International Journal of Engineering & Technology, 7 (4.11) (2018) 145-151

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Using K-Fold Cross Validation Proposed Models for Spikeprop Learning Enhancements

Falah Y.H. Ahmed[1]*, Yasir Hassan Ali[2], Siti Mariyam Shamsuddin[3]

[1]*Department of Information Science and Computing, Faculty of Information Sciences and Engineering (FISE), Management and Science University, 40100 Shah Alam, Selangor, Malaysia*

[2]*North Technical University, Technical Collage Mosul, Mosul, Iraq*

[3]*UTM Big Data Centre, Ibnu Sina Institute for Scientific and Industrial Research, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia*

**Corresponding author E-mail: falah_ahmed@msu.edu.my*

Abstract

Spiking Neural Network (SNN) uses individual spikes in time field to perform as well as to communicate computation in such a way as the actual neurons act. SNN was not studied earlier as it was considered too complicated and too hard to examine. Several limitations concerning the characteristics of SNN which were not researched earlier are now resolved since the introduction of SpikeProp in 2000 by Sander Bothe as a supervised SNN learning model. This paper defines the research developments of the enhancement Spikeprop learning using K-fold cross validation for datasets classification. Hence, this paper introduces acceleration factors of SpikeProp using Radius Initial Weight and Differential Evolution (DE) Initialization weights as proposed methods. In addition, training and testing using K-fold cross validation properties of the new proposed method were investigated using datasets obtained from Machine Learning Benchmark Repository as an improved Bohte's algorithm. A comparison of the performance was made between the proposed method and Backpropagation (BP) together with the Standard SpikeProp. The findings also reveal that the proposed method has better performance when compared to Standard SpikeProp as well as the BP for all datasets performed by K-fold cross validation for classification datasets.

Keywords: *SpikeProp, K-fold cross validation, reduce time error measurement, Spiking Neural Network and Backpropagation (BP).*

1. Introduction

Discriminate analysis, a traditional statistical classification procedure is built on the Bayesian decision theory of Backpropagation (BP). In these procedures, a fundamental probability model must be assumed in order to compute the posterior prospect upon which the judgment was decided. Various assumptions and conditions will be applied under, which the models are wisely developed and examined for effectiveness and perfection. Anyway, before the models can be successfully function, users must have a good knowledge and skills of both data properties and model capabilities. As neural networks do not mainly relay on prior knowledge of the data statistics, they have appeared as an important tool for classification. Differential Evolution (DE) is one of the standard evolutionary algorithms (EA) which works through repeated computational steps. D.E. is considered a stochastic powerful search algorithm introduced by [42].

Integrate-and-fire model was the first model to be applied in spikes over time to transfer information [2]. As the input spikes reach in time, the inner potential of the neuron will depend on the weights (postsynaptic potential, PSP) at the point of the input. The connections are excitatory and the incoming stimulus works to subsequently increase the PSP when the weights are positive. On the other hand, the inhibitory as a stimulus going through them will act to decrease the PSP with negative weights. An output spike is released when the postsynaptic potential reaches a threshold. As shown in Figure 1, the PSP is reset to its resting potential in the point of an output spike.

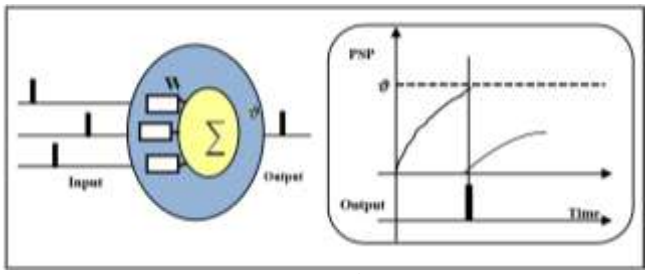

Fig. 1: Integrate-and-fire neuron [3]

The performance of SpikeProp is determined by the learning parameters of the BP network. Hence, it is necessary for this study to focus on finding ways to enhance the performance of SpikeProp by optimizing the back propagation initializing weights and the architectures of SNNs. Differential Evolution (DE) is one technique which can be used for this purpose.

2. Related Work

Some forms of optimization are required in the learning process for the purpose of the ANN interconnector's weights updating, inspired by how the biological swarm of animals acts in order to obtain a desirable goal for the group. One useful optimization technique is uses DE [4]. DE has been broadly applied to solve several world problems [5, 6], in addition to the introduction of the binary version of DE. There have been several advances and

Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

enhancements in this area [7-9]. DE has been used to derive universal function approximations for any analog function with random updating of weights [42].

For the first generations of ANN, the neurons are restricted to binary inputs and outputs. These binary impulses have a definite width, phase and time. These types could be considered driven by stabilized frequencies of the neuron. However, it has been discovered recently that neurons communicate by firing short electrical pulses which operate in a mode referred to as rate coding. Higher output signal can be achieved by firing at higher rate. Communication is performed using real spikes with two exclusive instants known as spiking time or no spiking time [37, 38]. The value of output of a neuron can be computed and the response of the network to the values of input is known or identified only after all the neurons have been fired with a communication window of this type. Each neuron can be modeled as it has a basic firing-rate and a continuously constant activation function. Spiking Neural Networks (SNN) as a model is considered to constitute the third generation of ANNs [10, 39, 40]. The output signals on the other hand can be continuously altered by variation in synaptic plasticity for all the three generations of neural networks. The basis for learning in all ANN is Synaptic plasticity, unless there is a non-variant activation function and accurate classification based on a certain vector input values which can be implemented with the help of a BP learning algorithm like gradient-descent [11, 12]. Spiking Neural Network (SNN) employs single spikes in time domain in order to communicate and to perform computation in a manner just as the real neurons react [13, 14].

