Neural Networks 122 (2020) 253–272

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

A review of learning in biologically plausible spiking neural networks

Aboozar Taherkhani [a,*], Ammar Belatreche [b], Yuhua Li [c], Georgina Cosma [d], Liam P. Maguire [e], T.M. McGinnity [e,f]

[a] School of Computer Science and Informatics, Faculty of Computing, Engineering and Media, De Montfort University, Leicester, UK
[b] Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne, UK
[c] School of Computer Science and Informatics, Cardiff University, Cardiff, UK
[d] Department of Computer Science, Loughborough University, Loughborough, UK
[e] Intelligent Systems Research Centre, Ulster University, Northern Ireland, Derry, UK
[f] School of Science and Technology, Nottingham Trent University, Nottingham, UK

ARTICLE INFO

Article history:
Received 28 February 2019
Received in revised form 17 September 2019
Accepted 23 September 2019
Available online 11 October 2019

Keywords:
Spiking neural network (SNN)
Learning
Synaptic plasticity

ABSTRACT

Artificial neural networks have been used as a powerful processing tool in various areas such as pattern recognition, control, robotics, and bioinformatics. Their wide applicability has encouraged researchers to improve artificial neural networks by investigating the biological brain. Neurological research has significantly progressed in recent years and continues to reveal new characteristics of biological neurons. New technologies can now capture temporal changes in the internal activity of the brain in more detail and help clarify the relationship between brain activity and the perception of a given stimulus. This new knowledge has led to a new type of artificial neural network, the Spiking Neural Network (SNN), that draws more faithfully on biological properties to provide higher processing abilities. A review of recent developments in learning of spiking neurons is presented in this paper. First the biological background of SNN learning algorithms is reviewed. The important elements of a learning algorithm such as the neuron model, synaptic plasticity, information encoding and SNN topologies are then presented. Then, a critical review of the state-of-the-art learning algorithms for SNNs using single and multiple spikes is presented. Additionally, deep spiking neural networks are reviewed, and challenges and opportunities in the SNN field are discussed.

1. Introduction

The human brain is a very complex system and is constructed of approximately 90 billion neurons (Azevedo et al., 2009). It is structurally organized by trillions of interconnected synapses. Information is transferred between neurons by electrical impulses called spikes. The effect of a spike, which is sent by a presynaptic neuron to a receiving neuron, depends on the strength of the synapse that connects the two neurons. The synaptic strengths and the connection pattern between neurons have a significant role in the information processing capability of nervous systems. The brain's processing ability to solve complex problems has inspired many researchers to investigate its processing function and learning mechanisms. Artificial Neural Networks (ANNs), as a powerful and flexible computing means to solve complex problems, have emerged as a result of this research on the brain's processing functionality.

ANNs are inspired by the biological nervous system and are successfully used in various applications (Hinton et al., 2012; Hinton, Osindero, & Teh, 2006; Hinton & Salakhutdinov, 2006). However, their high abstraction compared to their biological counterpart (Pham, Packianather, & Charles, 2008) and their inability to capture the complex temporal dynamics of biological neurons have resulted in a new area of ANNs where the focus is placed on more biologically plausible neuronal models known as Spiking Neural Networks (SNNs). Thanks to their ability to capture the rich dynamics of biological neurons and to represent and integrate different information dimensions such as time, frequency, and phase, SNNs offer a promising computing paradigm and are potentially capable of modelling complex information processing in the brain (Brette et al., 2007; Gerstner & Kistler, 2002; Hodgkin & Huxley, 1952; Izhikevich, 2004, 2006; Kasabov, Dhoble, Nuntalid, & Indiveri, 2013; Maass & Zador, 1999). SNNs are also potentially capable of dealing with large volumes of data and using trains of spikes for information representation (Kasabov et al., 2013). Additionally, SNNs are suitable for implementation on low power hardware.

It is broadly agreed that spikes (a.k.a pulses or action potentials), which represent short and sudden increases in the voltage of a neuron, are used to transfer information between neurons (Gerstner & Kistler, 2002). The encoding of information through

* Corresponding author.
E-mail address: aboozar.taherkhani@dmu.ac.uk (A. Taherkhani).

https://doi.org/10.1016/j.neunet.2019.09.036
0893-6080/

spikes is still a matter of debate in the computational neuroscience community. Previously, it was supposed that the brain encodes information through spike rates (Masquelier & Deco, 2013). However, neurobiological research findings have shown high speed processing in the brain that cannot be performed by a rate coding scheme alone (Brette, 2015). It has been shown that human visual processing can perform a recognition task in less than 100 ms by using neurons in multiple layers (from the retina to the temporal lobe). It takes about 10 ms processing time for each neuron. The time-window is thus too small for rate coding to occur (Thorpe, Delorme, & Van Rullen, 2001; Vreeken, 2003). The rapid information processing in the electro sensory system of electric fish (Heiligenberg, 1991) and in the auditory system of echo-locating bats (Kuwabara & Suga, 1993) are other examples of high speed information processing in biological nervous systems. High speed processing tasks can be performed using precise timing of spikes (Vreeken, 2003). Additionally, the firing of so many spikes in rate coding of a stimulus demands considerable energy and resources. Moreover, the precise timing of spikes has a higher information encoding capacity in a small set of spiking neurons (Paugam-Moisy & Bohte, 2012). Therefore, it seems clear that the precise timing of individual spikes, and not just the number of spikes or firing rate, is likely to convey information.

However, the exact learning mechanism in which a neuron is trained is an open question. Recently, biologists have found various forms of biological synaptic plasticity, which are governed by spikes (Feldman, 2012). These various forms of synaptic weight and delay learning (Lin & Faber, 2002) are compatible with the spiking neuron model, whereas there is considerable difficulty for their application in traditional models.

The activity of a biological neural system can be studied at various scales, levels and perspectives, for example genes and molecules, single-cell electrophysiology, multi neuron recordings, and cognitive neuroscience and psychophysics. Simulation and mathematical theories are used in the literature to link the various levels. In the bottom-up approach of investigating the biological nervous system, the knowledge of lower levels (such as properties of ion channels) is used to describe the higher level phenomena such as the generation of an action potential or memory formation. Hodgkin and Huxley's model of a biological neuron is an example of a bottom-up description of a neuron. In the biophysical neuron model, the properties of ion channels with different time constant and different dynamics in a cell membrane are modelled (Gerstner, Sprekeler, & Deco, 2012). The activity of a biological neuron system can be investigated in highly extended levels.

In this paper, the review starts from the level of a single neuron and then progresses to biologically plausible learning algorithms for a single neuron as well as populations of neurons. First the biological background of SNN learning algorithms is reviewed. The important elements of a learning algorithm such as the spiking neuron model, synaptic plasticity, information encoding and SNN topologies are then studied. Subsequently, state of the art learning algorithms for SNNs are reviewed. Finally, challenges and opportunities in the SNN field are discussed.

2. Biological background

Neurons represent the elementary processing units of the brain. They communicate by sending and receiving action potentials (Gerstner & Kistler, 2002). Neurons are connected to each other, through synapses, in an intricate pattern making specific structures. A review of the literature shows that the important considerations in the design of a learning algorithm for SNNs are as follows: neuron models, communication through synapses, the topology of the network, and the information encoding/decoding schemes. The following review discusses the impact of these aspects.

2.1. Spiking neuron models

In 1952, Hodgkin and Huxley (1952) performed experiments on the giant axon of the squid and built a four dimensional (4D) detailed conductance-based neuron model which can reproduce electrophysiological measurements. However, the intrinsic computational complexity of this model increases its computational cost. Consequently, more simple, phenomenological spiking neuron models are used for simulating large scale SNNs, neural coding and memory (Gerstner & Kistler, 2002). The biological plausibility and the implementation cost of various spiking neuron models are compared in Izhikevich (2004). The Leaky Integrate-and-Fire (LIF) model (Koch & Segev, 1998) and the Spike Response Model (SRM) (Gerstner, Kistler, Naud, & Paninski, 2014) are two popular 1D spiking neural models with low computational cost, but they offer poor biological plausibility compared with the Hodgkin and Huxley model. The 2D model of Izhikevich (Izhikevich, 2003) offers a good trade-off between biological plausibility and computational efficiency. Although it can produce various spiking dynamics, many of these characteristics such as Chaos and Bi-stability have not been used in current learning algorithms.

According to biological evidence, a neuron can operate as an integrator or Coincidence Detector (CD) (König, Engel, & Singer, 1996). In an integrator model the neuron integrates incoming Post Synaptic Potentials (PSP) in a longer time interval than in a CD. The integrator model takes advantage of the effect of not only input spikes with short inter spike intervals, but also input spikes that are relatively far from each other. However CDs use the effect of the input spikes that are near to each other (i.e. have short inter spike intervals) to generate the neuron total PSP (König et al., 1996). Learning algorithms that use an integrator model focus on synaptic weight plasticity, however, the CD learning algorithm exploits synaptic delay modulation to learn (Pham et al., 2008). Thus, there is potential to improve biological plausibility and computational ability of SNN learning algorithms by adjusting both synaptic weights and delays to construct new learning algorithms.

Leaky Integrate-and-Fire (LIF) neuron model: Detailed conductance-based neuron models (Hodgkin & Huxley, 1952) can reproduce electrophysiological signals to a high degree of accuracy, but they are computationally complex. Simple phenomenological spiking neuron models with low computational cost are highly popular for studies of neural coding, memory, and network dynamics. The LIF is a one dimensional spiking neural model with low computation cost (Gerstner et al., 2014), that is commonly adopted in the literature. The sub threshold dynamics of the LIF neuron are defined by the following equation:

$$\tau_m \frac{dv_m(t)}{dt} = -(v_m(t) - E_r) + R_m I(t) \tag{2.1}$$

where $v_m(t)$ is the membrane potential, τ_m is the membrane time constant, E_r is the membrane rest potential which is a constant, R_m is the membrane resistance, and $I(t)$ is the sum of the current supplied by the input synapses. $I(t)$ is calculated by the following equation:

$$I(t) = W \cdot S(t) \tag{2.2}$$

where $W = [w_1, w_2, \ldots, w_N]$ is the weight vector. $S(t) = [s_1(t); s_2(t); \ldots; s_N(t)]$ is the spatiotemporal input spike pattern containing N input spike trains, $s_i(t)$ *for* $i = 1, 2, \ldots, N$.

$$s_i(t) = \sum_f \delta(t - t_i^f) \tag{2.3}$$

where t_i^f is the firing time of the fth ($f = 1, 2, \ldots$) spike in the ith input spike train, $s_i(t)$. $s_i(t)$ is applied to the ith synapse, and $\delta(t)$ is a Dirac function.

When the membrane voltage, $v_m(t)$, reaches the threshold level, V_{th}, an output spike is generated, and the membrane voltage resets to the rest potential, E_r, and stays at the resting level for period of time t_{ref}, called the refractory period.

2.2. Synaptic plasticity

Synaptic plasticity (i.e. change in synaptic efficacy) is considered to be the biological underpinning of learning and memory. The exact relationship between microscopic synaptic properties and macroscopic functional consequences remains highly controversial (Morrison, Diesmann, & Gerstner, 2008). Unsupervised, supervised, and reinforcement learning are the three known types of learning strategies and the following details these approaches with a focus on synaptic plasticity. However, there is also biological evidence that the synaptic delay is not always constant and can be modulated during synaptic plasticity (Lin & Faber, 2002).

2.2.1. Unsupervised learning

Unsupervised learning is progressed according to local events, and the local events do not have any notion of the task to be solved, and also they do not have any notion of the change being 'good' or 'bad'. Learning simply involves an adaptation according to local activity. Hebb's in 1949 postulate, which describes how synaptic connections should be modified, has inspired many unsupervised approaches (Morrison et al., 2008). Unsupervised learning may be constructed from a combination of the following: (a) spontaneous growth or decay of weights in the absence of any activity (Turrigiano & Nelson, 2004); (b) effects caused by postsynaptic spikes alone independent of presynaptic spike arrival (Artola, Brocher, & Singer, 1990); (c) effects caused by presynaptic spikes, independent of postsynaptic variables — the case for short-term synaptic plasticity (Vasilaki & Giugliano, 2013); (d) effects caused by presynaptic spikes in conjunction with postsynaptic spikes or in conjunction with postsynaptic depolarization (Hebbian terms) (Clopath, Büsing, Vasilaki, & Gerstner, 2010); (e) all of the above effects may depend on the current value of the synaptic weight (Morrison et al., 2008), e.g. close to a maximum weight synaptic changes could become smaller (Morrison et al., 2008). Although Hebbian plasticity is used in the current SNN learning algorithms, other synaptic plasticity such as short-term plasticity is less used.

Spike Timing Dependent Plasticity (STDP) is a variant of the Hebbian unsupervised learning algorithm. This rule is proposed to describe the changes of a synaptic weight according to the relative timing of pre and postsynaptic spikes. According to STDP, a synaptic weight is potentiated if a presynaptic spike comes shortly before a postsynaptic spike. If the time interval between the pre- and postsynaptic spike is $t = t_{post} - t_{pre}$ and $t > 0$ then the synaptic weight will be potentiated. The magnitude of the potentiation is a function of t which decays exponentially with a time constant τ_+ and can be calculated by A_+e^{-t/τ_+}, where A_+ is the maximum synaptic change. Alternatively, if a pre synaptic spike occurs shortly after the postsynaptic spike, then the synaptic efficacy is decreased. The magnitude of the decrease can be calculated by A_-e^{t/τ_-}, where A_- represents the maximum depression, and τ_- is a time constant. Fig. 1 shows the synaptic changes as a function of time interval t. The shape of an STDP function does not have to be fixed across a network and different synapses can have differing shapes (parameters) for this function. According to physiological evidence, the generality of STDP is debated because the order of pre- and post-synaptic spikes is only important in some situations, depending on the presynaptic activity such as firing rate (Tetzlaff, Kolodziejski, Markelic, & Wörgötter, 2012).

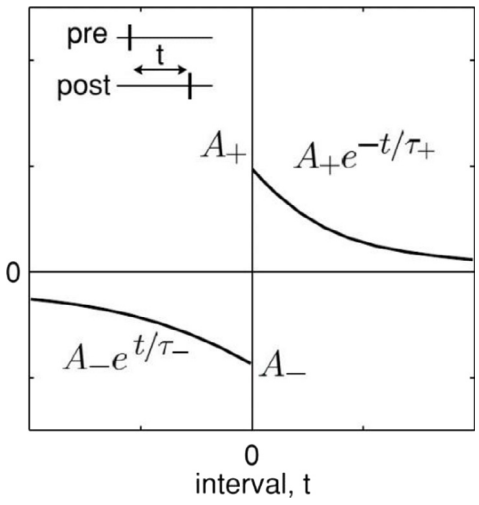

Fig. 1. STDP learning time window. If the post neuron fired after the presynaptic spike, the weight of synaptic connection from pre- to postsynaptic neuron is increased. The magnitude of change decreases as $\boldsymbol{A_+e^{-t/\tau_+}}$. Reverse order results in a decrease of the synaptic weight with magnitude $\boldsymbol{A_-e^{t/\tau_-}}$.
Source: The figure is adapted from González-Nalda (2009).

Biological experiments show that the standard pair-based STDP models (i.e. pre-before-post and post-before-pre) cannot give a full description of STDP in a biological neuron. There are experimental researches that investigate multiple-spike protocols (Morrison et al., 2008). Symmetric triplets STDP in the form of pre-post-pre and post-pre-post, which are a simple example of multiple-spike STDP, are investigated experimentally in Bi and Wang (2002), Froemke and Dan (2002), Froemke, Tsay, Raad, Long, and Dan (2006) and Wang, Gerkin, Nauen, and Bi (2005). There are different multiple-spike STDP models that are developed to predict the experimental results (Morrison et al., 2008). One simple triplet STDP model is developed in Pfister and Gerstner (2006).

Pfister and Gerstner (2006) implemented a triplet STDP with local variables called traces. The trace related to a presynaptic neuron j is shown by $x_j(t)$, and the post synaptic neuron 'i' corresponds to two traces $y_i^1(t)$ and $y_i^2(t)$ with fast and slow dynamics respectively ($\tau_1 < \tau_2$) as shown in Fig. 2 (Morrison et al., 2008). The LTD in the triplet STDP is similar to standard paired based STDP. A weight is depressed in proportion to the fast postsynaptic trace, $y_i^1(t)$ as shown by unfilled circles in Fig. 2. At the moment of a postsynaptic spike LTP is induced in proportion to the presynaptic trace $x_j(t)$ (similar to the standard pair based STDP) and the slow postsynaptic trace $y_i^2(t)$ as shown in Fig. 2 by filled circles.

