Assembly-based STDP: A New Learning Rule for Spiking Neural Networks Inspired by Biological Assemblies

Assembly-based STDP: A New Learning Rule for Spiking Neural Networks Inspired by Biological Assemblies

Vahid Saranirad
Intelligent Systems Research Centre
Ulster University
Derry, Northern Ireland, UK
saranirad-v@ulster.ac.uk

Shirin Dora
Department of Computer Science
Loughborough University
Loughborough, UK
s.dora@lboro.ac.uk

T.M. McGinnity
Intelligent Systems Research Centre
Ulster University
Derry, Northern Ireland, UK
tm.mcginnity@ulster.ac.uk

Damien Coyle
Intelligent Systems Research Centre
Ulster University
Derry, Northern Ireland, UK
dh.coyle@ulster.ac.uk

Abstract*— Spiking Neural Networks (SNNs), An alternative to sigmoidal neural networks, include time into their operations using discrete signals called spikes. Employing spikes enables SNNs to mimic any feedforward sigmoidal neural network with lower power consumption. Recently a new type of SNN has been introduced for classification problems, known as Degree of Belonging SNN (DoB-SNN). DoB-SNN is a two-layer spiking neural network that shows significant potential as an alternative SNN architecture and learning algorithm. This paper introduces a new variant of Spike-Timing Dependent Plasticity (STDP), which is based on the assembly of neurons and expands the DoB-SNN's training algorithm for multilayer architectures. The new learning rule, known as assembly-based STDP, employs trained DoBs in each layer to train the next layer and build strong connections between neurons from the same assembly while creating inhibitory connections between neurons from different assemblies in two consecutive layers. The performance of the multilayer DoB-SNN is evaluated on five datasets from the UCI machine learning repository. Detailed comparisons on these datasets with other supervised learning algorithms show that the multilayer DoB-SNN can achieve better performance on 4/5 datasets and comparable performance on 5th when compared to multilayer algorithms that employ considerably more trainable parameters.

Keywords—STDP, assembly of neurons, DoB, degree of belonging, spiking neural network, SNN

I. INTRODUCTION

Spiking Neural Networks (SNNs) are an alternative to sigmoidal neural networks, which add the concept of time into their operations. This third class of artificial neural networks consists of neurons that transmit information employing discrete impulses known as spikes. It has been shown that SNNs can mimic any feedforward network of sigmoidal neurons with lower power requirements [1][2]. However, supervised gradient-based training algorithms for sigmoidal neural networks can not directly be employed for SNNs due to the discrete nature of spikes.

One of the most popular training methods for SNNs is a biologically inspired rule known as Spike Timing Dependent Plasticity (STDP). STDP adjusts the synapse strength between two neurons based on the timing of each pair of consecutive presynaptic and postsynaptic spikes [3]. This learning rule consists of two opposite actions: 1) Long-term potentiation (LTP) strengthens the synapse when the postsynaptic neuron generates a spike shortly after a presynaptic spike. 2) Long term depression (LTD) weakens the synapse when a postsynaptic spike happens before a spike from the presynaptic neuron. Real-valued weights usually model synapses in SNNs, and accordingly, LTD and LTP are performed by decreasing and increasing the weights, respectively [4].

Several methods have used STDP to train spiking neural networks for classification problems. These approaches can be categorised into unsupervised, supervised and reinforcement learning. STDP has been used for unsupervised training of single layer SNNs [5][6][7]. Srinivasan et al. [7] integrated the firing frequency of post-synaptic neurons into STDP to improve the discrimination of samples from different classes with similar attributes. STDP is inherently meant for one layer of spiking neurons and can not be applied to multilayer architectures due to the absence of a mechanism to back-propagate errors across layers. However, Kheradpisheh et al. [8] used unsupervised STDP to train a multilayer convolutional SNN in a layer-by-layer manner to tackle this issue. After training each layer, they freeze its synaptic weights and disable its inhibitory strategies/neurons to provide enough spikes (information) to train the next layer. The output of this network's last fully connected layer is fed as a feature vector to a support vector machine classifier.

STDP is fundamentally an unsupervised procedure. However, it is the foundation of some supervised approaches such as SWAT [9], SEFRON [10], and ReSuMe [11] to train one-layer SNNs. Some methods used a combination of unsupervised and supervised STDP to train multilayer SNNs. Lee et al. [12] employed a layer-by-layer approach for a multilayer convolutional SNN. They trained the hidden convolutional layers with a layer-by-layer unsupervised STDP. However, a type of supervised STDP trains the last fully connected layer of this network, in which each neuron is associated with a class. According to this supervised rule, every training spike pattern is only employed to train the neuron allocated to the input pattern's class. Thiele et al. [13][14] used a similar technique with an end-to-end manner. They proposed a new scheme for end-to-end training of a deep convolutional SNN using dual accumulator neurons and STDP. In this biologically implausible method, each spiking neuron has two integrators with different thresholds; the one with the higher threshold trains the synapses with STDP, while the other integrator provides enough spikes for training in the subsequent layers.