This method of sending and receiving individual pulses is called pulse coding, where the transmitted information is carried by the pulse rate. Hence, this type of coding allows multiplexing of data [15]. For example, analysis of visual input in humans requires not more than 100ms for facial recognition. However, in [16] performed facial recognition using SNN with a minimum of 10 synaptic steps on the retina at the temporal lobe that allows nearly 10ms of the neurons to be performed. The processing time is short, yet, it is adequate to permit an averaging procedure that is required by pulse coding [3, 15, 16]. In fact, when computation speed is a problem, pulse coding technique is preferred [16].

3. Methodology

This experimental section can be divided into subsections, the contents of which vary according to the subject matter of the article. It must contain all the information about the experimental procedure and materials used to carry out experiments.

3.1. Initialization weights Defined by Differential Evolution (DE)

This segment discusses the creation of initialization weights performing D.E. strategies (Figure 2). D.E. operates on two sets of N operator vectors each of dimension D to form 2 N x D matrices. The first set is the hidden Weights, while the second set is the output weights. Spikeprop initial weights population were stored in each element of the two matrices. The operator vector weights may be composed of heterogeneous dataset from various observations, as in this study, where nine different standard datasets were applied. The first matrix make up the SpikeProp hidden weights (HiddnW), whereas the second matrix contains the SpikeProp output weights, (OutoutW) [21]. Initially, the D.E. starts by filling up the row matrices with vector weights and randomly generating operator values. Each vector weight (Wi) in both matrixes are randomly generated and sequentially treated for operation. Furthermore, the vector weight has three other vector weights which are Wc1, Wc2 and Wc3 that are randomly selected from the remaining vector weights of the same matrix of Wi. Yet, a new trial of vector weight generation is required to achieve a mutation steps. Eventually, these stages implement the D.E. for generating the initial weights iteratively in an optimum random manner.

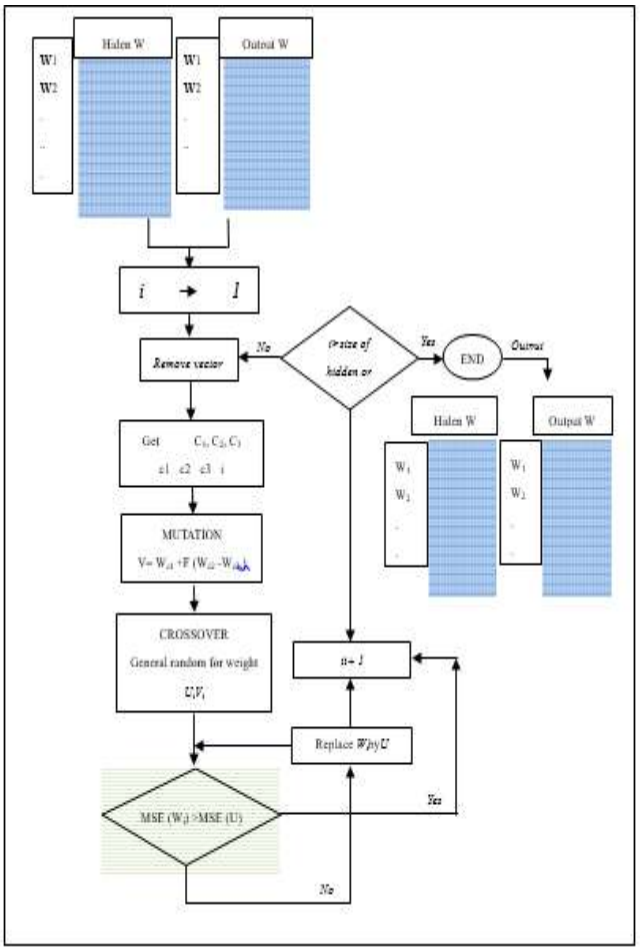

Fig. 2: Flowchart for the Differential Evolution created Initialization weights and Encoding Input Datasets Variables

In SpikeProp, an encoding process for dataset needs to be implemented to get an efficient encoding and cluster capacity expansion. The aim of this encoding method is to increase the distance cross the input patterns, which are related with respective input data. It has been observed that the synapses have a resolution delay of about 1ms. Therefore, the perceptive power of SpikeProp learning is limited to roughly this resolution. These encoding increases the temporal distance cross the data points and separates cluster widely [21]. As the aim of data input encoding, multifarious local receptive sets are used to distribute the input variable value over multiple input neurons [22-26]. The population code where the input variables are encoded with layered overlapping activation functions are studied in depth for applying real valued parameters.