The ability of the multiple spike STDP to model the synaptic plasticity of a biological neuron shows that the learning algorithms that work based on multiple spikes are more biologically plausible and multiple spike coding can be an appropriate choice for modelling the information processing in the brain.

Local dendritic depolarization related STDP: Several recent studies have investigated how a synapse location within the dendritic tree influences STDP. Dendritic mechanisms can produce different learning rules in different dendritic domains of the same neuron (Kampa, Letzkus, & Stuart, 2007; Letzkus, Kampa, & Stuart, 2006). Nearby synapses within dendritic branches contribute to local associative plasticity. Therefore, local dendritic depolarization is the main tool to manage the plasticity (Feldman, 2012). In other words local dendritic PSPs can change the STDP

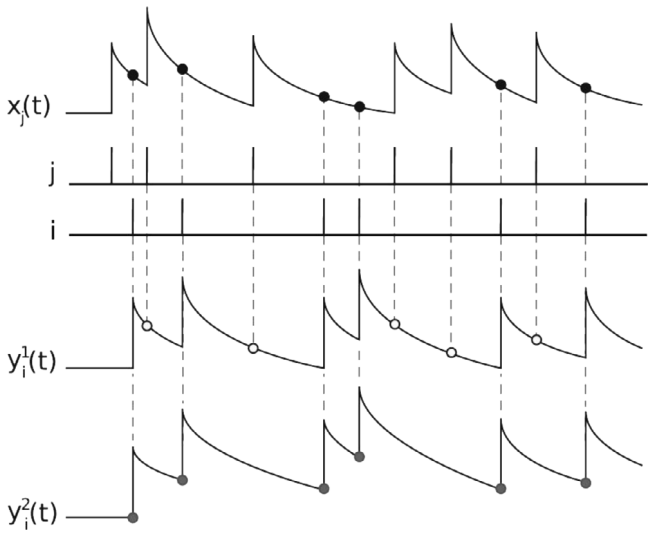

Fig. 2. Triplet STDP is governed by a trace corresponding to presynaptic spikes, $x_j(t)$, and the two fast ($y_i^1(t)$) and slow ($y_i^2(t)$) traces related to postsynaptic spikes. LTD (Long Term Depression) takes place at the time of a presynaptic spike in proportion to the momentary value of $y_i^1(t)$ as shown by unfilled circles. A synaptic weight is potentiated at the time of a postsynaptic spike in proportion to the momentary value of presynaptic trace ($x_j(t)$) and value of the slow trace $y_i^2(t)$ as shown by black filled circles.
Source: The figure is adapted from Morrison et al. (2008)

characteristic. This change can be in various forms. For example, anti-Hebbian LTD on cortical pyramidal cells is converted into Hebbian STDP depending on the location of the plasticity within the dendrite tree. The association of local dendritic depolarization and STDP may be useful in improving the information processing ability of neurons by specifying different synapses for different types of input information and by providing dynamic control over plasticity. Spike timing will be a main factor of plasticity in some circumstances; however, it will not be a prominent factor in others (Feldman, 2012). STDP has been used to design learning algorithms for spiking neural networks (Ponulak & Kasiński, 2010; Srinivasa & Cho, 2012). However, different biological characteristics of STDP are not considered in the learning algorithm, and it seems that the consideration of the new biological property of STDP (Local dendritic depolarization related STDP) might contribute to the design of a new learning algorithm which is more biologically plausible and has new interesting computational characteristics.

2.2.2. Supervised learning

There is evidence that confirms the existence of supervised or instruction-based learning in the brain (Carey, Medina, & Lisberger, 2005; Doya, 1999; Ito, 2000; Knudsen, 2002). One form of the supervised learning is governed by an instruction signal. It is believed that these signals are provided in the learning modules by sensory feedback (Carey et al., 2005; Knudsen, 2002) or by other neuronal assemblies (Doya, 1999; Ito, 2000). However, the exact mechanism of supervised learning in the brain is not clear (Sporea & Grüning, 2013). The cerebellum is thought to be the primary site for supervised learning in the brain (Jörntell & Hansel, 2006; Ponulak & Kasinski, 2011). Supervised learning at the level of a neuron has been demonstrated experimentally by Fregnac and Shulz (1999). Naturally, a few inputs to strong synapses can drive a neuron response and therefore drive learning of other input synapses. If these strong inputs are controlled for a target-specific task, they act as a teacher for the postsynaptic neuron.

2.2.3. Reinforcement learning

Behaviours are learnt not only through direct instructions, but more often by exploring available actions in the presence of reward signals. In reward-based or reinforcement learning the direction and amount of change of the learning free parameters depends on the presence or absence of a success signal (Schultz, Dayan, & Montague, 1997). Reinforcement learning initially is understood by psychological evidence. Recently, it has been shown that the concentration of dopamine which can act as a reward signal affects the synaptic change in various parts of the brain. Dopamine is a neuromodulator which is emitted by dopaminergic cells (Ponulak & Kasinski, 2011).

2.2.4. Delay learning

The existence of synaptic delay in the mammalian neocortex is proven by experimental research and depending on the type and location of the neurons, the synaptic delay could be as short as 0.1 ms and as long as 40 ms (Izhikevich, 2006; Paugam-Moisy, Martinez, & Bengio, 2008; Swadlow, 1992). The effect of the time delay on the processing ability of the nervous system is well established (Gilson, Bürck, Burkitt, & van Hemmen, 2012; Glackin, Wall, McGinnity, Maguire, & McDaid, 2010; Xu, Gong, & Wang, 2013; Xu, Zeng, et al., 2013). For instance, it has been shown that delays have an important role in the mammalian auditory pathway for perception of sound localization (Glackin et al., 2010). Additionally, according to biological evidence the synaptic delay can be changed according to input and output spikes (Lin & Faber, 2002). This biological evidence of delay learning provides an inspiration in developing new learning algorithms that exploit delay learning in addition to the usual weight learning.

2.3. Information encoding

How neurons encode information using spikes is one of the important questions discussed in neuroscience. It is assumed that neural information is conveyed either in the firing rate, or in the precise timing of spikes (temporal coding). Various forms of firing rate encoding exist, such as spike count, spike density, or population activity (Gerstner & Kistler, 2002). Although rate coding is commonly used in traditional ANNs, such an approach may not convey all the information related to a rapid processing task such as colour, visual information, odour and sound quality processing as the information encapsulated in the precise timing of spikes is ignored (Cariani, 2004; Hopfield, 1995; Mohemmed, Schliebs, Matsuda, & Kasabov, 2013).

Examples of temporal coding methods include time to first spike, Rank-Order Coding (ROC) (Rullen & Thorpe, 2001), latency code, phase coding, coding by synchrony (Ponulak & Kasinski, 2011), and polychronisation (which is a group of neurons time-locked to fire in various precise times (Izhikevich, 2006)). Yu, Tang, Tan, and Li (2013b) have shown that coding based on precise timing of spikes can convey more information than ROC which ignores the time differences between spikes. Additionally, ROC is more sensitive to noise, because, the rank of each spike is dependent on the rank of other spikes. If the rank of a single spike is changed as a result of a small noise component, then the ranks of the other spikes are subsequently changed. Consequently, the resulting pattern is completely different from the original pattern (Yu et al., 2013b).

2.4. SNN topologies

A common classification of SNN topologies considers three types of topologies, namely feed-forward, recurrent and hybrid networks. Synfire chain and fault-tolerant SNN proposed in Srinivasa and Cho (2012) are two examples of hybrid networks where

some subpopulations may be strictly feed-forward while others may have recurrent topologies.

According to statistical analysis, a cat's cerebral cortex structure can be considered a clustered network (Lameu et al., 2012). Each cluster is a scale-free network with highly connected hubs. Those hubs are strongly connected together, i.e. a high number of neurons in different hubs are connected (Lameu et al., 2012). Hazan and Manevitz (2012) have shown that specifying certain kinds of topological constraints, which have been claimed as reasonably biologically plausible, can restore robustness in SNNs (Hazan & Manevitz, 2012).

It is well established that the topology of SNNs in a brain dynamically changes during the learning process. Bassett et al. (2012) have shown that primary sensorimotor and visual regions have a relatively stiff core that changes little over time but they have flexible periphery regions which change more frequently (Bassett et al., 2012). Evolving spiking neural network (eSNN) (Belatreche, Maguire, & McGinnity, 2007; Wysoski, Benuskova, & Kasabov, 2008), dynamic evolving SNN (deSNN) (Kasabov et al., 2013), dynamic cluster formation using populations of spiking neurons (Belatreche & Paul, 2012), and the online supervised learning method with adaptive structure in Wang, Belatreche, Maguire, and McGinnity (2014) are examples of SNNs with dynamic topology. The evolving structure of SNNs enhances their processing ability as well as improving their biological plausibility.

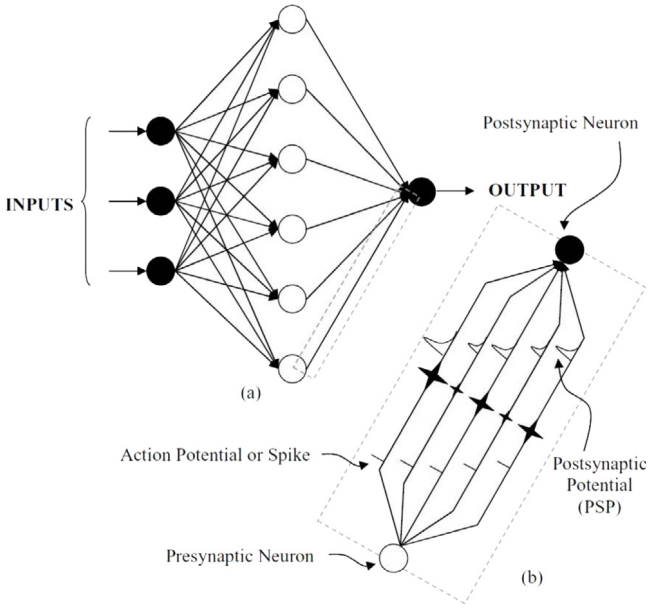

Fig. 3. (a) The Spiking neural network architecture used in Ghosh-Dastidar and Adeli (2007), McKennoch et al. (2006) and Silva and Ruano (2005); (b) each neuron is connected to the next neuron by multiple synapses with different delays.
Source: The figure is adapted from Ghosh-Dastidar and Adeli (2007).

3. Review of some state of the art learning algorithms for SNNs

In the previous section some biologically plausible elements that might be used to construct spiking neural network learning algorithms were introduced. This section presents a critical review of state of the art learning algorithms for spiking neurons. First, the learning algorithms that can train each neuron to fire a single spike are reviewed. Then, the learning algorithms that train a single neuron or a single layer of neurons to learn multiple spikes are discussed. After that, learning of multiple spikes in a multilayer spiking neural network are reviewed. The delay learning ability of spiking neuron is discussed in Section 3.4. Finally, recent Deep Spiking Neural Networks are reviewed.

3.1. Learning a single spike per neuron

SpikeProp (Bohte, Kok, & La Poutre, 2002) is one of the first supervised learning methods for spiking neurons. SpikeProp was inspired by the classical backpropagation algorithm. SpikeProp is a multilayer spiking neural network, and it is applied successfully to classification problems. In the network, two neurons are connected through multiple connections with different weights and delays (see Fig. 3). SpikeProp, much like other gradient-based methods, is based on the estimation of the gradient of an error function and thus has local minima problems. In addition, silent neurons or non-firing neurons are another problem which prevents the calculation of the gradient.

Back-propagation with momentum (McKennoch, Liu, & Bushnell, 2006; Xin & Embrechts, 2001), QuickProp (McKennoch et al., 2006; Xin & Embrechts, 2001), Resilient propagation (RProp) (Ghosh-Dastidar & Adeli, 2007; McKennoch et al., 2006; Silva & Ruano, 2005) Levenberg–Marquardt BP (Silva & Ruano, 2005), and the SpikeProp based on adaptive learning rate (Shrestha & Song, 2015) are various learning algorithms proposed to improve the performance of SpikeProp. In the first supervised learning algorithms for multilayer spiking neural networks for learning the precise timing, each neuron is trained to fire only a single spike. In the mentioned learning methods all neurons in the input, output and hidden layers can only fire a single spike. The mentioned learning algorithms depend on the neuron model used in the network.

Belatreche, Maguire, McGinnity, and Wu (2003) used evolutionary strategies to train both synaptic weights and delays for classification tasks. The proposed approach has good performance when compared to Spike-Prop, yet the training algorithm is time consuming.

In Pham et al. (2008) a self-organizing spiking neural network is designed for pattern clustering. The spike response neuron model with constant weights (which are chosen randomly) is used as a CD. The parameter of the CD is chosen so that the CD fires if its input synapses have spikes close to each other. The output layer is constructed of a two-dimensional grid. A CD is considered for each node in the grid. The number of neurons in the input layer is equal to the dimension of the input pattern. A Hebbian based rule is used to shift the synaptic delays. In other words, the synaptic delays are the learning parameters, and the proposed SOM (Self-organizing map) adjusts the synaptic delay of the winner neuron and the neurons near to the winner during adaptation stage. The Hebbian based learning is designed to adjust the synaptic delay so that the peaks of the input spike responses coincide with each other which cause the receiving neuron to fire. In the model only the first spike of an output is considered. The neuron threshold level is set to a small value at the beginning of the learning and is then increased during different learning epochs. The CDs with a high degree of coincidence can fire in response to a specific input. During learning, an input pattern is applied to the network and the neuron that fires first is considered as the winning neuron. Then the neurons that are in a neighbourhood of the winner are considered to contribute in a Hebbian learning process. The contribution of a neuron in the learning process is a function of its spatial distance from the winner. The function has a Gaussian shape and the winner is the centre of the function. The training procedure is stopped when the total change of the connection delays falls below a minimum value, or the delay change remains constant.

The SNN approach proposed in Pham et al. (2008) can be compared to the properties of a traditional SOM (Kohonen, 2013). In a basic SOM the neighbourhood Gaussian function has a time varying width which is decreased by the learning epochs. This shrinking property of the neighbourhood function is not used in the SNN approach proposed in Pham et al. (2008). According to this property the first stage of learning should cover a large number of neurons around the winner and as the learning progresses it is decreased. The appropriate neurons are chosen to become more selective to the specific input pattern by this method. Furthermore, in the SOM for SNNs a simple learning algorithm based on the delay shift is used, however to date there are various learning algorithms for SNN based on the weights learning such as the algorithms proposed in Bohte et al. (2002) and Mohemmed, Schliebs, Matsuda, and Kasabov (2012) that can be used to design SOM with higher computation ability than the computation ability of the algorithm mentioned in Pham et al. (2008).

Wysoski et al. (2008) proposed an evolving Spiking Neural Network (eSNN) by using a Hebbian-based training. eSNN changes its structure in order to respond optimally to different input visual patterns. It uses hierarchical layers and can be used for online learning applications. Rank order coding (ROC) and a simplified type of integrate-and-fire neuron model are used in eSNN. The algorithm only uses the information in the first spike of each input synapse and it ignores the information that is in the following spikes. In eSNN learning procedure each neuron can fire a single spike. A latency code is used to convert a real value related to input pattern to temporal information in the input spikes. Despite the application of the latency code, it uses a neuron model that works based on order of the input spikes and the precise time of input spikes is not too important for the neuron. Additionally, two different input patterns generated by the latency code can have same order of input spikes; however, they can have completely different temporal patterns relating to different classes. The other problem of eSNN is that it works only based on the first spike and the effect of the other input spikes is not reflected on the synaptic weights adjustment. So the network can only work on the single spike per input and it cannot capture the information related to spatiotemporal input pattern with multiple spikes in each input spike train. Although eSNN is composed of four layers, learning is only taking place in a single layer (layer L3 in Fig. 4).

In Kasabov et al. (2013) a dynamic eSNN (deSNN) was proposed to capture the information in more complex spatiotemporal input patterns composed of multiple spikes. deSNN (Kasabov et al., 2013) uses rank-order learning (similar to eSNN Wysoski et al., 2008) as well as the spike driven synaptic plasticity (SDSP) learning rule. deSNN initializes the input weights based on the rank of the first input spikes. The initial value of each synaptic weight is adjusted based on SDSP. SDSP adjusts each weight by using the other input spikes that come after the first one. A weight is potentiated when it receives an input spike, and it is depressed in each time step that it does not receive an input spike. By this strategy the information of the first spike as well as the following ones are captured (Kasabov et al., 2013).

The neuron model that is used in eSNN (Wysoski et al., 2008) and deSNN (Kasabov et al., 2013) is different from the standard biologically plausible neuron model such as LIF or SRM. The model works on the order of input spikes and it does not have a leakage in the generated PSP. The leakage can be an essential property of a biological neuron that is sensitive to the time interval and consequently it can be sensitive to temporal information inside the spatiotemporal input pattern. Although, each synaptic input in deSNN can have multiple spikes, the output can generate a single spike.

