Multi-Objective Differential Evolution of Evolving Spiking Neural Networks for Classification Problems

Abdulrazak Yahya Saleh, Siti Mariyam Shamsuddin, Haza Abdull Hamed

Multi-Objective Differential Evolution Of Evolving Spiking Neural Networks For Classification Problems

Abdulrazak Yahya Saleh*[1], Siti Mariyam Shamsuddin[1], and Haza Nuzly Abdull Hamed[2]

UTM Big Data Centre[1], Universiti Teknologi Malaysia (UTM), Skudai, 81310 Johor, Malaysia

Soft Computing Research Group[2], Faculty of Computing, Universiti Teknologi Malaysia (UTM), Skudai, 81310 Johor, Malaysia

* corresponding author: Abdulrazakalhababi@gmail.com

Abstract

Spiking neural network (SNN) plays an essential role in classification problems. Although there are many models of SNN, Evolving Spiking Neural Network (ESNN) is widely used in many recent research works. Evolutionary algorithms, mainly differential evolution (DE) have been used for enhancing ESNN algorithm. However, many real-world optimization problems include several contradictory objectives. Rather than single optimization, Multi-Objective Optimization (MOO) can be utilized as a set of optimal solutions to solve these problems. In this paper, MOO is used in a hybrid learning of ESNN to determine the optimal pre-synaptic neurons (network structure) and accuracy performance for classification problems simultaneously. Standard data sets from the UCI machine learning are used for evaluating the performance of this multi objective hybrid model. The experimental results have proved that the multi-objective hybrid of Differential Evolution with Evolving Spiking Neural Network (MODE-ESNN) gives better results in terms of accuracy and network structure.

Keywords. Differential Evolution, Evolutionary algorithms, Evolving spiking neural networks, Multi objective Optimization.

1 Introduction

Classification of patterns is vital to several data mining processes. Classification is one of the most commonly obverse processing tasks for a decision support system[1]. There are many areas in life which need classification such as medical diagnoses, medicine, science, industry, speech recognition and handwritten character recognition. Among feasible classifiers, Spiking neural network (SNN) plays an essential role in biological information processing [2].

Although there are many models of SNN, Evolving Spiking Neural Network (ESNN) is widely used in many recent research works. ESNN has several attractive advantages [3] including: simplicity, efficiency, trained by a fast one-pass learning algorithm. The evolving nature of model can be updated whenever new data becomes accessible with no requirement to retrain earlier existing samples. However, ESNN model is affected by its choice of parameters. The right selection of parameters allows the network to evolve towards reaching the best structure thus guaranteeing the best output. Of all the issues that require exploration in ESNN determining the optimal number of pre-synaptic neurons for a given data set is the most important one [4]. For these reasons, achieving an optimized trade-off between accuracy and the network structure is needed to find the best combination of parameters and pre-synaptic neurons.

Optimization has been used for enhancing ESNN algorithm. Multi-Objective Optimization (MOO) can be utilized as a set of optimal solutions to solve these problems. Every MOO solution appears to be a new trade-off between the objectives. The key objective of MOO is to improve ESNN optimal solutions of both structure and classification accuracy. MOO approach is preferred to traditional learning algorithms for the following reasons. First, as a result of using MOO a good performance of these learning algorithms can be achieved[5]. Second, various objectives are taken into consideration in the generation of multiple learning models. For example: accuracy, complexity[6-8], interpretability and accuracy[9], multiple error measures[10]. Third, it is superior to building learning ensembles to use models [8, 11, 12]. The important goal of MOO algorithm is to find a set of solutions from which the best one is chosen. Based on Tan *et al.*[13], the ability of evolutionary algorithms (EAs) to search for optimal solutions gives it the priority to be selected in MOO problems. EAs have the ability to explore different parts of the related algorithm in the optimal set because of the population-based algorithms.

In spite of the fact that one Multi-Objective Evolutionary Algorithms (MOEAs) was used and only for SpikeProp learning, most of the related work which use Multi-Objective Genetic Algorithms (MOGAs) is by Yaochu Jin *et al.* [14]. Both classification performance and connectivity of SNN with latency coding are optimized by MOGA. During optimization, both delay and weight between the two neurons are evolved. Furthermore, they minimize the classification error in percentage or the root mean square error for optimizing performance, and minimize the number of connections or the sum of delays for connectivity to explore the objectives influence on the connectivity and performance of SNNs. This is a very motivating finding. However, more experiments must be performed to verify this observation. No conclusion can be made on classification error should be used for classification. They have also revealed that complexities of the SNNs are equivalent, when the number of connections or the sum of delays is utilized to optimize connectivity. This study needs improvement to enhance its promising results.

Unlike previous study mentioned earlier in [14] and other single objective studies[15, 16], this paper deals with an improved method to obtain simple and accurate ESNN. The proposed method evolves toward optimal values defined by several objectives with model accuracy and ESNN's structure to improve performance for classification problems. The remaining parts of this paper are organized as follows: Methods including: Evolving Spiking Neural Network, Multi Objective Differential Evolution are presented in section 2. In addition, section 3 clarifies the proposed method multi objective hybrid of Differential Evolution with Evolving Spiking Neural Network (MODE-ESNN) used in this paper, Experimental design is discussed in section 4, section 5 explains in detail the results and discussion, and finally, section 6 concludes the paper with future works.

