Asian Journal of Research in Computer Science

9(4): 35-49, 2021; Article no.AJRCOS.69910
ISSN: 2581-8260

A Survey of Supervised Learning Models for Spiking Neural Network

Moses Apambila Agebure[1*], Paula Aninyie Wumnaya[2] and Edward Yellakuor Baagyere[1]

[1] *Department of Computer Science, School of Computing and Information Sciences, C. K. Tedam University of Technology and Applied Sciences, Ghana.*
[2] *Department of Electronics and Computer Hardware Technology, School of Computing and Information Sciences, C. K. Tedam University of Technology and Applied Sciences, Ghana.*

Authors' contributions

This work was carried out in collaboration among all authors. All authors designed the study, performed initial analysis and wrote the protocol. Author MAA wrote the first draft of the manuscript. Authors PAW and EYB reviewed and finalised the draft. All authors read and approved the final manuscript.

Article Information

DOI: 10.9734/AJRCOS/2021/v9i430228
Editor(s):
(1) Dr. Dariusz Jacek Jakóbczak, Koszalin University of Technology, Poland.
Reviewers:
(1) Reem Ibrahim Hasan, University of Information Technology and Communication, Iraq.
(2) Baltra Agusti Pramajuri, Indonesia.
(3) Anuj Kumar Dwivedi, Govt VBSD Girls PG College, India.
Complete Peer review History: http://www.sdiarticle4.com/review-history/69910

Received 10 April 2021
Accepted 14 June 2021
Published 17 June 2021

Review Article

ABSTRACT

There has been a significant attempt to derive supervised learning models for training Spiking Neural Networks (SNN), which is the third and most recent generation of Artificial Neural Network (ANN). Supervised SNN learning models are considered more biologically plausible and thus exploits better the computational efficiency of biological neurons and also, are less computationally expensive than second generation ANN. SNN models have also produced competitive performance in most tasks when compared to second generation ANNs. These advantages, coupled with the difficulty in adopting the well established learning models for second generation networks to train SNN due to the difference in information coding led to the recent introduction of supervised learning models for training SNN.

**Corresponding author: E-mail: magebure@cktutas.edu.gh;*

However, lack of comprehensive source of literature detailing strides made in this area, and the challenges and prospects of SNN serves as a hindrance to further exploration and application of SNN models. A comprehensive review of supervised learning methods in SNN is presented in this paper in which some widely used SNN neural models, learning models and their basic concepts, areas of applications, limitations, prospects and future research directions are discussed. The main contribution of this paper is that it presents and discusses trends in supervised learning in SNN with the aim of providing a reference point for those desiring further knowledge and application of SNN methods.

Keywords: Spiking neural network; supervised learning; classification; spike sequence learning; artificial neural network.

2010 Mathematics Subject Classification: 53C25; 83C05; 57N16.

1 INTRODUCTION

Artificial Neural Network (ANN) since its introduction in the 1940s [1], evolved over the years and has since become one of the most widely used class of machine learning algorithms. ANN models are biologically inspired computational models that seek to mimic the biological learning and information transfer processes in the human or more generally, the mammalian brain and are used for solving complex computational tasks that are difficult or impossible to be solved manually by humans [2–4]. Though the mathematical formulations used to model the behaviour of neurons in ANN vary, they all sought to mimic the learning principles and information transfer processes of biological neurons.

Contrary to modelling of information using continuous or binary values and activation function or Boolean gates in the first and second generation ANN [1], in Spiking Neural Networks, information is encoded using the precise timing of spikes and learning modelled around the time difference between spikes arriving at a neuron, which is identical to the mode of learning and information transfer in biological neurons [3]. Neural networks based on temporal coding are reported to be computationally less expensive which makes them suitable for implementation on hardware and also, enable efficient information transfer and processing than in rate-coded networks [5, 6].

Attempts have been made to leverage these advantages of SNN to solve supervised learning problems. The first supervised learning algorithm derived based on the timing of spikes was the SpikeProp [7], which employed the idea of Error Back propagation and gradient descent used in second generation ANN. Several modifications of the SpikeProp were later introduced as improvements [8, 9]. SNN training algorithms based on evolutionary algorithms have also been investigated [10, 11]. Evolving techniques similar to those used in second generation networks based on the evolving connectionist system [12] have akso been explored. The fourth and important category of learning algorithms for SNN are those that rely on the Herbian and anti-Herbian plasticity, which are considered the main mechanisms governing learning in biological neurons [13, 14]. Thus, learning algorithms based on this approach are considered the most biologically plausible among all categories of supervised SNN learning models. Most of these learning algorithms have been successfully applied to different benchmark and real-world classification tasks with performance comparable to some second generation networks and other algorithms such as those proposed in [13, 15, 16], were tested on spike sequence learning; a task that rate-coded networks can not perform.

Though supervised learning in SNN is an emerging field with great prospects, literature in this area is sparsed and scanty. This suggests that the need for research into supervised learning in SNN to validate and apply existing learning algorithms to emerging areas and also derive improved algorithms can not be overemphasised. Researchers as well as industry players desiring to explore this area would require a comprehensive presentation on existing learning algorithms, their prospects and challenges, main areas of application as well as pointers to future research directions. A comprehensive review of state-of-the-art supervised SNN learning algorithms is therefore presented in this paper.

This review is focused on algorithms derived for networks with upto two (2) layers. The papers are selected to reflect the key learning concepts that have been extensively explored since the inception of supervised learning in SNN. This design allowed us to present the concepts and key modifications and extensions made to improve their performance and adopt them to different learning tasks.

The following contributions are made in this paper:

- A review of state-of-the-art supervised SNN learning algorithms is made and categorised according to the main learning principles employed in each.
- The challenges and prospects of supervised learning in SNN is explored and discussed.
- Open Issues that require further exploration are also outlined.

The remaining sections of the paper is organised as follows: A description of some basic learning concepts employed in SNN and the most widely used SNN neural models are presented in Section II. This is followed by a review of selected supervised SNN learning algorithms in Section III. A dsicussion of relevant obsevrations from the reviewed literarure and pointers to future research is dedtailed in Section IV. The conclusion of the paper is presented in Section V.

