An Extensive Review of the Supervised Learning Algorithms for Spiking Neural Networks

Irshed Hussain and Dalton Meitei Thounaojam

Abstract A Spiking Neural Network (SNN) processes neural information through precise timing of spikes and is considered a brain-inspired computational model of the third generation of the artificial neural network. SNN has a set of biologically plausible spiking neurons that have proven effective in processing complex temporal and spatio-temporal data. In addition, SNNs are computationally powerful, energy-efficient as well as a dynamic systems. However, the formulation of efficient supervised learning algorithms for SNNs is challenging due to their inherently discontinuous and implicit non-linear mechanisms. It has become a significant challenge in this field. Moreover, there exist a few efficient supervised learning algorithms developed for SNN. This paper provides a thorough review of supervised learning algorithms developed for SNNs categorically. We have divided the supervised learning algorithms into several categories based on the core principles for optimisation, such as gradient rule, asymmetric supervised Hebbian learning, remote supervision, and metaheuristics.

Keywords Spiking neurons · Gradient rule · STDP · Remote supervision · Metaheuristics

1 Introduction

The Spiking Neural Network (SNN)—the "third generation" of Artificial Neural Network (ANN) [47]—overcomes the flaws in ANN such as biological plausibility, energy efficiency, and powerful computationally. It can mimic the human brain to a great extent. SNN is *biologically plausible* [58, 86], *energy-efficient* while simulating in neuromorphic hardware [20, 49, 86], and *computationally powerful* [19, 48] as it can approximate the same function which other generations of ANN do with very

I. Hussain (✉) · D. M. Thounaojam
National Institute of Technology Silchar, Silchar, Assam 788010, India
e-mail: ihussain.moon@gmail.com

D. M. Thounaojam
e-mail: dalton@cse.nits.ac.in

M. D. Borah et al. (eds.), *Big Data, Machine Learning, and Applications*, Lecture Notes in Electrical Engineering 1053, https://doi.org/10.1007/978-981-99-3481-2_6

less number of spiking neurons [48]. It primarily differs from SNN in information coding, synapse model, and neuron model. The former uses *rate coding* which is proven unlikely, and strong arguments are provided against the use of rate coding in the human brain by Thorpe et al. [65, 83]. The latter uses precise timing of *spikes* as information called *temporal coding* akin to the human brain [8, 11, 60]. There exists single, as well as multiple *synapse models* in SNN, implemented using the concept of kernel functions. In addition, SNN mainly uses Leaky-Integrate-and-Fire (LIF) [1, 7, 44, 72, 73, 85], Hodgkin-Huxley model [29–32], Spike Response Model (SRM) [18, 19, 43], and Izhikevich model [37] as *neuron model*. On the other hand, traditional ANN generally refers to neuron models as activation functions. There exist various linear as well as non-linear activation functions for traditional ANN, the most popular being the logistic or sigmoid activation function [25], ReLU [57], and softmax [23].

SNN can be defined as a finite set of N spiking neurons, a finite set of $S \subseteq N \times N$ synapses establishing a connection between elements of set N, synaptic weights between two synapses i and j, i.e., $w_{ij} \in \mathbb{R}$, a response function between i and j (where $(i, j) \in S$), $\Psi_{ij} : \mathbb{R} \rightarrow \mathbb{R}$, and a spike firing threshold Θ. Spiking neurons can mimic biological neurons where the relevant information between two spiking neurons is carried through the synapse(s) connected between any two neurons in terms of short electrical pulse with an amplitude about 10 mV and a duration about 1 ms [19, 60] called *action potential* or *spikes*. The *synaptic terminals* are those junctions where the exchange of information takes place between any two biological neurons through the receptive fields by the diffusion of *neurotransmitters*, and spiking neurons can mimic the simple form of this concept mathematically. Generally, information sender neuron is called the presynaptic neuron, and information receiver neuron is called the postsynaptic neuron. When presynaptic neuron(s) send information to a postsynaptic neuron, the internal state of the postsynaptic neuron changes. At rest, in a biological neuron membrane, the potential value remains at about [–65 mV to –70 mV] [19, 60] when there is an absence of input stimuli. Upon receiving input stimuli from presynaptic neuron(s), the value of the membrane potential of the postsynaptic neuron called Postsynaptic Potential (PSP) may increase or decrease according to the synapse model. An *excitatory synapse* model increases the PSP value which is called Excitatory Postsynaptic Potential (EPSP) [19, 49, 86] and an *inhibitory synapse* model decreases the PSP value which is called Inhibitory Postsynaptic Potential (IPSP) [19, 49, 86]. Note that a postsynaptic spiking neuron issues spike only when its PSP reaches a certain threshold and not at each propagation cycle like traditional ANN. The typical value of *threshold* in a biological neuron is about –55 mV [19, 49].

SNN has been used widely including in the task of classification and clustering. Its implementation in the hardware requires less energy. IBM's TrueNorth [9, 13], Intel Loihi [12], Tsinghua Tianjic [61], and DARPA Quad Copter [28] are energy-efficient neuromorphic hardware that use SNN. Note that it is *computationally powerful* [48] and can efficiently handle non-linear data with a single spiking neuron (works fine without hidden layer(s) as well as hidden neuron(s)), making the system computationally powerful. Since it can efficiently classify non-linear patterns without any hidden

layer(s), the synaptic load is less than conventional neural networks. Furthermore, SNN is used to develop the neuroprosthetic system where it exhibits properties that has the ability to adjust to the nonstationarity of the neuro-musculoskeletal system that is suitable to control neuro-prostheses [34, 64, 71].

2 Review of Supervised Learning Methods

The most challenging and crucial part of any supervised learning approach is hyper-parameter tuning to optimise the predicted output keeping in mind the target outputs. The optimising phenomenon is generally referred to as learning. With the flow of time, various supervised learning algorithms to train SNN have been developed utilising heterogeneous optimisation techniques. However, almost none of the algorithms is satisfactory, provided there is a fair trade between computational efficiency and biological plausibility. This section discusses the most popular supervised learning algorithms developed for SNN, thoroughly and categorically. Also, in [39, 46, 76, 90], a detailed review of different learning algorithms developed for SNN to train in a supervised manner using various approaches is presented lucidly.

