

## Review

## Deep learning in spiking neural networks

Amirhossein Tavanaei<sup>a,\*</sup>, Masoud Ghodrati<sup>b</sup>, Saeed Reza Kheradpisheh<sup>c</sup>,  
Timothée Masquelier<sup>d</sup>, Anthony Maida<sup>a</sup>

<sup>a</sup> School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA 70504, USA

<sup>b</sup> Department of Physiology, Monash University, Clayton, VIC, Australia

<sup>c</sup> Department of Computer Science, Faculty of Mathematical Sciences and Computer, Kharazmi University, Tehran, Iran

<sup>d</sup> CERCO UMR 5549, CNRS-Université de Toulouse 3, F-31300, France

## ARTICLE INFO

## Article history:

Received 7 June 2018

Received in revised form 2 December 2018

Accepted 3 December 2018

Available online 18 December 2018

## Keywords:

Deep learning

Spiking neural network

Biological plausibility

Machine learning

Power-efficient architecture

## ABSTRACT

In recent years, deep learning has revolutionized the field of machine learning, for computer vision in particular. In this approach, a deep (multilayer) artificial neural network (ANN) is trained, most often in a supervised manner using backpropagation. Vast amounts of labeled training examples are required, but the resulting classification accuracy is truly impressive, sometimes outperforming humans.

Neurons in an ANN are characterized by a single, static, continuous-valued activation. Yet biological neurons use discrete spikes to compute and transmit information, and the spike times, in addition to the spike rates, matter. Spiking neural networks (SNNs) are thus more biologically realistic than ANNs, and are arguably the only viable option if one wants to understand how the brain computes at the neuronal description level. The spikes of biological neurons are sparse in time and space, and event-driven. Combined with bio-plausible local learning rules, this makes it easier to build low-power, neuromorphic hardware for SNNs. However, training deep SNNs remains a challenge. Spiking neurons' transfer function is usually non-differentiable, which prevents using backpropagation.

Here we review recent supervised and unsupervised methods to train deep SNNs, and compare them in terms of accuracy and computational cost. The emerging picture is that SNNs still lag behind ANNs in terms of accuracy, but the gap is decreasing, and can even vanish on some tasks, while SNNs typically require many fewer operations and are the better candidates to process spatio-temporal data.

© 2018 Elsevier Ltd. All rights reserved.

## Contents

1. Introduction.....	47
2. Spiking Neural Networks: A biologically inspired approach to information processing.....	49
2.1. SNN architecture .....	49
2.2. Learning rules in SNNs.....	49
2.2.1. Unsupervised learning via STDP.....	49
2.2.2. Probabilistic characterization of unsupervised STDP.....	50
2.2.3. Supervised learning.....	50
3. Deep learning in SNNs.....	52
3.1. Deep, fully connected SNNs.....	53
3.2. Spiking CNNs.....	53
3.3. Spiking deep belief networks.....	55
3.4. Recurrent SNNs.....	56
3.4.1. Gated SNNs.....	56
3.4.2. Liquid state machines and reservoirs.....	57
3.5. Performance comparisons of contemporary models.....	58
4. Summary.....	58
References.....	59

\* Corresponding author.

E-mail address: [tavanaei@louisiana.edu](mailto:tavanaei@louisiana.edu) (A. Tavanaei).

## 1. Introduction

Artificial neural networks (ANNs) are predominantly built using idealized computing units with continuous activation values and a

set of weighted inputs. These units are commonly called ‘neurons’ because of their biological inspiration. These (non-spiking) neurons use differentiable, non-linear activation functions. The non-linear activation functions make it representationally meaningful to stack more than one layer and the existence of their derivatives makes it possible to use gradient-based optimization methods for training. With recent advances in availability of large labeled data sets, computing power in the form of general purpose GPU computing, and advanced regularization methods, these networks have become very deep (dozens of layers) with great ability to generalize unseen data and there have been huge advances in the performance of such networks.