2 Methods

This section reviews the vital foundation of evolving spiking neural network(ESNN) and discusses the evolutionary algorithms that have been utilized for enhancement. In the first part, an introduction of ESNN, neuron coding, learning method and ESNN design, and its algorithm is presented. The second part focuses on the algorithms which are used for improvement of ESNN. The concepts and methods of multi objective optimization (MOO) are highlighted. After that, the literature focuses on the working of Multi-Objective Differential Evolution(MODE) which is used to improve and enhance the performance of classification .

2. 1 Evolving Spiking Neural Network (ESNN)

Currently, several enhancements of SNN have been proposed. Wysoski improved one of these new models known as Evolving Spiking Neural Network (ESNN) [17]. Generally, ESNN used the principles of evolving connectionist systems (ECOS) where neurons are created incrementally[18, 19]. ESNN can learn data gradually by one-pass propagation of the data through creating and merging spiking neurons[20] making it possible to attain very fast learning in an ESNN[21]. The learning of ESNN has many good advantages as it can be applied incrementally, is adaptive and theoretically 'lifelong'. Therefore, the system can learn any new pattern via creating new output neurons, connecting them to the input neurons and merging with the similar ones[22]. This model stands on two principles: possibility of establishment of new classes and the merging of the similarities. The encoding method which is used for ESNN is the population as explained in [23]. The population distributes a single input value to multiple input neurons denotes as M. Each input neuron holds a firing time as input spikes. Firing times can be calculated which represent the input neuron *e* using the intersection of Gaussian function. Equations 1 and 2 have been used to calculate the centre and the width respectively with the variable interval of [Emin, Emax]. The width of each Gaussian receptive field is controlled by the parameter β.

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

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

Algorithm 1. ESNN Training Algorithm[24]

step 1*: Compute firing time of pre-synaptic neuron f from input sample*
step 2*: Prepare the initialization of neuron repository R*
Step 3*: Determine ESNN parameters (Sim, Mod and C) between the range [0,1] for each of them*
step 4*: for all input sample e related to the same output class do*
step 5*: Determine weight for all pre-synaptic neuron where:*

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

step 6*: Calculate*

$$PSP_{\max}(e) = \sum w_{fe} * Mod_e^{order(f)}$$

***step 7**: Acquire PSP threshold value* $\theta = PSP_{\max}(e) * C$

***step 8**: if the trained weight vector <= Sim of trained weight in R then*

***step** 9: Merge threshold value weight and with most similar neuron*

***step 10**:* $w = \frac{w(new) + w * T}{T + 1}$

***step 11**:* $\theta = \frac{\theta(new) + \theta * T}{T + 1}$

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)*

Thorpe model [25] is similar to the Fast Integrate and Fire Model used in our paper. Thorpe's model shows that the earliest spikes received by a neuron will get a better weight depending on the later spikes. The Post-Synaptic Potential (PSP) will fire and become disabled only if it beats the threshold value. The computation of PSP of neuron *e* can be shown in Equation 3,

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

where W_{fe} is considered as the weight of pre-synaptic neuron *f*; Mod *e* is the parameter factor of modulation with an interval of [0,1] and order (*f*) denotes the rank of spike emitted by the neuron. The order (*f*) starts with 0 if it spikes first among all pre-synaptic neurons and increases according to the firing time. Moreover, each training sample creates a new output neuron in the One-pass Learning algorithm. ESNN training steps can be understood from algorithm 1 as illustrated above in section 2. Fig. 1 depicts a simplified architecture of ESNN model which was explained in more detail in [3].

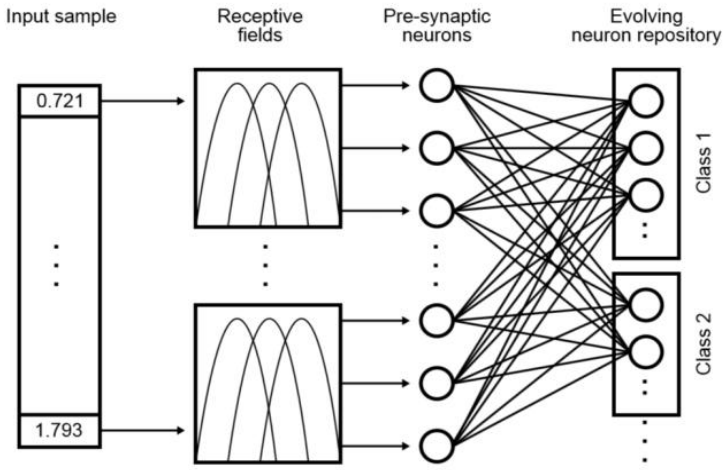

Fig. 1 A simplified architecture of ESNN [24]

The training starts with initialization of three ESNN parameters - modulation factor (*Mod*), proportion factor (*C*) and similarity value (*Sim*) in the interval [0, 1]. *Mod* is the modulation factor of the Thorpe neural model. The firing threshold θ is calculated as based on proportion factor (*C*) with a value between [0, 1]. As the training process continues, every sample produces an output neuron. The similarity of output neurons is calculated according to the Euclidean distance between the weight vector of the neurons. The parameter *Sim* controls the similarity distance. In the training stage, the output neuron stores the computed weight of all pre-synaptic neurons, a threshold value to determine when the output neuron will spike and the class label to which the input sample belongs. In the testing stage, the multiple pre-synaptic neurons encode each testing sample to spikes. After that, the PSP of the output class neurons is calculated. Once the neuron attains definite amount of spikes and the PSP exceeds the threshold value, it fires an output spike and becomes disabled. The testing sample belongs to the output class defined by the output neuron that fires first among all output neurons.