2 CONCEPT OF LEARNING IN SNN AND NEURAL MODELS

2.1 Concept of Learning in SNN

The main task in ANN learning is to fit the weights of synapses in a network given a set of training data with corresponding target labels (in supervised learning) or without target labels (in unsupervised learning), such that the network can approximate the expected output at the post-synaptic neuron when a new set of input data is presented to it. The weights are fitted using mathematical functions that minimise the difference between the actual output of the network and the target output. One key focus of research in this field is geared towards coming up with efficient learning models that adopt the same learning processes that occur in biological neurons. As a result, ANN has evolved rapidly over the years towards biologically plausible learning and is currently in its third generation.

The first generation of ANN was introduced in 1943, by McCulloch and Pitts [1]. In this generation, threshold gates also referred to as McCulloch-Pitts neurons or perceptrons were used as computational units. Other implementations in this generation include threshold circuits (multi-layer perceptrons), Boltzmann Machines [17], and Hopfield Nets [18]. Networks based on threshold gates are considered computationally robust with precise mathematical definition, and are generic for computing complex functions in small networks with binary input and output [19]. The computational neuron in this network applies a threshold function on the weighted sum of the input to generate an output, which is either 0 or 1. Though, networks in this generation served a critical purpose, their main limitation is that they could only produce binary output in response to inputs.

The second generation of ANN also referred to as rate-based networks vary from the first basically in two folds; first, they accept real-value input and generate real-value output, and secondly output are generated using activation functions. Some commonly used activation functions are the sigmoid functions defined as $f(x) = \left(\frac{1}{1+\exp^{-\theta^T x}}\right)$, and neurons based on radial basis functions. Sigmoidal networks can either be feed-forward or recurrent. Computation in these networks is in two phases; a sum of the weighted input values is calculated and an appropriate activation function is then applied to the weighted sum to generate the output. These networks are universal approximators for analog data and can also compute any arbitrary boolean function by employing thresholding at the output neuron, and in certain cases with fewer gates [19].

Evidence based on experiments conducted in the field of neuroscience points to the fact that biological neurons encode information via the precise timing of spikes and not necessarily only through the firing rates of neurons as in rate-coded networks [3, 5, 20–22]. Research in the third and most recent generation of ANN is focused on how to model computational neurons to simulate this timing (temporal coding) property of biological neurons. As explained earlier, a neural network may contain up to hundreds of millions of neurons connected to each other via synapses. These neurons emit short chemical or electrical signals referred to as action potentials or spikes to other connecting neurons. A set of spikes originating from a presynaptic neuron within a specified time frame forms a spike train. The number of spikes and most importantly the timing between them is what conveys information. The action potential of a presynaptic neuron is presynaptic potential and the response of a postsynaptic neuron to a spike arriving at its synapse at a given time, $t > 0$ is called postsynaptic potential (PSP). The difference between the potential of a postsynaptic neuron at rest (when it has not received spikes) and the potential when spikes arrive at its synapses is the neuron's membrane potential (potential difference).

Given that the resting potential of a postsynaptic neuron, i is u_r and its potential in a time course $t \geq 0$ is $u_i(t)$. At rest, $t = 0$, $u_i(t) = u_r$. That is when no spike has arrived at the synapses of neuron i. At a time, $t > 0$, where a spike arrives at the synapse from a presynaptic neuron j, the postsynaptic potential of i, $\varepsilon_{ij}(t)$ is calculated using equation (2.1) [3],

$$\varepsilon_{ij}(t) = u_i(t) - u_r. \tag{2.1}$$

If the value of $\varepsilon_{ij}(t)$ is positive, the postsynaptic potential is said to be excitatory (EPSP) and inhibitory (IPSP) when negative.

Each neuron in a network may be connected to several neurons, and thus can receive multiple spikes from these neurons. This implies that in SNN a postsynaptic neuron may be connected to and hence, receive spikes from multiple presynaptic neurons. In this case, the cumulative change of potential in the postsynaptic neuron is approximately the sum of the potentials evoked by the spikes from each presynaptic neuron. Consider a scenario where presynaptic neurons; $j = 1, 2, 3, ..., n$, each emitting spikes arbitrarily at time $t_j^{(f)}$. Where f $(f = 1, 2, 3, ..., f_s)$, is the f^{th} spike of neuron j. Summing over all spikes and neurons, $\varepsilon_{ij}(t)$ in equation (2.1) becomes $\sum_j \sum_f \varepsilon_{ij}\left(t - t_j^{(f)}\right)$. Thus, the total postsynaptic potential (PSP) is defined in equation (2.2)

$$u_i(t) = \sum_j \sum_f \varepsilon_{ij}\left(t - t_j^{(f)}\right) + u_r. \tag{2.2}$$

As shown in equation (2.2), when the total PSP of Neuron i reaches a given threshold, ϑ, it emits a spike (action potential) via its axon to the dendrites of adjoining neurons.

2.2 Spiking Neural Models

The dynamics of spiking neurons are modelled using biologically plausible mathematical neural models in which the well studied models include: Hodgkin-Huxley (HH) model [23], Integrate-and-Fire (IF) models [3, 4], Izhikerich's (Iz) model [3, 24], and Spike Response Model (SRM) [3, 4]. A brief description of these models is presented here, a detailed description can be found in [3, 4].

The HH model is considered the most biologically realistic model among the models listed above. The HH model, though biologically realistic is very difficult to simulate with SNN because of its complexity. HH models the transmission of information in biological neurons using electrical circuits and is defined by equation (2.3)

$$C\frac{du}{dt} = -g_{Na}m^3h\left(u - E_{Na}\right) - g_K n^4\left(u - E_K\right) - g_L\left(u - E_L\right) + I\left(t\right), \tag{2.3}$$

where C is the membrane capacitance; u the voltage across the capacitance C; the parameters g_{Na}, g_K, and g_L are the conductance of the various ion channels (Sodium (Na), potassium (K), and an undefined leaky channel); and m, h, and n are probabilities that determine the opening of the Na and K channels.