2.1 Learning by Finding the Gradient

The gradient-based method is well-known and is widely used as an optimising tool to train a neural network. In general, gradient or slope is used to find the direction of error in the continuous curve, making it easy to move in that particular direction to fine-tune the overall network error. However, the constraint is that the curve must be continuous, which means it can only provide continuous input. Therefore, it is complicated and challenging to apply this approach in SNN since all information processing happens in discrete forms. For presynaptic spike-times x_i, hidden neuron's spike-times y_j, synaptic delays for k synapses d^k, and predicted spike-times z_m, the change in synaptic weights is represented using Eqs. (1), (4), (5), and (6) applying gradient estimation approach. The change in weights between hidden and output layers Δw_{jm}^k is given in Eqs. (1) and (4):

$$\Delta w_{jm}^k = -\eta \times \delta_m \times \xi\left(t - y_j - d^k\right) \tag{1}$$

where $\xi(t)$ is the α-kernel which shapes the synapse, the definition is given in Eq. (2):

$$\xi(t) = \frac{t}{\tau_s} \exp\left(1 - \frac{t}{\tau}\right) H(t) \tag{2}$$

where $H(t)$ is the Heaviside function represented using Eq. (3), and τ_s is the synaptic time constant:

$$H(t) = \begin{cases} 1, & \text{if } t > 0 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

The value of δ_m is calculated for the lateral use in Eq. (4):

$$\delta_m = \frac{(\delta_m - z_m)}{\sum_{j=1}^{q} \sum_{k=1}^{r} w_{jm}^k \times \xi\left(z_m - y_j - d^k\right) \times \left(\frac{1}{z_m - y_j - d^k} - \frac{1}{\tau}\right)} \tag{4}$$

Now, the change in weights between the input and hidden layers Δw_{ij}^k is given in Eqs. (5) and (6):

$$\Delta w_{ij}^k = -\eta \times \delta_j \times \xi\left(t - x_i - d^k\right) \tag{5}$$

$$\delta_j = \frac{\sum_{m=1}^{s} \delta_m \sum_{k=1}^{r} w_{jm}^k \times \xi(z_m - y_j - d^k) \times \left(\frac{1}{z_m - y_j - d^k} - \frac{1}{\tau}\right)}{\sum_{i=1}^{p} \sum_{k=1}^{r} w_{ij}^k \times \xi\left(y_j - x_i - d^k\right) \times \left(\frac{1}{y_j - x_i - d^k} - \frac{1}{\tau}\right)} \tag{6}$$

The final change in synaptic weights Δw_{jm}^k calculated using the value of δ_j is given using Eq. (6), which is added to the initial synaptic weights w_{jm}^k to get the new synaptic weights. Thus, the training happens in the case of a gradient-based approach for SNN. There are $i = 1, 2, 3, \ldots, I$ input neurons, $j = 1, 2, 3, \ldots, J$ hidden neurons, and $m = 1, 2, 3, \ldots, M$ output or readout neurons present in the network.

The challenge of discontinuity is solved to some extent by Bohte et al. [5] by introducing probably the first popular supervised learning algorithm to train an SNN connected in feed-forward fashion and naming it as SpikeProp [5]. The exciting part of the SpikeProp is its similar analogy with the most popular backpropagation algorithm of ANN. SpikeProp eliminates discontinuity by allowing a single spike-time while discarding the lateral spikes. Although SNN smoothly work with non-linear classification problems if implemented efficiently, without the need of hidden layer(s) and hidden neuron(s), SpikeProp used hidden layers and thereby suffered from the heavy computational cost. The reason is that hidden layers increase the synaptic load in the architecture, and, as a result, more computational power is required.

SpikeProp uses the population coding scheme [19] combined with the concept of time-to-first-spike [19] firing, i.e., in every neuron, the first firing time is most important than the lateral ones. The utilisation of time-to-first-spike eliminates the discontinuity problem by omitting the lateral spike-times and considering only the early spike-time. It is observed that first spike-times are the most relevant in terms of information carrier [51]. Thus, the input, hidden, and readout (i.e., output neurons) are restricted to fire only a single spike. The SRM [19] is selected as the neuron model for providing the dynamics of the membrane potential. In addition, the synapses are connected in a one-to-one fashion between every pair of SRM neurons. The error

direction was investigated in SpikeProp by finding the slope since the usage of the time-to-first-spike as given in [5] turns discrete into the continuous nature of spiking. Although SpikeProp was a success to some extent, it lags behind in propping up weights if a neuron (postsynaptic neuron) no longer fires a spike after receiving the input stimuli.

Moreover, inhibitory neurons are not investigated properly and use a minimum value of learning rate. In [50], QuickProp and Rprop improve SpikeProp to some extent, and it is observed that the small value of the learning rate or step size used in SpikeProp can be increased to a large value that also leads to successful training to a certain extent. Note that the convergence rate in online mode, biological plausibility (since the synapses are not well-explored, which is less similar to the biological neurons), and the computational cost of SpikeProp is the flaw of this algorithm. In [6, 21, 50, 68, 91, 92], the convergence rate and multiple spiking nature are further investigated, which makes SpikeProp more generalised and a speedy algorithm than the previous version in [5]. However, the major problem of SpikeProp being a gradient-based supervised learning algorithm persists, that is, the stagnation at the local minimum, and it is a problem with any gradient-based optimisation algorithm.

The surge or sudden jumps present in any optimisation algorithm that uses gradient rule to determine the error-direction disturbs an optimisation algorithm's consistency. In addition, SpikeProp did not consider the mixture of inhibitory and excitatory neurons because, in this case, there is always a threat to the convergence of the algorithm, and it is also a barrier when we want a synapse model to be biologically more realistic. In [68], Shrestha et al. also explore some of the demerits of this kind along with the problem in formulating the loss function. Some other gradient-based supervised learning uses a slightly different concept by utilising extended delta learning rule developed in [53, 55]. In the algorithm, each spike-train is allowed to go through convolution with the suitable kernel function, distinguishing the algorithm from the others. The gradient descent approach-based SPAN algorithm proposed in [54] uses the concept of the spike-pattern association, which works with a single synapse connected in the form of α-shaped synaptic curve. It used the area under the curve to compute the overall loss in the network while training is an exciting feature.