A distinct historical landmark is the 2012 success of AlexNet (Krizhevsky, Sutskever, & Hinton, 2012) in the ILSVRC image classification challenge (Russakovsky et al., 2015). AlexNet became known as a deep neural network (DNN) because it consisted of about eight sequential layers of end-to-end learning, totaling 60 million trainable parameters. For recent reviews of DNNs, see LeCun, Bengio, and Hinton (2015a) and Schmidhuber (2015). DNNs have been remarkably successful in many applications including image recognition (He, Zhang, Ren, & Sun, 2016; Krizhevsky et al., 2012; Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2016), object detection (Girshick, Donahue, Darrell, & Malik, 2014; Long, Shelhamer, & Darrell, 2015), speech recognition (Hinton et al., 2012), biomedicine and bioinformatics (Mamoshina, Vieira, Putin, & Zharovonkov, 2016; Min, Lee, & Yoon, 2017), temporal data processing (Venna, Tavanaei, Gottumukkala, Raghavan, Maida, & Nichols, 2017), and many other applications (Hassabis, Kumaran, Summerfield, & Botvinick, 2017; Schmidhuber, 2015; VanRullen, 2017). These recent advances in artificial intelligence (AI) have opened up new avenues for developing different engineering applications and understanding of how biological brains work (Hassabis et al., 2017; VanRullen, 2017).

Although DNNs are historically brain-inspired, there are fundamental differences in their structure, neural computations, and learning rule compared to the brain. One of the most important differences is the way that information propagates between their units. It is this observation that leads to the realm of spiking neural networks (SNNs). In the brain, the communication between neurons is done by broadcasting trains of action potentials, also known as spike trains to downstream neurons. These individual spikes are sparse in time, so each spike has high information content, and to a first approximation has uniform amplitude (100 mV with spike width about 1 msec). Thus, information in SNNs is conveyed by spike timing, including latencies, and spike rates, possibly over populations (Gerstner, Kistler, Naud, & Paninski, 2014). SNNs almost universally use idealized spike generation mechanisms in contrast to the actual biophysical mechanisms (Hodgkin & Huxley, 1952).

ANNs, that are non-spiking DNNs, communicate using continuous valued activations. Although the energy efficiency of DNNs can likely be improved, SNNs offer a special opportunity in this regard because, as explained below, spike events are sparse in time. Spiking networks also have the advantage of being intrinsically sensitive to the temporal characteristics of information transmission that occurs in the biological neural systems. It has been shown that the precise timing of every spike is highly reliable for several areas of the brain and suggesting an important role in neural coding (Bair & Koch, 1996; Herikstad, Baker, Lachaux, Gray, & Yen, 2011; Mainen & Sejnowski, 1995). This precise temporal pattern in spiking activity is considered as a crucial coding strategy in sensory information processing areas (Butts et al., 2007; Golisch & Meister, 2008; Reinagel & Reid, 2000; Sinha et al., 2017; Victor, 2005) and neural motor control areas in the brain (Srivastava et al., 2017; Tang, Chehayeb, Srivastava, Nemenman, & Sober, 2014). SNNs have become the focus of a number of recent applications in many areas

of pattern recognition such as visual processing (Escobar, Masson, Vieville, & Kornprobst, 2009; Gupta & Long, 2007; Meftah, Lezoray, & Benyettou, 2010; Wysoski, Benuskova, & Kasabov, 2010), speech recognition (Kröger, Kannampuzha, & Neuschaefer-Rube, 2009; Liaw & Berger, 1998; Loisel, Rouat, Pressnitzer, & Thorpe, 2005; Näger, Storck, & Deco, 2002; Panchev & Wermter, 2004; Tavanaei & Maida, 2017a; Wade, McDaid, Santos, & Sayers, 2010), and medical diagnosis (Ghosh-Dastidar & Adeli, 2007; Kasabov et al., 2014). In recent years, a new generation of neural networks that incorporates the multilayer structure of DNNs (and the brain) and the type of information communication in SNNs has emerged. These deep SNNs are great candidates to investigate neural computation and different coding strategies in the brain.