The IF models are a simplified form of the HH model making them less complex and easier to simulate as compared to the HH models. The form of action potentials are not considered in IF models and spikes are only defined by their firing time, $t^{(f)}$. The IF model is defined by an electrical circuit modelled by equation (2.4), where C is the capacitor, R a resistor connected parallel to C and driven by the current $I\left(t\right)$

$$C\frac{du}{dt} = -\frac{1}{R}\left(u\left(t\right) - u_r\right) + I\left(t\right). \tag{2.4}$$

Introducing the membrane time constant of the neuron, $\tau_m = RC$, equation 2.4 can be rewritten as equation (2.5)

$$\tau_m\frac{du}{dt} = u_r - u\left(t\right) + RI\left(t\right). \tag{2.5}$$

Equation (2.5) defines the Leaky-Integrate-and-Fire (LIF) neuron model. The LIF neuron fires a spike at time $t^{(f)}$ when the condition $u\left(t^{(f)}\right) = \vartheta$ is satisfied with $u^{'}\left(t^{(f)}\right) > 0$. The potential is reset to a resting position $u_r < \vartheta$ after $t^{(f)}$.

There are other variants of the IF neuron model such as the Quadratic-Integrate-and-Fire and the Theta neuron models [3, 4].

The Iz model is defined by two differential equations as in equation (2.6) and (2.7). They are not as computationally complex as the HH model and thus represent a valuable trade-off between computational complexity and biological plausibility

$$\frac{du}{dt} = 0.04u\left(t\right)^2 + 5u\left(t\right) + 140 - w\left(t\right) + I\left(t\right), \tag{2.6}$$

$$\frac{dw}{dt} = a\left(bu\left(t\right) - w\left(t\right)\right), \tag{2.7}$$

with an after-spike resetting defined by Equation (2.8):

$$\text{if } u(t) \geq \vartheta; \qquad \text{then } \begin{cases} u(t) \leftarrow c \\ w(t) \leftarrow w + d \end{cases}, \tag{2.8}$$

where $w(t)$ is the membrane recovery variable and a, b, c, and d are dimensionless parameters.

The last amongst the most widely used neuron models is the Spike Response Model (SRM). The SRM is more generic when compared to the IF model, easier to understand and implement, and competes favourable with the HH model when simulating complex neuronal properties. The simplest form of the SRM is expressed in equation (2.9).

$$u_j(t) = \eta_j(t - \hat{t_j}) + \sum_{i\epsilon\Gamma_j} w_{ij}\varepsilon\big(t - \hat{t_i} - d_{ij}\big). \tag{2.9}$$

The kernel functions η_j and ε_{ij} model the reset of the potential after a spike and the membrane potential respectively, $\hat{t_j}$ is the last firing time of the postsynaptic neuron j, w_{ij} the synaptic weight, $\hat{t_i}$ the firing time of the presynaptic neuron i, Γ_j the set of presynaptic neurons connected to neuron j, and d_{ij} is the delay associated with axonal transmission.

3 SUPERVISED LEARNING IN SPIKING NEURAL NETWORK

Supervised learning, though has been successfully studied in second generation ANN for more than six (6) decades [25, 26], and has evolved to become one of the most sophisticated learning approaches in machine learning, its study in SNN and applications to real world tasks is still at the early stages. Existing supervised learning rules in SNN can be categorised with respect to the network structure (the number of layers in the network) they are derived for, which directly influence the type of task they can solve; the mathematical principles based on which they are derived, which influences the biological plausibility of a given rule; and the number of spikes neurons in a network can emit, this also determines the quantum of information flow in the network. A review of some major supervised learning models for spiking neural networks is presented below and categorised according to the learning principles employed. A summary of the reviewed papers is presented in Table 1.

3.1 Error Back Propagation and Gradient Descent Based Learning Models

The first successful supervised learning rule for spiking neural network called the SpikeProp, was introduced less than two (2) decades ago by [7]. The rule was derived for multi-layer networks with single spiking neurons mainly for classification tasks. The SpikeProp was derived using training mechanisms similar to the classical Error Back Propagation method used in second generation ANN. It relied on the definition of an error-function using the time difference between desired output spike time t_j^d and actual output spike time t_j^a of an output neuron j. The error-function is defined using the least mean squares error-function given as $E = \frac{1}{2}\sum_{j\in J}(t_j^a - t_j^d)^2$. The error is then back propagated into the hidden layers of the network and minimised by optimising the synaptic weights.

Following the successful implementation of the SpikeProp algorithm, several variants of it such as; Back Propagation (BP) with momentum [8], QuickProp [9], Resilient Propagation [9] were proposed as modifications of the original SpikeProp algorithm with the core aim of improving the convergence rate and classification accuracy. However, a major challenge in these rules, similar to the SpikeProp, is that output neurons could only emit at most a spike.

[27] on the other hand proposed a learning rule called the Tempotron in which input signals are classified based on the presence or absence of a spike in the output neuron. They defined error-functions based on the output neurons maximum voltage and threshold voltage contrary to desired and actual output spike times in the SpikeProp, in that, for input signals that the neuron is supposed to emit a spike in response but there is no spike the error is defined as $V_{th} - V(t_{max})$ and for input signals belonging to a class in which the output neuron is not supposed to emit a spike but there is a spike the error-function $V(t_{max}) - V_{th}$ is used. V_{th} denotes the threshold voltage and $V(t_{max})$ the maximum voltage recorded within the period Tms at time t_{max}. The gradient descent algorithm is used to minimise the error functions in the course of training and the synaptic weights are updated (i.e. increased or decreased) to induce or cancel a spike according to $\Delta w_i = \lambda \sum_{t_i<t_{max}} K(t_{max} - t_i)$.