The tempotron (Gütig & Sompolinsky, 2006), a neuron with supervised learning ability, can learn to separate two different classes. Tempotron learns to spike in response to '+' patterns and to be silent in response to '−' (there are two classes, and the two classes are shown by '+' and '−'). During training, if no output spike was elicited in response to a '+' pattern, each synaptic efficacy is increased. Conversely, if an output spike appears in response to a '−' pattern the synaptic efficacies are decreased. Although this model is considered to be biologically plausible, its processing ability is restricted to binary classification. Furthermore, the scalability aspect was not discussed and it is not clear how this method can be used to process real world datasets.

Yu, Tan, and Tang (2012), Yu et al. (2013b) and Yu, Tang, Tan, and Yu (2014) used Tempotron for pattern recognition and a latency code is used to encode input patterns. A three layer structure including encoding neurons, tempotron and a readout layer, is used. However, it has only a single learning layer in which each neuron is trained based on Tempotron. Each neuron can only fire once within the encoding window.

The learning algorithms mentioned in this section can train neurons to fire only a single spike in response to a set of inputs within a simulation time window. Some of them can learn spatiotemporal input patterns with multiple input spikes per input synapse.

3.2. Learning multiple spikes in a single neuron or a single layer of neurons

Multiple spikes significantly increase the richness of the neural information representation (Borst & Theunissen, 1999; Ponulak & Kasiński, 2010). Single-spike coding schemes limit the diversity and capacity of information transmission in a network of spiking neurons (Xu, Zeng, et al., 2013). Moreover, training a neuron to fire multiple spikes is more biologically plausible compared to a single-spike learning scheme (Sporea & Grüning, 2013; Xu, Gong, & Wang, 2013). Temporal encoding through multiple spikes transfers important information in biological neural assemblies and the information cannot be expressed by a single spike coding scheme or a rate coding scheme. Although the exact mechanisms of biological coding schemes in the brain are not well understood, the biological evidence shows that multiple spikes coding schemes have a pivotal role in the brain. For example, in the neuronal circuits of zebra fish brain, spatiotemporal spiking sensory inputs composed of spike trains are mapped to precise timing of spikes to execute well-timed motor sequences (Memmesheimer, Rubin, Ölveczky, & Sompolinsky, 2014). This biological evidence has motivated the development of learning algorithms for spiking neurons to fire multiple spikes with precise timings (Xu, Gong, & Wang, 2013).

Supervised Hebbian learning rules were used in Legenstein, Naeger, and Maass (2005) and Ruf and Schmitt (1997) for learning temporal patterns. Although this approach has interesting properties such as locality, scalability and the ability of on-line processing, the authors indicated that convergence cannot be guaranteed in a general case. Another problem with all supervised Hebbian methods is continuous synaptic change even if the neuron fires exactly at the right times.

Statistical methods optimize the weights in order to maximize the likelihood of getting postsynaptic spikes at the desired times (Pfister, Toyoizumi, Barber, & Gerstner, 2006). However, it is difficult to learn complex spike trains using statistical methods. The proposed method in Pfister et al. (2006) has not been applied to real world data processing.

ReSuMe (Ponulak, 2005; Ponulak & Kasiński, 2010) is another supervised learning algorithm that is based on a combination of STDP and anti-STDP learning windows to produce multiple

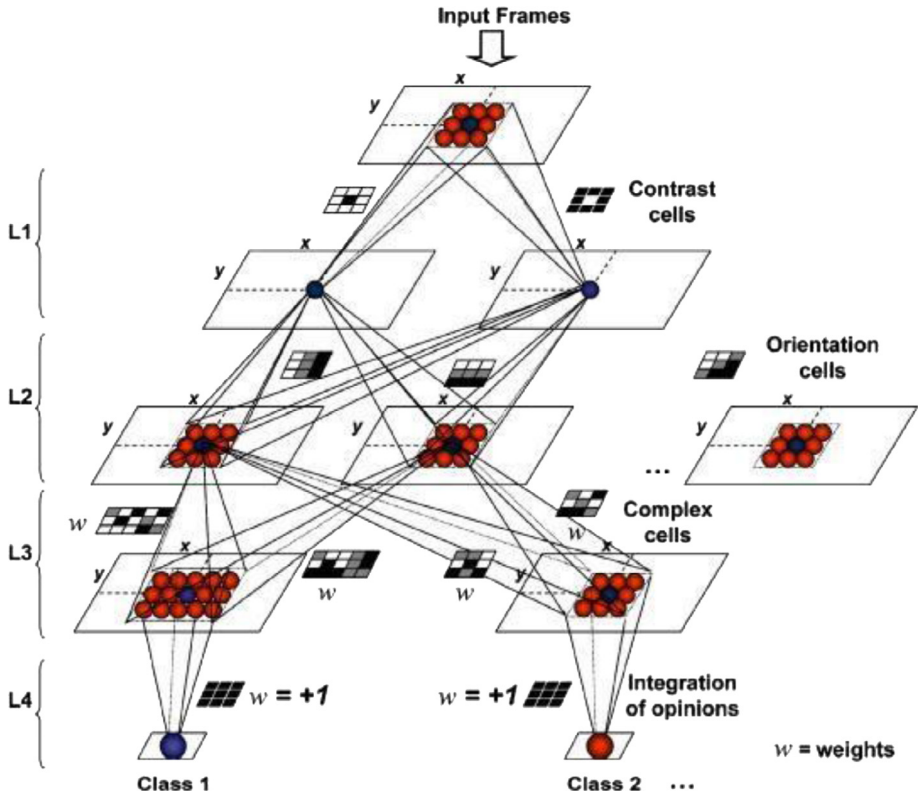

Fig. 4. Although eSNN (Wysoski et al., 2008) has a four-layer architecture, only the synaptic weights of the neurons in the third layer (L3) are trained and the other layers have constant synaptic weights.
Source: The figure is adapted from Wysoski et al. (2008)

desired output spikes. It is a biologically plausible supervised learning algorithm that is designed to produce a desirable output spike train in response to a spatiotemporal input spike pattern. The input and output spike sequences are generated randomly and the LIF neuron model is used in Ponulak (2005) and Ponulak and Kasiński (2010), however the learning algorithm does not depend on the model of spiking neuron. ReSuMe is based on the Widrow-Hoff rule which comes from a simple derivation of an error function in classical neural networks. The STDP is driven by using a remote teacher spike train to enhance appropriate synaptic weights to force the neuron to fire at desired times. Using remote supervised teacher spikes enables ReSuMe to overcome the silent neuron problem existing in the gradient based learning methods such as SpikeProp. The ability of online processing and locality are two remarkable properties of ReSuMe. The capabilities of ReSuMe have inspired research into new learning algorithms (Florian, 2012; Glackin, Maguire, McDaid, & Sayers, 2011; Mohemmed et al., 2013; Sporea & Grüning, 2013). ReSuMe works based on weight adjustment, and it is a well-known learning method.

Ponulak and Kasiński (2010) proposed ReSuMe to adjust the synaptic weights of a neuron to generate a desired spike train, $s_d(t)$, in response to a spatiotemporal input spike pattern $S(t) = [s_1(t); s_2(t); \ldots; s_N(t)]$. ReSuMe weight adjustment is based on the input, actual output and desired output spike trains. Ponulak incorporated precise spike times in the Widrow-Hoff rule and employed STDP and anti-STDP windows to adjust synaptic weights and enable supervised learning according to (3.4).

$$\frac{dw_i(t)}{dt} = [s_d(t) - s_o(t)][a + \int_0^{+\infty} Tw(s) s_i(t-s)\, ds] \tag{3.4}$$

where, $w_i(t)$, is the weight of the ith synapse at time t. The constant a is a non-Hebbian term. If the number of spikes in the actual output spike train, $s_o(t)$, is more (less) than the number of spikes in the desired spike train, $s_d(t)$, then the non-Hebbian term decreases (increases) the weights. This term speeds up the learning procedure. $Tw(s)$ is the learning window and like STDP it has an exponential function that decays with a time constant.

$$Tw(s) = \begin{cases} Ae^{-s/\tau} & \text{for } s \geq 0 \\ 0 & \text{for } s < 0 \end{cases} \tag{3.5}$$

Where τ is the exponential function decay time constant. The term A represents the amplitude of long term potentiation. The term $\int_0^{+\infty} Tw(s) s_i(t-s)\, ds$ represents the convolution of the learning window, $Tw(s)$, and the ith input spike train. As discussed in Ponulak and Kasiński (2010) the weight w_i is increased at the instance of spikes in the desired spike train, $s_d(t)$, if its input contains a spike shortly before a desired spike, whereas it is decreased if its input contains a spike shortly before an actual output spike. When the actual spike train approaches the desired spike train the weight increments and decrements compensate each other and the weights become stable. In ReSuMe the synapses do not have delays.

The Spike Pattern Association Neuron (SPAN) learning algorithm (Mohemmed et al., 2012) is similar to ReSuMe, in that it combines STDP and anti-STDP processes and is also derived from the Widrow-Hoff rule. The novelty of this algorithm is that it transforms spike trains into analogue signals such that common mathematical operations can be performed on them. SPAN can learn multiple desired spikes and can only train a single neuron (Fig. 5). Despite SPAN's similarity to ReSuMe, Mohemmed et al. (2012) did not compare its performance with ReSuMe. In Mohemmed et al. (2013) SPAN is extended to train a SNN consisting of multiple spiking neurons to perform a classification task. In the method the neurons construct a single layer of spiking neuron. Although SPAN can train a neuron to fire multiple spikes, only

a single desired spike is trained in Mohemmed et al. (2013) for spatiotemporal patterns corresponding to a class.

Florian (2012) proposed a learning algorithm to train a neuron to fire a desired spike train in response to a spatiotemporal input pattern. The algorithm is called Chronotron (Florian, 2012). Chronotron has two versions: I-learning with high biological plausibility and E-learning with high memory capacity and high computational cost. Florian et al. used a SRM for the neuron and a kernel function as a spike response. The total normalized PSP of a synapse j, $\lambda_j(t)$, is defined as the summation of all the kernel functions related to past presynaptic spikes in the synapse j until t. $\lambda_j(t)$ is called a normalized PSP because the effect of the weights is removed in the PSP and it only reflects the presynaptic spikes effect (number of input spikes and the time interval of the spikes). $\lambda_j(t)$ is used to construct a graphical illustration of the chronotron. This graphical illustration gives a good overview of the spike learning problem, so that it can be used to investigate the effect of the various parameters on the learning procedure. In E-learning, an error function, E, is calculated according to a desired spike train and the actual output of the neuron. The error function has a factor associated with the insertion, deletion and shift of the actual spikes toward the desired ones. The error is used to estimate the weight change, $\Delta w_j = -\frac{\partial E}{\partial w_j}$. After further simplification an approximate mathematical formulation for Δw_j is obtained. Δw_j is influenced by the synapse normalized PSP at the time of some actual output spikes that should be removed and some desired spikes that should be inserted and also it is influenced by the time shift that is necessary to shift the other actual output spikes towards the corresponding desired spikes. During the calculation of Δw_j some simplifications are applied; however the impact of such an approach is not justified (Florian, 2012). The intrinsic complexity and discontinuity of the dynamics of the spiking neuron implies that the traditional gradient descent method cannot easily be used whereas some heuristic methods inspired by the biological evidence should be employed.

The other version of the chronotron, I-learning, is inspired by ReSuMe and a weight is increased at the desired spike instant and it is decreased at the time of output spikes. In ReSuMe a weight change is managed by a decaying exponential function which is associated with the usual STDP learning window. However, I-learning is managed by the difference between two exponentials similar to the $\propto$ kernel with two time constants. One of the time constants is associated with the increasing part of the function and the other related to the decaying part. Chronotron does not have the non-Hebbian constant which is used in ReSuMe to speed up the learning procedure. Chronotron is a learning algorithm for a single neuron and it can train the neuron to fire multiple spikes at desired times in response to a corresponding spatiotemporal input pattern. Learning algorithms at the level of a neuron have been the subject of considerable recent research such as Tempoteron (Gütig & Sompolinsky, 2006), ReSuMe (Ponulak, 2005; Ponulak & Kasiński, 2010) and SPAN (Mohemmed et al., 2012). PSD (Precise-Spike-Driven Synaptic Plasticity) (Yu, Tang, Tan, & Li, 2013a) is another example of learning methods that can train a single neuron to fire multiple desired spikes.

The synaptic weight association training (SWAT) algorithm proposed in Wade, McDaid, Santos, and Sayers (2010) merges STDP and the Bienenstock–Cooper–Munro (BCM) (Jedlicka, 2002) learning rule to train neurons in a single layer of spiking neurons. In the BCM model, synaptic plasticity depends on the activity of the corresponding postsynaptic neuron. It potentiates a synaptic weight if the firing rate of the post synaptic neuron is higher than a threshold level, and it depresses the synaptic weight if the postsynaptic neuron firing rate is less than the threshold level. In SWAT, BCM is used to modulate the height of an STDP learning window to stabilize the weight adjustment governed by STDP. While STDP and BCM are used to train the output layer, the hidden layer in SWAT is used as a frequency filter to extract features from input patterns. The method only can use rate coding in the input and output patterns.

DL-ReSuMe (A Delay Learning-Based Remote Supervised Method for Spiking Neurons) (Taherkhani, Belatreche, Li, & Maguire, 2015b) integrates weight modulation with the synaptic delay shift to map a random spatiotemporal input spike pattern into a random desired output spike train in a supervised way. DL-ReSuMe can achieve up to 10% higher accuracy than ReSuMe and BPSL (Taherkhani, Belatreche, Li, & Maguire, 2014) (A biologically plausible supervised learning method for spiking neurons) at a much faster learning speed. DL-ReSuMe (Taherkhani et al., 2015b) can learn input spike trains of shorter duration and smaller mean frequency with a higher accuracy and much faster learning speed than ReSuMe. One interesting feature of DL-ReSuMe method is the ability to solve the silent window problem in a spatiotemporal input pattern.

DL-ReSuMe (Taherkhani et al., 2015b) adjusts each delay only once during learning, and after the change it stays fixed and its corresponding synaptic weight is continually changed in subsequent epochs to train the neuron to fire at desired times. However, it is possible that the single delay adjustment does not set the delay at an appropriate value and it needs more tuning. Specially, when multiple desired spikes are learnt by a neuron, the effect of delay adjustment on different desired spikes should be considered to train the delays precisely. Additionally, multiple changes of the weights alter the previous situation and it is more likely that the first adjustment of a delay which is appropriate for the previous weights is not suitable for the new configuration of the weights. So, the delays also should be updated based on the weight updates. EDL, an Extended Delay Learning based remote supervised method for spiking neurons (Taherkhani, Belatreche, Li, & Maguire, 2015a), was proposed to solve this problem of DL-ReSuMe by introducing multiple delay adjustments. The main property of EDL is its regular multiple adjustment of delay and weights. Irregular multiple adjustments of delays cause distraction not only in the delay training, but also in weight adjustments, and consequently it reduces the performance of the learning. Moreover, in EDL, the multiple delay adjustments become stable when the neuron reaches its goal.

The Liquid State Machine (LSM) provides an approach that consists of a dynamic filter. The dynamic filter is constructed by a recurrent SNN called reservoir. It maps input spike trains to an internal dynamic state which nonlinearly depend on current and previous inputs. The output of the LSM is fed to a readout layer (a simple classifier) which is trained to classify the internal dynamic state streams (Maass, Natschläger, & Markram, 2004). LSM can capture temporal information, so it can have promising results on the applications that deal with temporal information, i.e. the application that the exact sequence of the input occurring in time is important in addition to the value of the inputs (Verstraeten, Schrauwen, Stroobandt, & Van Campenhout, 2005). Because, the recurrent connections in the LSM give a short-term memory effect through the different loops generated in the recurrent network and it helps LSM to process temporal information in which the history of input is important (Verstraeten et al., 2005). Speech recognition, robot control, object tracking and EEG recognition are examples of tasks that are intrinsically temporal. Speech recognition of isolated digits (Verstraeten et al., 2005), real-time speech recognition (Schrauwen, D'Haene, Verstraeten, & Campenhout, 2008), and robotics (Joshi & Maass, 2004) are examples of applications that use LSM. LSM has comparable performance to state-of-the-art artificial intelligent systems on the mentioned tasks (Ju, Xu, Chong, & VanDongen, 2013). The mentioned learning algorithm for single neurons in Section 3.2 can be used as readout for a LSM.