2. 2 Multi-Objective Differential Evolution (MODE)

This section discusses MOO and Evolutionary algorithms. First, the concept of MOO is introduced followed by a discussion of the essential points of MOO methods. Then, the concept of the Evolutionary Algorithm (EA) mainly Differential Evolution(DE) is clarified. After that, a brief discussion of the MOEA for DE enhancement is explained. Finally, the related works of this study are discussed in-depth.

2. 2.1 Multi-Objective Optimization (MOO)

MOO adds a new concept to the previous concepts of optimization. The process of systematically optimizing a set of objective functions at the same time is known as multi objective optimization (MOO) or vector optimization[26]. According to Lu [27], the optimal solutions obtained by individual optimization of the objectives are not a feasible solution to the multi-objective problem. Most practical optimization problems need the synchronized optimization of more than one objective function.

2.2.2 Methods of MOO Algorithms

Typical methods aggregate the objectives into one objective function by using decision making before search. Conversely, the parameters of this function are assorted by the optimizer systematically. A number of optimization runs with various parameter settings are conducted with the intention of reaching a set of solutions. Essentially, this process is independent of the primary optimization algorithm. A number of examples of this class of techniques are the weighting method [28], the constraint method [28], goal programming [29]. In exchange for the different methods, these three common methods are briefly discussed here.

a. Weighting Method

All objectives are multiplied by weighting coefficient (w_i) After that, all new functions are added together to get a single cost function. Finally, Single Objective (SO) method can be used to solve this new single cost function . Mathematically, the latest function is written as

$$f(x) = \sum_{i=1}^{I} w_i f_i(x)$$

where $0 \le w_i \le 1$ and $\sum_{i=1}^{I} w_i = 1$ (4)

b. Constraint Method

Constraint Method depends on transforming *k-1* of the *k* objectives into constraints. The remaining objective, which can be selected subjectively, is the objective function of the resulting SO:

Maximize $y = f(x) = f_h(x)$ (5)

Subject to $e_i(x) = f_i(x) \ge \varepsilon_i$, $(1 \le i \le I, i \ne h)$, $x \in X_f$

The lower bounds, ε_i , are the parameters that are diverse by the optimizer to find multiple optimal solutions.

c. Goal Programming Method

According to Philipson and Ravindran [30], Goal Programming Method is considered as the most well-known method of solving MOPs. It was initially enhanced by [31] and[32]. This method depends on ranking the goals as created by the designer. Finally, the SO function is considered as the minimization of the deviations from these goals.

2.2.3 EA Algorithms

EAs have been utilized to optimize ESNNs, which have been used to solve the problems of optimization learning. The right classification measures are defined by the number of true positives (TPs) and the number of true negatives (TNs). Additionally, the misclassifications made by the classifier are determined by the number of false positives (FPs) and the number of false negatives (FNs). Many EAs types have been used for optimization, but DE algorithm is used in this study for many reasons, which will be explained in sections below:

a. Differential Evolution (DE)

In global optimization, DE is considered as one of the mainly powerful tools of global optimization in EAs [33]. Compared to some other competitive optimizers, DE has several strong advantages: it is much simpler to implement, has much better performance, has a small number of control parameters and has low space complexity [34]. DE, which belongs to the EAs, uses three steps: mutation, crossover and selection. In the stage of 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* in order to guarantee that DE has enough to work with. The fitness of every individual candidate is evaluated. Out of these individual candidates, one is randomly selected as the target candidate. The DE algorithm contains three parameters: size of population *Nindividuals*, mutation constant *F* and crossover constant *CR*.

2.2.4 The MODE Algorithm

According to Schaffer in [35], Vector evaluated genetic algorithm (VEGA) is considered as the first multi-objective proposed in the literature. VEGA is an adapted single-objective genetic algorithm with a modified selection method. Many literature surveys found provide more detail about the history and enhancement of multi objective with GA like Zitzler et al. [36] and Deb [37]. Unlike GAs, where binary encoding can be utilized, DE solutions are coded with real values. More recently, evolutionary algorithms have also been adapted to solve optimization problems with multiple conflicting objectives by approximating the Pareto set in such problems such as in[38]. A complete tutorial on evolutionary multi-objective optimization with DE can be found in [39].

3 The Proposed Hybridization of Multi-Objective DE based ESNN (MODE-ESNN)

This section discusses the proposed algorithm called MODE-ESNN which is a multi-objective optimization approach for ESNN training. This algorithm will simultaneously determine the ESNN structure (pre-synaptic neurons) and its corresponding model parameters by treating this problem as a multi-objective optimization problem. In MODE_ESNN, the pre-synaptic neuron of ESNN is represented as a candidate. DE and MOO are combined to carry out fitness evaluation and mating selection schemes.