The Tempotron as mentioned above is restricted to training a single neuron to emit 1 or no spikes in response to precise input spike times belonging to different classes within a given interval. Modelling the Tempotron around the presence or absence of spikes limit the ability of such neurons to convey information via its output to other neurons. As an improvement to the Tempotron, [28] proposed the Chronotron that relied on the distance between a neuron's actual output spike train and a desired spike train using the Victor-Purpora (VP) distance metric [29], which is "the minimum cost of transforming one spike train into the other by creating, removing or moving spikes" [28]. The cost function is defined as a modification of the VP distance. Two learning methods were investigated using their approach; the E-learning and I-learning rules. In the E-learning rule, the error function is minimised by performing piecewise gradient descent while the I-learning draws its inspiration from the E-learning and ReSuMe [13] rules in that a neuron is trained to emit target spike trains via synaptic changes that are proportional to synaptic currents at the timings of real and target output spikes.

More recently, [30] proposed a multi-layer single spiking learning rule. Using the neural model defined in equation (3.1), they projected that the time of the first spike of the output neuron can be estimated using equation (3.3) following (3.2). Equation (3.2) is mapped into the $z - domain$ as presented in equation (3.4)

$$\frac{dV^j(t)}{dt} = \sum_i w_{ji} \sum_r k(t - t_i^r), \tag{3.1}$$

where dV^j is the membrane potential of output neuron j, w_{ji} is the weight of the synapses connecting input neuron i to output neuron j, k models the membrane potential, and t_i^r is the firing time of presynaptic neuron i,

$$exp(t_o) = \frac{\sum_{i \epsilon I} w_i exp(t_i)}{\sum_{i \epsilon I} w_i - 1}, \tag{3.2}$$

$$t_o = \ln\left(\frac{\sum_{i \epsilon I} w_i exp(t_i)}{\sum_{i \epsilon I} w_i - 1}\right), \tag{3.3}$$

$$z_o = \frac{\sum_{i \epsilon I} w_i z_i}{\sum_{i \epsilon I} w_i - 1}, \tag{3.4}$$

where $z_o = exp(t_o)$, $z_i = exp(t_i)$, and w_i are the synaptic weights of input neurons that have spikes preceding t_o. They defined a cost function based on this and used Error Back propagation approach to optimise the synaptic weights. They evaluated their method on some classification task and reported interesting results. However, similar to the SpikeProp and Tempotron, single spiking neurons were used in this study.

Due to the limitations of single spike networks, there have been extensive research into extending some of these learning rules into multi-spiking rules as well as the proposition of new multi-spiking learning rules mainly for the purpose of data classification. This is desired because multi-spiking neurons can convey more information than single spiking neurons and are capable of learning any non-linearly separable data [31], which is a characteristic desired in all learning algorithms. In this regard, [32] and [33] respectively, modified the multi-layer SpikeProp learning rule to allow neurons in the input and hidden layers to fire multiple spikes. However, neurons in the output layer could only fire single spikes due to the difficulty associated with defining appropriate error functions for multiple spikes at the output neuron.

[34] proposed the first learning rule that allowed neurons in all layers of a multi-layer network to fire multiple spikes. The learning rule was derived using the error back propagation method. This rule, just like other error Back Propagation based rules is not biologically plausible, is difficult to train, particularly, when the number of spikes in the desired (output) spike train increases and also could only produce stable performance with a maximum of two (2) spikes per class label in the output neuron.

3.2 Evolving SNN and Evolutionary Algorithm Based Methods

The second group of SNN learning methods are those that employed evolving and/or evolutionary techniques for training. These come in two folds: Those that used evolving SNN (eSNN) architecture together with evolutionary algorithms for network feature and parameter optimisation [12, 35–37] and those that used evolutionary algorithms for direct synaptic weight optimisation [10,11,38–40].

The learning principles of eSNN is derived from the classical Evolving Connectionist Systems used in second generation neural networks. In eSNN, spiking neural models are used to model neural activities. To train an eSNN, an output neuron repository is initialised, and for each input sample belonging to a class, a new neuron is trained. The weight of the newly trained neuron is compared to weights of neurons in the repository and if there is an existing neuron in which the Euclidean distance from its weight to the weight of the new neuron is the least and less than a given threshold the two neurons are considered very similar and their weights and firing thresholds are merged according to specified rules. If no neuron is close enough to the new neuron it is added to the repository as a separate neuron. This process is carried out for all samples in the training set as the network evolves with each input sample. This process iteratively creates repositories; a repository for each class, which enable the acquisition of knowledge as and when it is available without the need to retrain with already learned samples [12,35–37].

[35] investigated an on-line learning procedure for SNN using a three (3) layer network. In their approach, an output map is created for each input sample propagated into the network and the weights between the output and hidden neurons are trained using equation (3.5)

$$\Delta w_{j,i} = mod^{order(a_j)} \tag{3.5}$$

where $w_{j,i}$ is the weight between neuron j of the hidden layer and neuron i of the output layer, $mod \in (0,1)$ is the modulation factor, $order(a_j)$ is the order of arrival of spikes from neuron j to neuron i. If there is an existing map that meets a similarity test with the newly trained map, the weights (W) and Postsynaptic threshold $(PSP_{threshold})$ are merged using equations (3.6) and (3.7) respectively.

$$W = \frac{W_{Map_{C(k)}} + N_{samples} W_{Map_{C(ksimilar)}}}{1 + N_{samples}} \tag{3.6}$$

$$PSP_{threshold} = \frac{PSP_{Map_{C(k)}} + N_{samples} PSP_{Map_{C(ksimilar)}}}{1 + N_{samples}} \tag{3.7}$$

$W_{Map_{C(k)}}$ and $PSP_{Map_{C(k)}}$ are weights and Postsnaptic Potential of neurons in the newly trained map while $W_{Map_{C(ksimilar)}}$ and $PSP_{Map_{C(ksimilar)}}$ are that of the most similar existing map, $N_{samples}$ is the number of instances that have already been used to train the network.

[36] on the other hand proposed an On-line eSNN (OeSNN) and investigated the suitability of combining data reduction techniques with on-line eSNN to regulate activities in the output repositories. In the OeSNN, in addition to finding existing neurons that met a distance criteria with newly trained neurons and merging their weights and threshold using similar mechanisms outlined in [35], if the distance criteria is not met, the newly trained neuron is added to the repository if the number of neurons in the repository at a particular point is less than a predefined repository size else the oldest neuron is replaced with the new one. This update mechanism was adopted to enforce the need for a restricted reservoir size and also eliminate the tendency to forget to discard outdated concepts when data streams are non-stationary. They further proposed and investigated other repository update mechanisms that eliminates the need for distance-based measures to determine which weights and thresholds to merge by employing data reduction techniques to determine candidate neurons for replacement whenever the upper boundary condition is met in a passive approach or a concept drift detection mechanism turns true in an active learning approach. The introduction of data reduction techniques was mainly to eliminate the shortfalls of relying on distance-based measures to determine candidate output neurons in repositories to merge or replace with newly trained neurons.