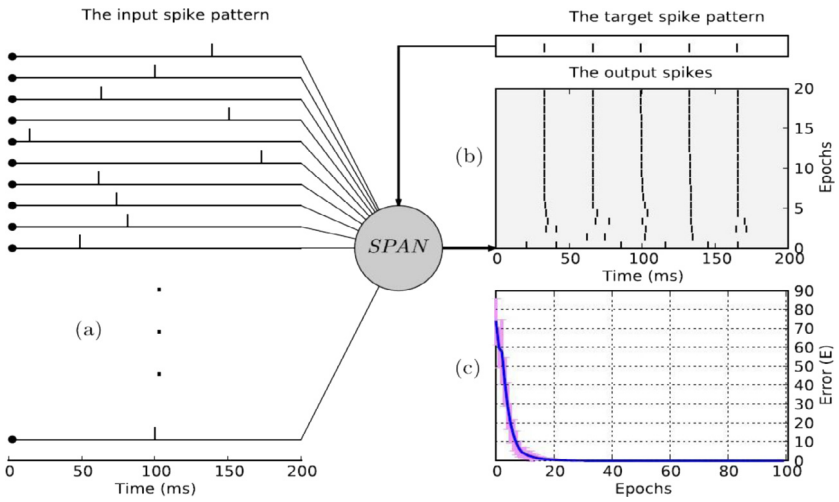

Fig. 5. SPAN trains a spiking neuron to map a spatio-temporal input pattern to an output spike train composed of a number of desired spikes. (b) The training effect on the development of the output spikes at desired times during different learning trials. (c) The reduction of the error between the actual output spike train and the desired spike train in addition to its standard deviation.
Source: The figure is adapted from Mohemmed et al. (2012)

The LSM proposed in Maass et al. (2004) has a fixed topology and fixed connection weights. Its biological plausibility and processing performance could be improved by employing other biologically plausible topologies, and other synaptic plasticity approaches such as short-term plasticity. Hazan and Manevitz (2012) implemented an LSM with various topologies to investigate the effect of the topological constraints on the LSM's robustness. Effects of the LIF and the Izhkevich neuron model (Izhikevich, 2003) are also investigated and shown to be qualitatively similar. Different types of read out methods such as Widrow & Hoff, Back-Propagation, SVM and Tempotron (Gütig & Sompolinsky, 2006) are used in Maass et al. (2004).

Paugam-Moisy et al. (2008) proposed a multi-timescale learning rule for SNNs where the reservoir has unsupervised synaptic plasticity driven by STDP and axonal conduction delays. Polychronous spiking patterns emerge inside the reservoir. A polychromous group is a specific group of fired spikes by neurons inside the reservoir. The spikes in a polychronous group are not synchronous, i.e. they are not fired in same time, however they have a time-locked firing pattern, i.e. there are a specific time intervals between the spikes. It is believed that in a biologically plausible SNN, where the synaptic plasticity of the network is governed by STDP and there are conduction delays between neurons, polychromous groups appear (Izhikevich, 2006; Paugam-Moisy et al., 2008). In Paugam-Moisy et al. (2008) it is supposed that applying input patterns from the same class can activate a specific polychronous group, however the polychronous group is not activated when patterns from other classes are applied. They proposed a supervised learning algorithm to adjust the delays related to readout neurons to learn firing patterns of different polychronous groups related to different classes. The SNN uses biologically plausible elements such as STDP synaptic plasticity and axonal conduction delays, and it is also used to classify a large dataset (US Postal Service: USPS, Hastie, Tibshirani, & Friedman, 2001). A layer of spiking neurons is used to classify an input pattern in a supervised manner. Only the delays of the readout neurons are adjusted during the supervised learning. The delays are adjusted to force the neuron related to the target class to fire before the neuron related to the other class. The learning method tries to maximize the time difference between the negative class and positive class during the classification of two classes. It has relatively poor performance in comparison with traditional classification methods. The synaptic weights of the readout neurons are not changed. The neurons in LSM can fire multiple spikes; however the readout neuron can fire a single spike. Using a readout that can fire multiple spikes can increase the biological plausibility of the method.

Although the mentioned supervised learning methods can train multiple output spikes, they work based on a single training neuron or single-layer of neurons. It is difficult to design a multilayer network of spiking neurons with supervised learning ability to fire multiple desired spikes, because the complexity of the learning task is increased by increasing the number of spikes and layers (Ghosh-Dastidar & Adeli, 2009; Xu, Gong, & Wang, 2013). In this situation the learning algorithm should control various neurons to generate different desired spikes. However, a real biological system is composed of a large number of interconnected neurons (Ghosh-Dastidar & Adeli, 2009; Sporea & Grüning, 2013; Xu, Gong, & Wang, 2013).

3.3. Learning multiple spikes in a multilayer spiking neural network

A multilayer neural network has higher information processing ability than a single layer of neurons. It has been shown that a perceptron multilayer neural network has a higher processing ability than a single layer of neurons. A single layer perceptron is limited to the classification of linearly separable patterns. However, a multilayer feedforward perceptron neural network can overcome the limitation of the single layer network (Haykin, 2009). The higher processing ability of a multilayer neuronal network is not only proven in the classical neuronal network, but is also confirmed in the spiking neural network. Sporea and Grüning (2013) have shown that a multilayer spiking neural network can perform a nonlinear separable logical operation, i.e. XOR, however the task cannot be accomplished without the hidden layer neurons. So single neurons or single-layer of neurons cannot simulate the learning of a biological neural network with a high processing ability, and designing a learning algorithm for a multilayer network of spiking neuron with the ability of firing multiple spikes is essentially required (Ghosh-Dastidar & Adeli, 2009; Sporea & Grüning, 2013; Xu, Gong, & Wang, 2013).

Bichler, Querlioz, Thorpe, Bourgoin, and Gamrat (2012) introduced a two-layer network for spiking neuron capable of extraction of temporally overlapping features directly from unfiltered Address-Event Representation (AER) silicon retina data, using only a simple, local STDP rule and 10 parameters in all for the

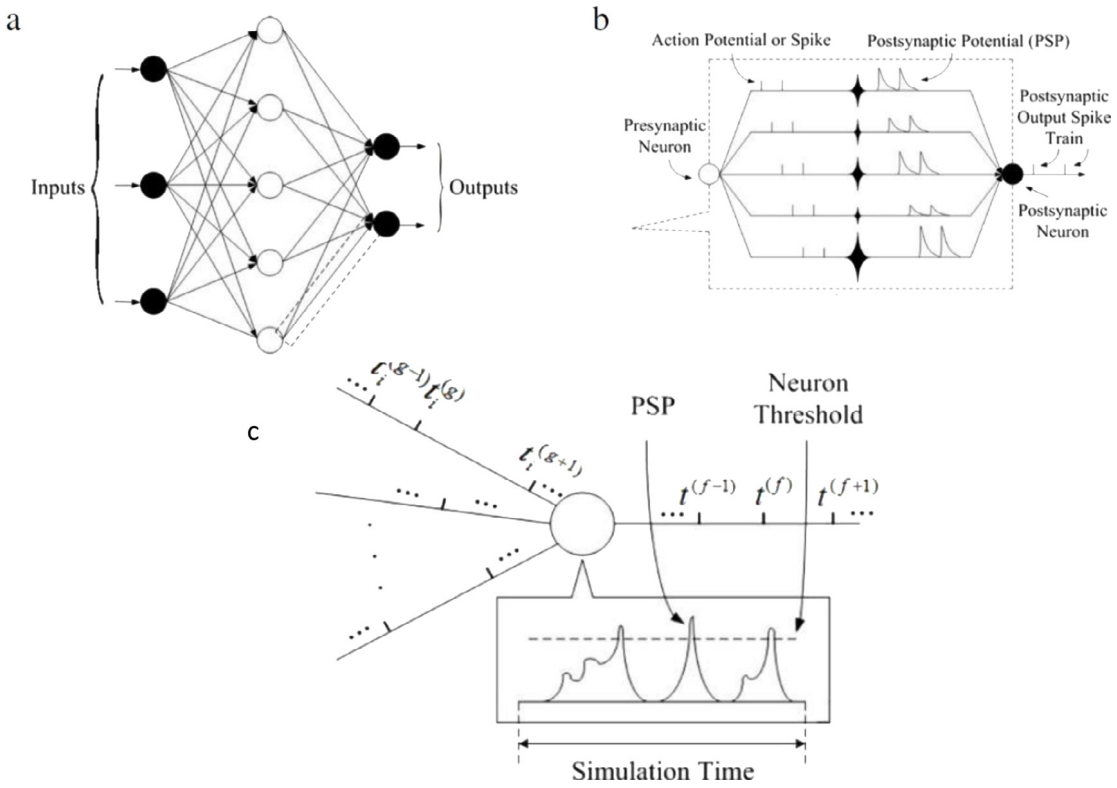

Fig. 6. (a) The structure of the multilayer neural network proposed by Xu, Gong, and Wang (2013) to train multiple spikes (b) In the network two neurons are connected by multiple sub connection with different delays (c) each neuron in the hidden layer or output layer can fire multiple spikes.
Source: The figure is adapted from Xu, Gong, and Wang (2013)

neurons (LIF neuron is used in this method). Although the proposed learning algorithm is unsupervised, its 10 parameters are optimized through a supervised manner. A simplified form of STDP is used in the method. In the special form of STDP used in Bichler et al. (2012), during LTD all the input weights are reduced by a constant value, regardless of the time difference between presynaptic spike and post synaptic spike times, i.e. $t_{post} - t_{pre}$. However, in a biologically plausible form of STDP, the amplitude of a weight adjustment depends exponentially on the time difference. Additionally, the weight adjustment of the neurons in the hidden layer is independent of the activity of the output neurons.

The online learning algorithm for SNNs proposed in Wang et al. (2014) is used to classify real valued input data. It is composed of three layers. In the input layer the real valued data are encoded to the precise timing of spikes through population coding scheme. The structure of the hidden layer is changed dynamically by using a clustering algorithm. STDP and anti-STDP are used to train neurons in the output layer. The method (Wang et al., 2014) uses latency coding in the first layer and the time to the first spike in the hidden layer, and it uses rating coding in the output layer to associate the applied input pattern to the class of the neuron that fire more spikes. Additionally, the training in the hidden and output layers operate independently. Consequently, it is hard for the network to find appropriate interaction between the activity of the different layers. Moreover, it is not explained whether adding and removing neurons to the hidden layer is a property of the biological neural network or this characteristic is only added to the artificial neural network to improve its performance.

In the previous gradient descent learning algorithms for network of spiking neurons like SpikeProp, QuickProp (McKennoch et al., 2006) and RProp (Ghosh-Dastidar & Adeli, 2007), each neuron in the input, hidden and output layers can only fire a single spike. Booij and tat Nguyen (2005) and Ghosh-Dastidar and Adeli (2009) extended the multilayer SpikeProp (Bohte et al., 2002) to allow each neuron in the input and hidden layers to fire multiple spikes. However, each output neuron can fire only a single spike. Xu, Gong, and Wang (2013) proposed the first supervised learning method based on the classical error backpropagation method that can train a multilayer network of spiking neurons to fire multiple spikes (Fig. 6). In this supervised learning method all the neurons in the input, hidden and output layers can fire multiple spikes. The network has multiple neurons at the output layer and each of the neurons can learn to fire their desired spike trains.

SpikeProp (Bohte et al., 2002), QuickProp (McKennoch et al., 2006), RProp (Ghosh-Dastidar & Adeli, 2007), and the methods proposed by Booij and tat Nguyen (2005) and Ghosh-Dastidar and Adeli (2009) adjust synaptic weights of a SNN to train each output neuron to fire at a desired time. In the methods, the sum of the square error for output neurons is considered as the error function. i.e. $E = 0.5 \sum_{j \in J} (t_j^a - t_j^d)^2$ where j is the index of the jth output neuron from the output neuron set, J. t_j^a and t_j^d are the firing times of the first actual output spike and the desired spike of the jth output neuron, respectively.

The difficulty of designing an algorithm which trains multiple spikes in the output neurons is summarized as follows: there is not a constant number of actual output spikes, especially when working with the multilayer neural network with multiple spikes for each neuron. In this situation it is difficult to construct an error function. Secondly, when the number of spikes is increased as a result of increasing the number of neurons, the interfering effect of the various desired spikes on the weight changes is also increased and consequently the learning becomes difficult. When the weights are adjusted to train a neuron to learn a spike, this adjustment affects the neuron that has been learnt from other desired spikes. This effect can be referred to the interfering effect. So, when the number of the spikes is increased, the limited

resources (learning parameters) should be adjusted in response to many demands (desired spikes). In other words, different desired spikes may lead to adjustment of learning parameters in opposite directions and finding an optimum value to satisfy the different situations is difficult. In order to overcome the first problem, Xu, Gong, and Wang (2013) supposed that the number of actual spikes and the number of desired spikes are identical, i.e. they are equal to F_j for jth output neuron, and they calculate the error function theoretically according to this assumption. They use $E = 0.5 \sum_{j=1}^{N} \sum_{f=1}^{F_j} (t_j^{(f)} - \hat{t}_j^{(f)})^2$ as the error function to calculate the square error of the output neurons, where N is the number of output neurons in the output layer, and $t_j^{(f)}$ and $\hat{t}_j^{(f)}$ are the firing times of the fth actual output spike and desired spike of the jth output neuron, respectively. Then during the experiment, if the number of spikes in the actual spike train is not equal to the number of spikes in the desired spike train, they consider all the spikes in the spike train that has the lower number of spikes, say n_l, and they only consider the first n_l spikes of the longer spike train. It does not seem to be an effective method to solve the first difficulty, because the learning method might adjust the network learning parameters for a desired spike which already has an actual output spike at the desired time. Consider a situation that the number of actual output spikes is higher than the number of desired spikes and there is a high number of early actual output spikes. The high number of the early actual output spikes causes that the method associates an early undesired actual spike to a far desired spike and overlook the actual spike that already is at the desired time. Finding a method that works based on local time event or temporal event can improve the method. Xu, Gong, and Wang (2013) believe that their learning algorithm overcomes the second problem by using the principle of "the Bigger PSP, the Bigger Adjustment (BPBA)" in their learning. According to the BPBA principle at the instant of weight adjustment, t, the changes of each synapse are in proportion to the height of the PSP produced in the synapse at the time t. This is equivalent to what happens during STDP. In STDP the presynaptic spike near to the post synaptic spike has a big PSP and their synaptic weights also are changed by a big value. Xu, Gong, and Wang (2013) defined an error function based on the time difference between actual output spike trains and desired spike trains. Then the derivation of the error function with respect to the synaptic weights is calculated, and a learning rule with a large mathematical equation for changing the synaptic weights is extracted. The constructive effect of one of the mathematical elements is described according to BPBA principle; they do not investigate the effect of other mathematical elements in the learning rule. It is not clear what happens if some of the mathematical elements in the learning rule are simplified, similar to what is done in Gütig and Sompolinsky (2006) for the chronotron. The simulation results in Xu, Gong, and Wang (2013) have shown that ReSuMe is more efficient and accurate than the proposed method of Xu et al. for simulation times that are less than 600 ms. However the algorithm can be used to train a multilayer neural network. If ReSuMe is improved for learning a multilayer SNN, it might have good accuracy and efficiency compared to the method that is proposed in Xu, Gong, and Wang (2013).

Gradient based methods suffer from various problems. A sudden jump in the network training error, called surge, is considered as one of the major problems, and it can cause failure in learning (Shrestha & Song, 2015). The non-monotonic and nonlinear behaviour of the neuron membrane potential make it difficult to minimize the constructed error and consequently leads to these surges during training (Shrestha & Song, 2013, 2015; Takase et al., 2009). The problem increases when the output neurons are trained to fire more than a single spike. Additionally, construction of an error function becomes difficult, because, the number of actual output spikes is not usually the same as the number of desired spikes during learning (Xu, Gong, & Wang, 2013). Although, training multiple spikes is difficult, it is a biologically plausible coding scheme (Sporea & Grüning, 2013; Xu, Gong, & Wang, 2013) and it can convey more neural information (Borst & Theunissen, 1999; Ponulak & Kasiński, 2010). This implies that the decoding scheme has an important effect on the learning tasks. After investigation of gradient based methods and their application in Adeli and Hung (1994), Ghosh-Dastidar and Adeli (2007) and Hung and Adeli (1993, 1994), it is concluded that the application of STDP which works based on local events is worth further investigation to design learning algorithms for a multilayer network of spiking neurons (Ghosh-Dastidar & Adeli, 2009).