However, the common problem is aforementioned to persist. Therefore, moving in a different direction in the search for another approach becomes necessary. This approach is primarily based on the concept of Hebbian learning, especially asymmetric Hebbian learning which is discussed in the next section.

2.2 Asymmetric Supervised Hebbian Learning

The Spike-Time-Dependent Plasticity (STDP) is a biological process that optimises the information processing mechanism among neurons. It is considered the asymmetric form of Hebbian learning that adjusts the synaptic efficacy or weights between neurons, considering the timing of a neuron's spike-time (relative timings) and input spike-times. The correlation (temporal) between pre- and postsynaptic spiking neu-

rons is taken into consideration. The plasticity generally means change; the meaning maps to the synapse change (the change happens in terms of synaptic efficacy). Like any other synaptic plasticity mechanism, along with the development and fine-tuning of neuronal circuits while in the brain's development phase, it is believed that STDP handles the learning and storing corresponding information inside the brain [4, 69]. It explains activity-dependent development partially regarding two different concepts: the Long-Term Potential (LTP) and the other is the Long-Term Depression (LTD). When the repeated presynaptic spike arrives a few milliseconds before the postsynaptic spikes, it is referred to as the LTP. On the other hand, when the repeated presynaptic spike comes after the postsynaptic spikes, it is referred to as the LTD.

The learning window which is also called the STDP-function varies for different synapse models. The rapid change in the learning window's value forces the time scale to be represented to the millisecond. Although it primarily learns in an unsupervised manner and is considered a partial learning algorithm, most researchers combine STDP with a concept called anti-STDP to train in a supervised manner. There are various supervised algorithms for SNN which are developed using the STDP. However, a few are successful to some extent, both computationally and biologically.

The time difference Δt between presynaptic spike (t_{pre}) and postsynaptic spike (t_{post}) is represented as $\Delta t = (t_{\text{pre}} - t_{\text{post}})$. The change in synaptic weights for excitatory synapse $w_{\text{excitatory}}$ is given in Eq. (7). The exponentially decaying shape shown in Fig. 1 indicates the dependency on the time difference of spikes, i.e., Δt:

$$\Delta w_{\text{excitatory}} = \begin{cases} \mathcal{A}^{+} \exp(\frac{\Delta t}{\tau^{+}}), & \forall \Delta t < 0 \\ \mathcal{A}^{-} \exp(-\frac{\Delta t}{\tau^{-}}), & \forall \Delta t > 0 \\ 0, & \forall \Delta t = 0 \end{cases} \tag{7}$$

where $\mathcal{A}^{+}$, and $\mathcal{A}^{-}$ represents constant value (usually taken as 1.0) for the LTP and LTD, respectively. The values of τ^{+} and τ^{-} known as time constants shape the curve for LTP and LTD, respectively.

In [70], a learning algorithm for SNN is proposed in which STDP and anti-STDP are used to fit the algorithm in a supervised paradigm. In this algorithm, multiple spiking activity is used where each spiking neuron can fire multiple spikes at a different time step. The architecture of the network is feed-forward, having hidden layers. The demerit of the algorithm is the negligence of the precise spike-times produced by the neurons present in the hidden layers at the time of training. Qiang et al. proposed an algorithm that uses temporal coding to represent real-valued continuous information in the form of discrete spikes to train SNN in a supervised manner [93]. In [81], Aboozar et al. proposed the supervised learning that is biologically plausible called the BPSL algorithm, which is capable of firing multiple target spikes from a spiking neuron. Although it is referred to as the biologically plausible algorithm, there is a lack of proper implementation in the synapse model when essential biological elements are considered.

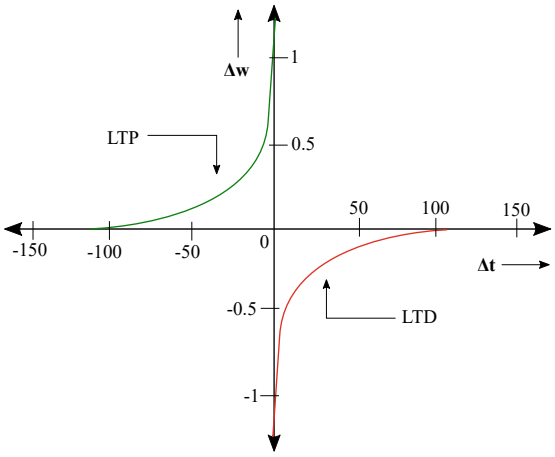

Fig. 1 The learning window for STDP (relation between synaptic weights and spike-time difference) where LTP is represented by the left curve, and LTD is represented by the right curve

In [88], John et al. proposed a supervised algorithm using synaptic weight association training and called it SWAT. It is used to classify the non-linear feed input patterns into their respective target patterns. There is an exciting feature present in the SWAT algorithm that uses the dynamic synapse model [10, 84], which is capable of working in terms of the mechanism of long-term plasticity. SWAT has biological properties to some extent. However, the major drawback of SWAT is the computational cost since it has a huge synaptic load to be dealt with having high computational power. The increase in synaptic load results from a huge number of connections formed due to the presence of many hidden neurons in the network topology. Therefore, it is challenging for a computer with moderate computational power to adjust and fine-tune many network parameters. As far as SWAT training is concerned, it is trained using the STDP algorithm transforming into the supervised paradigm.

Tempotron algorithm, proposed in [24], which trains SNN in a supervised manner, came with a slightly different picture. It allows a neuron to learn spike firing decisions (whether to issue a spike or not) when its cell membrane is updated with the potential of incoming input stimuli from several presynaptic neurons. The working of Tempotron's response is like a switch "on" or "off" akin to a digital system. Instead of precise spike-time learning, Tempotron decides the ability of a neuron's firing (acts like a decider). This algorithm also lags behind when there is the question of a balanced trade-off between biological plausibility and computational efficiency. In addition, Tempotron can be used only in a single-layered network topology which is a barrier for multilayered network topology. Also, it is restricted to 0 or 1 as output which does not encode information in precise spike timing.