The firing time of an input neuron is calculated using the intersection of Gaussian function as defined in (1). The Gaussian mean μiis calculated in (2) and its width is computed in (3) with the input variable interval of I_{max} and I_m. I_{min} and I_m define the minimum and maximum input values. The parameter controls the width of each Gaussian receptive field where $1 \leq \beta \leq 2$.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2} 2 \quad (1)$$

$$\mu_i = I_{max} + \frac{2_i - 3}{2} \cdot \frac{I_{max} - I_n}{M-2} \quad (2)$$

And width

$$\sigma = \frac{I_{max} - I_m}{M-2} \quad (3)$$

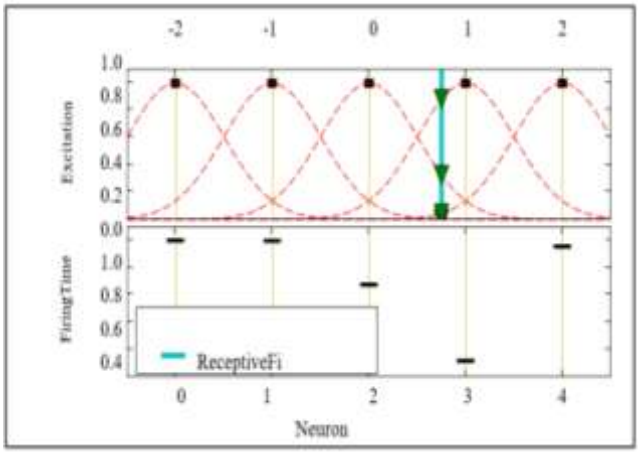

Fig. 3: Population Encoding Method [33, 34]

Figure 3 schematically illustrates population Encoding Method. This illustration has been redrawn from [34]. For the diagram, (β = 2) was used and the input interval $I_{max} - I_m$ was set to [−2.0, 2.0] as for the neurons input (M = 5) was used. In this example, the input value was defined as 0.70ms and five firing times were calculated based on the Gaussian intersections. Both spike encoding methods have been tested in several applications such as visual recognition [16, 28], audio recognition [27] and speech recognition [29].

3.2. K-Fold Cross Validation

K-fold cross validation is the validation experiments in which training, examination and testing will be applied. In the first experiment, 90% of the random datasets are used for training while the remaining 10% will be used for testing. In the second experiment, a completely different 90% of the random datasets is used for training while the remaining 10% is used for testing. The experiments are repeated with a completely different 90% of the datasets is used for training and the remaining 10% of the datasets is used for testing as illustrated in Figure 4, where 10 experiments will be performed subsequently. Assuming that the datasets chosen for training and testing are completely random and the process of N-fold cross validation is ergodic, the correct output is the average of the outputs of all experiments [30].
Sensitivity of validation process are determined by two main factors. The first factor is in model selection. That is each experiment must select the 90% of the datasets for training in a completely random manner. The second factor is in performance evaluation. For the creation of N-fold portion from the dataset, it is assumed that for a single N experiment, N-1 folds will be utilized for training and testing (Figure 4). The true error is evaluated as the average error rate as it is calculated in (4).

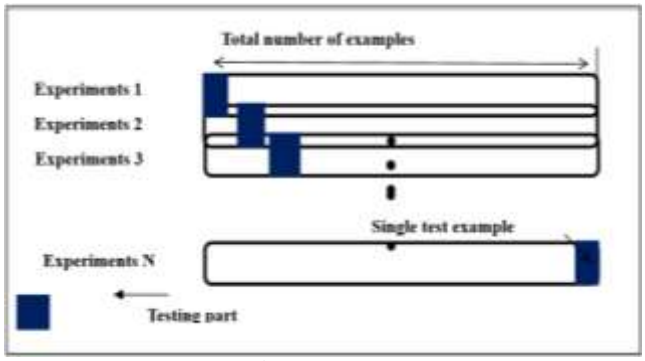

Fig. 4: K-fold cross validation [31]

3.3. K-Fold Cross Validation

Classification accuracy depends on the datasets used in the testing. The datasets used for training and the whole process of training and testing (done using random selection of datasets). The accuracy vale is calculated as:

$$Accuracy = \sum_R \frac{1 - \left| t_{j_2}^a - t_{j_2}^d \right|}{\max(t) - \min(t)} / n_r, \quad (4)$$

where R is the set of records, t_r is the target output value for record r, o_r is the prediction generated by the network for record r and n_r is the number of records.
For multiple outputs models, the accuracy is considered as the simple average of the accuracies of individual outputs.

3.4. Dataset Collection

Several experiments were conducted on the datasets in order to evaluate the performances of all the proposed methods. All datasets were downloaded from the machine learning benchmark repository which contains information data relating to several complex environmental problems in qualitative analytical chemistry. Such a topics have been the major concern in many studies related to Artificial Neural Networks (ANNs), Spiking neural networks (SNNs), machine learning [32] and Spikeprop implementation [21]. The datasets used in this study are sourced from XOR, breast tumor, BTX, heart functions, hepatitis, Pima Indian diabetes, liver care, iris problem databases. It may be possible to get good SpikeProp algorithm by hybridizing two good techniques models, just the way we could get a good strain and results by cross breeding two good genes. Therefore, this paper looks at the Hybridization implementation is combination of DE Weights Initialization and RIW Table 1 shows the dataset collection used in this work.

Table 1: Description of datasets

Data Set	Attributes	Classes	Samples	Input	Output	Training	Testing
Breast Cancer	9	2	683	9	2	538	145
BTX	19	3	512	19	7	407	105
Diabetes	8	2	768	8	2	613	155
Heart	13	2	297	13	2	240	57
Hepatitis	19	2	155	19	2	123	32
Iris	4	3	150	4	3	120	30
Liver	6	2	345	6	2	276	69