STDP and anti-STDP were used in Sporea and Grüning (2013) as the first biologically plausible supervised learning algorithm for classification of real world data using a multilayer spiking neural network. Each neuron in the input, hidden and output layers of the network can fire multiple spikes. In the learning algorithm, the output spikes produced by the hidden neurons are not considered during training of the learning parameter of the hidden neurons. In other words the output spikes of the hidden neuron are not considered during the STDP and anti-STDP related to the hidden neuron input weight adjustments. However, in a biological neuron, the pre and post synaptic spikes of a neuron are usually used in STDP. Additionally, the spikes fired by the hidden neurons have a significant effect on a training task in a multilayer spiking neural network. Applying the hidden neuron output spikes during the learning of the hidden neuron can lead to a more biologically plausible learning algorithm, and improve the accuracy of the method. Another negative aspect of the learning method used for the hidden layer in Sporea and Grüning (2013) is that the same method was used for adjusting the input weights of both inhibitory and excitatory neurons. However, different weight adjustment strategies that reflect appropriately the effect of positive and negative PSP produced by the excitatory and inhibitory neurons in a hidden layer can be used to improve the performance of the method.

A biologically plausible supervised learning algorithm for spiking neural networks is proposed in Taherkhani, Belatreche, et al. (2018) and Taherkhani, Cosma, and McGinnity (2018). It uses the precise timing of multiple spikes which is a biologically plausible coding scheme to transmit the information between neurons. The learning parameters of neurons in the hidden layer and output layer are adjusted in parallel to train the network. It uses biological concept such as STDP, anti-STDP and delays learning to train the network. Simulation results show that the proposed method in Taherkhani, Belatreche, et al. (2018) and Taherkhani, Cosma, and McGinnity (2018) has improved performance compared to the fully supervised algorithm that trains multiple spikes in all layers proposed in Sporea and Grüning (2013). The improvement of the proposed method is achieved because of some significant properties of the method. First, it has used the firing times of spikes fired by the hidden neurons to train the input weights of the hidden neurons. However, in Sporea and Grüning (2013) the firing time of a hidden neuron is not considered. Another property of the proposed method is the application of different appropriate approaches to adjust the weights related to inhibitory and excitatory neurons in a hidden layer. Delay learning increases the complexity of the learning method and consequently the running time, however, it can improve the performance of the method. Additionally, delays are a biologically plausible property and it is a natural property of a real biological system.

The various algorithms discussed in this section are summarized in Table 1 according to their ability to train a single desired spike or multiple desired spikes, and their structure (a single

neuron, a single layer of neurons, and multiple layers of spiking neurons). The learning algorithms are highly concentrated on learning a single spike (the second row) or on a single neuron (the second column) which are comparably simple tasks. The rate coding, which works based on multiple spikes without considering the precise timing of the spikes, commonly is used in traditional ANNs. After finding the biological evidence that proves the encoding of information in the precise timing of the spikes, learning algorithms are devised to capture the information in the precise time of spikes. The first learning algorithms for spiking neurons such as SpikeProp (Bohte et al., 2002) work based on a single spike. Other examples of the learning algorithms that learn a single spike are shown in the first row of Table 1. However, the learning algorithms which work based on multiple spikes (the third and the fourth rows of Table 1) train a more biologically plausible learning task. Investigation of biological neurons shows that the synaptic plasticity models working based on multiple spikes are a better representation of their biological counterparts based on experimental data (Bi & Wang, 2002; Froemke & Dan, 2002; Froemke et al., 2006; Morrison et al., 2008; Seth, 2015; Wang et al., 2005; Zuo, Safaai, Notaro, Mazzoni, Panzeri, & Diamond, 2015). Additionally, multiple spikes convey more information (Borst & Theunissen, 1999; Ponulak & Kasiński, 2010; Xu, Gong, & Wang, 2013). The fourth column of the table contains examples of learning algorithms for multilayer networks which are biologically plausible learning algorithms. Although, the multilayer learning algorithms that can train multiple spikes are more biologically plausible, training multiple spikes is a difficult learning task for a multilayer network. Therefore, finding a learning algorithm for multilayer network of spiking neurons to train multiple desired spikes remains a challenging task.

3.4. Learning algorithms with delay leaning ability

Experimental research has proven the existence of synaptic delays in biological neural (Katz & Miledi, 1965; Minneci, Kanichay, & Silver, 2012; Parnas & Parnas, 2010). The synaptic delay influences the processing ability of the nervous system (Gilson et al., 2012; Glackin et al., 2010; Xu, Gong, & Wang, 2013). There is biological evidence that the synaptic delay can be modulated (Boudkkazi, Fronzaroli-Molinieres, & Debanne, 2011; Lin & Faber, 2002). 'Delay Selection' (Ghosh-Dastidar & Adeli, 2007, 2009) and 'Delay Shift' (Adibi, Meybodi, & Safabakhsh, 2005) are two approaches that have been developed for delay learning in SNNs. Two neurons are connected by multiple synapses (sub connections) with various time delays in the delay selection method. The weights related to appropriate delays are increased and the weights of connections with unsuitable delays for learning desired spikes are decreased. For example, SpikeProp (Bohte et al., 2002) used a 'Delay Selection' method.. Similarly, other research (Ghosh-Dastidar & Adeli, 2007, 2009; McKennoch et al., 2006; Shrestha & Song, 2015; Sporea & Grüning, 2013; Xu, Gong, & Wang, 2013) has constructed multiple synapses with different synaptic weights and delays between two neurons. Using sub-connections with various synaptic delays improves the learning performance by producing output spikes at different desired times. However, the number of learning parameters (synaptic weights) is increased and consequently the computational cost is increased.

The delay shift approach is used to train a CD. A CD fires in response to coincident input spikes. Synaptic delay adjustment in the model is an essential characteristic of the learning algorithm (Adibi et al., 2005; Pham et al., 2008). Pham et al. (2008) proposed a self-organizing delay shift method to train a CD. A Radial Basis Function (RBF) network is trained in Adibi et al. (2005) using a delay shift approach. In this method a synaptic delay vector makes the centre of the RBF neuron. If input spike patterns are close to this centre, they will fire the neuron. In a CD, the weights are considered constant and are not modified. However, the synaptic weight modulation has a dominant effect on the processing ability of spiking neural networks. DL-ReSuMe (Taherkhani et al., 2015b), EDL (Taherkhani et al., 2015a), and the supervised method proposed in Taherkhani, Belatreche, et al. (2018) and Taherkhani, Cosma, and McGinnity (2018) for multilayer SNN are other examples that use delay shift method for training.

3.5. Deep Spiking Neural Networks

Deep learning methods have achieved successful performance in different applications especially in image recognition (Hinton & Salakhutdinov, 2006; LeCun, Bengio, & Hinton, 2015). The ability of deep learning to integrate the feature extraction and feature learning processes enables it to be applied to challenging problems that are difficult to be solved by traditional machine learning methods, since traditional machine learning methods have separate feature engineering and feature learning processes (Taherkhani, Belatreche, et al., 2018; Taherkhani, Cosma, & McGinnity, 2018). Although, deep learning architectures have shown promising results, their processing ability is far below their biological counterpart, the brain.

In a Deep Neural Network (DNN), several processing layers are staked to make a deep multilayer structure. All or most of the stacked layers are trained to increase the selectivity ability of the overall deep learning method and make a robust representation of the input data on different layers (LeCun et al., 2015). The deep learning methods that work with raw input data are considered as representation learning methods. In a representation deep learning method the raw inputs are applied to a network and the network extracts representations which are required for classification or detection from the raw input data (LeCun et al., 2015).

3.5.1. Backpropagation for supervised learning

Backpropagation for supervised learning method has an important role in the successful results of classical deep learning methods (LeCun et al., 2015). Consequently, appropriate backpropagation learning algorithms for Deep Spiking Neural Network (DSNN) can result in successful performance for DSNNs. Zenke and Ganguli (Zenke & Ganguli, 2018) have proposed a voltage based gradient decent approach, called SuperSpike, to train a multilayer network of deterministic integrate-and-fire neurons to process spatiotemporal spiking patterns. They proposed a biologically plausible strategy for credit assignment in a deep SNN. In the gradient approach, the partial derivative of the hidden neurons is approximated by the product of a nonlinear function of postsynaptic voltage and related filtered presynaptic spike train. Using a nonlinear function of the postsynaptic voltage, instead of the postsynaptic spike train as credit assignment for hidden neurons, overcomes the problem of silent neuron in the network without injecting noise which might reduce the method accuracy. Additionally, SuperSpike used synaptic eligibility traces to import temporal activity in the credit assignment rule. Experiments carried out by Neftci, Mostafa, and Zenke (2019) revealed that SuperSpike does not have good performance for large multilayer SNN. Additionally, it is not applied for real world applications.

Mostafa (2018) proposed a backpropagation learning method for SNNs composed of multiple layers. Nonleaky integrate-and-fire neurons were used in this network. The neuron synaptic current kernels are exponentially decaying functions. The time of the first spike is considered as the desired output. A differentiable function has been constructed to relate the times of input

Table 1
Learning algorithms for spiking neural networks.

	Single neuron	Single layer	Multilayer
A single spike		Evolving spiking neural network (eSNN) (Belatreche et al., 2007; Wysoski et al., 2008), Tempotron (Yu et al., 2012, 2013b, 2014), Multiple SPAN (Mohemmed et al., 2013) LSM (Paugam-Moisy et al., 2008), Pattern Clustering (Pham et al., 2008)	SpikeProp (Bohte et al., 2002), Backpropagation with momentum (McKennoch et al., 2006; Xin & Embrechts, 2001), QuickProp (McKennoch et al., 2006; Xin & Embrechts, 2001), Resilient propagation (RProp) (Ghosh-Dastidar & Adeli, 2007; McKennoch et al., 2006; Silva & Ruano, 2005), Levenberg–Marquardt BP (Silva & Ruano, 2005), The SpikeProp based on adaptive learning rate (Shrestha & Song, 2015), Evolutionary strategy (Belatreche et al., 2003)
A single output spike & multiple spikes in input or hidden layers	Tempotron (Gütig & Sompolinsky, 2006) Supervised Hebbian learning (Ruf & Schmitt, 1997)	deSNN (Kasabov et al., 2013)	(Booij & tat Nguyen, 2005), (Ghosh-Dastidar & Adeli, 2009)
Multiple spikes	ReSuMe (Ponulak & Kasiński, 2010), SPAN (Mohemmed et al., 2012), Chronotron (Florian, 2012), Supervised Hebbian learning (Legenstein et al., 2005), PSD (Yu et al., 2013a) BPSL (Taherkhani et al., 2014), DL-ReSuMe (Taherkhani et al., 2015b), EDL (Taherkhani et al., 2015a)	Statistical methods (Pfister et al., 2006)	(Sporea & Grüning, 2013), Unsupervised (Bichler et al., 2012), (Xu, Gong, & Wang, 2013), (Taherkhani, Belatreche, et al., 2018; Taherkhani, Cosma, & McGinnity, 2018).

spikes to the time of the first spike of an output neuron using a transformation. The input spikes that have causal relation with an output spike are considered, i.e. the input spikes that fire before the output spike time. Then weights are updated using a derivable cost function on spike time. In this method the time of the first spike is considered, and it prevents the neuron to fire multiple times.

3.5.2. Spiking convolutional neural networks

CNN is a well-known DL method. The early layers in CNN were inspired from neuron responses in the primary visual cortex (V1). Primary visual features, such as oriented edges, are detected by the neurons in the V1 area. In conventional CNN, input images are convolved with kernels or filters to extract features related to the edges in early layers and to extract more abstract features, i.e. features related to shapes in higher level layers. In the traditional CNN the parameters of the kernel filters are often trained using error backpropagation methods. There exist a number of Spiking CNNs (SCNNs) which utilize hand-crafted convolution kernels and these SCNNs have been applied in a number of classification tasks (Kheradpisheh, Ganjtabesh, & Masquelier, 2016; Kheradpisheh, Ganjtabesh, Thorpe, & Masquelier, 2018; Masquelier & Thorpe, 2007; Tavanaei, Masquelier, & Maida, 2016; Zhao, Ding, Chen, Linares-Barranco, & Tang, 2015). In hand-crafted convolution kernels, the kernel parameters are fixed whilst the network is being trained.

Kheradpisheh et al. (2018) have proposed a SCNN where in the first layer Difference of Gaussians (DoG) filters are used to extract edges, and hand-crafted DOG filters were used in the first layer. DOG filters were used to encode the contrast in the input image to latency spike code. Positive or negative contrasts are detected by applying a DOG filter over different parts of input images. The DOG cells emit spikes depending on the contrast, and they fire spikes in such way that the order in the spikes contains information about the input image contrast. It is suggested that this rank order code can be used to perform V1 like edge detection (Delorme, Perrinet, & Thorpe, 2001; Kheradpisheh et al., 2018). After the first layer there are three pairs of convolutional and pooling layers. The convolutional layers are trained by STDP. Additionally, there is a lateral inhibition mechanism in all convolutional layers such that when a neuron is fired, it inhibits other local neurons by resetting their potentials to zero, and prevents their firing for the current applied input image. Moreover, each neuron can only fire once during recognition of an input image. The learning in a subsequent convolutional layer begins when the learning in the current convolutional layer is finalized. The SCNN (Kheradpisheh et al., 2018) used max pooling layers which allow the propagation of its first emitted input spike. Despite the rank order coding of input information in the hand-crafted DOG layer, in the pooling layer only the time of the first spike is considered and information in the other spikes is ignored. The final pooling layer in the SCNN is a global max pooling layer which pools on overall neuronal maps of the last convolutional layer. The output of the global pooling layer is feed to a non-spiking classifier, i.e. a linear SVM classifier, to determine the category of the applied input. Therefore, supervised learning takes place in a non-spiking method, i.e. SVM. Additionally, instead of using spiking activity of the last convolution layer, the threshold of the neurons in the last convolutional layer was set to infinite and the final potentials are used for the next steps. The performance of the SCNN on MNIST data was 98.4%. There exist other SCNNs where all convolutional layers are trained instead of using fixed initial Gabor filters (Tavanaei, Masquelier, & Maida, 2018).

Illing, Gerstner, and Brea (2019) have shown that localized receptive fields, i.e. Gabor filters, improve the accuracy of a networks compared to all-to all connectivity. They have used integrate-and-fire neurons and STDP to train a shallow SNN that contains a single hidden layer and a readout layer. The hidden layer has fixed weights which correspond to random Gabor filters, and they used a STDP based learning rule to train the readout layer. The SNN reached a testing accuracy comparable to other deep SNN on MNIST, i.e. 98.6%. Illing et al. (2019) suggest that novel biologically plausible deep learning methods should reach better performance than their shallow counterpart networks. Additionally, novel biologically plausible deep learning methods should be tested on more complicated data than MNIST.

3.5.3. Semi-supervised learning methods

Panda and Roy (2016) proposed a convolutional Auto-Encoder (AE) learning method for SNN. AE is a well-known unsupervised learning method in classical neural networks. AE maps input to a

lower dimensional feature space, then it reconstructs the input data from the low dimensional features. The first procedure is called encoding and the last procedure that reconstructs the input from the low dimension features is called decoding. It is supposed that AE can extract robust and discriminative features in a low dimensional feature space that can be used to improve classification accuracy. Panda and Roy (2016) have used a backpropagation algorithm for layer wise training of different convolutional layers. They have used the membrane potential of spiking neurons to construct an error function. The convolutional layers are trained independently and are staked in a network. Additionally, they have used an average pooling layer after each convolutional layer. The pooling layer calculates the average of membrane potential of convolutional neurons within a sampling window. Panda and Roy (2016) have used a backpropagation method to train the final fully connected output layer. The output layer maps extracted features from multiple convolutional and pooling layers to corresponding spiking labels using labelled training data. The CSNN achieved a high classification accuracy of 99.08% on MNIST dataset.

Lee, Panda, Srinivasan, and Roy (2018) proposed a learning algorithm for SCNN. The SCNN pretrains convolutional kernels using STDP in a layer wise manner. During the pretraining procedure, synaptic weights and neuronal thresholds are trained. At the time of a post-spike, the shared kernel weights are adjusted. Average STDP induced weight adjustments are applied when there are multiple post-neuronal spikes in a feature map. Additionally, at the time of a post-spike in a feature map, the firing thresholds of all the neurons in the feature map are uniformly increased. The threshold of neurons in feature maps exponentially decay over non-spiking periods. It is supposed that this threshold adjustment resembles homoeostasis in biological neurons. After the pretraining procedure, all the layers in the SCNN are trained using a gradient descent BP algorithm.