Evolutionary algorithms were later introduced into this learning paradigm to enable feature selection and network parameter optimisation [41–45]. In each evolution of the evolutionary algorithms in these studies, a sub feature space is selected and used to train an eSNN and the feature space, network parameters and performance are adapted by the evolutionary algorithm. This process is repeated in each evolution and the appropriate search operators applied according to the optimisation principles of the chosen evolutionary algorithm. The introduction of evolutionary algorithms is reported to have led to an improvement in performance of eSNNs [12]. Several evolutionary algorithms have been explored in this regard and notable among these is the work presented by [41] in which the Versatile Quantum-inspired Evolutionary Algorithm (vQEA) [46] is used in combination with eSNN learning mechanisms for a simultaneous selection of relevant feature subsets and optimisation of eSNN parameters following the wrapper approach [47].

Different from the use of hierarchical multi-model Estimation of Distribution Algorithm (hMM-EDA) in [41], [42], proposed a similar eSNN model but employed a Quantum inspired Particle Swamp Optimiser (QiPSO) for both feature selection and parameter optimisation. A modified version of the QiPSO dubbed the Dynamic QiPSO, in which the classical PSO algorithm is combined with QiPSO to enable a parallel exploration of the search space was proposed in [43,44]. [45], also, proposed a hybrid learning scheme that merged the Harmony Search (HS) algorithm [48] and eSNN architecture for data classifications tasks. In their case the HS algorithms was used to search for optimal values of the main learning parameters of the eSNN, which include the similarity value, modulation factor and proportion factor. Unlike the afore mentioned studies, the method proposed by [45] did not support dataset feature selection.

Contrary to the above approach to learning where evolutionary algorithms are hybridised with eSNN, the methods introduced by [10, 11, 38–40], used the evolutionary methods as a means of training synaptic weights. [38], made use of Differential Evolution, which is a novel minimisation technique that has the capability to solve non-differentiable, non-linear, and multimodal objective functions. Their approach starts by randomly initialising a defined number of sub-populations of weights in [-1 1]. All sub-populations are then evolved independently in parallel and in each generation the weights undergo mutation and selection. This evolutionary process is continued until a stopping criterion is met. They tested their approach using three (3) benchmark problems: The XOR problem, Diabetes and Iris datasets problems, which are classification problems and reported good results that are comparable to that of multilayer perceptrons and also trained with fewer network weight when compared to the SpikeProp method (6 weights against 320 for SpikeProp in the XOR problem).

[10], also proposed a novel spiking neural network learning method using an evolutionary strategy. They optimised both synaptic weights and delays by minimising an error function defined as: $E = \sum_{t}^{T}\sum_{t}(t_o^a(t) - t_o^t(t))^2$; where $t_o^a(t)$ and $t_o^t(t)$ are the actual and target spike times respectively, of an output neuron for a pattern t, and the total number of patterns in the training set denoted by T. They asserted that their choice of an evolutionary strategy over other methods such as the genetic algorithms is due to its ability to process real values without the need to convert them into binary form. They used Gaussian and Cauchy mutation schemes, and tournament selection and elitism to create new population for succeeding generations.

[11, 39, 40] employed nature-inspired meta-heuristic algorithms to search for optimal synaptic weights that maximised the classification ability of the SNN. These papers followed a similar concept for neural signal generation where input patterns x^i are transformed into signals of the form $I = x\dot{w}\gamma$, where w is the weight connecting pre and post-synaptic neurons and γ is a gain factor that assists the neuron to fire. Following this, the Cuckoo Search algorithm [49], Artificial Bee Colony algorithm [50], and Bat algorithm (BA) [51] were utilised by [52], [39], and [11] respectively, to search for optimal synaptic weights.

3.3 Spike Timing Dependant Plasticity Based Methods

The Remote Supervised Method (ReSuMe) [53], is one of the first biologically plausible learning rules derived for single layer networks in which neurons in the network can emit multiple spikes. The learning rule is derived using STDP (Hebbian) and anti-STDP (anti-Hebbian) processes [2]. The rule is governed by equation (3.8)

$$\frac{d}{dt}w_{oi}(t) = [S_d(t) - S_o(t)]\left[a_d + \int_0^{\infty} a_{di}(s)S_i(t-s)ds\right], \tag{3.8}$$

where $S_d(t)$ and $S_o(t)$ denote the target and actual output spikes, respectively, a_d is a non-Hebbian term that controls the amount of synaptic weight change, and $a_{di}(s)S_i(t-s)$ is a kernel.

This rule implements a spiking version of the Widrow-Hoff algorithm for rate-based neurons [26]. ReSuMe was derived to overcome some key functional limitation of the first supervised learning rule, the SpikeProp [7]. These include, the minimisation of the error between the target and actual spikes without the need for gradient calculations, which eliminated the difficulty in defining cost functions and the computational requirements of gradient decent based methods. Unlike the SpikeProp, which is tied to the SRM model, ReSuMe is neural model independent and can be used to simulate any neural model. Also, neurons in the network could fire multiple spikes thus, making it suitable for solving different supervised learning tasks such as spike sequence learning and data classification. However, with the ReSuMe, convergence is not guaranteed when the number of spikes in the output spike train exceeds one (1).