3.5. DE Initialization Weights Enhancement

For proposed method, initialization weights are decided through DE chromosomes (agents) which will give information whether they are near or far away from the optimum position. Those DE chromosomes (agents), which are near the optimum position are chosen and a circle containing these DE chromosomes (agents) are crated. For the second iteration, DE chromosomes (agents) are distributed in the circle. Again, DE chromosomes (agents) will give information whether they are near or far away from the opti-

mum position. Only those DE chromosomes (agents), which are near the optimum position are chosen.

```
  Begin,
Step 1: Initialization. Set the generation number for
the weights  W=0, and randomly initialize a population
of  N target individuals,  Wi = (w1, ,w2, , . . . ,wN,),
with Wi, = (w1i, ,w2i, , . . . ,wni,), i=1,2, . . . ,
N uniformly distributed in the range [Wmin, Wmax].
Step 2: Evaluate each target individual in P= (w1, ,
w2, , . . . ,wN ).
Step 3: Generate N trial individuals. For i=1 to N repeat
Steps3.1-3.2.
      Step 3.1: Mutation. Generate a mutant vector of
     the weights  Vi, = (v1i, ,v2i, , . . . ,vni,G) for
     wi,G, i=1,2,…, N by using a mutation operator.
      Step 3.2: Crossover. Generate a trial vector of
     the weights  Ui, = (u1i, ,u2i, , . . . ,uni, ) for
     wi, , i=1,2,…, N by applying a crossover operator
     on Vi, and wi.
Step 4: Selection for next generation weights. Evaluate
each Ui, ,i=1,2,…, N, and determine the members of the
target population of the next generation weights  by
using the one-to-one selection scheme.
Step 5: Increment the generation count weights Wi=Wi+1.
If a termination criterion is not satisfied, go to
step3; Otherwise, stop the procedure and output Wbest
and f(Wbest).
        End.
```

Fig. 5: DE Weights Initialization procedure

A smaller circle is then drawn. The procedure is repeated until a very small circle (elemental circle), enclosing the optimum position is found. This elemental circle is in fact the best initial weights for SpikeProp. SpikeProp which uses D.E. initialization weights give better result than standard Spikeprop. The process of obtaining the output initialization weights using DE for Spikeprop is illustrated in Figure 6. The algorithm employed is shown in Figure 5, Hybridization of DE and RIW for Spikeprop Learning Enhancements

Cross breeding shown good strain and results by two good genes, it is believed that to get good SpikeProp algorithm, the hybridizing of two good techniques models have to be used. Therefore, this study aims to look at the Hybridization implementation in combination with the DE initialization weights and RIW. The strength of DE is the fact that it can sense the optimum point even though the optimum point is far away. When the optimum point is near, this optimum point can be identified quickly and accurately by using RIW. DE has no active knowledge sharing. In other words, the swarm particles do not communicate interactively with one another.

The process can be considered as a cleverly contrived trial and error. When implementing DE, the path of Weights Initialization will be more direct and more efficient if there is active communication between one direction and another points; also between the optimal point and direction points. DE provides the intelligence for Weights Initialization that will behave as a moving DE chromosome.

In Weights Initialization by DE component will determine the initialization weights which are then fine-tuned by using the DE component. Improved Spikeprop that uses DE initialization weights can rely on RIW to calculate the initial weights, which are then stored in the hidden weights matrix and the output weights matrix. Since Weights Initialization by DE possesses the good traits of RIW and DE, it has better performance than the standard SpikeProp. Figure 6 illustrates the process of Hybridization implementation methods.

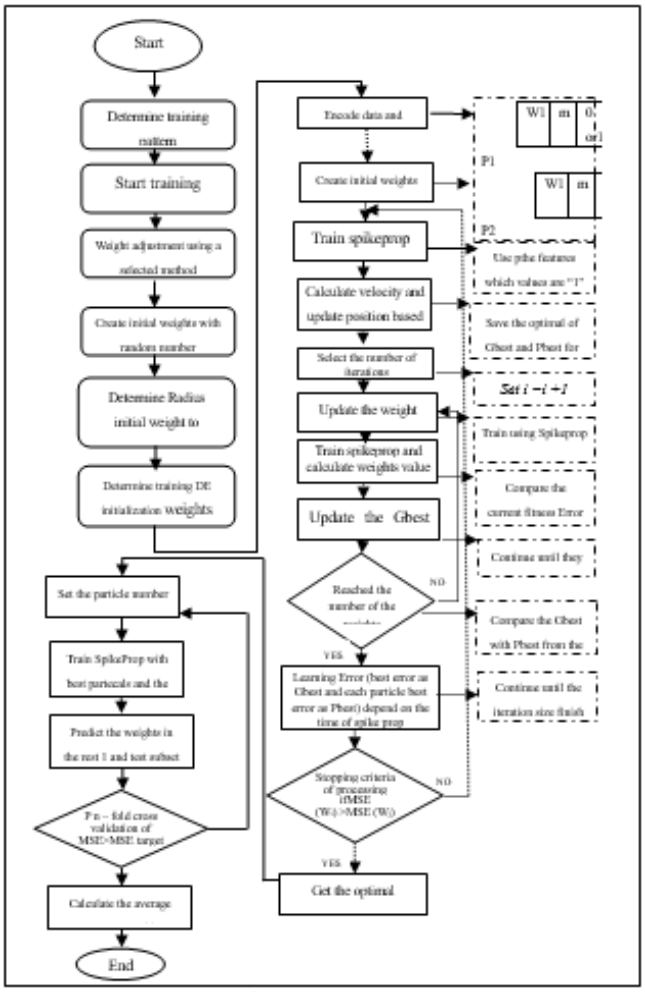

Fig. 6: Flowchart of DE Weights Initialization and RIW

3.6. Proposed Radius Initial Weight (RIW)

To bring about acceleration in SpikeProp, the actual weights are assigned for random initialization values in the range between [-1, 1]. When the initial weight values are small, the hidden nodes and the output nodes will cease to exist, decelerating the learning process. Input values within the range [-1,1] should be chosen carefully while the activation derivative function is allowed to use appropriate small values [36] for BP to converge to its optimum solution. Whether be it BP or SpikeProp, it is prudent to use the same initialization weight matrices for all layers of the network [35].