The Deep Belief Networks (DBNs) proposed by Hinton and Salakhutdinov (2006) and the Deep Boltzmann Machines (DBMs) proposed by Srivastava and Salakhutdinov (2014) are two classical deep learning methods that have an unsupervised pretraining procedure. They have a layer-wise pretraining procedure and the deep structures of the networks enable them to learn high-level representation of features. The Restricted Boltzmann Machine (RBM) is a two-layer neural network that constitutes the building block of DBNs and DBMs. RBMs are trained by a method called Contrastive Divergence. Neftci, Das, Pedroni, Kreutz-Delgado, and Cauwenberghs (2014) have proposed a spiking version of Contrastive Divergence to train a spiking RBM composed of integrate-and-fire neurons. Neftci et al. (2014) have used a STDP for Contrastive Divergence. The accuracy of the spiking DBN on the MNIST data has been compared with feed-forward fully connected multi-layer SNNs and SCNNs in Tavanaei, Ghodrati, and Reza (2019), and the comparison results show that the spiking DBN has lower accuracy than the other SNNs.

3.5.4. Converted classical DNN as DSNN

There are a group of DSNNs which are directly constructed from the conventional DNN. In the group of DSNNs, first a classical DNN which is composed of neurons with continuous activation values is trained, then the classical DNN is deployed to a DSNN (Cao, Chen, & Khosla, 2015; Rueckauer, Lungu, Hu, & Pfeiffer, 2017). By this approach the state-of-the-art methods for training DNN can be used to construct DSNN to achieve a competitive performance. This conversion might cause loss of accuracy in the DSNN compared to the original DNN. Different techniques (such as introducing extra limitations on neuron firing rates or network parameters, weight scaling, adding noise, and using probabilistic weights) have been used to reduce the loss of accuracy during the conversion from ANN to SNN (Shrestha, 2018). The generated DSNN needs a large number of time steps to perform the input–output mapping, and the constructed DSNN cannot capture temporal dynamics of spatiotemporal data (Lee, Sarwar, & Roy, 2019). Tavanaei et al. (2019) compared a number of DSNNs, and their result shows that deep SNNs which are converted from a classical DNN can achieve the highest accuracy on MNIST data compared to methods that train DSNNs directly. DSNNs which are constructed by converting a classical DNN to DSNN usually use rate coding to encode an analog output of a classical neuron. The rate coding can mask the temporal information that can be processed by DSNN (Lee et al., 2019; Mostafa, 2018).

3.5.5. Deep recurrent spiking neural networks

Recurrent neural networks are a class of neural networks whose internal states evolve with time, and they have been used in temporal processing tasks such as noisy time series prediction, language translation, and automatic speech recognition (Bellec, Salaj, Subramoney, Legenstein, & Maass, 2018; Neftci et al., 2019). Training large RNNs is a difficult task because during training, functions with long-range temporal and spatial dependencies should be optimized. The non-derivable dynamics of Spiking RNNs increase training difficulties. Training a deep RNN is a challenging task, as the dependency in space is extended in addition to the dependency in time (Neftci et al., 2019). Th gradient descent method is a powerful method to train learning parameters in RNNs. In a multilayer network with hidden units the adjustment of learning parameters (credit assignment) for hidden units are obtained using the chain rule of derivatives. Special and temporal credit assignments are two problems that should be solved to train a RNN. The spatial credit assignment is a common problem for RNNs and multi-layer perceptrons, and it is assigned spatially across layers to update learning parameters.

In the gradient descent method, a method similar to the spatial credit assignment is used for temporal credit assignment through unrolling the network in time. This method is called Backpropagation Through Time (BPTT). The non-linearity in the activity of spiking neurons makes it difficult to apply the classical BBTT to SNN. Different methods have been proposed to address this challenge. Using biologically inspired local learning rules, converting trained classical NNs to SNNs, smoothing the network to make it continuously derivable, and using approximated gradient descent for SNN are different techniques that have been used to overcome the problem related to nonlinearity in gradient descent learning methods for SNNs (Neftci et al., 2019).

The smoothing method can be used to overcome nonlinearity issues of the gradient decent learning method for SNNs. Smoothing models are used in SNN to construct a network with well-behaved gradients. For instance Huh and Sejnowski (2018) used an extended version of integrate-and-fire model in which the nonlinearity of a neuron is replaced by a continuous-valued gating function. Then a RNN with those neurons was trained using a standard BPTT. Another group of smoothing models are probabilistic models which use the log-likelihood of a spike train as a smooth quantity. Although there are some probabilistic models that can learn precise output spike timing (Gardner, Sporea, & Grüning, 2015), probabilistic models usually used rate-coding at output level and firing probability densities. These properties reduce their ability to capture temporal information. Although BBTT is a standard learning method for recurrent neural networks, BBTT is not a biologically realistic method. Because in BBTT, error signals should be transmitted backward in time and space. Bellec et al. (2019) proposed a biologically plausible learning method using locally available information to make a

biologically plausible approximation of BBTT. They have used feedforward eligibility traces of synapses that are available in a real time at the location of synapses to design the learning method (Bellec et al., 2019).

4. Challenges and opportunities

Recently, classical DNNs have achieved successful results in different applications. Despite this, DSNNs are still in the developmental stage and more advanced learning algorithms are required to train them. The main challenge in training DSNNs is to find methods to train neurons in a hidden layer of DSNN in interaction with an output layer and other hidden layers. Supervised learning using a backpropagation method is a common approach to train a classical deep (multilayer) artificial neural network (ANN). However, the nonlinearity discontinuity of SNN activities makes it difficult to adopt existing backpropagation methods to train SNNs (Tavanaei et al., 2019). Additionally, it is not biologically plausible to train a DSNN with the error backpropagation method (Illing et al., 2019), and backpropagation learning methods do not mimic the learning of the human brain (Whittington & Bogacz, 2019). It is an important challenge to understand what are the best biologically plausible network architectures and learning rules to train DSNNs for information processing. Local learning rules inspired by STDP are more biological plausible and can be investigated to design new learning algorithms for SNNs. One interesting structural property of a biological NN is the recurrent connections in the network. BBTT (Neftci et al., 2019) is a classical method which has been applied to train a RNN. Finding a biologically plausible alternative for BBTT to train RNN using biologically plausible local learning rule can be consider as an important challenge for SNNs.

Neural encoding is an important aspect of learning in a SNN (Wu et al., 2019). Finding appropriate encoding mechanisms for spiking activity and designing compatible learning algorithms for the encoding mechanisms are new challenges in the SNN field. An optimization method for output spike train encoding has been proposed in Taherkhani, Cosma, and Mcginnity (2019), and it has been shown that optimizing the output encoding during the learning phase in classification tasks can increase classification accuracy by 16.5% compared to when using a non-adapted output encoding. Xu, Yang, and Zeng (2019) have shown that the time distances of input spikes related to actual and desired output spikes have an important effect on the accuracy of SNN, and finding an optimal time interval for input spikes to be involved in the synaptic weight adjustments can improve learning accuracy by 55% in a SNN. Wu et al. (2019) have shown that input spike encoding has a significant effect on improving the accuracy of a learning algorithm for SNNs. In a classification task a simple output encoding is to assign an applied input to the class corresponding to the output neuron that has the highest firing rate. Orchard et al. (2015) have assigned an applied input to the class related to the output neuron that fires first, and they have reported this output encoding has achieved a good performance. On other hand, Diehl et al. (2015) have shown that an increase in the number of output spikes in the output encoding will increase the classification accuracy. A population of neurons can be used instead of a single neuron for each class to reduce the variance of the output (Pfeiffer & Pfeil, 2018). Therefore, spike encoding has a prominent effect on the performance of SNNs and appropriate encoding strategies should be employed. Additionally, learning algorithms should be compatible with the selected encoding strategy. This is a challenge in biologically plausible learning methods because it is not clear which encoding method (or methods) is used in the brain.

Time plays an important role in activity of SNNs. SNNs generate spatiotemporal spike patterns whereas classical NNs work with spatial activation. Consequently, SNNs need a specific loss function that generates an error related to time which is different from the loss function of a classical NN. There are a considerable number of DSNNs that use rate-based approach (Diehl & Cook, 2015; Eliasmith et al., 2012; Guerguiev, Lillicrap, & Richards, 2017; Mesnard, Gerstner, & Brea, 2016; Neftci, Augustine, Paul, & Detorakis, 2016). However, these DSNNs are close to classical neural networks which work based on continuous values, and they neglect the information carried by individual spikes that can lead to a fast computation (Zenke & Ganguli, 2018). For instance, a widely used method to convert a real value to a spike train is using the spike train that is drawn from a Poisson process with a firing rate in proportion to the real value. In this conversion only the average firing rate of the spike train is important and the precise timing of spikes is not considered (Bellec et al., 2019). This can limit the capability of SNNs to process the precise timing of spikes.

DSNNs are often used for processing the data which has been used by classical DNNs. For instance, image samples in the MNIST and CIFAR10 datasets which are constructed from pixels with continuous values have been used for a long time with classical DNN, and DSNNs cannot currently outperform classical DNNs on this data. The nature of these benchmarks is close to the activation of classical neural network (i.e. they have continuous values) and they can directly be used by the methods. However, such data should be converted into spike trains before it can be used by DSNN. This conversion might destroy some parts of information in the images, and it can result in a reduction of accuracy of DSNN (Bellec et al., 2019). Therefore, applying DSNNs for processing new datasets that have properties which are compatible with SNNs can lead to improved performance rather than when processing these datasets using non spiking neural networks. Research on this type of data can lead to the emergence of new SNN that can perform processing tasks which are difficult for conventional DNN. For instance the data obtained by event-based cameras in Ramesh et al. (2019) or the spiking activity that is recorded from real biological nervous system (Maggi, Peyrache, & Humphries, 2018) originate from original spatiotemporal spiking activity and they can be a good candidate to be used by DSNN. There has been progress in event-based vision and audio sensors (Liu, Delbruck, Indiveri, Whatley, & Douglas, 2015; Pfeiffer & Pfeil, 2018) and the data extracted from them can be processed by DSNNs. However, currently, there is a lack of appropriate benchmark data for evaluating SNNs. SNNs have the ability to achieve good performance when trained on suitable datasets that have properties which are compatible with SNNs, and currently there is an urgent need to develop such benchmark datasets.

A major capability of classical deep learning methods, especially deep CNNs, is their capability for hierarchical feature discovery, where discriminative, abstract, and invariant features are extracted. Bio-inspired SNNs have brain-like representation ability, and they potentially have higher representation capability than traditional rate-based networks (Maass, 1996; Tavanaei et al., 2019). The SNNs ability to process temporal data can be mixed with the hierarchical feature representation capability of classical deep neural networks to construct a neural network with a high processing ability.

Advancements in regularization methods for training classical deep neural networks have improved the performance of these methods. However, SNNs have different characteristics, and finding appropriate regularization methods for learning parameter adjustment in SNN can be another interesting challenge to improve the performance of DSNNs.

Processing of the future big data, which is exponentially expanding as a result of advancements in technology, demands

huge computing resources. This demand requires a novel scalable computing framework. Neuromorphic hardware is believed to be a paradigm that can potentially satisfy such a demand (Neftci, 2018), because they mimic the brain which has significant processing ability. The brain can perform information processing with a high level of robustness, efficiency, and adaptivity. Neuromorphic engineering tries to produce hardware that can emulate the cognitive and adaptive ability of the brain. The exact principles of computation in the brain, which contains a massively parallel self-organizing neural architecture, have not yet been discovered (Douglas & Martin, 2004). In a situation such as real-time adaptability, autonomy, or privacy, it is required that computation performs close to sensors (Neftci, 2018). In this case computing systems such as neuromorphic hardware with low power consumption have advantage.

Biological neurons have constraints on communication, and power consumption, and biological neurons communicate through sparse spiking activity which reduce energy consumption (Gerstner & Kistler, 2002). Spike events consume energy and a low number of spikes in sparse spiking activity means a low power consumption. The state of other activities in the neuron such as membrane potentials, synaptic states, and neurotransmitter concentrations are local to the neuron. The sparse communication and the local processing generate a highly scalable structure. A computational strategy that works based on locally available information and sparse global communication is required to construct neuromorphic hardware with a scalable structure to solve real-world problems. Designing algorithms using local events for learning precise timing of spikes in a SNN can be useful for designing a learning strategy for neuromorphic hardware. SNNs are compatible for implementation in hardware as they can be more energy efficient compared to classical neural networks that work based on continuous values. Additionally, the nature of information communication in SNNs, which is based on spike activities, is close to the binary processing in a hardware platform. However, most of the state-of-the art machine learning methods work on non-local information that restricts their scalability when they are implemented in hardware. Consequently, the methods cannot perform online and incremental fast and energy efficient learning, similar to the learning observed in humans and animals. New biological plausible learning algorithms for SNNs are required to design neuromorphic hardware that performs accurate high speed and scalable low energy computation.

In summary, there is a number of challenges for designing learning algorithms for DSNN. Designing learning algorithms to train hidden neurons in an interconnected SNN by overcoming the discontinues and nonlinear behaviour of SNN to improve their accuracy is one of the main challenges. It is believed that local learning rules such as STDP performed in the biological nervous system can be a good option to design new biologically plausible learning algorithms for SNNs. Another challenge in training SNNs is neural encoding. It has been shown that neural encoding has significant effect on performance of a SNN. However, it is a challenge to find what is the exact encoding mechanism in the brain and how to design a learning algorithm to be compatible with the encoding mechanism. Another challenge in designing learning algorithms for SNN is the existence of time in the activity of a SNN. SNNs directly work with time which is a new characteristic compared to traditional NN. This offers an ability to work directly with time, however it causes discontinuity in neuron activity that increases the difficulty of learning. Another challenge is to make new dataset that use the ability of spiking times, such as the data generated from event-based vision and audio sensors. Designing biologically plausible learning algorithms that work based on local events offers a good opportunity to design learning algorithms for power efficient neuromorphic hardware.

5. Conclusion

In this paper, the biological background of spiking neurons was first considered, and then state of the art learning algorithms for SNNs critically reviewed. According to the literature, there are different mathematical models for biological neurons. The models simulate behaviour of a biological neuron in different level of details. LIF model is a simple one dimensional model of a biological neuron that needs less computational effort for modelling a biological neuron and Hodgkin and Huxley is a four dimensional model that can simulate the dynamic of a biological neuron with more details. It has a high computational cost.

The review shows that synaptic plasticity is supposed to be the base of learning in the brain and there are different models of synaptic plasticity for a biological neuronal system. The models try to simulate the behaviour of biological synapses based on experimental data. Literature review shows that standard pair-based STDP model was used to design biological plausible learning algorithms for spiking neurons. However, the standard pair-based STDP model is not the only model for the synaptic plasticity. It has been shown that multiple-spike STDP models are biologically plausible models, and they can predict experimental data captured from biological neurons with a higher precision. It is not clear how multiple-spike STDP models can be used to design learning algorithm for artificial neuronal network for machine learning purpose. Application of the multi-spike STDP models can lead to design of a more biologically plausible learning algorithm for spiking neuron and potentially leads to design a more powerful intelligent system.

According to the review, delays are a natural property of a biological neural network. On the other hand information conveys between neurons through precise timing of spikes. Delays can have direct effect on the precise timing of spikes. Therefore, designing a learning algorithm that merges the usual weight adjustment methods with a proper delay learning approach, can lead to a more biologically plausible learning algorithm with a higher processing ability.

Traditional neural networks work based on rate coding. The idea of encoding of information in precise timing of spikes motivated research into the development of learning algorithms such as SpikeProp that work based on a single spike per neuron. However, coding scheme based on precise timing of multiple spikes can convey more temporal information between neurons and it is more biologically plausible. However, designing a learning algorithm for a network of spiking neurons that conveys information between neurons through precise timing of multiple spikes is a difficult task and it demands more research.

A single biological neuron has interestingly different learning characteristics and many of the learning ability of a single biological neuron are not considered in their artificial counterparts. Various learning algorithms for single neurons were reviewed in this paper. The review shows that designing new learning algorithms for a single neuron with new biological properties is an ongoing research, and it can lead to generation of new biologically plausible learning algorithms with higher processing abilities. The review also shows that multilayer neuronal networks have higher processing ability compared to a single neuron or single layer of neurons. It is not clear how the different learning algorithms for a single neuron with new biologically plausible characteristics can be extended to train a multilayer network of spiking neurons. A challenging task remains to design a more biologically plausible learning algorithm for multilayer spiking neural networks.

Acknowledgements

Dr. Aboozar Taherkhani, Dr. Georgina Cosma, and Prof. T.M. McGinnity acknowledge the financial support of The Leverhulme Trust, UK (Research Project Grant No: RPG-2016-252, Title: Novel Approaches for Constructing Optimized Multimodal Data Spaces).

References

Adeli, H., & Hung, S. (1994). An adaptive conjugate gradient learning algorithm for efficient training of neural networks. *Applied Mathematics and Computation*, *62*(1), 81–102.