Conventional STDP is based on the relation between the spike-timing of neurons on two sides of a synaptic weight. This weight updating process does not guarantee high

performance on a machine learning task. According to neuroscientific research [15], synapse adjustment in the human brain is also influenced by a third factor known as neuromodulation. Biological neuromodulators are released when a task is completed successfully (reward) or when an experience is unexpected (novelty). Neuromodulation inspired a more powerful method for training SNNs known as Neo-Hebbian STDP. Three-factor STDP states that if the pre-synaptic neuron fires before or shortly after the post-synaptic neuron, a flag (eligibility trace) is set on the corresponding synapse, signalling that it is eligible for changes. However, the modification is only performed if a neuromodulator (third factor) arrives simultaneously or shortly after the flag [16]. The Neo-Hebbian rule has been utilised in several studies to train SNNs [17][18][19]. Mozafari et al. [17] introduced a variant of three-factor STDP called reward-modulated STDP to train a convolutional SNN, which can be considered a reinforcement learning algorithm. The final layer of their network contains one neuron that has been pre-assigned to each of the classes. The neuron with the earliest spike determines the network's decision in response to the input samples. Comparing this decision with the input pattern's class, a reward/punishment signal is sent to the convolutional layers as a neuromodulator for STDP. A reward signal updates the weights with normal STDP, but the punishment neuromodulator adjusts the weights using a learning rule known as anti-STDP in which LTP and LTD are swapped.

Neuroscientific researches suggest that the human brain allocates a neuronal assembly to each class in a real-life object classification task [15][20][21]. An assembly of neurons is a group of neurons assigned to a certain piece of cognitive data, such as a memory, a concept, or a phrase [20]. Recently Saranirad et al. [22] introduced a new type of spiking neural network inspired by biological assemblies, known as DoB-SNN. All the above SNNs allocate each output neuron to a class before the training process. However, instead of a crisp pre-training allocation, DoB-SNN associates Degree of Belongings (DoBs) to each neuron, which estimates neurons' spiking frequencies in response to the input patterns from each class. The training algorithm of DoB-SNN estimates the DoBs while adjusting the synaptic weights and forms an assembly of neurons for each class. These assemblies are trained to have relatively higher spiking activity for the input patterns from their class and lower activity for other classes. DoBs enable DoB-SNN to employ supervised STDP and anti-STDP to train the synaptic weights. DoB-SNN showed better or comparable performance than some state-of-the-art SNNs while using considerably fewer network parameters. However, this method is used to train only one layer of spiking neurons.

In this paper, a new variant of STDP, entitled assembly-based STDP, is introduced to extend the DoB-SNN algorithm to multilayer architectures with supervised training of all layers. Assembly-based STDP only performs STDP for the neurons from the same assembly and builds strong connections between the assemblies from the same class in different layers. Performance evaluation on multiple classification datasets indicates that adding another layer to DoB-SNN using assembly-based STDP improves the performance of the network. Furthermore, comparative results reveal that three-layer DoB-SNN can match or outperform other multilayer algorithms which exhibit significantly more trainable parameters.

The remainder of the paper is organised as follows. The training algorithm and network architecture of a three-layer DoB SNN are first described in Section II. Next, in Section III, the performance of multilayer DoB-SNN is evaluated on multiple datasets and compared with some of the supervised SNNs. Finally, Section IV is allocated to discussion and conclusions.

II. Spiking Neural Network with Degree of Belonging

Figure (1) represents the three-layer architecture of DoB-SNN. The network consists of an input layer for spike patterns X_n from the class $c_n \in [1, \dots, N_c]$ where N_c is the number of classes in the input dataset. The input layer is followed by a hidden and an output layer of fully connected LIF neurons. Neurons in the hidden and output layer are equipped with values in the range [0,1], known as Degree of Belonging (DoB). DoB of j^{th} neuron $\left(\eth_k^j\right)$ is an estimation of the average spiking frequency of neuron j for the training inputs from class k. DoBs of neuron j satisfy the following condition.

$$\sum_{k=1}^{N_c} \eth_k^j = 1 \tag{1}$$

which indicates that sum of all DoBs of each neuron for all classes is equal to 1. In this paper, an assembly of each class is defined as a group of neurons which are associated with that class based on their DoBs. Neurons are in the assembly of the class for which they have the highest DoB among all classes. Also, neurons that never were the most active for any of the classes and therefore have equal values of DoB for all classes are termed unassigned, meaning that they are not a member of any class's assembly.