Following the uncertainty regarding convergence in ReSuMe with respect to multiple spikes in an output spike train, [15, 16], modified it to include delay learning as defined in equation (3.9), which they called Delay Learning-ReSuMe (DL-ReSuMe) and Extended Delay Learning-ReSuMe (EDL-ReSuMe), respectively. Their methods resulted in a significant improvement in both the accuracy and time requirements compared to the ReSuMe. DL and EDL-ReSuMe, like the traditional ReSuMe, are single layer learning rules and also suffered a significant loss in performance when the learning period, T is extended beyond 500ms under their experimental settings.

$$\frac{d}{dt}w_{oi}(t) = [S_d(t) - S_o(t)]\left[a_d + \int_0^{\infty} Tw(s)s_i(t-dt_i-s)ds\right], \tag{3.9}$$

where dt_i defines the synaptic delay associated with synapse i.

Another learning method that made use of the Widrow-Hoff rule is the Spike Pattern Association Neuron (SPAN) [54]. Instead of adopting the Widrow-Hoff rule into the spatio-temporal domain as done in the ReSuMe, [54] rather convolved the temporal coded spike times into analog signals and directly applied the Widrow-Hoff rule to adjust synaptic weights in the course of training. They successfully applied the SPAN to spike sequence learning and classification tasks in the presence of noise.

Though the ReSuMe and its variants have been successfully applied to a variety of tasks, their efficient application to classification tasks is hindered by the number of network layers they are defined for, particularly on non-linearly separable datasets. It has been asserted that single layer networks are unable to efficiently handle complex classification tasks [31, 34].

[31], proposed the first biologically plausible multi-layer network learning rule in which all neurons could fire multiple spikes. They combined the error function definition used in gradient descent to the learning process used in ReSuMe to form the new learning rule. Though this learning rule is more biologically plausible than the rule presented in [34]. This learning rule also recorded a drop in performance when the number of spikes in the desired spike train increases.

Table 1: **Summary of Reviewed Papers**

Article	Neural Model	Learning Method	Num. of Layers	Input Coding	Output Coding (# of Spikes)	Tasks
[7] (SpikeProp)	SRM	Gradient descent	2	Population coding	Precise timing (1)	Classification
[8] (BP with Momentum)	SRM	Gradient descent	2	Population coding	Precise timing (1)	Classification
[9] (QuickProp)	SRM	Error-gradient	2	Binary/Population coding	Precise timing (1)	Classification
[9] (Resilient Prop)	SRM	Gradient descent	2	Binary/Population coding	Precise timing (1)	Classification
[27] (Tempotron)	LIF	Gradient descent	1	Population coding	Presence or absence of a spike (1)	Classification
[28] (Chronotron)	SRM	Piecewise gradient descent	1	Gaussian distribution	multiple (1–3)	Spike pattern & classification
[30]	non-LIF	Gradient descent	2	Population coding	Time of first spike (1)	Classification
[32] (MuSpiNN)	SRM	Gradient descent	2	Population coding	Precise timing (1)	Classification
[33]	SRM	Gradient descent	2	Spike time coding & Poisson process	Time of first spike	Spike pattern classification
[34] (GMSES)	SRM	Gradient descent	2	Population (GRF)	Precise timing (2)	Classification
[35]	Thorpe	Evolving	2	Rank order population coding	Time of first spike	Classification
[36]	Thorpe	Evolving + data reduction	1	Population (GRF)	Time of first spike	Classification
[41]	Thorpe	Evolving + vQEA for feature selection and parameter optimisation	1	GRF population	Time of first spike	Classification
[42]	Thorpe	Evolving + QiPSO for feature selection and parameter optimisation	1	GRF population	Time of first spike	Classification
[43, 44]	Thorpe	Evolving + DQiPSO for probabilistic feature selection and parameter optimisation	1	Population (GRF)	Time of first spike	Classification
[45]	Thorpe	Evolving + HSA for optimal network parameter selection	1	Population (GRF)	Time of first spike	Classification
[38]	SRM	Differential Evolutionary	1	Normalised attribute values over learning interval	Number of spikes	Classification
[10]	SRM	Evolutionary strategy	2	Sparse & 1-Dimensional coding	Precise timing (1)	Classification
[39]	Iz	Cuckoo search algorithm	1	Input current	Multiple spikes (firing rate)	Classification
[40]	LIF	Artificial Bee Colony	1	Input current	Multiple spikes (fireing rate)	Classification
[11]	LIF & Iz	Bat algorithm	1	Input current	multiple spikes (firing rate)	Classification
[13]	LIF, HH, & Iz (model independent)	STDP/anti-STDP	1		Precise timing (multiple)	Spike sequence & Classification
[15]	LIF	STDP/anti-STDP with delay shift	1	Random spike train using Poisson process	Precise timing (multiple)	Spike pattern
[16]	LIF	STDP/anti-STDP with delay shift and multi-delay-value adjustmet	1	Random spike train using Poisson process	Precise timing (multiple)	Spike pattern
[54]	LIF	Gradient descent	1	Ramdom generated spikes	Evenly spaced multiple spikes	Spike pattern association & classification
[31]	LIF, HH, & Iz (model independent)	STDP/anti-STDP/Gradient descent	2	Population (GRF)	Precise timing (2)	Classification
[14]	LIF	STDP/anti-STDP	2	Population (GRF)	Precise timing (multiple)	Classification

More recently, [14], presented another multi-layer multi-spiking learning rule. They adopted and modified the ReSuMe learnig method to enable all neurons in all layers of a multi-layer network to fire multiple spikes. Unlike the rule presented in [31], they considered spikes fired by neurons in the hidden layer when training hidden layer weights. Similar to the multi-spiking rules proposed earlier, they assessed the performance of the rule by applying it to some benchmark datasets and the results recorded were comparable to some second generation networks.

4 DISCUSSION

From the literature presented above, supervised learning models in SNN are derived to solve two (2) main categories of tasks including spike sequence leanring and data classification. The

experimental results reported in these areas showed that supervised learning in SNN is a viable alternative to traditional ANN due to its lesser computational requirements and comparable classificatiction performance, which when well exploited can change the dynamics of leraning in ANN.
Spike sequence learning which is demostrated to be suitable for implementation in neuromorphic systems due to its scalable nature and closeness to biological neural behaviour is another plus in SNN [13], a task which is not possible in traditional ANN. Though spike sequence learning is demostrated as an important attribute of SNN, it is only single layer learning models based on the WidrowHoff rule that have been demostrated to be most suitable for such tasks [13, 16, 16]. Learning models that are derived using optimisation and other meta-heuristics techniques [7, 10, 38], are not suitable for spike sequence learning because it is extremely difficult to define appropriate cost functions to minimise errors in spike trains due to their discrete nature.