Other more supervised learning algorithms are primarily based on STDP; a few of the most used are discussed in this literature review. J. Wang et al. proposed the OSNN algorithm in [89] which is an online supervised learning algorithm for SNN. The OSNN has an adaptive network structure trained in an online fashion using

supervised learning patterns. In [56, 80], a supervised learning algorithm is proposed where the concept of STDP and anti-STDP is used to make the algorithm work as supervised learning. It is well-known that STDP primarily works in an unsupervised fashion. It is not considered a fully functional learning algorithm due to its plasticity updating mechanism, which changes the sign of synaptic efficacy instead of updating a fair value based on all presynaptic neurons' spike firing times. It is a barrier to STDP-based supervised learning. Note that the Hebbian approach-based supervised learning algorithm has a common problem, which is the continuous change in synapse parameters even if neuron fire spikes exactly match the target spikes. Thus, there is a need for some extra work for adding additional learning rules or constraints to the original algorithm to provide stability. Moreover, in supervised Hebbian learning, all undesired timings of the spike are usually suppressed by the "teaching signal" during the training phase. Therefore, corelation happens only between pre- and postsynaptic spikes, around the desired timings of the spike. Since this type of corelation is absent in all other circumstances, synaptic strength cannot be weakened even if a neuron fires spikes at undesired times during the testing phase.

It is observed from the literature that spiking neurons have the ability to successfully classify non-linear patterns into their respective target classes without using any hidden layer(s), and this powerful feature of spiking neurons is not implemented in the aforementioned learning algorithms except SEFRON proposed in [38]. SEFRON did not use any hidden layer. However, it was successful in classifying the non-linear patterns, thereby decreasing the synaptic load. It explores the computational power to a certain extent by utilising a single spiking neuron. However, we analysed and observed that the number of network parameters could be reduced to half, keeping the classification accuracy unhampered, which we experimented successfully.

2.3 Learning with Remote Supervision

Ponulak et al. [62] proposed a distinguished learning algorithm called ReSuMe that is based on the concept of "remote supervision". It is argued that ReSuMe eliminates the significant drawbacks found in the supervised Hebbian learning approach. Apart from this, ReSuMe also implements some exciting features. The primary principle is to impose the input-output characteristics into the SNN for yielding the target spike trains in response to the corresponding input spikes. Unlike supervised Hebbian learning, ReSuMe does not directly feed the desired signals to the learning neurons. Nevertheless, it can co-determine the synaptic connection's plasticity. The algorithm ReSuMe also uses the supervised Hebbian approach for learning, but its "remote supervision" feature primarily distinguishes it from the others that use the supervised Hebbian learning approach. The concept of "remote supervision" is biologically justifiable based on an experimentally observed neurophysiological phenomenon—heterosynaptic plasticity [63, 75, 87]. The working rule of ReSuMe is briefly explained in Eq. (8):

$$\frac{\mathrm{d}}{\mathrm{d}t}w(t) = \left[\mathcal{S}^d(t) - \mathcal{S}^l(t)\right]\left[a + \int_0^\infty \mathcal{W}(s) \times \mathcal{S}^{in}(t-s)\mathrm{d}s\right] \tag{8}$$

where $\mathcal{S}^d(t)$, $\mathcal{S}^l(t)$, and $\mathcal{S}^{in}(t)$ represent the desired, presynaptic (input), and postsynaptic (output) spike trains, respectively. The parameter a denotes the amplitude of the contribution (non-correlated) to the $\frac{d}{dt}w(t)$, and the convolution function given in Eq. (8) is the modification (Hebbian-like) of w. The value of s represents the time-delay between spikes of synaptic sites and over s, and the integral kernel $\mathcal{W}(s)$ is defined as shown in Eq. (8). A positive value of a corresponds to the excitatory synapses where the shape of $\mathcal{W}(s)$ becomes similar to the STDP rules, and a negative value of a corresponds to the inhibitory synapses where the shape of $\mathcal{W}(s)$ becomes similar to the anti-STDP rules.

The exciting merit of ReSuMe is its independence from the spiking neuron models. Therefore, it can work with a variety of spiking neuron models. Also, ReSuMe can learn the target temporal as well as spatio-temporal spikes efficiently. In addition, it converges quickly towards the optimum value. There exist algorithms that explore ReSuMe in a better manner, such as in [77, 78], the ReSuMe algorithm is further investigated, where synaptic delays were added. The delay used is the static constant values is not random. In addition, in [79], multiple neurons are successfully trained using the training rules of ReSuMe instead of training a single neuron. However, ReSuMe has many disadvantages despite the advantageous features: ReSuMe claims to be suitable for online learning, but due to the fixed network topology, it is not adaptive to the incoming stimuli. Also, ReSuMe is unable to predict inputs just after single usage of the training patterns. Although ReSuMe is biologically plausible, local behaviour restricts its learning ability.

Another exciting supervised learning algorithm that works on the ReSuMe principle developed to train SNN is called Chronotron, proposed by Florian et al. in [16]. The Chronotron is experimented with using three different models: first is the gradient descent learning (called gradient descent E-learning) where delta learning rule is used, second is the I-learning where gradient descent E-learning and ReSuMe learning rule are combined and used. The third one is the ReSuMe learning rule. Supervised learning is implemented using a sophisticated distance metric called VictorPurpora in Chronotron, an exciting feature of the algorithm. However, Chronotron trained the synaptic efficacies in a batch mode by fixing the network topology, making it unsuitable for online learning.

2.4 Learning with Metaheuristics

Heuristic methods are used as a powerful and comprehensive tool for solving challenging optimisation problems. Although heuristics provide "good balanced" solutions relatively very close to the global optimum in affordable cost and time, their

design and development become complicated as they depend on "problem-specific" characteristics [59]. Therefore, to solve the flaw mentioned above, metaheuristics came into existence [22]. Metaheuristics are "problem-agnostic" rather than "problem-specific" and have become remarkably popular in many optimisation areas, such as developing learning algorithms for ANN. However, the power of metaheuristics is significantly less explored and experimented with within the case of SNN. In this section, the metaheuristic approaches which are used to train SNN are briefly discussed.