Adibi, P., Meybodi, M. R., & Safabakhsh, R. (2005). Unsupervised learning of synaptic delays based on learning automata in an RBF-like network of spiking neurons for data clustering. *Neurocomputing*, *64*(0), 335–357. http://dx.doi.org/10.1016/j.neucom.2004.10.111.

Artola, A., Brocher, S., & Singer, W. (1990). Different voltage-dependent thresholds for inducing long-term depression and long-term potentiation in slices of rat visual cortex. *Nature*, *347*(6288), 69–72. http://dx.doi.org/10.1038/347069a0.

Azevedo, F. A., Carvalho, L. R., Grinberg, L. T., Farfel, J. M., Ferretti, R. E., Leite, R. E., et al. (2009). Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology*, *513*(5), 532–541.

Bassett, D. S., Wymbs, N. F., Rombach, M. P., Porter, M. A., Mucha, P. J., & Grafton, S. T. (2012). Core-periphery organisation of human brain dynamics. arXiv preprint arXiv:1210.3555.

Belatreche, A., Maguire, L., & McGinnity, M. (2007). Advances in design and application of spiking neural networks. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, *11*(3), 239–248. http://dx.doi.org/10.1007/s00500-006-0065-7.

Belatreche, A., Maguire, L., McGinnity, M., & Wu, Q. (2003). A method for supervised training of spiking neural networks. In *Paper presented at the Proc. IEEE conf. cybernetics intelligence–challenges and advances* (pp. 39–44).

Belatreche, A., & Paul, R. (2012). Dynamic cluster formation using populations of spiking neurons. In *Paper presented at the Neural networks (IJCNN), the 2012 international joint conference on* (pp. 1–6).

Bellec, G., Salaj, D., Subramoney, A., Legenstein, R., & Maass, W. (2018). Long short-term memory and learning-to-learn in networks of spiking neurons. In *Paper presented at the Advances in neural information processing systems* (pp. 787–797). Retrieved from http://arxiv.org/abs/1803.09574.

Bellec, G., Scherr, F., Hajek, E., Salaj, D., Legenstein, R., & Maass, W. (2019). Biologically inspired alternatives to backpropagation through time for learning in recurrent neural nets. (No. 1) arxiv. Retrieved from https://arxiv.org/pdf/1901.09049.pdf.

Bi, G., & Wang, H. (2002). Temporal asymmetry in spike timing-dependent synaptic plasticity. *Physiology & Behavior*, *77*(4–5), 551–555. http://dx.doi.org/10.1016/S0031-9384(02)00933-2.

Bichler, O., Querlioz, D., Thorpe, S. J., Bourgoin, J., & Gamrat, C. (2012). Extraction of temporally correlated features from dynamic vision sensors with spike-timing-dependent plasticity. *Neural Networks*, *32*(Aug.), 339–348.

Bohte, S. M., Kok, J. N., & La Poutre, H. (2002). Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, *48*(1), 17–37.

Booij, O., & tat Nguyen, H. (2005). A gradient descent rule for spiking neurons emitting multiple spikes. *Information Processing Letters*, *95*(6), 552–558. http://dx.doi.org/10.1016/j.ipl.2005.05.023.

Borst, A., & Theunissen, F. E. (1999). Information theory and neural coding. *Nature Neuroscience*, *2*(11), 947–957.

Boudkkazi, S., Fronzaroli-Molinieres, L., & Debanne, D. (2011). Presynaptic action potential waveform determines cortical synaptic latency. *The Journal of Physiology*, *589*(5), 1117–1131.

Brette, R. (2015). Philosophy of the spike: Rate-based vs. spike-based theories of the brain. *Frontiers in Systems Neuroscience*, *9*(Nov.), 151.

Brette, R., Rudolph, M., Carnevale, T., Hines, M., Beeman, D., Bower, J. M., et al. (2007). Simulation of networks of spiking neurons: A review of tools and strategies. *Journal of Computational Neuroscience*, *23*(3), 349–398. http://dx.doi.org/10.1007/s10827-007-0038-6.

Cao, Y., Chen, Y., & Khosla, D. (2015). Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, *113*(1), 54–66. http://dx.doi.org/10.1007/s11263-014-0788-3.

Carey, M. R., Medina, J. F., & Lisberger, S. G. (2005). Instructive signals for motor learning from visual cortical area MT. *Nature Neuroscience*, *8*(6), 813–819.

Cariani, P. A. (2004). Temporal codes and computations for sensory representation and scene analysis. *IEEE Transactions on Neural Networks*, *15*(5), 1100–1111.

Clopath, C., Büsing, L., Vasilaki, E., & Gerstner, W. (2010). Connectivity reflects coding: A model of voltage-based STDP with homeostasis. *Nature Neuroscience*, *13*(3), 344–352.

Delorme, A., Perrinet, L., & Thorpe, S. J. (2001). Networks of integrate-and-fire neurons using rank order coding B: Spike timing dependent plasticity and emergence of orientation selectivity. *Neurocomputing*, *38–40*, 539–545. http://dx.doi.org/10.1016/S0925-2312(01)00403-9.

Diehl, P. U., & Cook, M. (2015). Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in Computational Neuroscience*, *9*, 1–9. http://dx.doi.org/10.3389/fncom.2015.00099.

Diehl, P. U., Neil, D., Binas, J., Cook, M., Liu, S. C., & Pfeiffer, M. (2015). Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *Proceedings of the international joint conference on neural networks*. 2015-Septe. http://dx.doi.org/10.1109/IJCNN.2015.7280696.

Douglas, R. J., & Martin, K. A. C. (2004). Neuronal circuits of the neocortex. *Annual Review of Neuroscience*, *27*(1), 419–451. http://dx.doi.org/10.1146/annurev.neuro.27.070203.144152.

Doya, K. (1999). What are the computations of the cerebellum the basal ganglia and the cerebral cortex? *Neural Networks*, *12*(7–8), 961–974. http://dx.doi.org/10.1016/S08936080(99)00046-5.

Eliasmith, C., Stewart, T. C., Choo, X., Bekolay, T., Dewolf, T., Tang, Y., et al. (2012). A large-scale model of the functioning brain. *Science*, *338*, 1202–1205.

Feldman, D. E. (2012). The spike-timing dependence of plasticity. *Neuron*, *75*(4), 556–571.

Florian, R. V. (2012). The chronotron: A neuron that learns to fire temporally precise spike patterns. *PLoS One*, *7*(8), e40233.

Fregnac, Y., & Shulz, D. E. (1999). Activity-dependent regulation of receptive field properties of cat area 17 by supervised hebbian learning. *Journal of Neurobiology*, *41*(1), 69–82.

Froemke, R. C., & Dan, Y. (2002). Spike-timing-dependent synaptic modification induced by natural spike trains. *Nature*, *416*(6879), 433–438.

Froemke, R. C., Tsay, I. A., Raad, M., Long, J. D., & Dan, Y. (2006). Contribution of individual spikes in burst-induced long-term synaptic modification. *Journal of Neurophysiology*, *95*(3), 1620–1629. http://dx.doi.org/10.1152/jn.00910.2005.

Gardner, B., Sporea, I., & Grüning, A. (2015). Learning spatiotemporally encoded pattern transformations in structured spiking neural networks. *Neural Computation*, *27*(12), 2548–2586. http://dx.doi.org/10.1162/NECO_a_00790.

Gerstner, W., & Kistler, W. M. (2002). *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge University Press.

Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

Gerstner, W., Sprekeler, H., & Deco, G. (2012). Theory and simulation in neuroscience. *Science*, *338*(6103), 60–65. http://dx.doi.org/10.1126/science.1227356.

Ghosh-Dastidar, S., & Adeli, H. (2007). Improved spiking neural networks for EEG classification and epilepsy and seizure detection. *Integrated Computer-Aided Engineering*, *14*(3), 187–212.

Ghosh-Dastidar, S., & Adeli, H. (2009). A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection. *Neural Networks*, *22*(10), 1419–1431. http://dx.doi.org/10.1016/j.neunet.2009.04.003.

Gilson, M., Bürck, M., Burkitt, A. N., & van Hemmen, J. L. (2012). Frequency selectivity emerging from spike-timing-dependent plasticity. *Neural Computation*, *24*(9), 2251–2279.

Glackin, C., Maguire, L., McDaid, L., & Sayers, H. (2011). Receptive field optimisation and supervision of a fuzzy spiking neural network. *Neural Networks*, *24*(3), 247–256.

Glackin, B., Wall, J. A., McGinnity, T. M., Maguire, L. P., & McDaid, L. J. (2010). A spiking neural network model of the medial superior olive using spike timing dependent plasticity for sound localization. *Frontiers in Computational Neuroscience*, *4*.

González-Nalda, P. (2009). STDP learning time window. Retrieved from https://lsi.vc.ehu.eus/pablogn/investig/pres%20Pulsos%20Ekaitz%20110330/STDP.png.

Guerguiev, J., Lillicrap, T. P., & Richards, B. A. (2017). Biologically feasible deep learning with segregated dendrites. *eLife*, *6*, 1–28. http://dx.doi.org/10.7554/eLife.22901.

Gütig, R., & Sompolinsky, H. (2006). The tempotron: A neuron that learns spike timing–based decisions. *Nature Neuroscience*, *9*(3), 420–428.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The elements of statistical learning: Data mining, inference, and prediction*. Retrieved from http://www-stat.stanford.edu/~tibs/ElemStatLearn/.

Haykin, S. S. (2009). *Neural networks and learning machines*. New York: Prentice Hall.

Hazan, H., & Manevitz, L. M. (2012). Topological constraints and robustness in liquid state machines. *Expert Systems with Applications*, *39*(2), 1597–1606.

Heiligenberg, W. (1991). *Neural nets in electric fish*. MIT Press.

Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, *29*(Nov.).

Hinton, G. E., Osindero, S., & Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527–1554. http://dx.doi.org/10.1162/neco.2006.18.7.1527.

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507. http://dx.doi.org/10.1126/science.1127647.

Hodgkin, A. L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, *117*(4), 500.

Hopfield, J. (1995). Pattern recognition computation using action potential timing for stimulus representation. *Nature*, *376*(6535), 33–36.

Huh, D., & Sejnowski, T. J. (2018). In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Gradient descent for spiking neural networks* (pp. 1433–1443). Curran Associates, Inc, Retrieved from http://papers.nips.cc/paper/7417-gradient-descent-for-spiking-neural-networks.pdf.

Hung, S., & Adeli, H. (1993). Parallel backpropagation learning algorithms on cray Y-MP8/864 supercomputer. *Neurocomputing*, *5*(6), 287–302.

Hung, S., & Adeli, H. (1994). Object-oriented backpropagation and its application to structural design. *Neurocomputing*, *6*(1), 45–55.

Illing, B., Gerstner, W., & Brea, J. (2019). Biologically plausible deep learning — but how far can we go with shallow networks? *Neural Networks*, *118*, 90–101. http://dx.doi.org/10.1016/j.neunet.2019.06.001.

Ito, M. (2000). Mechanisms of motor learning in the cerebellum1. *Brain Research*, *886*(1–2), 237–245. http://dx.doi.org/10.1016/S0006-8993(00)03142-5.

Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, *14*(6), 1569–1572.

Izhikevich, E. M. (2004). Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, *15*(5), 1063–1070.

Izhikevich, E. M. (2006). Polychronization: Computation with spikes. *Neural Computation*, *18*(2), 245–282.

Jedlicka, P. (2002). Synaptic plasticity metaplasticity and BCM theory. *Bratislavské Lekárske Listy*, *103*(4/5), 137–143.

Jörntell, H., & Hansel, C. (2006). Synaptic memories upside down: Bidirectional plasticity at cerebellar parallel fiber-purkinje cell synapses. *Neuron*, *52*(2), 227.

Joshi, P., & Maass, W. (2004). Movement generation and control with generic neural microcircuits. In *Biologically inspired approaches to advanced information technology* (pp. 258–273). Springer.

Ju, H., Xu, J., Chong, E., & VanDongen, A. M. J. (2013). Effects of synaptic connectivity on liquid state machine performance. *Neural Networks*, *38*(Feb.), 39–51. http://dx.doi.org/10.1016/j.neunet.2012.11.003.

Kampa, B. M., Letzkus, J. J., & Stuart, G. J. (2007). Dendritic mechanisms controlling spike-timing-dependent synaptic plasticity. *Trends in Neurosciences*, *30*(9), 456–463.

Kasabov, N., Dhoble, K., Nuntalid, N., & Indiveri, G. (2013). Dynamic evolving spiking neural networks for on-line spatio- and spectro-temporal pattern recognition. *Neural Networks*, *41*(May), 188–201. http://dx.doi.org/10.1016/j.neunet.2012.11.014.

Katz, B., & Miledi, R. (1965). The measurement of synaptic delay, and the time course of acetylcholine release at the neuromuscular junction. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, *161*(985), 483–495.

Kheradpisheh, S. R., Ganjtabesh, M., & Masquelier, T. (2016). Bio-inspired unsupervised learning of visual features leads to robust invariant object recognition. *Neurocomputing*, *205*(Sep.), 382–392. http://dx.doi.org/10.1016/j.neucom.2016.04.029.

Kheradpisheh, S. R., Ganjtabesh, M., Thorpe, S. J., & Masquelier, T. (2018). STDP-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, *99*(Mar.), 56–67. http://dx.doi.org/10.1016/j.neunet.2017.12.005.

Knudsen, E. I. (2002). Instructed learning in the auditory localization pathway of the barn owl. *Nature*, *417*(6886), 322–328.

Koch, C., & Segev, I. (1998). *Methods in neuronal modeling: From ions to networks*. MIT press.

Kohonen, T. (2013). Essentials of the self-organizing map. *Neural Networks*, *37*(Jan.), 52–65. http://dx.doi.org/10.1016/j.neunet.2012.09.018.

König, P., Engel, A. K., & Singer, W. (1996). Integrator or coincidence detector? the role of the cortical neuron revisited. *Trends in Neurosciences*, *19*(4), 130–137. http://dx.doi.org/10.1016/S0166-2236(96)80019-1.

Kuwabara, N., & Suga, N. (1993). Delay lines and amplitude selectivity are created in subthalamic auditory nuclei: The brachium of the inferior colliculus of the mustached bat. *Journal of Neurophysiology*, *69*(5), 1713.

Lameu, E., Batista, C., Batista, A., Iarosz, K., Viana, R., Lopes, S., et al. (2012). Suppression of bursting synchronization in clustered scale-free (rich-club) neuronal networks. *Chaos-Woodbury*, *22*(4), 043149.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(436), http://dx.doi.org/10.1038/nature14539.

Lee, C., Panda, P., Srinivasan, G., & Roy, K. (2018). Training deep spiking convolutional neural networks with STDP-based unsupervised pre-training followed by supervised fine-tuning. *Frontiers in Neuroscience*, *12*(435), http://dx.doi.org/10.3389/fnins.2018.00435.

Lee, C., Sarwar, S. S., & Roy, K. (2019). Enabling spike-based backpropagation in state-of-the-art deep neural network architectures. *113*(1), 54–66, This paper is avilable Online in: https://arxiv.org/abs/1903.06379v3.

Legenstein, R., Naeger, C., & Maass, W. (2005). What can a neuron learn with spike-timing-dependent plasticity? *Neural Computation*, *17*(11), 2337–2382.

Letzkus, J. J., Kampa, B. M., & Stuart, G. J. (2006). Learning rules for spike timing-dependent plasticity depend on dendritic synapse location. *The Journal of Neuroscience*, *26*(41), 10420–10429.

Lin, J., & Faber, D. S. (2002). Modulation of synaptic delay during synaptic plasticity. *Trends in Neurosciences*, *25*(9), 449–455.

Liu, S., Delbruck, T., Indiveri, G., Whatley, A., & Douglas, R. (2015). *Event-based neuromorphic systems*. John Wiley & Sons.

Maass, W. (1996). Lower bounds for the computational power of networks of spiking neurons. *Neural Computation*, *8*(1), 1–40. http://dx.doi.org/10.1162/neco.1996.8.1.1.

Maass, W., Natschläger, T., & Markram, H. (2004). Computational models for generic cortical microcircuits. In *Computational neuroscience: A comprehensive approach (vol. 18)* (pp. 575–605).

Maass, W., & Zador, A. (1999). Computing and learning with dynamic synapses. *Pulsed Neural Networks*, *6*(May), 321–336.

Maggi, S., Peyrache, A., & Humphries, M. D. (2018). An ensemble code in medial prefrontal cortex links prior events to outcomes during learning. *Nature Communications*, *9*(1), http://dx.doi.org/10.1038/s41467-018-04638-2.

Masquelier, T., & Deco, G. (2013). Learning and coding in neural networks. In *Principles of neural coding* (pp. 513–526). CRC Press.