With regards to data classification, all categories of learning models were successfully applied and each produced very competitve experimental results when compared to some second generation networks and other machine learning algorithms on benchmark classification tasks. Within the different categories of SNN learning models considered in this paper, they all produced relatively comparable performace across all datasets used to test them. The benchmark classification problems that have been extensively used to test most of the models are the XOR problem and the Fisher Iris dataset [55].

There is a general assertion in literature that sought to link the number of layers in a network and the number of spikes neurons can emit to the performance of learning models on classification tasks [31]. However, experimental results recorded in studies that proposed models for training multilayer networks with multispiking neurons such as [14,31] are not better than single spiking models for both single and multilayer layer networks such as [10, 40]. This suggests that there is a need to conduct comprehensive evaluatory studies to confirm if indeed multilayer networks with multispiking neurons are superior to networks with other structures and at what point and complexity of data the difference is significant. Another issue that needs attention is the lack of significant attempt to investigate the impact data centric problems such as the class imbalance problem and outliers have on the performance of SNN models.

5 CONCLUSION

SNN is the newest generation of ANN. It presents so many advantages over classical ANN including its closeness to biological neural activities in terms of information encoding and neural activity modelling. There is a good number of research work that sought to leverage on the advantages of SNN to solve supervised learning tasks. These research works are however sparse in literature and with so many unresolved critical questions. This paper therefore surveyed state-of-the-art supervised learning models for SNN, highlighted their limitations and provided foresights for future research in the form of open issues. The surveyed learning models are grouped with respect to concepts based on which learning is done and include; those that used methods similar to classical error back propagation technique used in second generation ANN; those that relied on evolutionary algorithms and evolving neural network approach; and the final category are those that employed biological plausible mechanisms such as the spike timing dependant plasticity approach. It is established that, methods employing spike timing dependant plasticity approach allowed neurons in the networks to emit more spikes than the other methods. A key issue that need further exploration is to experimentally establish the impact of desired output spike train and network layers on classification accuracy of SNN models. This will confirm the assessing made in literature that multi-spiking learning models for multi-layered networks are required to solve certain classification tasks.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

[1] McCulloch W, Pitts W. A Logical Calculus of the Ideas Immanent in Nervous Activity. Bulletin of Matheamtical Biophysics. 1943;5:115-133.

[2] Hebb DO. The organization of behavior. J. Cogn. Neurosci. 1949;99:70.

[3] Gerstner W, Kistler MW. Spiking Neuron Models: Single Neurons, Populations, Plasticity. Cambridge University Press; 2002.

[4] Ponulak F, Kasinski A. Introduction to spiking neural networks: Information processing, learning and applications. Acta Neurobiologiae Experimentalis. 2011;71(4):409-33.

[5] Butts DA, Weng C, Jin J, Yeh CI, Lesica NA, Alonso JM, Stanley GB. Temporal precision in the neural code and the timescales of natural vision. Nature. 2007;449:92-95.

[6] Pfeiffer M, Pfeil T. Deep learning with spiking neurons: opportunities and challenges. Frontiers in Neuroscience. 2018;12.

[7] Bohte SM , Kok JN, La Poutré H. Error-backpropagation in temporally encoded networks of spiking neurons. Neurocomputing. 2002;48(1-4):17-37.

[8] Xin J, Embrechts MJ. Supervised learning with spiking neural Networks. In International Joint Conference on Neural Networks. 2001;3(3):1772-1777.

[9] McKennoch S, Liu DLD, Bushnell L. Fast Modifications of the SpikeProp Algorithm. The 2006 IEEE International Joint Conference on Neural Network Proceedings. 2006;3970-3977.

[10] Belatreche A, Maguire LP, McGinnity M, Wu QX. Evolutionary Design of Spiking Neural Networks. New Mathematics and Natural Computation. 2006;2(3):237-253.

[11] Turkson RE, Liu S, Baagyere EY, Eghan MJ. Using meta-heuristic algorithm in spiking neural network for pattern recognition tasks. In 2019 16th International Computer Conference on Wavelet Active Media Technology and Information Processing. 2019;22-28.

[12] Schliebs S, Kasabov N. Evolving spiking neural network A survey. Evolving Systems. 2013;4(2):87-98.

[13] Ponulak F, Kasiński A. Supervised learning in spiking neural networks with ReSuMe: Sequence learning, classification, and spike shifting. Neural Computation. 2010;22(2):467-510.

[14] Taherkhani A, Belatreche A, Li Y, Maguire LP. A supervised learning algorithm for learning precise timing of multiple spikes in multilayer spiking neural networks. IEEE Transactions on Neural Networks and Learning Systems. 2018;29(11):5394-5407.

[15] Taherkhani A, Belatreche A, Li Y, Maguire L. DL-ReSuMe: A delay learning-based remote supervised method for spiking neurons. IEEE Transactions on Neural Networks and Learning Systems. 2015;26(12):3137-3149.

[16] Taherkhani A, Belatreche A, Li Y, Maguire L. EDL: An Extended Delay Learning Based Remote Supervised Method for Spiking Neurons. In Neural Information Processing, Ser. Lecture Notes in Computer Science. 2015;9490(11):190-197.

[17] Hinton GE. Learning multiple layers of representation. Trends in Cognitive Sciences. 2007;11(10):428-434.

[18] Hopfield JJ. Computational Abilities. Biophysics. 1982;79:2554-2558.

[19] Maass W. Networks of spiking neurons: The third generation of neural network Models. Neural Networks. 1997;10(9):1659-1671.

[20] Markram H, übke JL, Frotscher M, Sakmann B. Regulation of synaptic efficacy by coincidence of postsynaptic aps and epsps. Science. 1997;275(10):213-215.

[21] Rullen RV, Thorpe SJ. Rate coding versus temporal order coding: What the retinal ganglion cells tell the visual cortex. Neural Computation. 2001;13(6):1255-1283.