Metaheuristics such as evolutionary algorithms are mathematically simpler and can work on the real numbers directly, and do not waste time encoding these real numbers into other formats. Therefore, most of the complex classification problems want this strategy. In [67], an evolving network of spiking neurons is proposed, which is based on the Thorpe model [82] called eSNN. The advantages of eSNN include the fast real-time simulation achieved at a low computational cost in a large network architecture. Also, without retaining past data, the model can accumulate knowledge at the time of data arrival. The usage of fuzzy rules for yielding the inference engine is an exciting feature of eSNN. However, eSNN has many disadvantages, such as the "infinite repository" problem. For each new arrival of patterns in online fashion, its repository of neurons grows infinitely. Also, due to the usage of averaging synaptic weights with rank order, eSNN cannot handle input patterns having the same rank (despite having different spike-times), as well as rank order, and also can increase the number of neurons in the network, which may lead to the loss of relevant stored information.

In [14], synaptic efficacies of SNN were optimised to reduce the overall network error using evolutionary techniques where the concept of "self-regulatory" (called the algorithm as SRESN) is appropriately implemented, which regulates the learning process. The current stored knowledge can automatically evolve the output layer neurons based on training patterns. SRESN can add a neuron, change network parameters, or forgo learning from samples based on the class-specific and sample knowledge stored in the network. Thus, SRESN works in a "self-regulatory" mode of learning. This method has both online and offline modes of training. However, SRESN does not use synaptic delays, which is an essential factor, to provide better computational cost compromising the biological plausibility.

Evolutionary methods are also used to improve the gradient-based SpikeProp algorithm [5], which uses the Particle Swarm Optimisation (PSO) technique [41], and it is referred to as SpikeProp-PSO. It enhanced the learning process of SpikeProp using the angle-driven dependency-learning rule. However, it increases the computational cost. Also, it is biologically less plausible since the biological elements present in synapses are neglected.

Differential Evolution (DE) [74] is a powerful optimisation tool known for its simplicity and good performance, which is combined with eSNN [67] to develop

another supervised learning algorithm called DEPT-ESNN [66]. The primary goal of DEPT-ESNN is to select the optimum number of eSNN parameters such as modulation factor, similarity factor, and threshold. In DEPT-ESNN, DE plays a vital role by providing suitable values for the mentioned eSNN parameters adaptively rather than trial-and-error. The advantage of DEPT-ESNN includes its simple implementation and generalisation. But biological elements present in synapses are not considered, which makes DEPT-ESNN biologically less plausible.

Although metaheuristic approaches are a bit time-consuming and can work with a single spiking scheme, they have many advantages that are not achievable using other optimisation approaches. Therefore, there is a need for more exploration of metaheuristic approaches to develop an efficient learning algorithm compatible with SNN. Note that other powerful metaheuristics such as Genetic Algorithm (GA) [3, 26, 33] and Grey Wolf Optimisation (GWO) [52] are neither explored nor successfully experimented directly, providing a properly balanced trade-off between the computational cost and biological plausibility, with SNN trained in the supervised manner.

The aforementioned supervised learning algorithms, irrespective of the approach used, did not explore the synapse model thoroughly which is found from the literature. Although in some algorithms such as [5, 77–79] synaptic delays [40] is used, those are constant synaptic delays, and wherever the usage of the mixture of excitatory neurons and inhibitory neurons is observed, those are not appropriately implemented like GABA-switch [17, 45]. Synaptic delays are significant when biological plausibility is concerned. In the GABA-switch mechanism, switching from excitatory neuron to inhibitory and vice versa happens randomly. The robustness of an algorithm is tested against noise, and in the biological process, the presence of noise is evident while sharing information among neurons [15]. Therefore, it should be robust for a model to be biologically plausible, which is less explored as far as SNN is concerned. Another important phenomenon observed in a biological neuron is the spontaneous firing of spikes [27, 42] which is almost neglected in most of the synapse models of an SNN architecture.

Moreover, there is a lack of a balanced trade-off between the computational cost and biological plausibility in almost all the aforementioned supervised learning algorithms developed to train an SNN topology. A balanced trade-off between the computational cost and biological plausibility is essential in the case of SNN because if the computational complexity is very high, it is difficult to handle a high-dimensional dataset.

Tables 1 and 2 show a brief summary of the gradient and STDP-based supervised learning algorithms, and a brief summary of remote-supervision and metaheuristic supervised learning algorithms.

Table 1 A brief summary of gradient and STDP-based supervised learning algorithms

Approach	Algorithm	Advantages	Disadvantages
Gradient		(1) Able to solve complex non-linear classification problems	(1) When a post-synaptic neuron stops firing/responding to its corresponding input patterns, there is no mechanism using which synaptic weights can be "propped up"
	SpikeProp [5] and Variants [6, 21, 50, 68, 91, 92]	(2) Computationally powerful for classification	(2) Even though neurons fire at most one spike due to the time-to-first-spike encoding, the synaptic load is very high; mathematically challenged
		(3) QuickProp improves the convergence speed using momentum, Rprop also seems to speed up SpikeProp	(3) Only excitatory neurons with a simple synapse model are used, and arbitrary values of synaptic delays are used—a barrier to biological plausibility
Common problem: Gradient-based optimisation algorithms may be stuck at the local minimum			
STDP	SWAT [88]	(1) Uses dynamic synapse model (2) Can handle large non-linear datasets	(1) Huge synaptic load—it is computationally very costly (2) More number of network parameters to adjust; very less biological elements are used in the synapse model
	Tempotron [24]	(1) Applicable to a wide range of input classes, and it is flexible to information encoding scheme	(1) Suitable only for single-layered network topology. (2) Lack of precise spike-timing information due to the restricted output either as 0 or 1 during a predetermined interval; less biologically plausible
	SEFRON [38]	(1) Lower computational cost (2) Less network parameters are to be adjusted as there is no hidden layer(s)	(1) Stability and robustness are not assured (2) Computational complexity can be reduced to half keeping classification accuracy unhampered; less biologically plausible
	Others [56, 70, 80, 81, 89, 93] [77–79]	(1) Can be used for a wide range of classification problems including large datasets. In [77–79], synaptic delays are considered. [93] performs well with the MNIST dataset	(1) Learning rule is based on STDP, not a fully functional supervised learning algorithm. Synaptic delays considered in [77–79] are not random. A higher value of constant synaptic delays can affect learning. In [70], spikes fired by hidden neurons are neglected while training
Common problem: STDP is not considered a fully supervised learning algorithm, also stability cannot be guaranteed			