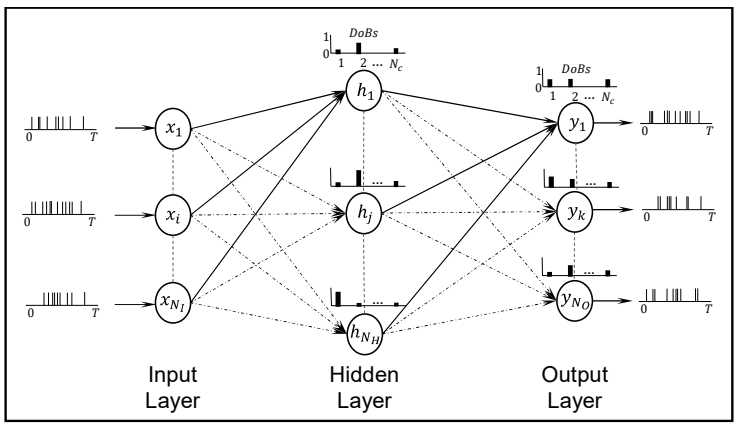

Figure 1. The architecture of three-layer DoB-SNN

A. Training Algorithm

Synapses between the input and hidden layer connect input neurons to spiking neurons, while weights between the hidden and output layer have presynaptic neurons with DoBs. Accordingly, the learning rules for training these two layers are different. The hidden layer is trained by conventional STDP; however, a new variant of STDP called assembly-based STDP is introduced for the output layer. The following two sections describe the training algorithms for the hidden and output layers.

1) Hidden Layer

Synaptic weights in the hidden layer are initially set to random values from the interval [-1,1], and DoBs are initialised equally as given

$$\eth_k^j = \frac{1}{N_c} \quad \forall j \in [1, \dots, N_H], k \in [1, \dots, N_c] \tag{2}$$

after presenting an input pattern X_n from the class c_n. The spiking frequency f_j of j^{th} neuron in the hidden layer in response to X_n is computed as the total number of spikes during simulation time. Based on the spiking frequency, the normalised spiking frequency of neuron j in this layer is given by

$$g_j = \frac{f_j}{\sum_{m=1}^{N_H} f_m} \tag{3}$$

The learning algorithm for the hidden layer identifies three neurons for learning as follows:

$\boldsymbol{J_1}$**:** Neuron with the highest spiking frequency among all neurons in this layer, given by:

$$J_1 = \text{argmax}_j \left(g_j\right) \quad \forall j \in [1, \dots, N_H] \tag{4}$$

$\boldsymbol{J_2}$**:** Neuron with the highest spiking frequency in the assembly of class c_n, given by:

$$J_2 = \text{argmax}_j \left(g_j\right) \;\; \forall j |\, c_n = \text{argmax}_k(\eth_k^j) \tag{5}$$

$\boldsymbol{J_3}$ **:** Neuron with the highest spiking frequency among unassigned neurons, given by:

$$J_3 = \text{argmax}_j \left(g_j\right) \;\; \forall j |\, \eth_k^j = \frac{1}{N_c} \;\; \forall k \in [1, \dots, N_c] \tag{6}$$

Following each training pattern, only DoBs of neuron J_1 are updated as follows.

$$\Delta\eth_k^{J_1} = \begin{cases} \frac{1-\eth_{c_n}^{J_1}}{1+\exp(-g_{J_1})} & k = c_n \\ -\frac{1-\Delta\eth_{c_n}^{J_1}}{N_c-1} & k \neq c_n \end{cases} \tag{7}$$

Updating DoBs of neuron J_1 with equation (7) increases its belonging to the assembly of class c_n, while decreasing the DoBs for other classes in a way that satisfies the condition in equation (1).

Without loss of generality, we assume that the network has been trained using several training patterns. In this situation, neurons based on their DoBs are either unassigned or associated with one of the class assemblies. After presenting a new sample X_n from the class c_n, two different scenarios may happen. The following sections explain how the synaptic weights will be updated in each situation.

Scenario A: In this Scenario, Neuron J_1 is unassigned or associated with the class c_n. If J_1 is unassigned, its DoBs will be updated by equation (7), and therefore, it will join the assembly of class c_n, and if J_1 is already associated with the class c_n, its association is strengthened using equation (7). In both cases, the most active neuron for a training pattern from c_n is associated with c_n, and accordingly, it is the desired scenario. In this case, the synaptic weights connecting J_1 to the input layer are updated using STDP, provided that f_{J_1} is less than α_h. The amount of change in the synaptic weight (Δw_{ij}) connecting presynaptic neuron i to postsynaptic neuron j using exponential STDP is given by

$$\Delta w_{ij} = \begin{cases} +e^{\frac{-|t_i - t_j|}{\tau}} & t_i \geq t_j \\ -e^{\frac{-|t_i - t_j|}{\tau}} & t_i < t_j \end{cases} \tag{8}$$

where t_i and t_j depict the firing time of the neurons i and j respectively, and τ is the time constant for STDP. This modification in weights will increase the activity of J_1 for the spike patterns from c_n, which is compatible with this neuron's DoBs. α_h is a fixed threshold as the maximum allowed frequency for a neuron to be updated by STDP update and prohibits J_1 from dominating the training process. Section III explains how this parameter is determined.