Masquelier, T., & Thorpe, S. J. (2007). Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS Computational Biology*, *3*(2), 0247–0257. http://dx.doi.org/10.1371/journal.pcbi.0030031.

McKennoch, S., Liu, D., & Bushnell, L. G. (2006). Fast modifications of the spikeprop algorithm. In *Paper presented at the IJCNN'06. International joint conference on neural networks* (pp. 3970–3977).

Memmesheimer, R., Rubin, R., Ölveczky, B., & Sompolinsky, H. (2014). Learning precisely timed spikes. *Neuron*, *82*(4), 925–938. http://dx.doi.org/10.1016/j.neuron.2014.03.026.

Mesnard, T., Gerstner, W., & Brea, J. (2016). Towards deep learning with spiking neurons in energy based models with contrastive hebbian plasticity. arXiv:1612.03214.

Minneci, F., Kanichay, R. T., & Silver, R. A. (2012). Estimation of the time course of neurotransmitter release at central synapses from the first latency of postsynaptic currents. *Journal of Neuroscience Methods*, *205*(1), 49–64.

Mohemmed, A., Schliebs, S., Matsuda, S., & Kasabov, N. (2012). SPAN: Spike pattern association neuron for learning spatio-temporal spike patterns. *International Journal of Neural Systems*, *22*(04).

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. http://dx.doi.org/10.1016/j.neucom.2012.08.034.

Morrison, A., Diesmann, M., & Gerstner, W. (2008). Phenomenological models of synaptic plasticity based on spike timing. *Biological Cybernetics*, *98*(6), 459–478.

Mostafa, H. (2018). Supervised learning based on temporal coding in spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *29*(7), 3227–3235. http://dx.doi.org/10.1109/TNNLS.2017.2726060.

Neftci, E. O. (2018). Data and power efficient intelligence with neuromorphic learning machines. *iScience*, *5*(July), 52–68. http://dx.doi.org/10.1016/j.isci.2018.06.010.

Neftci, E., Augustine, C., Paul, S., & Detorakis, G. (2016). Event-driven random back-propagation: Enabling neuromorphic deep learning machines. *Frontiers in Neuroscience*, *11*(324), http://dx.doi.org/10.3389/fnins.2017.00324.

Neftci, E., Das, S., Pedroni, B., Kreutz-Delgado, K., & Cauwenberghs, G. (2014). Event-driven contrastive divergence for spiking neuromorphic systems. *Frontiers in Neuroscience*, *7*(8), 1–14. http://dx.doi.org/10.3389/fnins.2014.00272.

Neftci, E. O., Mostafa, H., & Zenke, F. (2019). Surrogate gradient learning in spiking neural networks. Retrieved from https://arxiv.org/abs/1901.09948.

Orchard, G., Meyer, C., Etienne-Cummings, R., Posch, C., Thakor, N., & Benosman, R. (2015). Hfirst: A temporal approach to object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(10), 2028–2040. http://dx.doi.org/10.1109/TPAMI.2015.2392947.

Panda, P., & Roy, K. (2016). Unsupervised regenerative learning of hierarchical features in spiking deep networks for object recognition. In *Paper presented at the IEEE international joint conference on neural networks* (pp. 299–306). Retrieved from https://arxiv.org/abs/1602.01510.

Parnas, I., & Parnas, H. (2010). Control of neurotransmitter release: From Ca2 to voltage dependent G-protein coupled receptors. *Pflügers Archiv-European Journal of Physiology*, *460*(6), 975–990.

Paugam-Moisy, H., & Bohte, S. (2012). Computing with spiking neuron networks. In *Handbook of natural computing* (pp. 335–376). Springer.

Paugam-Moisy, H., Martinez, R., & Bengio, S. (2008). Delay learning and polychronization for reservoir computing. *Neurocomputing*, *71*(7), 1143–1158.

Pfeiffer, M., & Pfeil, T. (2018). Deep learning with spiking neurons: Opportunities and challenges. *Frontiers in Neuroscience*, *12*, http://dx.doi.org/10.3389/fnins.2018.00774.

Pfister, J. P., & Gerstner, W. (2006). Triplets of spikes in a model of spike timing-dependent plasticity. *The Journal of Neuroscience*, *26*(38), 9673–9682. http://dx.doi.org/10.1523/JNEUROSCI.1425-06.2006.

Pfister, J., Toyoizumi, T., Barber, D., & Gerstner, W. (2006). Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning. *Neural Computation*, *18*(6), 1318–1348.

Pham, D., Packianather, M., & Charles, E. (2008). Control chart pattern clustering using a new self-organizing spiking neural network. *Proceedings of the Institution of Mechanical Engineers, Part B (Management and Engineering Manufacture)*, *222*(10), 1201–1211.

Ponulak, F. (2005). *ReSuMe—New supervised learning method for spiking neural networks. (no. 1)*. Institute of Control and Information Engineering, Poznan University of Technology, Retrieved from http://d1cie.put.poznan.pl/pracownicy/prac_15/Publikacje/ReSuMe_FP_TechRep_2005a.pdf.

Ponulak, F., & Kasiński, A. (2010). Supervised learning in spiking neural networks with ReSuMe: Sequence learning, classification, and spike shifting. *Neural Computation*, *22*(2), 467–510.

Ponulak, F., & Kasinski, A. (2011). Introduction to spiking neural networks: Information processing, learning and applications. *Acta Neurobiologiae Experimentalis*, *71*(4), 409–433.

Ramesh, B., Yang, H., Orchard, G. M., Le Thi, N. A., Zhang, S., & Xiang, C. (2019). DART: Distribution aware retinal transform for event-based cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *8828*, 1. http://dx.doi.org/10.1109/tpami.2019.2919301.

Rueckauer, B., Lungu, I., Hu, Y., & Pfeiffer, M. (2017). Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, *11*, 682. http://dx.doi.org/10.3389/fnins.2017.00682.

Ruf, B., & Schmitt, M. (1997). Learning temporally encoded patterns in networks of spiking neurons. *Neural Processing Letters*, *5*(1), 9–18.

Rullen, R. V., & Thorpe, S. J. (2001). Rate coding versus temporal order coding: What the retinal ganglion cells tell the visual cortex. *Neural Computation*, *13*(6), 1255–1283.

Schrauwen, B., D'Haene, M., Verstraeten, D., & Campenhout, J. V. (2008). Compact hardware liquid state machines on FPGA for real-time speech recognition. *Neural Networks*, *21*(2–3), 511–523. http://dx.doi.org/10.1016/j.neunet.2007.12.009.

Schultz, W., Dayan, P., & Montague, P. R. (1997). A neural substrate of prediction and reward. *Science*, *275*(5306), 1593–1599.

Seth, A. (2015). Neural coding: Rate and time codes work together. *Current Biology*, *25*(3), R110–R113. http://dx.doi.org/10.1016/j.cub.2014.12.043.

Shrestha, S. B. (2018). SLAYER : Spike layer error reassignment in time. In *Paper presented at the Advances in neural information processing systems* (pp. 1412–1421). Retrieved from http://papers.nips.cc/paper/7415-slayer-spike-layer-error-reassignment-in-time.

Shrestha, S. B., & Song, Qing (2013). Weight convergence of SpikeProp and adaptive learning rate. In *Paper presented at the Communication, control, and computing (Allerton), 2013 51st annual allerton conference on* (pp. 506–511) http://dx.doi.org/10.1109/Allerton.2013.6736567.

Shrestha, S. B., & Song, Q. (2015). Adaptive learning rate of SpikeProp based on weight convergence analysis. *Neural Networks*, *63*(Mar.), 185–198. http://dx.doi.org/10.1016/j.neunet.2014.12.001.

Silva, S. M., & Ruano, A. E. (2005). Application of Levenberg–Marquardt method to the training of spiking neural networks. In *Paper presented at the Neural networks and brain, 2005. ICNN & B'05. International conference on (vol. 3)* (pp. 1354–1358).

Sporea, I., & Grüning, A. (2013). Supervised learning in multilayer spiking neural networks. *Neural Computation*, *25*(2), 473–509.

Srinivasa, N., & Cho, Y. (2012). Self-organizing spiking neural model for learning fault-tolerant spatio-motor transformations. *IEEE Transactions on Neural Networks and Learning Systems*, *23*(10), 1526–1538.

Srivastava, N., & Salakhutdinov, R. (2014). Multimodal learning with deep Boltzmann machines. *Journal of Machine Learning Research (JMLR)*, *15*(1), 2949–2980.

Swadlow, H. A. (1992). Monitoring the excitability of neocortical efferent neurons to direct activation by extracellular current pulses. *Journal of Neurophysiology*, *68*(2), 605–619.

Taherkhani, A., Belatreche, A., Li, Y., & Maguire, L. P. (2014). A new biologically plausible supervised learning method for spiking neurons. In *Paper presented at the Proc. ESANN* (pp. 11–16).

Taherkhani, A., Belatreche, A., Li, Y., & Maguire, L. P. (2015a). In S. Arik, T. Huang, W. K. Lai, & Q. Liu (Eds.), *EDL: An extended delay learning based remote supervised method for spiking neurons* (pp. 190–197). Springer International Publishing, Retrieved from http://dx.doi.org/10.1007/978-3-319-26535-3_22.

Taherkhani, A., Belatreche, A., Li, Y., & Maguire, L. P. (2015b). DL-ReSuMe: A delay learning based remote supervised method for spiking neurons. *IEEE Transactions on Neural Networks and Learning Systems*, *26*(12), 3137–3149. http://dx.doi.org/10.1109/TNNLS.2015.2404938.

Taherkhani, A., Belatreche, A., Li, Y., Member, S., & Maguire, L. P. (2018). A supervised learning algorithm for learning precise timing of multiple spikes in multilayer. *IEEE Transactions on Neural Networks and Learning Systems*, *29*(11), http://dx.doi.org/10.1109/TNNLS.2018.2797801.

Taherkhani, A., Cosma, G., & McGinnity, T. M. (2018). Deep-FS: A feature selection algorithm for deep boltzmann machines. *Neurocomputing*, *322*(Dec.), 22–37. http://dx.doi.org/10.1016/j.neucom.2018.09.040.

Taherkhani, A., Cosma, G., & Mcginnity, T. M. (2019). Optimization of output spike train encoding for a spiking neuron based on its spatiotemporal input pattern. *IEEE Transactions on Cognitive and Developmental Systems*, 1. http://dx.doi.org/10.1109/TCDS.2019.2909355.

Takase, H., Fujita, M., Kawanaka, H., Tsuruoka, S., Kita, H., & Hayashi, T. (2009). Obstacle to training SpikeProp networks — cause of surges in training process —. In *Paper presented at the Neural networks, 2009. IJCNN 2009. International joint conference on* (pp. 3062–3066). http://dx.doi.org/10.1109/IJCNN.2009.5178756.

Tavanaei, A., Ghodrati, M., & Reza, S. (2019). Deep learning in spiking neural networks. *Neural Networks*, *111*(Mar.), 47–63. http://dx.doi.org/10.1016/j.neunet.2018.12.002.

Tavanaei, A., Masquelier, T., & Maida, A. S. (2016). Acquisition of visual features through probabilistic spike-timing-dependent plasticity. In *Proceedings of the international joint conference on neural networks, 2016-Octob* (pp. 307–314). http://dx.doi.org/10.1109/IJCNN.2016.7727213.

Tavanaei, A., Masquelier, T., & Maida, A. (2018). Representation learning using event-based STDP. *Neural Networks*, *105*(Sep.), 294–303. http://dx.doi.org/10.1016/j.neunet.2018.05.018.

Tetzlaff, C., Kolodziejski, C., Markelic, I., & Wörgötter, F. (2012). Time scales of memory, learning, and plasticity. *Biological Cybernetics*, *106*(11–12), 715–726.

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

Turrigiano, G. G., & Nelson, S. B. (2004). Homeostatic plasticity in the developing nervous system. *Nature Reviews Neuroscience*, *5*(2), 97–107.

Vasilaki, E., & Giugliano, M. (2013). Emergence of connectivity motifs in networks of model neurons with short-and long-term plastic synapses. arXiv preprint arXiv:1301.7187.

Verstraeten, D., Schrauwen, B., Stroobandt, D., & Van Campenhout, J. (2005). Isolated word recognition with the liquid state machine: A case study. *Information Processing Letters*, *95*(6), 521–528. http://dx.doi.org/10.1016/j.ipl.2005.05.019.

Vreeken, J. (2003). *Spiking neural networks, an introduction: Technical Report UU-CS, (2003-008)*, (pp. 1–5).

Wade, J. J., McDaid, L. J., Santos, J. A., & Sayers, H. M. (2010). SWAT: A spiking neural network training algorithm for classification problems. *IEEE Transactions on Neural Networks*, *21*(11), 1817–1830.

Wang, J., Belatreche, A., Maguire, L. p., & McGinnity, T. M. (2014). An online supervised learning method for spiking neural networks with adaptive structure. *Neurocomputing*, *144*(Nov.), 526–536. http://dx.doi.org/10.1016/j.neucom.2014.04.017.

Wang, H., Gerkin, R. C., Nauen, D. W., & Bi, G. (2005). Coactivation and timing-dependent integration of synaptic potentiation and depression. *Nature Neuroscience*, *8*(2), 187–193.

Whittington, J. C. R., & Bogacz, R. (2019). Theories of error back-propagation in the brain. *Trends in Cognitive Sciences*, *23*(3), 235–250. http://dx.doi.org/10.1016/j.tics.2018.12.005.

Wu, J., Chua, Y., Zhang, M., Yang, Q., Li, G., & Li, H. (2019). Deep spiking neural network with spike count based learning rule. Retrieved from https://arxiv.org/abs/1902.05705.

Wysoski, S. G., Benuskova, L., & Kasabov, N. (2008). Fast and adaptive network of spiking neurons for multi-view visual pattern recognition. *Neurocomputing*, *71*(13), 2563–2575.

Xin, J., & Embrechts, M. J. (2001). Supervised learning with spiking neural networks. In *Paper presented at the Neural networks, 2001. Proceedings. IJCNN'01. International joint conference on (vol. 3)* (pp. 1772–1777).

Xu, B., Gong, Y., & Wang, B. (2013). Delay-induced firing behavior and transitions in adaptive neuronal networks with two types of synapses. *Science China Chemistry*, *56*(2), 222–229.

Xu, Y., Yang, J., & Zeng, X. (2019). An optimal time interval of input spikes involved in synaptic adjustment of spike sequence learning. *Neural Networks*, *116*(Aug.), 11–24. http://dx.doi.org/10.1016/j.neunet.2019.03.017.

Xu, Y., Zeng, X., Han, L., & Yang, J. (2013). A supervised multi-spike learning algorithm based on gradient descent for spiking neural networks. *Neural Networks*, *43*(Jul.), 99–113. http://dx.doi.org/10.1016/j.neunet.2013.02.003.

Yu, Q., Tan, K. C., & Tang, H. (2012). Pattern recognition computation in a spiking neural network with temporal encoding and learning. In *Paper presented at the Neural networks (IJCNN), the 2012 international joint conference on* (pp. 1–7).

Yu, Q., Tang, H., Tan, K. C., & Li, H. (2013a). Precise-spike-driven synaptic plasticity: Learning hetero-association of spatiotemporal spike patterns. *PLoS One*, *8*(11), http://dx.doi.org/10.1371/journal.pone.0078318.

Yu, Q., Tang, H., Tan, K. C., & Li, H. (2013b). Rapid feedforward computation by temporal encoding and learning with spiking neurons. *IEEE Transactions on Neural Networks and Learning Systems*, *24*(10), 1539–1552. http://dx.doi.org/10.1109/TNNLS.2013.2245677.

Yu, Q., Tang, H., Tan, K. C., & Yu, H. (2014). A brain-inspired spiking neural network model with temporal encoding and learning. *Neurocomputing*, *138*(August), 3–13. http://dx.doi.org/10.1016/j.neucom.2013.06.052.

Zenke, F., & Ganguli, S. (2018). SuperSpike: Supervised learning in multilayer spiking neural networks. *Neural Computation*, *30*(6), 1514–1541. http://dx.doi.org/10.1162/neco_a_01086.

Zhao, B., Ding, R., Chen, S., Linares-Barranco, B., & Tang, H. (2015). Feedforward categorization on AER motion events using cortex-like features in a spiking neural network. *IEEE Transactions on Neural Networks and Learning Systems*, *26*(9), 1963–1978. http://dx.doi.org/10.1109/TNNLS.2014.2362542.

Zuo, Y., Safaai, H., Notaro, G., Mazzoni, A., Panzeri, S., & Diamond, M. (2015). Complementary contributions of spike timing and spike rate to perceptual decisions in rat S1 and S2 cortex. *Current Biology*, *25*(3), 357–363. http://dx.doi.org/10.1016/j.cub.2014.11.065.