MODE_ESNN starts by collecting, normalizing and reading the dataset. The maximum number of iterations are then set and the candidate size is computed. In addition, the pre-synaptic neurons and parameters of ESNN are determined randomly. A population of the hybrid method is then generated and initialized. Every candidate is evaluated for each iteration based on Differential Evolution algorithm. The proposed method stops after the maximum iterations is reached. Fig. 2 describes the schematic representation of MODE_ESNN.

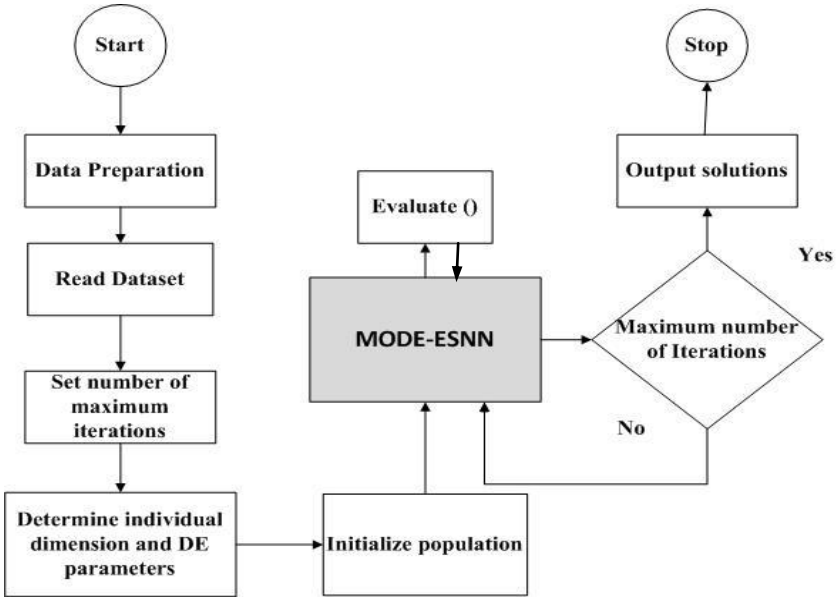

Fig. 2 Schematic representation of the proposed MODE_ESNN

The main procedure of proposed MODE-ESNN is presented below:

1. Generate an initial population *P(t)* at *t*=0 of size *N*, where each candidate represents ESNN and where *t* is the number of the actual iteration.
2. Use the random selection technique for obtaining the initial values for both pre-synaptic neurons, which are considered as the structure of ESNN and the parameters of the model.
3. Determine candidate dimension and assign DE parameters values.
4. While stopping criterion is not met do:
   a) Use ESNN algorithm to find the candidate fitness.
   b) Achieve results of pre-synaptic neurons and ESNN parameters.
   c) evaluate the candidates of population *Q(t)* according to its accuracy value.
   d) Determine the best candidate according to the best parents' values.
   e) While size of population *P(t+1)* is < *N* do:
      I. Calculate mutated candidates according to Equation 6

$$V_i^{t+1} = X_{r1}^t + F.(X_{r2}^t - X_{r3}^t) \quad (6)$$

      the r_1,r_2 and r_3 are randomly selected 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

      II. Perform crossover and selection for each candidate. The target individual candidate is mixed with the mutated vector, using the following scheme, to yield the trial vector *u* by updating the pre-synaptic neuron and ESNN parameters vector simultaneously using Equation 7

$$u_{ij}^{t+1} = \begin{cases} v_{ij}^{t+1}, rand(j) \le CR \quad or \quad j = randn(i) \\ x_{ij}^{t+1}, rand(j) \le CR \quad or \quad j \ne randn(i) \end{cases} \quad (7)$$

      where j=1,2,….., *Nvariables*, rand (j) $\in$ [0,1] is the j_{th} evolution of a uniform random generator number $\in$ r. CR $\in$ [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 verifies that u_i^{t+1} gets at least one element from u_i^{t+1} . If not, no new parent candidate would be produced and the population would not alter.
   f) End while
   g) Sort population *P(t+1)* according to their fitness value.
   h) *t*=*t*+1
5. End while

4 Experimental Design

This section presents the experiments of study on hybrid learning of ESNN network based on multi-objective method. Many techniques can be used for validation, but k-fold cross-validation is used in this paper. The advantage of k-fold cross-validation over hold-out validation is that all observations are used for both training and testing, and each observation is used for testing exactly once. 10-fold cross-validation is commonly used[40]. In order to evaluate the effectiveness of the proposed method, a detailed empirical study is carried out on seven different data sets which is explained in detail below.

4.1 Data Preparation

Several real-world data sets from UCI repository are used in this study to represent some of the most challenging problems in machine learning. Many researchers have used these data sets as a benchmark in validating the performance of their algorithms. The key characteristics of these problems and their associated learning tasks are summarized in Table 1.