Scenario B: In this scenario, J_1 is associated with a class other than c_n and therefore it is an undesired situation. Since J_1 is the most active neuron for an input pattern from class c_n, which is incompatible with its highest DoB, this neuron's activity for c_n is reduced using anti-STDP. According to anti-STDP, when the presynaptic neuron generates a spike just before the postsynaptic spike, the weight of the synapse is reduced. Conversely, the weight is increased if a presynaptic spike happens quickly after the presynaptic neuron fires. Anti-STDP can be formulated as

$$\Delta w_{ij} = \begin{cases} -e^{\frac{-|t_i - t_j|}{\tau}} & t_i \geq t_j \\ +e^{\frac{-|t_i - t_j|}{\tau}} & t_i < t_j \end{cases} \tag{9}$$

In addition, neuron J_2, the most active neuron in the assembly of c_n, fired with a lower frequency than J_1. Accordingly, the synaptic weights of J_2 are updated using STDP (equation (8)) to increase its activity for the input patterns from c_n, provided that f_{J_2} is more than β_h. The fixed frequency threshold β_h prevents the training algorithm from updating the weights of neurons with low frequencies using STDP. Therefore β_h protects the network from losing previously stored data. Furthermore, if no neuron is associated with the assembly of c_n, the unassigned neuron J_3 gets an STDP update. This weight modification for J_3 increases its activity for the input patterns from c_n and increases the chance of this neuron of joining the assembly of c_n, while presenting the subsequent training patterns to the network.

After training the hidden layer, all unassigned neurons are removed from the network. These neurons are untrained and might undermine the network's overall performance. Then, all the weights and DoBs in this layer are frozen before training

the output layer. The training algorithm for this layer is described in more detail in [22].

2) Output Layer

The scenarios considered by the training algorithm for updating weights and DoBs in the output layer are the same as those described for updating the hidden layer but weights are updated using a different STDP rule. Since the training of the output layer starts after the hidden layer, neurons in the hidden layer have already been trained to spike compatible with their DoBs. The training algorithm for the hidden layer is designed to form assembly of neurons for each of the classes in the dataset. The training forces neurons to have relatively higher spiking activity for the patterns from their class and spike with lower frequencies for other classes. To expand these assemblies to the output layer, we use the estimated DoBs in the hidden layer for training the output layer. For this purpose, a new variant of STDP is introduced in this paper, entitled as Assembly-based STDP.

As equation (8) implies, STDP updates the weights connected to all the presynaptic neurons only based on the times of the presynaptic and postsynaptic spikes. In Assembly-Based STDP, only presynaptic neurons from the assembly of input pattern's class are updated using STDP. When an input pattern X_n from the class c_n is presented to the network, the weights corresponding to the presynaptic neurons are updated as follows.

$$\Delta w_{ij} = \begin{cases} +e^{\frac{-|t_i - t_j|}{\tau}} & t_j \geq t_i,\ c_n = \text{argmax}_k(\eth_k^i) \\ -e^{\frac{-|t_i - t_j|}{\tau}} & t_j < t_i,\ c_n = \text{argmax}_k(\eth_k^i) \\ 0 & c_n \neq \text{argmax}_k(\eth_k^i) \end{cases} \quad (10)$$

Assembly-based STDP controls the plasticity and lets STDP only occur between the neurons from the same class. This learning rule builds strong connections between assemblies from the same class. Additionally, the anti-STDP form of assembly-based STDP is given by

$$\Delta w_{ij} = \begin{cases} -e^{\frac{-|t_i - t_j|}{\tau}} & t_j \geq t_i,\ c_n = \text{argmax}_k(\eth_k^i) \\ +e^{\frac{-|t_i - t_j|}{\tau}} & t_j < t_i,\ c_n = \text{argmax}_k(\eth_k^i) \\ 0 & c_n \neq \text{argmax}_k(\eth_k^i) \end{cases} \quad (11)$$

As explained in the previous section, anti-STDP is employed when the neuron with the highest spiking frequency (J_1) in the output layer is associated to a class other than c_n. In this case, the anti-STDP rule in equation (11) reduces the weights between neuron J_1 in the output layer and the assembly of c_n in the hidden layer. This type of anti-STDP leads to weak or negative connections between the assemblies from two different classes in the hidden and output layer.