Table 2 A brief summary of remote-supervision and metaheuristic-based supervised learning algorithms

Approach	Algorithm	Advantages	Disadvantages
Remote Supervision		(1) Solves the flaw in SpikeProp and is independent of the spiking neuron models	(1) Due to the fixed network topology not adaptive to incoming stimuli
	ReSuMe [62]	(2) Efficiently learns target temporal and spatio-temporal spike-patterns	**(2)** After only single use of the training patterns, it is impossible to predict inputs
		(3) Quickly converges towards optimised values	(3) Moderately biologically plausible; but local behaviour restricts its learning ability
	Chronotron [16]	(1) Supervised learning is implemented using a sophisticated distance metric called VictorPurpora	(1) Trained the synaptic efficacies in batch mode by fixing the network topology, making it unsuitable for online learning
Common problem: Network topology should be fixed before training, which is not adaptive			
Metaheuristic	PSO-SpikeProp [2]	(1) Enhanced learning process of SpikeProp using angle-driven dependency-learning rule	(1) Poor performance; increase in computational cost and biological elements present in synapses are not considered
	DEPT-ESNN [66]	(1) Simple implementation and generalisation	(1) Moderate performance; biologically not plausible
	eSNN[67]	(1) Fast real-time simulation provided low computational cost in case of large network architecture	(1) For each new arrival of patterns in online fashion, its repository of neuron grows infinitely
		(2) Without retaining past data, the model is capable of accumulating knowledge in case of data arrival	(2) Due to the usage of averaging synaptic weights with rank order, it cannot handle input patterns having the same rank
		(3) The exciting feature is the fuzzy rule generation	**(3)** The rank order might lead to an increase in the number of neurons in the network, which can lead to information loss
Common problem: Very time consuming; not guaranteed to find the optimal solution, but finds near-optimal solution			

3 Conclusion

The supervised learning algorithms discussed in this paper, irrespective of the approach used, did not explore the synapse model thoroughly that is found from the literature. Although in some algorithms such as [5, 77–79] synaptic delays [40] are used, those are constant synaptic delays, and wherever the usage of the mixture of excitatory neurons and inhibitory neurons is observed, those are not appropriately implemented like GABA-switch [17, 45]. Synaptic delays are significant when biological plausibility is concerned. In the GABA-switch mechanism, switching from excitatory neuron to inhibitory and vice versa happens randomly. The robustness of an algorithm is tested against noise, and in the biological process, the presence of noise is evident while sharing information among neurons [15]. A model should be robust to be biologically plausible, which is less explored in the case of SNN. Another important phenomenon observed in a biological neuron is the spontaneous firing of spikes [27, 42] which is almost neglected in most of the synapse models of an SNN architecture.

Although metaheuristic approaches are a little time-consuming and can work with a single spiking scheme, they have many advantages that are not achievable using other optimisation approaches. Therefore, there is a need for more exploration of metaheuristic approaches to develop an efficient learning algorithm compatible with SNN. Note that other powerful metaheuristics such as Genetic Algorithm (GA) [3, 26, 33] and Grey Wolf Optimisation (GWO) [52] are rarely explored successfully and experimented directly, providing a properly balanced trade-off between the computational cost and biological plausibility, with SNN trained in the supervised manner except [35] and [36].

References

1. Abbott LF (1999) Lapicque's introduction of the integrate-and-fire model neuron (1907). Brain Res Bull 50(5–6):303–304
2. Ahmed FY, Shamsuddin SM, Hashim SZM (2013) Improved spikeprop for using particle swarm optimization. Math Probl Eng
3. Baluja S, Caruana R (1995) Removing the genetics from the standard genetic algorithm. In: Machine learning proceedings. Elsevier, pp 38–46
4. Bi GQ, Poo MM (2001) Synaptic modification by correlated activity: Hebb's postulate revisited. Annu Rev Neurosci 24(1):139–166
5. Bohte SM, Kok JN, La Poutre H (2002) Error-backpropagation in temporally encoded networks of spiking neurons. Neurocomputing 48(1–4):17–37
6. Booij O, tat Nguyen H (2005) A gradient descent rule for spiking neurons emitting multiple spikes. Inf Process Lett 95(6):552–558
7. Brunel N, Van Rossum MC (2007) Lapicque's 1907 paper: from frogs to integrate-and-fire. Biol Cybern 97(5–6):337–339
8. Cariani PA (2004) Temporal codes and computations for sensory representation and scene analysis. IEEE Trans Neural Netw 15(5):1100–1111
9. Cassidy A, Sawada J, Merolla P, Arthur J, Alvarez-lcaze R, Akopyan F, Jackson B, Modha D (2016) Truenorth: A high-performance, low-power neurosynaptic processor for multi-sensory

