared and normalized, the novel hybrid algorithm DE-ESNN is trained depending on the finalized dataset to classify the dataset into their classes. The training process of ESNN is explained above in algorithm 1 and the training process of the hybrid DE-ESNN is explained in algorithm 2. There are many important parameters which can control the result of training in ESNN. The value of Modulation factor (Mod) , the threshold parameter and the value of neuron similarity (Sim) should be between 0 and 1. The selected values through many experiments for these parameters are indicated in Table 1

Table 1: Selected values of parameters in ESNN

Parameter	Normal Range	Iris data set	Breast cancer dataset	Hepatitis Dataset	Heart Dataset	Ionosphere Dataset
Mod	0-1	0.9	0.9	0.9	0.9	0.9
Sim	0-1	0.1	0.1	0.1	0.1	0.1
Threshold	0-1	0.75	0.1	0.45	0.75	0.75

Moreover, DE has at the same time another parameters. According to [28], the selected values for DE parameters are shown in table 2

Table 2: Selected values of parameters in DE

Parameter	Description	Selected values
F	(Mutation) constant	0.9
CR	Crossover constant	0.8
Nindividuals	Size of population	20
Nvariables	Dimensionality of problem	2
T_MAX	Maximal number of iterations	1000

Our proposed approach DE-ESNN is described in Algorithm 2. DE-ESNN works when each candidate holds the pre-synaptic neuron value. This value is considered as a candidate's vector value. Updating the candidate's vector value means updating the pre-synaptic neuron value. The process begins by initializing the population and all candidates' vector values with random values. Then, in every iteration, all candidates' vector values are updated based on adding weighted difference of two vectors to third, mixing new candidate's vector with target vector to yield trial vector. Finally, Replacing target vector with trial vector if latter is strictly superior .This updating process is repeated in every iteration until stopping criteria is met, for example a desired fitness value or a maximum number of iterations. The training process continues until satisfactory fitness is achieved by the best candidate vector value. When the training ends, the values are used to calculate the classification accuracy for the training patterns. The same set of values is used then to test ESNN using the test patterns

Algorithm 2. DE-ESNN Training Algorithm

step1: *Initialize candidates*
Step2: *Read data set*
Step3: *Set ESNN parameters Mod = [0,1], C = [0,1] and Sim = [0,1]*
Step 4: *Initialize neuron repository R*
Step5: *for Every candidate do*
Step6: *Encode spikes by Gaussian formula into firing time of pre-synaptic neuron f*
Step7: *for all input sample e belong to the same output class do*
Step8: *Set weight for all pre-synaptic neuron where:*

$$wf = Mod_e^{order(f)}$$

Step9: *Calculate*

$$PSP\max(e) = \sum wfe * Mod_e^{order(f)}$$

Step 10: *Get PSP threshold value*

$$\theta = PSP_{\max(e)} * c$$

Step 11: *if the trained weight vector <= Sim of trained weight in R then*
Step 12: *Merge weight and threshold value with most similar neuron*
Step 13:

$$w = \frac{w(new) + w * N}{N + 1}$$

Step 14:

$$\theta = \frac{\theta(new) + \theta * N}{N + 1}$$

where N is number of merge before
Step 15: *else*
Step 16: *Add new neuron to output neuron repository R*
Step 17: *end if*
Step 18: *end for (Repeat to all input samples for other output class)*
Step19: *calculate fitness*
Step20: *Update DE individuals*
Step21: *Do the same processes (6-20) for all iterations*
Step22: *Do testing according the best fitness value*
Step23: *end*

5. Results and Discussion

This section presents the results of the novel hybrid DE-ESNN and its comparative study with ESNN and BPNN. The experiments are conducted using iris, breast cancer, hepatitis, heart and ionosphere datasets. All experiments are based on ten runs. The results of comparison have been analyzed based on their classification performance. Table 3 shows the results of comparison of DE-ESNN, ESNN and BPNN for all chosen datasets in both training and testing data. As it can be seen, the results reported have demonstrated better accuracy of DE-ESNN in the training phase for all datasets except breast cancer. In addition, the findings show that the accuracy of DE-ESNN in the testing phase has yielded better performance for iris, heart and ionosphere datasets. However, the accuracy of DE-ESNN is not convincing in the rest datasets with values of 91.18 % for breast cancer and 51.61% for Hepatitis because the number of attributes and instances are a lot. Moreover, as illustrated in Table 3, testing accuracy values signify that DE-ESNN has resulted in better convergence, compared to ESNN and BPNN for iris, heart and ionosphere datasets and less in other datasets depending on the nature of their distribution.