[22] Van Rullen R, Guyonneau R, Thorpe SJ. Spike times make sense. Trends in neurosciences. 2005;28(1):1-4.

[23] FitzHugh R. Impulses and Physiological States in Theoretical Models of Nerve Membrane. Biophysical Journal. 1961;1(6):445-466.

[24] Izhikevich EM. Simple model of spiking neurons. IEEE Transactions on Neural Networks. 2003;14(6):1569-1572.

[25] Rosenblatt FF. The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review. 1958;65(6):386-408.

[26] Widrow B, Hoff EM. Adaptive Switching Circuits. 1960;4:96-104.

[27] Gütig R, Sompolinsky H. The tempotron: a neuron that learns spike timingbased decisions. Nature Neuroscience. 2006;9:420428.

[28] Florian RV. The chronotron: A neuron that learns to fire temporally precise spike patterns. PLoS ONE. 2012;7:8.

[29] Victor J, Purpura K. Metric-space analysis of spike trains: Theory, algorithms and application. Network: Computation in Neural Systems. 1997;8:127-164, 05.

[30] Mostafa H. Supervised Learning Based on Temporal Coding in Spiking Neural Networks. IEEE Transactions on Neural Networks and Learning Systems. 2018;29(7):3227-3235.

[31] Sporea I, Grüning A. Supervised learning in multilayer spiking neural networks. Neural Computation. 2013;25(2):473-509.

[32] Ghosh-Dastidar S, Adeli H. A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection. Neural Networks. 2009;22(10):1419-1431.

[33] Booij O, Tat Nguyen H. A gradient descent rule for spiking neurons emitting multiple spikes. Information Processing Letters. 2005;95(6) SPEC. ISS:552-558.

[34] Xu Y, Zeng X, Han L, Yang J. A supervised multi-spike learning algorithm based on gradient descent for spiking neural networks. Neural networks. 2013;43C:99-113.

[35] Simei GW, Lubica B, Nikola K. Adaptive Learning Procedure for a Network of Spiking Neurons and Visual Pattern Recognition. Advanced Concepts for Intelligent Vision Systems. ACIVS 2006. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg. 2006;4179:1133-1142.

[36] Lobo JL, Laña I, Del Ser Bilbao J, M. N., Kasabov N. Evolving Spiking Neural Networks for online learning over drifting data Streams. Neural Networks. 2018;108:1-19.

[37] Kasabov NK. Evolving Spiking Neural Networks. Berlin, Heidelberg: Springer Berlin Heidelberg. 2019;169-199.

[38] Pavlidis N, Tasoulis DK, Plagianakos VP, Vrahatis MN. Spiking Neural Network Training Using Evolutionary Algorithms. In Proc. of International Joint Conference on Neural Networks (IJCNN'05), 2005. Poznań University of Technology, Institute of Control and Information Engineering. 2005;2190-2194.

[39] Vázquez RA. Training spiking neural models using cuckoo search Algorithm. In 2011 IEEE Congress of Evolutionary Computation (CEC). 2011;679-686.

[40] Vázquez RA, Garro BA. Training spiking neural models using artificial bee colony; 2015.

[41] Schliebs S, Defoin-Platel M, Kasabov N. Integrated feature and parameter optimization for an evolving spiking neural network. In Advances in Neuro-Information Processing, M. Köppen, N. Kasabov, and G. Coghill, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg. 2009;1229-1236.

[42] Hamed HNA, Kasabov N, Shamsuddin SM. Integrated feature selection and parameter optimization for evolving spiking neural networks using quantum inspired particle swarm optimization. In 2009 International Conference of Soft Computing and Pattern Recognition. 2009;695-698.

[43] Hamed HNA, Kasabov N, Shamsuddin SM. Probabilistic Evolving Spiking Neural Network Optimization Using Dynamic Quantum-inspired Particle Swarm Optimization. Australian Journal of Intelligent Information Processing Systems. 2010;11(1).

[44] Hamed HNA, Kasabov N, Shamsuddin SM. Quantum-inspired particle swarm optimization for feature selection and parameter optimization in evolving spiking

neural networks for classification tasks. In Evolutionary Algorithms. 2011;133-148.

[45] Yusuf ZM, Hamed HNA, Yusuf LM, Isa MA. Evolving spiking neural network (esnn) and harmony search algorithm (hsa) for parameter optimization. In 2017 6th International Conference on Electrical Engineering and Informatics (ICEEI). 2017;1-6.

[46] Defoin Platel M, Schliebs S, Kasabov N. A versatile quantum-inspired evolutionary algorithm. In 2007 IEEE Congress on Evolutionary Computation. 2007;423-430.

[47] Kohavi R, John GH. Wrappers for feature subset selection. Artificial Intelligence. 1997;97(1):273-324.

[48] Geem ZW, Kim J, Loganathan G. A new heuristic optimization algorithm: Harmony search. Simulation. 2001;76:60-68, 02.

[49] Yang X, Suash Deb. Cuckoo search via lvy flights. In 2009 World Congress on Nature Biologically Inspired Computing (NaBIC). 2009;210-214.

[50] Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (abc) algorithm. Journal of Global Optimization. 2007;39:459-471, 11.

[51] Yang XS. A new metaheuristic Bat-inspired Algorithm. Studies in Computational Intelligence. 2010;284:65-74.

[52] Vázquez RA, Garro BA. Training spiking neurons by means of particle swarm optimization. In Advances in Swarm Intelligence, Y. Tan, Y. Shi, Y. Chai, and G. Wang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg. 2011;242-249.

[53] Ponulak F. Supervised learning in spiking neural networks with ReSuMe Method. Unpublished doctoral dissertation, Poznán University of Technology, Poland; 2006. Available: http://d1. cie. put. poznan. pl/˜ fp

[54] Mohemmed A, Schliebs S, Matsuda S, Kasabov N. Span: Spike pattern association neuron for learning spatio-temporal spike patterns. International Journal of Neural Systems. 2012;22(04).

[55] Dua D, Graff C. UCI machine learning repository; 2017. Available:http://archive.ics.uci.edu/ml