As in the hidden layer, this layer considers two frequency thresholds α_o and β_o with different values from the thresholds in the hidden layer for updating weights. After training of the output layer all unassigned neurons are removed from this layer. Now, in the testing stage, when a sample X_n from the class c_n is presented to the network, the predicted class $\hat{c}_n$ depends on the DoBs of the most active neuron (J_1) in the output layer as

$$\hat{c}_n = \text{argmax}_k(\eth_k^{J_1}) \quad (12)$$

III. Performance Evaluation

The proposed three-layer DoB-SNN with assembly based-STDP is applied on five numerical datasets from the UCI machine learning repository [23], including breast cancer, Iris, Ionosphere, Pima diabetes, and liver disorders. The training algorithm for three-layer DoB-SNN includes four hyperparameters α_h, β_h, α_o, and β_o. Nested Cross-Validation(nCV) [24] is employed to optimise these frequency thresholds. Figure (2) represents an example of a 5×5 nCV diagram used in this paper, which consists of two nested loops. The outer loop is a typical CV to evaluate model performance. The inner loop splits the outer training data into several folds and computes the network's accuracy for every hyperparameter range. The best set of parameters is the one with maximum mean accuracy across all inner folds. Finally, the network is trained with outer training data using the best parameters resulting from the inner loop and is tested on the outer test data, producing the final CV results for the model. Table I describes the details of datasets and the size of nCV used for each dataset.

TABLE I. Description of Datasets and the Size of nCV Used for Each Dataset

Data Set	# Classes	# Samples	# Features	# Outer Loop Folds	# Inner Loop Folds
Iris	3	150	4	5	5
Breast Cancer	2	683	9	10	10
Pima diabetes	2	768	9	10	10
Ionosphere	2	351	34	5	5
Liver disorders	2	345	6	5	5

Three-layer DoB-SNN is compared with two-layer DoB-SNN as well as three other SNNs, including SWAT [9], SRESN [25], and SpikeProp [26] (Table II). The results of two-layer DoB-SNN, SWAT, SRESN, and SpikeProp have been reproduced from [22]. Also, the simulation parameters of three-layer DoB-SNN have been set as in [22]. Three-layer DoB-SNN was implemented in Python 3.7 utilising computing resources provided by the Northern Ireland High-Performance Computing (NI-HPC) facility.

The metrics used for comparison are the number of network parameters and the classification accuracy for the test dataset. The number of trainable parameters is equal to $(N_I \times N_O)$ and $(N_I \times N_H + N_H \times N_O)$ for two- and three-layer architectures, respectively. SpikeProp uses 16 synapses between every pre- and postsynaptic neurons, and therefore its number of trainable parameters is $16 \times (N_I \times N_H + N_H \times N_O)$. It should be noted that DoB-SNN uses rate coding while other methods employ population coding, and as a result, they have more input neurons. The classification accuracy is given by

$$Accuracy = \frac{\text{Number of correctly classified samples}}{\text{Total number of samples}} \quad (17)$$

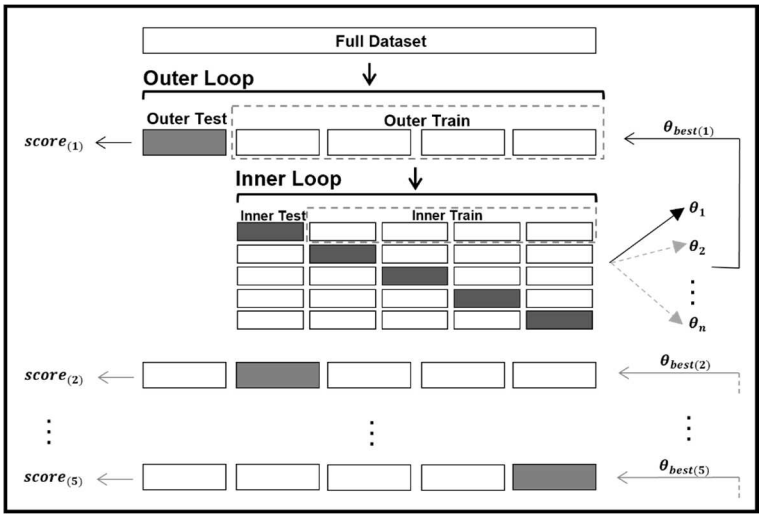

Figure 2. An example diagram of 5×5 nested cross-validation used in this paper: θ represents a hyperparameter vector

Results are reported as average and standard deviation of accuracies across all outer folds. Additionally, one-way ANOVA [27] is used to determine whether the classification accuracy of different training methods were significantly different. After proving statistical significance in each case, pairwise comparisons are performed using Fisher's Least Significant Difference (LSD) approach [28].