perception, action, and cognition. In: Proceedings of the government microcircuits applications and critical technology conference. Orlando, FL, USA, pp 14–17
10. Choquet D, Triller A (2013) The dynamic synapse. Neuron 80(3):691–703
11. Comsa IM, Fischbacher T, Potempa K, Gesmundo A, Versari L, Alakuijala J (2020) Temporal coding in spiking neural networks with alpha synaptic function. In: ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 8529–8533
12. Davies M, Srinivasa N, Lin TH, Chinya G, Cao Y, Choday SH, Dimou G, Joshi P, Imam N, Jain S et al (2018) Loihi: a neuromorphic manycore processor with on-chip learning. IEEE Micro 38(1):82–99
13. DeBole MV, Taba B, Amir A, Akopyan F, Andreopoulos A, Risk WP, Kusnitz J, Otero CO, Nayak TK, Appuswamy R et al (2019) Truenorth: accelerating from zero to 64 million neurons in 10 years. Computer 52(5):20–29
14. Dora S, Subramanian K, Suresh S, Sundararajan N (2016) Development of a self-regulating evolving spiking neural network for classification problem. Neurocomputing 171:1216–1229
15. Faisal AA, Selen LP, Wolpert DM (2008) Noise in the nervous system. Nat Rev Neurosci 9(4):292–303
16. Florian RV (2012) The chronotron: A neuron that learns to fire temporally precise spike patterns. PLOS ONE 7:1–27. https://doi.org/10.1371/journal.pone.0040233
17. Ganguly K, Schinder AF, Wong ST, Poo MM (2001) Gaba itself promotes the developmental switch of neuronal gabaergic responses from excitation to inhibition. Cell 105(4):521–532
18. Gerstner W (1995) Time structure of the activity in neural network models. Phys Rev E 51(1):738
19. Gerstner W, Kistler WM (2002) Spiking neuron models: Single neurons, populations, plasticity. Cambridge University Press
20. Gerstner W, Kistler WM, Naud R, Paninski L (2014) Neuronal dynamics: from single neurons to networks and models of cognition. Cambridge University Press
21. Ghosh-Dastidar S, Adeli H (2009) A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection. Neural Netw 22(10):1419–1431
22. Glover F (1977) Heuristics for integer programming using surrogate constraints. Decision Sci 8(1):156–166
23. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press
24. Gütig R, Sompolinsky H (2006) The tempotron: a neuron that learns spike timing-based decisions. Nat Neurosci 9(3):420–428
25. Han J, Moraga C (1995) The influence of the sigmoid function parameters on the speed of backpropagation learning. In: International workshop on artificial neural networks. Springer, pp 195–201
26. Haupt RL, Ellen Haupt S (2004) Practical genetic algorithms. Wiley Online Library
27. Häusser M, Raman IM, Otis T, Smith SL, Nelson A, Du Lac S, Loewenstein Y, Mahon S, Pennartz C, Cohen I et al (2004) The beat goes on: spontaneous firing in mammalian neuronal microcircuits. J Neurosci 24(42):9215–9219
28. Hewitt J (2014) Darpa's new autonomous quadcopter is powered by a brain-like neuromorphic chip. https://www.extremetech.com/extreme/193532-darpas-new-autonomous-quadcopter-is-powered-by-a-brain-like-neuromorphic-chip, online. Accessed 5 Nov
29. Hodgkin AL, Huxley AF (1952) A quantitative description of membrane current and its application to conduction and excitation in nerve. J Physiol 117(4):500–544
30. Hodgkin AL, Huxley AF, Katz B (1952) Measurement of current-voltage relations in the membrane of the giant axon of loligo. J Physiol 116(4):424
31. Hodgkin AL, Huxley AF (1952) The components of membrane conductance in the giant axon of loligo. J Physiol 116(4):473
32. Hodgkin AL, Huxley AF (1952) Currents carried by sodium and potassium ions through the membrane of the giant axon of loligo. J Physiol 116(4):449

33. Holland JH et al (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press
34. Huber J, Lisiński P, Kasiński A, Kaczmarek M, Kaczmarek P, Mazurkiewicz P, Ponulak F, Wojtysiak M (2004) Therapeutic effects of spinal cord and peripheral nerve stimulation in patients with the movement disorders. Artif Organs 28(8):766
35. Hussain I, Thounaojam DM (2020) Spifog: an efficient supervised learning algorithm for the network of spiking neurons. Sci Rep 10(1):1–11
36. Hussain I, Thounaojam DM (2021) Wolif: An efficiently tuned classifier that learns to classify non-linear temporal patterns without hidden layers. Appl Intell 51(4):2173–2187
37. Izhikevich EM (2003) Simple model of spiking neurons. IEEE Trans Neural Netw 14(6):1569–1572
38. Jeyasothy A, Sundaram S, Sundararajan N (2018) Sefron: a new spiking neuron model with time-varying synaptic efficacy function for pattern classification. IEEE Trans Neural Netw Learn Syst 30(4):1231–1240
39. Kasiński A, Ponulak F (2006) Comparison of supervised learning methods for spike time coding in spiking neural networks. Int J Appl Math Comput Sci 16(1):101–113
40. Katz B, Miledi R (1965) The measurement of synaptic delay, and the time course of acetylcholine release at the neuromuscular junction. Proc R Soc London Ser B Biol Sci 161(985):483–495
41. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks, vol 4. IEEE, pp 1942–1948
42. Kerschensteiner D (2014) Spontaneous network activity and synaptic development. Neurosci 20(3):272–290
43. Kistler WM, Gerstner W, Hemmen JLV (1997) Reduction of the hodgkin-huxley equations to a single-variable threshold model. Neural Comput 9(5):1015–1045
44. Lapicque L (1907) Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polarization. Journal de Physiologie et de Pathologie Generalej 9:620–635
45. Lee SW, Kim YB, Kim JS, Kim WB, Kim YS, Han HC, Colwell CS, Cho YW, Kim YI (2015) Gabaergic inhibition is weakened or converted into excitation in the oxytocin and vasopressin neurons of the lactating rat. Molecular Brain 8(1):1–9
46. Lobo JL, Del Ser J, Bifet A, Kasabov N (2020) Spiking neural networks and online learning: an overview and perspectives. Neural Netw 121:88–100
47. Maass W (1997) Networks of spiking neurons: the third generation of neural network models. Neural Netw 10(9):1659–1671
48. Maass W (1997) Noisy spiking neurons with temporal coding have more computational power. In: Advances in neural information processing systems 9: Proceedings of the 1996 conference, vol 9. MIT Press, p 211
49. Maass W, Bishop CM (2001) Pulsed neural networks. MIT Press
50. McKennoch S, Liu D, Bushnell LG (2006) Fast modifications of the spikeprop algorithm. In: The 2006 IEEE international joint conference on neural network proceedings. IEEE, pp 3970–3977
51. Minneci F, Kanichay RT, Silver RA (2012) Estimation of the time course of neurotransmitter release at central synapses from the first latency of postsynaptic currents. J Neurosci Methods 205(1):49–64
52. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
53. Mohemmed A, Schliebs S, Matsuda S, Kasabov N (2011) Method for training a spiking neuron to associate input-output spike trains. In: Engineering applications of neural networks. Springer, pp 219–228
54. Mohemmed A, Schliebs S, Matsuda S, Kasabov N (2012) Span: Spike pattern association neuron for learning spatio-temporal spike patterns. Int J Neural Syst 22(04):1250012
55. Mohemmed A, Schliebs S, Matsuda S, Kasabov N (2013) Training spiking neural networks to associate spatio-temporal input-output spike patterns. Neurocomputing 107:3–10
56. Mostafa H (2017) Supervised learning based on temporal coding in spiking neural networks. IEEE Trans Neural Netw Learn Syst 29(7):3227–3235