Table 1: Description of data sets

Dataset	Attributes	Classes	Samples
Appendicitis	7	2	106
Haberman	3	2	306
Heart	13	2	297
Hepatitis	19	2	155
Ionosphere	34	2	351
Iris	4	3	150
Liver	6	2	345

4.2 The Learning Phase of the Proposed MODE-ESNN

Initially, the data set has been divided randomly into ten subsets of equal size. One subset is used as the testing data set, and the other nine subsets are used as the training data sets. The training and testing processes are repeated so that all the subsets are used as a testing data set. The training process of the hybrid MODE-ESNN is explained in section 3. There are many important parameters which can control the result of training in MODE-ESNN. The various parameter settings of the proposed method are tabulated in Table 2.

Table 2: Parameter settings for MODE-ESNN algorithms

Parameter	MODE-ESNN
Population size	30
Maximal number of iterations	1000
(Mutation) constant F	0.9
Crossover constant CR	0.8
Dimensionality of problem	2

The performance of the algorithms is evaluated by conducting analysis on ten evaluations. In this experiment, the evolutionary process of the proposed algorithm is analyzed and the performance is evaluated accordingly.

The objective of this evaluation is to establish the effectiveness of the proposed method in designing ESNN network. This involves both the training of the ESNN network and the evolved structure (pre-synaptic neurons) of the ESNN network. The capabilities of the proposed method have been investigated through a comparison with each other that has been applied for classification problems. In order to evaluate the classification performance of the proposed method, the comparisons are conducted and compared with the standard ESNN, DE-ESNN, DEPT-ESNN and other data mining methods. The results of all proposed methods in

terms of all measures are presented in Table 3, Table 4 and Table 6. In these tables, the best results are highlighted in bold font. The results for all data sets involved are analyzed based on structure of ESNN (number of pre-synaptic neurons), parameters values (Mod, Sim and Threshold), sensitivity (SEN), specificity (SPEC), geometric means (GM), negative predictive value (NPV), positive predictive value (PPV), average site performance (ASP) and classification accuracy (ACC) for all data sets. The results of the proposed methods for each data set are analyzed and presented in the following section.

5 Results and Discussion

This section displays the results of the proposed method MODE-ESNN. The results of MODE-ESNN are measured in terms of number of pre-synaptic neurons, Mod, Sim and Threshold. The experiments are run 10 times in the training and testing for all data sets, respectively as shown in Tables 3.

Table 3: Results of number of pre-synaptic neurons and parameters (Mod, Sim and Threshold) for MODE-ESNN

Data set		Pre-synaptic neurons	MOD	SIM	Threshold
Appendicitis	Mean	12.00	0.9535	0.4196	0.5518
	SD	0.00	2.3406	5.8500	1.0000
Haberman	Mean	10.00	0.7991	0.8234	0.0899
	SD	9.61	0.0682	0.0897	0.0207
Heart	Mean	45.00	0.1877	0.5456	0.7761
	SD	11.38	0.1848	0.1550	0.0263
Hepatitis	Mean	23.00	0.6988	0.5088	0.5728
	SD	8.23	0.2128	0.0694	0.0492
Ionosphere	Mean	46.00	0.5623	0.5645	0.7877
	SD	9.49	0.0792	0.0121	0.0815
Iris	Mean	18.00	0.6005	0.4941	0.4824
	SD	12.44	0.1734	0.1863	0.2662
Liver	Mean	17.00	0.6839	0.4824	0.7800
	SD	10.40	0.2614	0.1531	0.2447

The results of MODE-ESNN have revealed the values of pre-synaptic neurons and parameters (Mod, Sim and Threshold) for all data sets. From Table 3, the values of different parameters: Mod, Sim and Threshold are illustrated for all data sets. In addition, the results of pre-synaptic neurons also shown simultaneously because the multi-objective technique is used here. Table 3 shows the average results of the proposed algorithm in terms of parameter *Mod*. The *Mod* parameter importance comes from its representation for the connection weight in ESNN model. If a high value was selected, it means most weights will have a connection value which reflects a well presented weight patterns according to their output class. On the contrary, selecting too low values leads to ending up with most connections assigned with the zero weight value due to the nature of weight computation. The findings show the ability of the proposed method to produce the optimum values of *Mod* parameter which is different from data set to another. The nature of data set

can affect the value of parameter. This clear effect is due to the correlation between parameter optimization and classification accuracy as integrated using the known Wrapper approach. Knowledge discovery from these results of *Mod* parameter shows that there is an important impact of high values of decision of the output classes of instances.