In regard to the scientific motivation, it is well accepted that the ability of the brain to recognize complex visual patterns or identify auditory targets in a noisy environment is a result of several processing stages and multiple learning mechanisms embedded in deep spiking networks (Felleman & Van Essen, 1991; Freiwald & Tsao, 2010; Serre, 2014). In comparison to traditional deep networks, training deep spiking networks is in its early phases. It is an important scientific question to understand how such networks can be trained to perform different tasks as this can help us to generate and investigate novel hypotheses, such as rate versus temporal coding, and develop experimental ideas prior to performing physiological experiments. On the other hand, SNNs provide neural architectures to better process spatio-temporal data, especially in an online mode (Kasabov, 2012, 2014) and temporal binary activation of digital systems (Vreeken, 2003; Yu, Tang, Tan, & Yu, 2014). Knowledge representation in time and space makes SNNs unique to perform brain-like computations and to understand the brain data/activity in a spatio-temporal pattern. More details about the knowledge representation provided by SNNs have been discussed in Kasabov (2018).

In regard to the engineering motivation, SNNs have some advantages over traditional neural networks in regard to implementation in special purpose hardware. At the present, effective training of traditional deep networks requires the use of energy intensive high-end graphic cards. Spiking networks have the interesting property that the output spike trains can be made sparse in time. An advantage of this in biological networks is that the spike events consume energy and that using few spikes which have high information content reduces energy consumption (Stone, 2018). This same advantage is maintained in hardware (Carrillo et al., 2013, 2012; Merolla et al., 2014; Seo et al., 2011). Thus, it is possible to create low energy spiking hardware which is highly responsive to event-based sensors based on the property that spikes are sparse in time (Pfeiffer & Pfeil, 2018).

An important part of the learning in deep neural models, both spiking and non-spiking, occurs in the feature discovery hierarchy, where increasingly complex, discriminative, abstract, and invariant features are acquired (Bengio, 2009). Given the scientific and engineering motivations mentioned above, deep SNNs provide appropriate architectures for developing an efficient, brain-like representation. Also, pattern recognition in the primate's brain is done through multi-layer neural circuits that communicate by spiking events. This naturally leads to interest in using artificial SNNs in applications that brains are good at, such as pattern recognition (Maass, 2015). Bio-inspired SNNs, in principle, have higher representation power and capacity than traditional rate-coded networks (Maass, 1997). Furthermore, SNNs allow a type of bio-inspired learning (weight modification) that depends on the relative timing of spikes between pairs of directly connected neurons in which the information required for weight modification is locally available. This local learning resembles the remarkable learning that occurs in many areas of the brain (Chavez-Noriega, Halliwell, & Bliss, 1990; Liu, Pu, & Poo, 2005; Rozenberg, Bck, & Kok, 2011; Seung, 2003; Song, Pan, & Păun, 2013).

The spike trains are represented formally by sums of Dirac delta functions and do not have derivatives. This makes it difficult to use derivative-based optimization for training SNNs, although very recent work has explored the use of various types of substitute or approximate derivatives (Huh & Sejnowski, 2017; Lee, Delbruck, & Pfeiffer, 2016). This raises a question: How are neural networks in the brain trained if derivative-based optimization is not available? Although spiking networks have theoretically been shown to have Turing-equivalent computing power (Maass, 1996), it remains a challenge to train SNNs, especially deep SNNs using multi-layer learning. In many existing spiking networks, learning is restricted to a single layer, for example (Beyeler, Dutt, & Krichmar, 2013; Masquelier & Thorpe, 2007; Tavanaei, Masquelier, & Maida, 2016b). Equipping spiking networks with multi-layer learning is an open area that has potential to greatly improve their performance on different tasks. The main core of the previous research is based on the fact that coding with the timing of spikes carries useful information and exhibits great computational power in biological systems (Butts et al., 2007; Reinagel & Reid, 2000; Srivastava et al., 2017).

Here, we review recent studies in developing deep learning models in SNNs with the focus on: (1) describing the SNNs' architectures and their learning approaches; (2) reviewing deep SNNs composed of feedforward, fully connected spiking neural layers; (3) spiking convolutional neural networks; (4) reviewing spiking restricted Boltzmann machines and spiking deep belief networks; (5) reviewing recurrent SNNs; and (6) providing a comprehensive summary comparing the performance of recent deep spiking networks. We hope that this review will help researchers in the area of artificial neural networks to develop and extend efficient and high-performance deep SNNs and will also foster a cross-fertilization in future experimental and theoretical work in neuroscience.