Table 3: Classification Accuracy of Chosen Datasets

The dataset	Algorithm	Classification accuracy	
		Training set%	Testing set%
Iris data set	DE-ESNN	**99.17**	**93.33**
	ESNN	94.17	90.00
	BPNN	90	86
Breast cancer	DE-ESNN	97.62	91.18
	ESNN	**99.42**	**98.24**
	BPNN	95.43	96.32
Hepatitis	DE-ESNN	88.7	51.61
	ESNN	**100**	**84.21**
	BPNN	78.23	54.84
Heart	DE-ESNN	**97.47**	**62.71**
	ESNN	68.49	52.54
	BPNN	75.10	61.3
Ionosphere	DE-ESNN	**96.08**	**71.43**
	ESNN	93.24	70.00
	BPNN	95	68

It can be clarified that Fig.3 illustrates the result of the hybrid DE-ESNN for the optimal number of pre-synaptic neurons for all datasets. It is clear that for each

dataset, differential evolution search for the optimum number of pre-synaptic depending on the best fitness accuracy. Fig .3 demonstrates the movement of searching the best values for the pre-synaptic neurons in achieving the best accuracy.

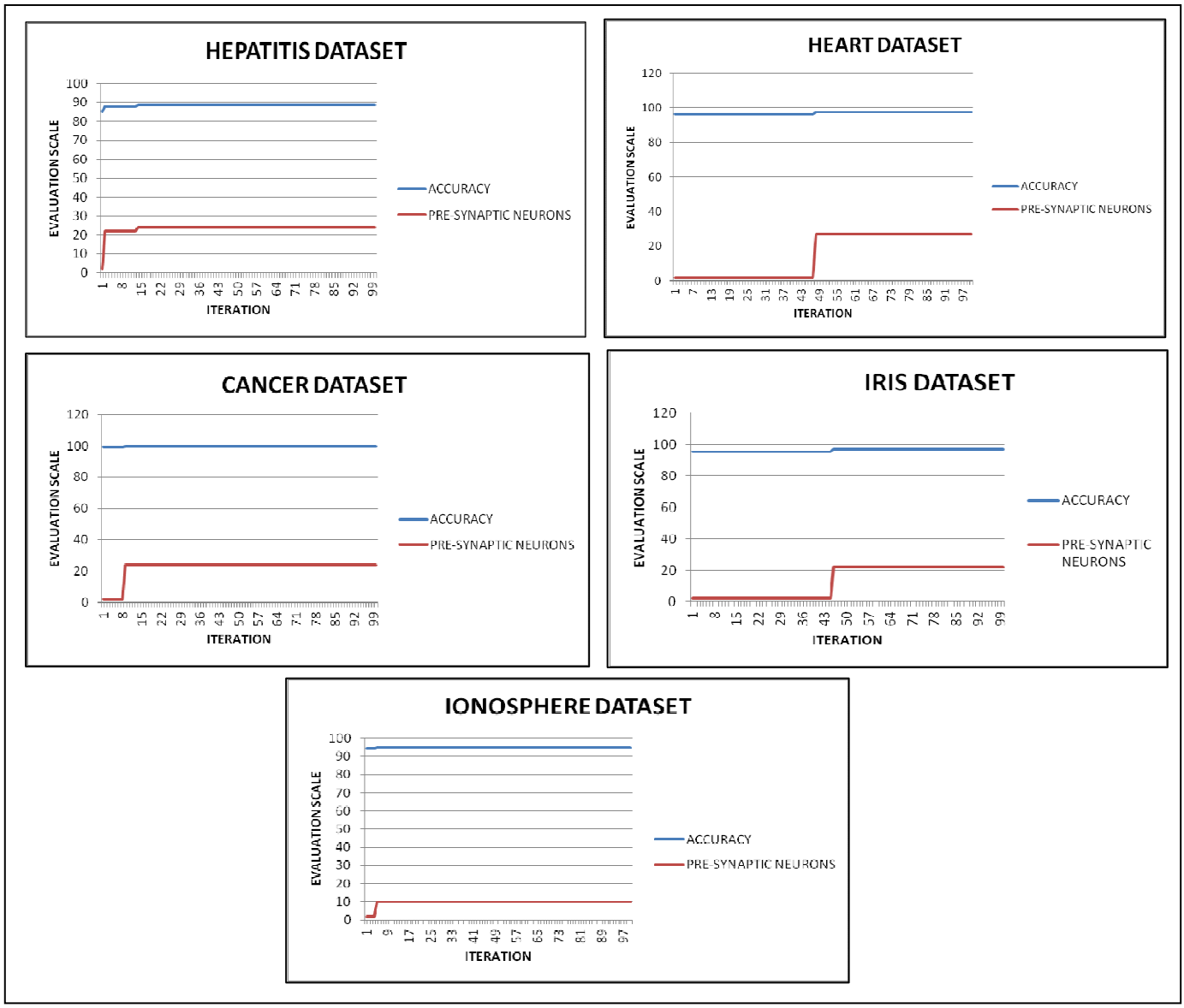

Fig .3 The optimum pre-synaptic and accuracy for all datasets

From the plot in Fig .3, this curve started to leap in a very fast way starting from the second iteration for hepatitis dataset with an accuracy of 85.483871. Moreover, the same scenario also occurs for ionosphere dataset in the first four iterations with an accuracy of 94.3038 and eight iterations for cancer dataset with

an accuracy of 99.452555. Similarly, iris datasets generate 45 iterations with an accuracy of 95.57522, and 47 iterations for heart datasets with an accuracy of 96.62921.

It can be noticed clearly that the range of the optimum pre-synaptic neurons is between 22 and 27 neurons for four datasets from the fifth selected datasets. This means that the best accuracy results can be founded when we select the value of pre-synaptic neurons. The optimum value of pre-synaptic for ionosphere dataset was 10.

6. Conclusion and Future Works