57. Nair V, Hinton GE (2010) Rectified linear units improve restricted boltzmann machines. In: ICML
58. Natschläger T, Ruf B (1998) Spatial and temporal pattern analysis via spiking neurons. Netw: Comput Neural Syst 9(3):319–332
59. Parejo JA, Ruiz-Cortés A, Lozano S, Fernandez P (2012) Metaheuristic optimization frameworks: a survey and benchmarking. Soft Comput 16(3):527–561
60. Paugam-Moisy H, Bohte SM (2012) Computing with spiking neuron networks. Handb Nat Comput 1:1–47
61. Pei J, Deng L, Song S, Zhao M, Zhang Y, Wu S, Wang G, Zou Z, Wu Z, He W et al (2019) Towards artificial general intelligence with hybrid tianjic chip architecture. Nature 572(7767):106–111
62. Ponulak F, Kasiński A (2010) Supervised learning in spiking neural networks with resume: sequence learning, classification, and spike shifting. Neural Comput 22(2):467–510
63. Bi GQ (2002) Spatiotemporal specificity of synaptic plasticity: Cellular rules and mechanisms. Biol Cybern 87(5–6):319–332
64. Rigelsford J (2001) Control of movement for the physically disabled. Ind Robot: Int J
65. Rullen RV, Thorpe SJ (2001) Rate coding versus temporal order coding: what the retinal ganglion cells tell the visual cortex. Neural Comput 13(6):1255–1283
66. Saleh AY, Shamsuddin SM, Hamed HNA (2017) A hybrid differential evolution algorithm for parameter tuning of evolving spiking neural network. Int J Comput Vis Robot 7(1–2):20–34
67. Schliebs S, Kasabov N (2013) Evolving spiking neural network-a survey. Evol Syst 4(2):87–98
68. Shrestha SB, Song Q (2015) Adaptive learning rate of spikeprop based on weight convergence analysis. Neural Netw 63:185–198
69. Sjostrom PJ, Rancz EA, Roth A, Hausser M (2008) Dendritic excitability and synaptic plasticity. Physiol Rev 88(2):769–840
70. Sporea I, Grüning A (2013) Supervised learning in multilayer spiking neural networks. Neural Comput. 25(2):473–509
71. Stanic U, Davis R, General consideration in the clinical application of electrical stimulation. International FES Society web page. http://www.ifess.org
72. Stein RB (1965) A theoretical analysis of neuronal variability. Biophys J 5(2):173–194
73. Stein RB (1967) Some models of neuronal variability. Biophys J 7(1):37–68
74. Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359
75. Bonhoeffer T, Staiger V, Aertsen AM (1989) Synaptic plasticity in rat hippocampal slice cultures: Local hebbian conjunction of pre and postsynaptic stimulation leads to distributed synaptic enhancement. Proc Nat Acad Sci USA 86(20):8113–8117
76. Taherkhani A, Belatreche A, Li Y, Cosma G, Maguire LP, McGinnity TM (2020) A review of learning in biologically plausible spiking neural networks. Neural Netw 122:253–272
77. Taherkhani A, Belatreche A, Li Y, Maguire LP (2015) Dl-resume: a delay learning-based remote supervised method for spiking neurons. IEEE Trans Neural Netw Learn Syst 26(12):3137–3149
78. Taherkhani A, Belatreche A, Li Y, Maguire LP (2015) Edl: an extended delay learning based remote supervised method for spiking neurons. In: International conference on neural information processing, pp 190–197
79. Taherkhani A, Belatreche A, Li Y, Maguire LP (2015) Multi-dl-resume: Multiple neurons delay learning remote supervised method. In: 2015 international joint conference on neural networks (IJCNN), pp 1–7
80. Taherkhani A, Belatreche A, Li Y, Maguire LP (2018) A supervised learning algorithm for learning precise timing of multiple spikes in multilayer spiking neural networks. IEEE Trans Neural Netw Learn Syst 29(11):5394–5407
81. Taherkhani A, Belatreche A, Li Y, Maguire LP et al (2014) A new biologically plausible supervised learning method for spiking neurons. In: ESANN
82. Thorpe S, Delorme A, Rullen RV (2001) Spike-based strategies for rapid processing. Neural Netw 14(6):715–725. https://doi.org/10.1016/S0893-6080(01)00083-1

83. Thorpe S, Fize D, Marlot C (1996) Speed of processing in the human visual system. Nature 381(6582):520–522
84. Tsodyks MV, Markram H (1996) Plasticity of neocortical synapses enables transitions between rate and temporal coding. In: International conference on artificial neural networks. Springer, pp 445–450
85. Vazquez RA, Cachón A (2010) Integrate and fire neurons and their application in pattern recognition. In: 2010 7th international conference on electrical engineering computing science and automatic control. IEEE, pp 424–428
86. Vreeken J (2003) Spiking neural networks, an introduction
87. Hui-zhong WT, Zhang LI, Bi GQ, Poo MM (2000) Selective presynaptic propagation of long-term potentiation in defined neural networks. J Neurosci 20(9):3233–3243
88. Wade JJ, McDaid LJ, Santos JA, Sayers HM (2010) Swat: a spiking neural network training algorithm for classification problems. IEEE Trans Neural Netw 21(11):1817–1830
89. Wang J, Belatreche A, Maguire L, McGinnity TM (2014) An online supervised learning method for spiking neural networks with adaptive structure. Neurocomputing 144:526–536
90. Wang X, Lin X, Dang X (2020) Supervised learning in spiking neural networks: a review of algorithms and evaluations. Neural Netw 125:258–280. https://doi.org/10.1016/j.neunet.2020.02.011, https://www.sciencedirect.com/science/article/pii/S0893608020300563
91. Xu Y, Yang J, Zhong S (2017) An online supervised learning method based on gradient descent for spiking neurons. Neural Netw 93:7–20
92. Xu Y, Zeng X, Han L, Yang J (2013) A supervised multi-spike learning algorithm based on gradient descent for spiking neural networks. Neural Netw 43:99–113
93. Yu Q, Tang H, Tan KC, Yu H (2014) A brain-inspired spiking neural network model with temporal encoding and learning. Neurocomputing 138:3–13