## 2. Spiking Neural Networks: A biologically inspired approach to information processing

The introduction of SNNs in the last few decades, as a powerful third generation neural network (Maass, 1997), has encouraged many studies with the focus on biologically motivated approaches for pattern recognition (Ghosh-Dastidar & Adeli, 2009b; Kasabov, Dhoble, Nuntalid, & Indiveri, 2013). SNNs were originally inspired by the brain and the communication scheme that neurons use for information transformation via discrete action potentials (spikes) in time through adaptive synapses. In a biological neuron, a spike is generated when the running sum of changes in the membrane potential, which can result from presynaptic stimulation, crosses a threshold. The rate of spike generation and the temporal pattern of spike trains carry information about external stimuli (Gerstner & Kistler, 2002; Rieke, 1999) and ongoing calculations. SNNs use a very similar process for spike generation and information transformation. In the following sections, we explain the details of SNN architectures and learning methods applied to these types of networks.

### 2.1. SNN architecture

An SNN architecture consists of spiking neurons and interconnecting synapses that are modeled by adjustable scalar weights. The first step in implementing an SNN is to encode the analog input data into the spike trains using either a rate based method (Gerstner & Kistler, 2002; Gerstner et al., 2014), some form of temporal coding (Bohte, 2004; Hopfield et al., 1995), or population coding (Bohte, La Poutré, & Kok, 2002b). As stated earlier, a biological neuron in the brain (and similarly in a simulated spiking neuron) receives synaptic inputs from other neurons in the neural network. Biological neural networks have both action potential generation

dynamics and network dynamics. In comparison to true biological networks, the network dynamics of artificial SNNs are highly simplified. In this context, it is useful to assume that the modeled spiking neurons have pure threshold dynamics (in contrast to, e.g., refractoriness, hysteresis, resonance dynamics, or post-inhibitory rebound properties). The activity of pre-synaptic neurons modulates the membrane potential of postsynaptic neurons, generating an action potential or spike when the membrane potential crosses a threshold. Hodgkin and Huxley were the first to model this phenomenon (Hodgkin & Huxley, 1952). Specifically, they created a model of action potential generation from the voltage gating properties of the ion channels in the squid cell membrane of the squid axon. After the Hodgkin and Huxley model with extensive biological details and high computational cost (Gerstner & Kistler, 2002; Hodgkin & Huxley, 1952; Kistler, Gerstner, & van Hemmen, 1997), diverse neuron models have been proposed such as the spike response model (SRM) (Jolivet, Timothy, & Gerstner, 2003), the Izhikevich neuron model (Izhikevich et al., 2003), and the leaky integrated-and-fire (LIF) neuron (Delorme, Gautrais, Van Rullen, & Thorpe, 1999). The LIF model is extremely popular because it captures the intuitive properties of external input accumulating charge across a leaky cell membrane with a clear threshold.

Spike trains in a network of spiking neurons are propagated through synaptic connections. A synapse can be either excitatory, which increases the neuron's membrane potential upon receiving input, or inhibitory, which decreases the neuron's membrane potential (Kandel, Schwartz, Jessell, Siegelbaum, & Hudspeth, 2013). The strength of the adaptive synapses (weights) can be changed as a result of learning. The learning rule of an SNN is its most challenging component for developing multi-layer (deep) SNNs, because the non-differentiability of spike trains limits the popular backpropagation algorithm.

### 2.2. Learning rules in SNNs

As previously mentioned, in virtually all ANNs, spiking or non-spiking, learning is realized by adjusting scalar-valued synaptic weights. Spiking enables a type of bio-plausible learning rule that cannot be directly replicated in non-spiking networks. Neuroscientists have identified many variants of this learning rule that falls under the umbrella term spike-timing-dependent plasticity (STDP). Its key feature is that the weight (synaptic efficacy) connecting a pre- and post-synaptic neuron is adjusted according to their relative spike times within an interval of roughly tens of milliseconds in length (Caporale & Dan, 2008). The information used to perform the weight adjustment is both local to the synapse and local in time. The following subsections describe common learning mechanisms in SNNs, both unsupervised and supervised.