In this paper, a novel hybrid DE-ESNN was proposed to find out the optimal number of pre-synaptic neurons for a given dataset which is required before the structure of ESNN. A comparative study has been done between DE-ESNN, ESNN and BPNN to show the effectiveness of DE-ESNN. All DE-ESNN , ESNN and BPNN have been used to perform the classification on five standard data sets. The results show that DE-ESNN is capable to classify data set with accuracy better mostly than the ESNN and BPNN algorithm. The accuracy of ESNN in testing phase gave promising results compared to the accuracy in ESNN and BPNN. Moreover, DE-ESNN found out the optimum pre-synaptic before structure of ESNN which is considered as the complexity. Hybridization of a multi differential evolution MODE-ESNN with an ESNN algorithm may be a good topic to be explored to get more results. In our future work, we will conduct the statistical tests to further validate the significant of the above findings.

Acknowledgements

This work is supported by The Ministry of Higher Education (MOHE) under Research Acculturation Collaborative Effort (RACE) Grant Scheme (00M09). The authors would like to thanks Research Management Centre (RMC), Universiti Teknologi Malaysia (UTM) for the support in R & D, Soft Computing Research Group (SCRG) for the inspiration in making this study a success. The authors would also like to thank the anonymous reviewers who have contributed enormously to this work.

References

1. Bahrammirzaee, A., *A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems.* Neural computing & applications, 2010. **19**(8): p. 1165-1195.

2. Gerstner, W. and W.M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. 2002: Cambridge Univ Pr.

3. Wysoski, S.G., L. Benuskova, and N. Kasabov. *Adaptive learning procedure for a network of spiking neurons and visual pattern recognition*. in *Advanced Concepts for Intelligent Vision Systems*. 2006. Springer.

4. Kasabov, K. and N. Kasabov, *Evolving connectionist systems: the knowledge engineering approach*. 2007: Springer.

5. Hamed, A. and H. Nuzly, *Novel Integrated Methods of Evolving Spiking Neural Network and Particle Swarm Optimisation*, 2012, AUT University.

6. KASABOV, N. and S. SOLTIC, *KNOWLEDGE EXTRACTION FROM EVOLVING SPIKING NEURAL NETWORKS WITH RANK ORDER POPULATION CODING.* International Journal of Neural Systems, 2010. **20**(06): p. 437-445.