For the Iris dataset, three-layer DoB-SNN slightly improved the performance of two-layer DoB-SNN and outperformed all other methods. One-way ANOVA indicates that at least one approach performs significantly different from others ($\rho < 1 \times 10^{-4}$). Fisher's pairwise analysis shows that the three-layer DoB-SNN has significantly better performance than SpikeProp ($\rho < 0.05$) and SWAT ($\rho < 5 \times 10^{-5}$). Additionally, the two DoB-SNN architectures used for the Iris dataset employ fewer training parameters than others.

Regarding breast cancer, adding another layer to DoB-SNN improved its accuracy by 0.6%. The three-layer DoB-SNN has reached the best accuracy while using considerably fewer trainable parameters. ANOVA showed a significant difference among the methods ($\rho < 5 \times 10^{-4}$). Pairwise comparison indicated that three-layer DoB-SNN only outperforms SWAT significantly ($\rho < 5 \times 10^{-5}$).

For the Pima diabetes dataset, SpikeProp performs better than others, and one-way ANOVA analysis indicates that not all the approaches have equal performance ($\rho < 5 \times 10^{-17}$). However, pairwise statistical analysis confirmed that SpikeProp did not significantly outperform two-layer ($\rho > 0.1$) and three-layer DoB-SNN ($\rho > 0.2$).

TABLE II
COMPARISON OF DOB-SNN WITH SRESN, SPIKEPROP, AND SWAT ON FIVE UCI BENCHMARK CLASSIFICATION DATASETS

Data Set	Algorithm	Architecture	# Trainable Parameters	Testing Accuracy (%)
Iris	**3L-DoBSNN**	**5-(5-8)-(8-10)**	**65-120**	**97.79(0.62)**
	2L-DoBSNN	5-(5-8)	25-40	97.75(0.92)
	SWAT	24-312-3	936	93.88(1.80)
	SRESN	24-(5-11)	120-264	97.01(0.73)
	SpikeProp	25-10-3	4480	96.13(0.83)
Breast Cancer	**3L-DoBSNN**	**10-(6-9)-(6-7)**	**96-153**	**97.95(1.48)**
	2L-DoB-SNN	10-(6-9)	(60-90)	97.35(1.66)
	SWAT	54-702-2	1404	95.66 (0.08)
	SRESN	54-(9-13)	486-702	97.10(0.20)
	SpikeProp	55-15-2	13680	97.04 (0.53)
Pima diabetes	3L-DoBSNN	10-(16-19)-10	320-380	76.63(1.07)
	2L-DoBSNN	10-(16-19)	160-190	76.57(1.17)
	SWAT	54-702-2	1404	72.11(1.38)
	SRESN	54-(10-13)	540-702	70.06 (1.82)
	SpikeProp	**55-20-2**	**16640**	**77.38(1.03)**
Ionosphere	3L-DoBSNN	**35-(17-19)-(8-11)**	**731-874**	**91.10(1.22)**
	2L-DoBSNN	35-(17-19)	595-665	89.78(1.26)
	SWAT	204-2652-2	5304	90.04(1.87)
	SRESN	204-(15-21)	3060-4284	88.52(1.07)
	SpikeProp	205-25-2	82800	86.89(2.00)
Liver disorders	**3L-DoBSNN**	**7-(18-20)-(9-10)**	**288-340**	**74.20(2.50)**
	2L-DoBSNN	7-(18-20)	126-140	70.33(4.65)
	SWAT	36-468-2	936	60.43(2.72)
	SRESN	36-(7-10)	252-360	60.17(1.78)
	SpikeProp	37-15-2	9360	64.23(3.92)

Considering Ionosphere, three-layer DoB-SNN outperforms all the methods. One-way ANOVA represents that at least one SNN has performed significantly different from others ($\rho < 0.005$). LSD indicates that three-layer DoB-SNN significantly outperformed SRESN ($\rho < 0.05$) and SpikeProp ($\rho < 5 \times 10^{-4}$) while using considerably fewer network parameters.

Finally, the comparative results for Liver disorders demonstrate that adding another layer to DoB-SNN using assembly-based STDP increases its testing accuracy by 3.87% for this dataset. One-way ANOVA indicates that at least one method performs significantly different from others ($\rho < 5 \times 10^{-6}$). The LSD pairwise analysis shows that three-layer DoB-SNN significantly outperforms SWAT ($\rho < 5 \times 10^{-6}$), SRESN ($\rho < 5 \times 10^{-6}$), and SpikeProp ($\rho < 5 \times 10^{-4}$).