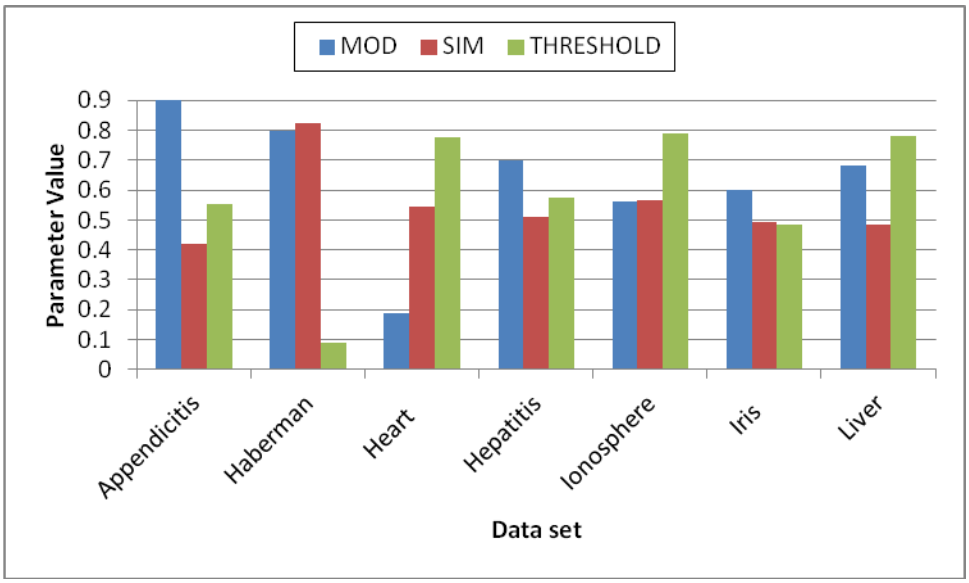

Fig. 3 A comparison of the MODE-ESNN for 10-fold cross-validation in terms of parameter analysis

From the opposite position, the value of parameter *Sim* is important for the model ESNN. Higher values of *Sim* means more neurons with the similarity range that are merged while lower values means else. For almost all data sets the results are shown in Table 3 and Fig. 3. It is noticed that no specific value can be applied to all problems because each problem has its own combination of parameters. Generally, each data set requires specific analysis to understand the problem it represents. Fast decision making demands a comprehensive understanding. Knowledge discovery from these results shows that the high values of Sim parameter leads to establishing better network architecture. This is due to the effect of the parameter Sim in controlling the merge rate of output neurons.

Fig. 3 shows that the average results of the proposed algorithm in terms of parameter *Threshold*. The importance of the *Threshold* parameter is in its function of controlling the PSP threshold. The different values of *Threshold* parameter from data set to another for the proposed method reflects the influence of nature of data set and the hybrid method which control the process of the algorithm. Knowledge discovery from these results of *Threshold* parameter shows that there is a main influence of parameter values of making the decision. Higher values means more spikes and time for this decision. However, if there are smaller values, few spikes are enough to fire an output spike and determine the class of instance.

Moreover, the performance of MODE_ESNN is evaluated in terms of sensitivity, specificity, geometric mean and accuracy. Table 4 illustrates the results of MODE_ESNN in terms of SEN, SPE, GM, NPV, PPV, ASP and ACC for all data sets, respectively.

From Table 4, sensitivity gives high results for Appendicitis, Haberman, Heart, Hepatitis and Iris data sets as well as the specificity values of Ionosphere and Iris data sets. Similarly, the geometric mean gives high results in Iris data set. The observation from the results in Tables 4 has indicated that MODE_ESNN produced high results on accuracy for four out of seven data sets.

Table 4: Results of sensitivity, specificity, geometric mean, NPV, PPV, ASP and accuracy for MODE-ESNN

Data set		SEN	SPE	GM	NPV	PPV	ASP	ACC
Appendicitis	Mean	**83.50**	28.30	48.61	57.90	**97.80**	**83.68**	**73.00**
	SD	0.12	0.42	0.23	0.28	0.05	0.15	10.59
Haberman	Mean	**92.90**	11.70	32.97	45.50	**77.60**	**79.40**	**72.00**
	SD	0.15	0.22	0.18	0.15	0.03	0.07	10.09
Heart	Mean	**100**	22.16	47.08	**81.20**	**76.40**	**81.65**	58.20
	SD	0.00	6.22	0.00	0.16	0.08	0.07	4.53
Hepatitis	Mean	**99.00**	36.00	59.70	52.60	55.70	**70.05**	54.00
	SD	0.03	0.13	0.06	0.16	0.04	0.05	19.99
Ionosphere	Mean	15.30	**85.40**	36.15	**82.90**	**80.10**	**73.00**	**69.55**
	SD	0.13	0.11	0.12	0.07	0.17	0.16	6.30
Iris	Mean	89.00	95.00	91.95	99.40	92.90	95.85	89.71
	SD	0.25	0.07	0.13	0.01	0.075	0.04	3.32
Liver	Mean	44.80	55.1	49.68	66.5	49.00	59.55	50.57
	SD	0.18	0.12	0.147	0.05	0.05	0.06	7.26

Additionally as in Table 4, MODE_ESNN performs high NPV on all data sets except Haberman. For positive predictive value (PPV), MODE_ESNN generates high values for all data sets except Hepatitis and Liver. However, ASP gives the highest results in all data sets except Liver. For negative predictive value (NPV), MODE_ESNN generates high values for Heart, Ionosphere and Iris data sets.

As shown below, the proposed method has been evaluated with several data mining algorithms by using KEEL tool. KEEL software is an open source Data Mining tool widely used in research and real life applications. More details are presented in [41, 42]. The methods are steady-state genetic algorithm for extracting fuzzy classification rules from data(SGERD),CO-evolutionary rele Extractor (CORE), hierarchical decision rules (HIDER) and tree analysis with randomly generated and evolved trees (TARGET)[43]. Additional details concerning these algorithms are provided in Table 5.