Acceleration can be further improved by reducing the weight values by an amount equal to the value of an average random initialization weight in each column on the weight matrices. Therefore, this method is believed to be depends on the initial weight values of BP algorithm. The following describes the steps for the proposed methods.

a) $v_{ij}(old)$, $w_{ij}(old)$ are the small random values between the hidden layer and output layer in the range of [-1, 1].

b) Calculating the range weight of the two matrices $m1_i$ and $m2_j$, for each column.

$$m1_i = \sum_{i=1}^{n} v_{ij'},$$
$$where, \quad j = 1,\ldots,n \tag{5}$$

$$m2_i = \sum_{i=1}^{m} w_{ij} \ ,$$
$$where, \quad j = 1,\ldots,n \tag{6}$$

c) To find the value of new weights vector:

$$v_{ij}(new) = \frac{v_{ij}(old)}{m1_i} \tag{7}$$

$$w_{ij}(new) = \frac{w_{ij}(old)}{m2_i} \tag{8}$$

The Spiking Neuron Network (SNN) is trained by error backpropagation (SpikeProp), as in (5) or in (6). On the other hand, the synaptic weights are adjusted accordingly to minimize the network error. Therefore, in SpikeProp the actual weights are assigned to random initialization values followed the radius rule as depicted in (7) and (8). The input values range on the other hand will differ from each of dataset used.

$InRadius = (InputMax - InputMin) \div$
$W_{j,k}(ol$ are the small random values between the hidden layer and output layer in the range of [-1,1].

$$W_{j,k}(new) = InRadius * \left(2 * \frac{X}{input - 1}\right) * W_{j,k}(old) \tag{9}$$

where X = Round (input/output).
The following example rounds X to two values for Input / output after the decimal.
Referring in (9), the selection of the X is carried out in such a manner as soon as the input dataset is approximately similar to that of output dataset, choose X = 2. But, if the input dataset is generally more than twice the output dataset, choose X = 6. Correct choice of the numbers results in getting the solution in the radius point (central arithmetic).

4. Results and Discussion

The results of the proposed enhanced Spikeprop algorithm are presented. The design of the proposed method is given follows by the explanation on the SpikeProp algorithm. The performance evaluation and comparisons have been implemented in terms of the convergence error, average error, and accuracy. With the intention to investigate the performance of the planned algorithms, nine standard datasets are considered in this study. They are real world datasets (section 3), which are unlike the respect number of available samples (32-484), attributes (3-56) and classes (2-10). These standard datasets have been used in the previous studies [31].

4.1. Analysis of the Standard SpikeProp and the Proposed Method Using K-Fold Cross-Validation

K-fold cross-validation is away better over hold-out validation in all observations used for both training and testing, as every single observation used for testing were exactly once [31]. In justification of the k-fold cross-validation, the folds are carefully chosen so that the mean response value is roughly equal in all the folds. However, in the event of a dichotomous classification, every fold contains around the exact amounts of the other two kinds of class labels. The proposed algorithms are evaluated by 10-fold cross validation. Initially, the dataset required to be divided randomly into equal size of ten subsets. Every single subset is used as a testing dataset, yet the other nine subsets are used as a training datasets. The training and testing processes are recurrent so that the rest of all the subsets are used as a testing dataset. The training set on the other hand, is applied to train the network to get the optimal solutions, whereas the testing set is used to test the generalization performance of the planned methods. However, this was not observed by any individual SpikeProp network when the training process. The performance of the algorithms is evaluated by conducting four different error measurements.
The results of these methods proposed are based on error measurements and these are presented in next. The average of RMSE is better than MSE for training the standard SpikeProp in eight datasets as explained in Table 2. It demonstrated that the standard SpikeProp has produced the smallest error on BTX, Iris, Heart, Diabetes, breast cancer, liver and Hepatitis dataset respectively.

Table 2: Results for Standard Spikeprop for 10-Fold Cross-Validation

	Training SpikeProp				Testing SpikeProp			
	MSE	RMSE	MAPE	MAD	MSE	RMSE	MAPE	MAD
Breast Cancer	0.5613	0.5069	44.2247	1.0138	0.5711	0.4984	43.2685	0.9968
BTX	1.6796	0.2204	35.5672	1.5434	2.1596	0.2659	43.8155	1.8615
Diabetes	0.4423	0.4007	32.7159	0.8015	0.5045	0.4305	35.2056	0.8610
Heart	0.4393	0.3719	31.8171	0.7439	0.3874	0.3545	31.4742	0.7091
Hepatitis	0.7101	0.5665	49.0714	1.1331	0.7250	0.5768	52.1207	1.1536
Iris	0.7071	0.3339	35.5990	1.0019	0.5947	0.2809	34.9229	0.8427
Liver	0.6895	0.5296	42.2933	1.0593	0.7143	0.5377	42.7806	1.0754