IV. Discussion and Conclusion

In this paper, a new variant of STDP, known as assembly-based STDP, has been proposed to extend the training algorithm to multilayer architectures. The new learning rule lets STDP happen only between the neurons from the same class and builds strong connections between the assemblies from the same class in consecutive layers. Assembly-based STDP has been employed to train a three-layer DoB-SNN. Performance evaluation on five UCI datasets indicated that adding another layer to DoB-SNN has improved its performance. The new three-layer network outperformed or achieved comparable accuracies with other approaches in the comparison while having considerably fewer network parameters.

Both DoBs and truth values in the fuzzy logic range between 0 and 1 and use values of partial truth. These two concepts may seem similar at first, but fuzzy logic assigns a value to a set based on the membership functions, while DoBs are adjusted incrementally depending on their spiking activity in response to the samples from different classes without any membership functions or fuzzy rules. There are some methods which combine the concept of fuzzy logic with spiking neural networks [29][30][31][32][33]. These approaches employ the idea of fuzzy logic to create frequency-based receptive fields for neurons [29][30], encode the input data into spike patterns [31][32], or cluster the input patterns [33]. However, none of those above algorithms used fuzzy logic to categorise neurons into different class assemblies, and therefore they are not similar to DoB-SNN.

Assembly-based STDP has been employed to add one layer to the DoB-SNN. Using this new variant of STDP, more layers can be added to the network and build multilayer fully connected or convolutional architectures. Future work will involve implementing multilayer DoB-SNN for more challenging datasets. Additionally, the proposed training algorithm for DoB-SNN is layerwise. In a future work, we will be introducing an end-to-end training algorithm for DoB-SNN.

Acknowledgment

This project was supported by Dr George Moore PhD scholarship in intelligent data analytics. We are grateful for access to the Tier 2 High Performance Computing resources provided by the Northern Ireland High Performance Computing (NI-HPC) facility funded by the UK Engineering and Physical Sciences Research Council (EPSRC), Grant No. EP/T022175. Damien Coyle is grateful for funding UKRI Turing AI Fellowship 2021-2025 funded by the EPSRC (grant number EP/V025724/1).

References

[1] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural Networks*, vol. 111. Elsevier Ltd, pp. 47–63, 01-Mar-2019.

[2] W. Maass, "Fast sigmoidal networks via spiking neurons," *Neural Comput.*, vol. 9, no. 2, pp. 279–304, Feb. 1997.

[3] H. Markram, "A history of spike-timing-dependent plasticity," *Front. Synaptic Neurosci.*, vol. 3, no. AUG, pp. 1–24, 2011.

[4] Y. Dan and M.-M. Poo, "Spike Timing-Dependent Plasticity: From Synapse to Perception," *Physiol. Rev.*, vol. 86, no. 3, pp. 1033–1048, Jul. 2006.

[5] D. Querlioz, O. Bichler, P. Dollfus, and C. Gamrat, "Immunity to device variations in a spiking neural network with memristive nanodevices," *IEEE Trans. Nanotechnol.*, vol. 12, no. 3, pp. 288–295, 2013.

[6] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Front. Comput. Neurosci.*, vol. 9, no. AUGUST, pp. 1–9, 2015.

[7] G. Srinivasan, S. Roy, V. Raghunathan, and K. Roy, "Spike timing dependent plasticity based enhanced self-learning for efficient pattern recognition in spiking neural networks," *Proc. Int. Jt. Conf. Neural Networks*, vol. 2017-May, pp. 1847–1854, 2017.

[8] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "STDP-based spiking deep convolutional neural networks for object recognition," *Neural Networks*, 2018.

[9] J. J. Wade, L. J. McDaid, J. A. Santos, and H. M. Sayers, "SWAT: A Spiking Neural Network Training Algorithm for Classification Problems," *IEEE Trans. Neural Networks*, vol. 21, no. 11, pp. 1817–1830, Nov. 2010.

[10] A. Jeyasothy, S. Sundaram, and N. Sundararajan, "SEFRON: A New Spiking Neuron Model With Time-Varying Synaptic Efficacy Function for Pattern Classification," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 30, no. 4, pp. 1231–1240, Apr. 2019.

[11] F. Ponulak and A. Kasiński, "Supervised Learning in Spiking Neural Networks with ReSuMe: Sequence Learning, Classification, and Spike Shifting," *Neural Comput.*, vol. 22, no. 2, pp. 467–510, Feb. 2010.

[12] C. Lee, G. Srinivasan, P. Panda, and K. Roy, "Deep Spiking Convolutional Neural Network Trained With Unsupervised Spike-Timing-Dependent Plasticity," *IEEE Trans. Cogn. Dev. Syst.*, vol. 11, no. 3, pp. 384–394, Sep. 2019.

[13] J. C. Thiele, O. Bichler, and A. Dupret, "A Timescale Invariant STDP-Based Spiking Deep Network for Unsupervised Online Feature Extraction from Event-Based Sensor Data," in *2018 International Joint Conference on Neural Networks (IJCNN)*, 2018, vol. 2018-July, pp. 1–8.