Analysis are accomplished on five benchmark data sets from UCI repository (Appendicitis, Haberman, Ionosphere, Iris and Liver). Results of Accuracy performance are shown in Table 6.

Table 5: Description of KEEL data mining algorithm used[43]

Algorithm	Description
CORE	A CO-evolutionary algorithm which employs as a fitness measure a combination of the true positive rate and the false positive rate.
SGERD	A steady state GA which generates a pre-specified number of rules per class following a GCCL approach
HIDER	A method which iteratively creates rules that cover randomly selected examples of the training set.
TARGET	A GA where each chromosome represents a complete decision tree.

As in Table 6, the proposed method MODE-ESNN outperforms other data mining methods for all data sets in terms of accuracy performance.

Table 6: Accuracy Performance of the proposed methods with data mining methods

Dataset	Appendicitis	Haberman	Ionosphere	Iris	Liver
ESNN	100	66.92	91.96	99.33	75.13
DE-ESNN	100	76.26	96.01	99.41	100
DEPT-ESNN	100	84.78	96.2	99.33	100
MODE-ESNN	**100**	**92.3**	**99.4**	**99.78**	**100**
CORE	89.2	76.14	59.22	96.51	63.28
TARGET	88.47	74.07	83.19	87.03	65.08
HIDER	92.56	76.83	97.97	97.48	73.58
SGERD	87.52	74.07	82.36	87.03	65.08

A careful examination of the results to sketch a significant ending: the classification performance of the proposed methods approximately gives best results compared to the classification performance of other methods on different datasets. Accordingly, it can be believed that the inadequate performance in some methods are affected by factors such as: noise, samples, high-dimensionality, missing values, multiple classes and imbalanced classes.

6 Conclusion and Future Works

A hybrid MODE-ESNN method was proposed in this paper to determine the optimal number of pre-synaptic neurons as well as the parameters for a given dataset simultaneously. A comparative study has been conducted between MODE-ESNN and ESNN, DE-ESNN and DEPT-ESNN with other data mining methods to show the performance improvement of ESNN. Both MODE-ESNN and other methods have been used to perform the classification on standard data sets. The results show that MODE-ESNN is able to classify data set with mostly better accuracy than the other algorithms. Moreover, MODE-ESNN uncovered the optimum parameters of ESNN which is considered important for good accuracy. Addi-

tionally, MODE-ESNN mostly shows better results in sensitivity, PPV, ASP and other measurements. For future work, enhancement of the hybridization of a multi differential evolution MODE-ESNN with an ESNN algorithm may be a good trend to be explored to obtain more results. In another direction, these hybrid multi objective evolutionary algorithms with ESNN motivate researchers to investigate the usefulness of integration of other new meta heuristic algorithms to enhance ESNN.

Acknowledgements This work is supported by The Ministry of Higher Education (MOHE) under FRGS (4F347). The authors would like to thank Research Management Centre (RMC), Universiti Teknologi Malaysia (UTM) for the support in R & D, UTM Big Data Centre, Soft Computing Research Group (SCRG) for the inspiration in making this study a success.

References

1. Ahmed, F.Y.H., S.M. Shamsuddin, and S.Z.M. Hashim, *Improved SpikeProp for Using Particle Swarm Optimization.* Mathematical Problems in Engineering, 2013. **2013**: p. 13.
2. Gerstner, W. and W.M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. 2002: Cambridge university press.
3. 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.
4. Hamed, *Novel Integrated Methods of Evolving Spiking Neural Network and Particle Swarm Optimisation*, 2012, Auckland University of Technology.
5. Abbass, H.A., *Speeding up backpropagation using multiobjective evolutionary algorithms.* Neural Computation, 2003. **15**(11): p. 2705-2726.
6. Igel, C., *Multi-objective Model Selection for Support Vector Machines*, in *Evolutionary Multi-Criterion Optimization*, C. Coello Coello, A. Hernández Aguirre, and E. Zitzler, Editors. 2005, Springer Berlin Heidelberg. p. 534-546.
7. Jin, Y., *Multi-objective machine learning*. Vol. 16. 2006: Springer.
8. Jin, Y., T. Okabe, and B. Sendhoff. *Neural network regularization and ensembling using multi-objective evolutionary algorithms*. in *Evolutionary Computation, 2004. CEC2004. Congress on*. 2004. IEEE.
9. Jin, Y., B. Sendhoff, and E. Körner, *Evolutionary Multi-objective Optimization for Simultaneous Generation of Signal-Type and Symbol-Type Representations*, in *Evolutionary Multi-Criterion Optimization*, C. Coello Coello, A. Hernández Aguirre, and E. Zitzler, Editors. 2005, Springer Berlin Heidelberg. p. 752-766.
10. Fieldsend, J.E. and S. Singh, *Pareto evolutionary neural networks.* Neural Networks, IEEE Transactions on, 2005. **16**(2): p. 338-354.
11. Abbass, H.A. *Pareto neuro-evolution: Constructing ensemble of neural networks using multi-objective optimization*. in *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*. 2003. IEEE.
12. Chandra, A. and X. Yao, *DIVACE: Diverse and accurate ensemble learning algorithm*, in *Intelligent Data Engineering and Automated Learning–IDEAL 2004*. 2004, Springer. p. 619-625.