4.2. Analysis of the DE Initialization Weights Using K-Fold Cross-Validation

The result of proposed method (Hybridization of DE initialization weights and RIW**)**. In this model, the structure of Model 1 (DE-SpikeProp) is implemented using DE weights particles initialization and radius initial weight using K-Fold Cross-Validation. In proposed method, we notice that the performance measurement is much better than the standard SpikeProp, BP and previous proposed method when the hybridization takes place as shown in Table 3. The results in this Model from Table 3, the errors measurement are present as (MSE, RMSE, MAPE and MAD). As shows from Table 3, the effectiveness results of this merging model in the prediction of 8 datasets within 250 iterations for training and testing. The values of errors measurement are compared between merging method, SpikeProp standard and BP. The comparison was made, it is gives better results in RMSE better in BTX, Iris, Heart, Liver and Hepatitis is being compared. In error of MSE is less better of Heart, Iris, Liver, Hepatitis and BTX. However, the MSE gives relativity a better than RMSE in Diabetes and Breast Cancer data problems as shown in Table 3 from training part. Even though, the MAD is bigger than RMSE for all dataset problems.

Table 3: Results for Spikeprop based on DE Initialization Weights using for 10-Fold Cross-Validation

	Training Model 5				Testing Model 5			
	MSE	RMSE	MAPE	MAD	MSE	RMSE	MAPE	MAD
Breast Cancer	0.3581	0.3632	35.5693	0.7265	0.3503	0.3559	34.7571	0.7118
BTX	0.9561	0.1598	26.9413	1.1189	1.5238	0.2167	37.4711	1.5174
Diabetes	0.2564	0.2613	24.3281	0.5227	0.2881	0.2837	26.5072	0.5675
Heart	0.3048	0.2967	27.9364	0.5934	0.3055	0.2997	28.2429	0.5995
Hepatitis	0.5054	0.4650	43.0645	0.9301	0.5096	0.4698	42.4454	0.9396
Iris	0.3140	0.2279	32.9549	0.6838	0.3010	0.2185	29.9411	0.6555
Liver	0.3625	0.3416	32.0119	0.6833	0.3766	0.3527	32.9843	0.7054

4.3. Analysis of the Proposed Method Radius Initial Weight (RIW)

The temporal encoded networks of spiking neurons are depends on the estimations that every neurons fire minimum one time. However, the weight limitation approach introduced in this method is to guarantee that neurons fire is not later than the maximal network delay and initial weights enable networks to converge to zero error rapidly. The proposed RIW creates the input neurons which are more efficient and results in a lesser number of value for first start firing time (this will explain in more details in next section). The experiments in Table 4 show the error measurements of training and testing (MSE, RMSE MAPE and MAD) datasets. The standard SpikeProp algorithm has been significantly enhanced not only to become more reliable and efficient, but also in a way where the networks can be designed in smaller sizes. This proposed method of RIW gives the input number values for all datasets either 2 or 6; the selective value depends on the testing during the implementation. When the number of inputs is near to the output number of the same dataset, we can choose the value of 2. However, if the input numbers are double of the output numbers, the chosen value is 6. This will be illustrated in more details in the next sections. From Table 4, RMSE has shown better results for error generalization on BTX, Iris, Diabetes and Heart, and less competitive in Liver, Breast Cancer, Hepatitis and XOR. In regards to MSE, the results are better for Diabetes, Heart and Breast Cancer, and less competitive in Liver, Hepatitis, Iris and BTX but XOR still portrait the same values for RMSE.

Table 4: Results of Spikeprop based on Radius Initial Weight (RIW) for 10- Fold Cross-Validation

	Training				Testing			
	MSE	RMSE	MAPE	MAD	MSE	RMSE	MAPE	MAD
Breast Cancer	0.4942	0.4619	41.4347	0.9238	0.5110	0.4758	42.4348	0.9517
BTX	1.6512	0.2065	36.1897	1.4460	1.7368	0.2317	43.9632	1.6223
Diabetes	0.3523	0.3376	28.7361	0.6753	0.4064	0.3697	30.4818	0.7394
Heart	0.4247	0.3514	29.2199	0.7029	0.3711	0.3240	31.7256	0.6481
Hepatitis	0.5443	0.5003	47.3719	1.0006	0.5008	0.4625	53.5463	0.9257
Iris	0.5865	0.3215	37.3042	0.9645	0.1217	0.2995	18.8008	1.9787
Liver	0.5258	0.4443	37.9292	0.8886	0.5567	0.4622	37.8640	0.9245
XOR	4.375	2.489	21.4644	2.4890	4.375	2.4890	21.4644	2.4890

4.4. Result and Analysis Comparison In Terms of Accuracy Using K-Fold Cross-Validation

The experiments are conducted on ten independent runs for training and testing datasets respectively (refer to Table 2, 3 and 4 and Figure 7). As shown in Table 5, for training, the proposed methods give better results for Breast Cancer datasets with standard SpikeProp and other method except for method of RIW. Similarly, to the BTX datasets the outcomes are better for standard SpikeProp sand other proposed methods except for RIW.

The accuracy for the DE Initialization weights to enhance SpikeProp is better than standard SpikeProp and other method except RIW, also in Liver datasets. But, in Hepatitis and Iris datasets, the results are not convincing for RIW and DE Initialization weights, so standard is not better SpikeProp and other methods except DE Initialization weights in Diabetes and Heart. However, RIW is better than standard SpikeProp and other methods. In RIW the accuracy is better than standard SpikeProp and BP in all dataset. For DE Weights Initialization the accuracy is better than standard SpikeProp, RIW and BP in all eight datasets as using the K-Fold Cross-Validation. On the other hand, hybridization of DE Initialization weights and RIW, is better than DE Initialization weights and standard SpikeProp for all eight datasets. Moreover, it is also better than Model 1 in Diabetes, Heart, Hepatitis and Iris. Finally, Figure 7 shows better accuracy for all the proposed methods and standard SpikeProp in all datasets.