7. Fernandez Caballero, J.C., et al., *Sensitivity versus accuracy in multiclass problems using memetic Pareto evolutionary neural networks.* Neural Networks, IEEE Transactions on, 2010. **21**(5): p. 750-770.

8. da Silva, A.J., N.L. Mineu, and T.B. Ludermir, *Evolving Artificial Neural Networks Using Adaptive Differential Evolution.* Advances in Artificial Intelligence - Iberamia 2010, 2010. **6433**: p. 396-405.

9. Mineu, N.L., et al., *Topology Optimization for Artificial Neural Networks using Differential Evolution*, in *2010 International Joint Conference on Neural Networks Ijcnn 2010*. 2010, Ieee: New York.

10. Dheeba, J. and S.T. Selvi, *An Improved Decision Support System for Detection of Lesions in Mammograms Using Differential Evolution Optimized Wavelet Neural Network.* Journal of Medical Systems, 2012. **36**(5): p. 3223-3232.

11. Zhou, S.W., et al., *Parameters selection of SVM for function approximation based on Differential Evolution - art. no. 1270*. Proceedings of the International Conference on Intelligent Systems and Knowledge Engineering. 2007, Paris: Atlantis Press. 1270-1270.

12. Wysoski, S., L. Benuskova, and N. Kasabov, *On-line learning with structural adaptation in a network of spiking neurons for visual pattern recognition.* Artificial Neural Networks–ICANN 2006, 2006: p. 61-70.

13. Bohte, S.M., J.N. Kok, and H. La Poutre, *Error-backpropagation in temporally encoded networks of spiking neurons.* Neurocomputing, 2002. **48**(1): p. 17-37.

14. Abdull Hamed, H., et al. *String pattern recognition using evolving spiking neural networks and quantum inspired particle swarm optimization*. 2009. Springer.

15. Thorpe, S. *How can the human visual system process a natural scene in under 150ms? experiments and neural network models*. 1997. D-Facto public, ISBN.

16. Schliebs, S., et al., *Integrated feature and parameter optimization for an evolving spiking neural network: Exploring heterogeneous probabilistic models.* Neural Networks, 2009. **22**(5): p. 623-632.

17. Abbass, H.A., R. Sarker, and C. Newton. *PDE: a Pareto-frontier differential evolution approach for multi-objective optimization problems*. in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*. 2001. IEEE.

18. Das, S. and P.N. Suganthan, *Differential evolution: A survey of the state-of-the-art.* Evolutionary Computation, IEEE Transactions on, 2011. **15**(1): p. 4-31.

19. Noman, N. and H. Iba, *Accelerating differential evolution using an adaptive local search.* Evolutionary Computation, IEEE Transactions on, 2008. **12**(1): p. 107-125.

20. Gong, W., Z. Cai, and C.X. Ling, *DE/BBO: a hybrid differential evolution with biogeography-based optimization for global numerical optimization.* Soft Computing, 2010. **15**(4): p. 645-665.

21. Ilonen, J., J.-K. Kamarainen, and J. Lampinen, *Differential evolution training algorithm for feed-forward neural networks.* Neural Processing Letters, 2003. **17**(1): p. 93-105.

22. Storn, R. and K. Price, *Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces.* Journal of global optimization, 1997. **11**(4): p. 341-359.

23. Grosan, C. and A. Abraham, *Hybrid evolutionary algorithms: methodologies, architectures, and reviews*, in *Hybrid evolutionary algorithms*. 2007, Springer. p. 1-17.

24. Ahmed, F.Y., S.M. Shamsuddin, and S.Z.M. Hashim, *Improved SpikeProp for using Particle Swarm Optimization (PSO).*

25. Duda, R.O. and P.E. Hart, *Pattern classification and scene analysis*. Vol. 3. 1973: Wiley New York.

26. Mangasarian, O.L., W.N. Street, and W.H. Wolberg, *Breast cancer diagnosis and prognosis via linear programming.* Operations Research, 1995. **43**(4): p. 570-577.

27. Blake, C. and C.J. Merz, *{UCI} Repository of machine learning databases.* 1998.

28. Storn, R., *Differential Evolution Research – Trends and Open Questions*, in *Advances in Differential Evolution*, U. Chakraborty, Editor. 2008, Springer Berlin Heidelberg. p. 1-31.