[14] J. C. Thiele, O. Bichler, and A. Dupret, "Event-Based, Timescale Invariant Unsupervised Online Deep Learning With STDP," *Front. Comput. Neurosci.*, vol. 12, no. June, pp. 1–13, Jun. 2018.

[15] N. Frémaux and W. Gerstner, "Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules," *Front. Neural Circuits*, vol. 9, no. JAN2016, 2015.

[16] W. Gerstner, M. Lehmann, V. Liakoni, D. Corneil, and J. Brea, "Eligibility Traces and Plasticity on Behavioral Time Scales:

Experimental Support of NeoHebbian Three-Factor Learning Rules," *Front. Neural Circuits*, vol. 12, no. July, pp. 1–16, Jul. 2018.

[17] M. Mozafari, S. R. Kheradpisheh, T. Masquelier, A. Nowzari-Dalini, and M. Ganjtabesh, "First-spike-based visual categorization using reward-modulated STDP," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 29, no. 12, pp. 6178–6190, Dec. 2018.

[18] Z. Bing, Z. Jiang, L. Cheng, C. Cai, K. Huang, and A. Knoll, "End to end learning of a multi-layered snn based on r-stdp for a target tracking snake-like robot," *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2019-May, pp. 9645–9651, 2019.

[19] Z. Bing, I. Baumann, Z. Jiang, K. Huang, C. Cai, and A. Knoll, "Supervised learning in SNN via reward-modulated spike-timing-dependent plasticity for a target reaching vehicle," *Front. Neurorobot.*, vol. 13, no. May, pp. 1–17, 2019.

[20] G. L. Gerstein, P. Bedenbaugh, and A. M. H. J. Aertsen, "Neuronal assemblies," *IEEE Trans. Biomed. Eng.*, vol. 36, no. 1, pp. 4–14, 1989.

[21] C. H. Papadimitriou, S. S. Vempala, D. Mitropolsky, M. Collins, and W. Maass, "Brain computation by assemblies of neurons," *Proc. Natl. Acad. Sci.*, vol. 117, no. 25, pp. 14464–14472, Jun. 2020.

[22] V. Saranirad, T. M. McGinnity, S. Dora, and D. Coyle, "DoB-SNN: A New Neuron Assembly-Inspired Spiking Neural Network for Pattern Classification," *Proc. Int. Jt. Conf. Neural Networks*, vol. 2021-July, 2021.

[23] D. Dua and C. Graff, "UCI machine learning repository," 2017.

[24] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," *Int. Jt. Conf. Artif. Intell.*, 1995.

[25] S. Dora, K. Subramanian, S. Suresh, and N. Sundararajan, "Development of a Self-Regulating Evolving Spiking Neural Network for classification problem," *Neurocomputing*, vol. 171, pp. 1216–1229, Jan. 2016.

[26] S. M. Bohte, J. N. Kok, and H. La Poutre, "SpikeProp: Backpropagation for networks of spiking neurons," *Neurocomputing*, vol. 48, no. 1–4, p. 17, 2002.

[27] R. A. Fisher, "On the'probable error'of a coefficient of correlation deduced from a small sample," *Metron*, vol. 1, pp. 1–32, 1921.

[28] R. A. Fisher, "The design of experiments," 1949.

[29] C. Glackin, L. McDaid, L. Maguire, and H. Sayers, "Implementing Fuzzy Reasoning on a Spiking Neural Network," in *Artificial Neural Networks - ICANN 2008*, vol. 5164 LNCS, no. PART 2, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 258–267.

[30] C. Glackin, L. Maguire, L. McDaid, and H. Sayers, "Receptive field optimisation and supervision of a fuzzy spiking neural network," *Neural Networks*, vol. 24, no. 3, pp. 247–256, Apr. 2011.

[31] M. Bavandpour, S. Bagheri-Shouraki, H. Soleimani, A. Ahmadi, and B. Linares-Barranco, "Spiking neuro-fuzzy clustering system and its memristor crossbar based implementation," *Microelectronics J.*, vol. 45, no. 11, pp. 1450–1462, Nov. 2014.

[32] D. Tang, T. Y. Tang, J. Botzheim, N. Kubota, and T. Yamaguchi, "Fuzzy Spiking Neural Network for Abnormality Detection in Cognitive Robot Life Supporting System," in *2015 IEEE Symposium Series on Computational Intelligence*, 2015, pp. 130–137.

[33] Y. Bodyanskiy and A. Dolotov, "Image processing using self-learning fuzzy spiking neural network in the presence of overlapping classes," in *2008 11th International Biennial Baltic Electronics Conference*, 2008, pp. 213–216.