13. Tan, K.C., T.H. Lee, and E.F. Khor, *Evolutionary algorithms with dynamic population size and local exploration for multiobjective optimization.* Evolutionary Computation, IEEE Transactions on, 2001. **5**(6): p. 565-588.

14. Jin, Y., R. Wen, and B. Sendhoff, *Evolutionary multi-objective optimization of spiking neural networks*, in *Artificial Neural Networks–ICANN 2007*. 2007, Springer. p. 370-379.

15. Saleh, A.Y., et al., *A Novel hybrid algorithm of Differential evolution with Evolving Spiking Neural Network for pre-synaptic neurons Optimization.* Int. J. Advance Soft Compu. Appl, 2014. **6**(1).

16. Saleh, A.Y., S.M. Shamsuddin, and H.N.B.A. Hamed, *Parameter Tuning of Evolving Spiking Neural Network with Differen-tial Evolution Algorithm.* International Conference of Recent Trends in Information and Communication Technologies, 2014: p. 13.

17. 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.

18. Kasabov, N., et al., *Dynamic evolving spiking neural networks for on-line spatio- and spectro-temporal pattern recognition.* Neural Networks, 2013. **41**: p. 188-201.

19. Schliebs, S. and N. Kasabov, *Evolving spiking neural network—a survey.* Evolving Systems, 2013. **4**(2): p. 87-98.

20. Kasabov, N.K., *NeuCube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data.* Neural Networks, 2014. **52**: p. 62-76.

21. Kasabov, N., *Evolving spiking neural networks and neurogenetic systems for spatio-and spectro-temporal data modelling and pattern recognition*, in *Advances in Computational Intelligence*. 2012, Springer. p. 234-260.

22. Kasabov, N., et al., *Evolving spiking neural networks for personalised modelling, classification and prediction of spatio-temporal patterns with a case study on stroke.* Neurocomputing, 2014. **134**: p. 269-279.

23. 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.

24. Hamed, et al., *String Pattern Recognition Using Evolving Spiking Neural Networks and Quantum Inspired Particle Swarm Optimization*, in *Neural Information Processing*, C. Leung, M. Lee, and J. Chan, Editors. 2009, Springer Berlin Heidelberg. p. 611-619.

25. 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.

26. Marler, R.T. and J.S. Arora, *Survey of multi-objective optimization methods for engineering.* Structural and multidisciplinary optimization, 2004. **26**(6): p. 369-395.

27. Yen, G.G. and H. Lu, *Dynamic multiobjective evolutionary algorithm: adaptive cell-based rank and density estimation.* Evolutionary Computation, IEEE Transactions on, 2003. **7**(3): p. 253-274.

28. Cohon, J.L., *Multiobjective programming and planning, 1978.* Academic, New York, 2010.

29. Steuer, R., *Multiple criteria optimization: theory, computation, and application. 1986*. Willey, New York, 1986.

30. Philipson, R. and A. Ravindran, *Application of goal programming to machinability data optimization.* Journal of Mechanical Design, 1978. **100**(2): p. 286-291.

31. Charnes, A. and W.W. Cooper, *Management models and industrial applications of linear programming*. Management Science, 1957. **4**(1): p. 38-91.

32. Ijiri, Y., *Management goals and accounting for control*. Vol. 3. 1965: North Holland Pub. Co.

33. 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.

34. 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.

35. Schaffer, J.D., *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*, in *Proceedings of the 1st International Conference on Genetic Algorithms*1985, L. Erlbaum Associates Inc. p. 93-100.

36. Zitzler, E., M. Laumanns, and S. Bleuler, *A tutorial on evolutionary multiobjective optimization*, in *Metaheuristics for multiobjective optimisation*. 2004, Springer. p. 3-37.

37. Deb, K., et al., *Scalable test problems for evolutionary multiobjective optimization*. 2005: Springer.

38. Qasem, S.N. and S.M. Shamsuddin, *Memetic elitist pareto differential evolution algorithm based radial basis function networks for classification problems.* Applied Soft Computing, 2011. **11**(8): p. 5565-5581.

39. Mezura-Montes, E., M. Reyes-Sierra, and C.A.C. Coello, *Multi-objective optimization using differential evolution: a survey of the state-of-the-art*, in *Advances in differential evolution*. 2008, Springer. p. 173-196.

40. Do, K.-A. and C. Ambroise, *Analyzing microarray gene expression data*. Wiley, 2004. **14**: p. 1080-1087.

41. García, S., J. Luengo, and F. Herrera, *A Data Mining Software Package Including Data Preparation and Reduction: KEEL*, in *Data Preprocessing in Data Mining*. 2015, Springer. p. 285-313.

42. Derrac, J., et al. *Using KEEL software as a educational tool: A case of study teaching data mining*. in *Next Generation Web Services Practices (NWeSP), 2011 7th International Conference on*. 2011. IEEE.

43. Alcalá, J., et al., *Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework.* Journal of Multiple-Valued Logic and Soft Computing, 2011. **17**: p. 255-287.