Table 5: Accuracy for 10- Fold Cross-Validation

	Testing			
	SpikeProp	RIW	DE Initialization	BP ANN
Breast Cancer	47.55	51.30	57.99	31.42
BTX	59.65	67.77	70.75	41.57
Diabetes	56.59	62.83	75.99	26.72
Heart	55.26	61.85	75.49	26.24
Hepatitis	40.76	45.90	56.55	14.36
Iris	25.24	31.20	50.06	14.02
Liver	6.469	12.84	25.41	2.39

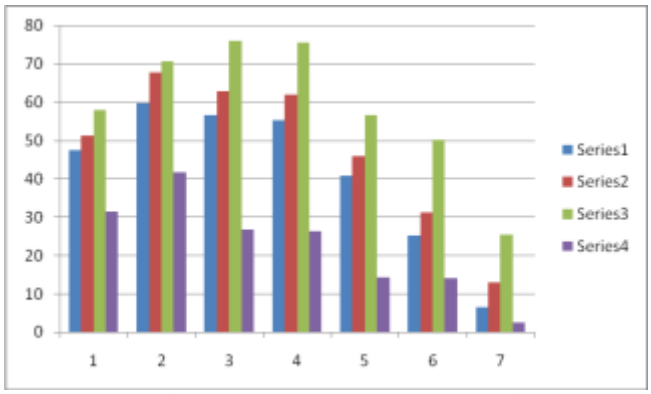

Fig. 7: Accuracy of the proposed methods and SpikeProp for testing in K-Fold Cross-Validation

5. Conclusion

The experiments are divided into one evaluation schemes: K-fold cross-validation. The development of SpikeProp algorithm was done using Spikeprop based on Radius Initial Weight (RIW), Spikeprop Learning based on DE Weights by Initialization and Hybrid of RIW and DE Weights by Initialization were in addition successfully tested on Breast Cancer, BTX, Diabetes, Heart, Hepatitis Iris as well as Liver datasets. The empirical analysis was done

by comparing the results obtained from all the experiments. In addition, the analysis was conducted based on the results achieved from each dataset.

The accuracy of the proposed methods has been formally analyzed for classification is K-Fold Cross Validation. This study shows that SpikeProp and BP with Proposed methods provide feasible results in terms of accuracy. The proposed DE Weights by Initialization is the best method using hold-out validation for all datasets and RIW in k-fold cross-validation in almost datasets. In terms of errors in time (ms), the findings show that DE Weights by Initialization and RIW are better than SpikeProp and also the BP standard for every dataset.

References

[1] Duda, R. O., Hart, P. E., & Stork, D. G. (1973). Pattern classification and scene analysis. John Wiley and Sons.

[2] Gerstner, W., & Kistler, W. M. (2002). Spiking neuron models: An introduction. Cambridge University Press.

[3] Kasabov, N. (2012). Evolving, probabilistic spiking neural networks and neurogenetic systems for spatio- and spectro-temporal data modelling and pattern recognition. Natural Intelligence: The INNS Magazine, 1(2), 23-37.

[4] Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39-43.

[5] Xu, J., Lam, A. Y., & Li, V. O. (2011). Chemical reaction optimization for task scheduling in grid computing. IEEE Transactions on Parallel and Distributed Systems, 22(10), 1624-1631.

[6] Kennedy, J., & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation, pp. 4104-410.

[7] Khanesar, M. A., Teshnehlab, M., & Shoorehdeli, M. A. (2007). A novel binary particle swarm optimization. Proceedings of the IEEE Mediterranean Conference on Control and Automation, pp. 1-6.

[8] Srinivasan, T. R., & Shanmugalakshmi, R. (2012). Neural approach for resource selection with PSO for grid scheduling. International Journal of Computer Applications, 53(11):37-41.

[9] Yuan, X., Nie, H., Su, A., Wang, L., & Yuan, Y. (2009). An improved binary particle swarm optimization for unit commitment problem. Expert Systems with Applications, 36(4), 8049-8055.

[10] Grüning, A., & Sporea, I. (2011). Supervised learning of logical operations in layered spiking neural networks with spike train encoding. Neural Processing Letters, 36(2), 117-134.

[11] Yu, J., Xi, L., & Wang, S. (2013). An improved particle swarm optimization for evolving feedforward artificial neural networks. Neural Processing Letters, 26(3), 217-231.

[12] Gerstner, W., Kempter, R., van Hemmen, J. L., & Wagner, H. (1999). Hebbian learning and spiking neurons. Physical Review E, 59(4), 4498-4514.

[13] Belatreche, A., & Paul, R. (2012). Dynamic cluster formation using populations of spiking neurons. Proceedings of the IEEE International Joint Conference on Neural Networks, pp. 1-6.

[14] Ferster, D., & Spruston, N. (1995). Cracking the neuronal code. Science, 270(5237), 756-757.

[15] Gerstner, Kempter, Hemmen and Wagner. (1999). Hebbian learning of pulse timing in the Barn Owl Auditory System in Maass. Berlin: MIT-Press.

[16] Thorpe, S., Delorme, A., & Van Rullen, R. (2001). Spike-based strategies for rapid processing. Neural Networks, 14(6–7), 715-725.

[17] Bohte, S. M., Kok, J. N., & Poutré, J. A. L. (2000). Spike-prop: Error-backpropagation in multi-layer networks of spiking neurons. Proceedings of the 8th European Symposium on Artificial Neural, pp. 1-6.

[18] Xin, J., & Embrechts, M. J. (2001). Supervised learning with spiking neural networks. Proceedings of International Joint Conference on Neural Networks, pp. 1772-1777.

[19] Moore, S. C. (2002). Back-propagation in spiking neural networks. Master thesis, University of Bath.

[20] Tiňo, P., & Mills, A. J. (2005). Learning beyond finite memory in recurrent networks of spiking neurons. Neural Computation, 18(3), 591-613.

[21] Bohte, S. M., Kok, J. N., & La Poutre, H. (2002). Error-backpropagation in temporally encoded networks of spiking neurons. Neurocomputing, 48(41-44), 17–37.

[22] Baldi, P., & Heiligenberg, W. (1988). How sensory maps could enhance resolution through ordered arrangements of broadly tuned receivers. Biological Cybernetics, 59(4-5), 313–318.

[23] Eurich, C. W., & Wilke, S. D. (2000). Multi-dimensional encoding strategy of spiking neurons. Neural Computation. 12(17), 1519–1529.

[24] Pouget, A., Deneve, S., Ducom, J. C., & Latham, P. E. (1999). Narrow vs. wide tuning curves: What's best for a population code? Neural Computation, 11(1), 85–90.

[25] Snippe, H. P., & Koenderink, J. J. (1992). Discrimination thresholds for channelcoded systems. Biological Cybernetics. 66(6), 543–551.

[26] Zhang, G., Hu, M. Y., Patuwo, B. E., & Indro, D. C. (1999). Artificial neural networks in bankruptcy prediction: General framework and cross-validation analysis. European Journal of Operational Research, 116(1), 16–32.

[27] Wysoski, S. G., Benuskova, L., & Kasabov, N. (2006). On-line learning with structural adaptation in a network of spiking neurons for visual pattern recognition. Proceedings of the International Conference on Artificial Neural Networks, pp. 61-70.

[28] Wysoski, S. G., Benuskova, L., & Kasabov, N. (2007). Text-independent speaker authentication with spiking neural networks. Proceedings of the 17th International Conference on Artificial Neural Networks, pp. 758-767.

[29] Loiselle, S., Rouat, J., Pressnitzer, D., & Thorpe, S. (2005). Exploration of rank order coding with spiking neural networks for speech recognition. Proceedings of the IEEE International Joint Conference on Neural Networks, pp. 2076-2080.

[30] McLachlan, G., Do, K. A., & Ambroise, C. (2005). Analyzing microarray gene expression data. John Wiley and Sons.

[31] Qasem, S. N., & Shamsuddin, S. M. (2012). Radial basis function network based on time variant multi-objective particle swarm optimization for medical diseases diagnosis. Applied Soft Computing, 11(11), 1427–1438.

[32] Yu, J., Xi, L., & Wang, S. (2007). An improved particle swarm optimization for evolving feedforward artificial neural networks. Neural Processing Letters, 26(3), 217-231.

[33] Kasabov, N. (2009). Integrative connectionist learning systems inspired by nature: Current models, future trends and challenges. Natural Computing, 8(2), 199-218.

[34] Schliebs, S., Defoin-Platel, M., Worner, S., & Kasabov, N. (2009). Integrated feature and parameter optimization for an evolving spiking neural network: Exploring heterogeneous probabilistic models. Neural Networks, 22(5-6), 623-632.

[35] Schraudolph, N. N., & Graepel, T. (2002). Towards stochastic conjugate gradient methods. Proceedings of the IEEE 9th International Conference on Neural Information Processing, pp. 853-856.

[36] Fukuoka, T., Tokunaga, A., Kondo, E., Miki, K., Tachibana, T., & Noguchi, K. (1998). Change in mRNAs for neuropeptides and the GABAA receptor in dorsal root ganglion neurons in a rat experimental neuropathic pain model. Pain, 78(1), 13-26.

[37] Panda, P., & Roy, K. (2016). Unsupervised regenerative learning of hierarchical features in spiking deep networks for object recognition. Proceedings of the IEEE International Joint Conference on Neural Networks, pp. 1–8.

[38] Kheradpisheh, S. R., Ganjtabesh, M., & Masquelier, T. (2016). Bio-inspired unsupervised learning of visual features leads to robust invariant object recognition. Neurocomputing, 205, 382-392.

[39] Tavanaei, A., & Maida, A. (2017, November). Bio-inspired multilayer spiking neural network extracts discriminative features from speech signals. Proceedings of the International Conference on Neural Information Processing, pp. 899-908.

[40] Liu, T., Liu, Z., Lin, F., Jin, Y., Quan, G., & Wen, W. (2017, November). Mt-spike: A multilayer time-based spiking neuromorphic architecture with temporal error backpropagation. Proceedings of the 36th International Conference on Computer-Aided Design, pp. 450-457.

[41] Tavanaei, A., & Maida, A. S. (2017). Bp-stdp: Approximating backpropagation using spike timing dependent plasticity, https://arxiv.org/pdf/1711.04214.pdf.

[42] Saleh, A. Y., Hamed, H. N. B. A., Shamsuddin, S. M., & Ibrahim, A. O. (2017). A new hybrid k-means evolving spiking neural network model based on differential evolution. Proceedings of the International Conference of Reliable Information and Communication Technology, pp. 571-583.