#### 2.2.1. Unsupervised learning via STDP

As stated above, unsupervised learning in SNNs often involves STDP as part of the learning mechanism (Caporale & Dan, 2008; Markram, Gerstner, & Sjöström, 2011). The most common form of biological STDP has a very intuitive interpretation. If a presynaptic neuron fires briefly (e.g.,  $\approx 10$  ms) before the postsynaptic neuron, the weight connecting them is strengthened. If the presynaptic neuron fires briefly after the postsynaptic neuron, then the causal relationship between the temporal events is spurious and the weight is weakened. Strengthening is called long-term potentiation (LTP) and weakening is called long-term depression (LTD). The phrase "long-term" is used to distinguish between very transient effects on the scale of a few ms that are observed in experiments.

Formula (1) idealizes the most common experimentally observed STDP rule for a single pair of spikes obtained by fitting to experimental data (Dan & Poo, 2006).

$$\Delta w = \begin{cases} Ae^{-\frac{(t_{pre} - t_{post})}{\tau}} & t_{pre} - t_{post} \leq 0 \quad A > 0 \\ Be^{-\frac{(t_{post} - t_{pre})}{\tau}} & t_{pre} - t_{post} > 0 \quad B < 0 \end{cases} \quad (1)$$



$w$  is the synaptic weight.  $A > 0$  and  $B < 0$  are usually constant parameters indicating learning rates.  $\tau$  is the time constant (e.g., 15 ms) for the temporal learning window. The first of the above cases describes LTP while the second describes LTD. The strength of the effect is modulated by a decaying exponential whose magnitude is controlled by the time-constant-scaled time difference between the pre- and postsynaptic spikes. Rarely, do artificial SNNs use this exact rule. They usually use a variant, either to achieve more simplicity or to satisfy a convenient mathematical property.

Besides the temporally and spatially local weight change described in Eq. (1), STDP has known important temporally accumulated network-level effects. For instance, STDP affects a neuron's behavior in response to repeated spike patterns embedded in a possibly stochastic spike train. A neuron (equipped with STDP-trained synapses) in coincidence with similar volleys of spikes is able to concentrate on afferents that consistently fire early (shorter latencies) (Guyonneau, VanRullen, & Thorpe, 2005; Song, Miller, & Abbott, 2000). Spike trains in many areas of the brain are highly reproducible. Guyonneau et al. (2005) have shown that presenting repeated inputs to an SNN equipped with STDP shapes neuronal selectivity to the stimulus patterns within the SNN. Specifically, they showed that the response latency of the postsynaptic potential is decreased as STDP proceeds. Reducing the postsynaptic latency results in faster neural processing. Thus, the neuron responds faster to a specific input pattern than to any other. In fact, the STDP rule focuses on the first spikes of the input pattern which contain most of the information needed for pattern recognition. It has been shown that repeating spatio-temporal patterns can be detected and learned by a single neuron based on STDP (Masquelier, Guyonneau, & Thorpe, 2008; Masquelier & Kheradpisheh, 2018). STDP can also solve difficult computational problems in localizing a repeating spatio-temporal spike pattern and enabling some forms of temporal coding, even if an explicit time reference is missing (Masquelier et al., 2008; Masquelier, Guyonneau, & Thorpe, 2009). Using this approach, more complex networks with multiple output neurons have been developed (Kheradpisheh, Ganjtabesh, & Masquelier, 2016a; Masquelier & Thorpe, 2007, 2010; Tavanaei & Maida, 2017b).

### 2.2.2. Probabilistic characterization of unsupervised STDP

Many studies offer evidence that at least an approximate Bayesian analysis of sensory stimuli occurs in the brain (Doya, 2007; Körding & Wolpert, 2004; Mozer, Pashler, & Homaei, 2008; Rao, Olshausen, & Lewicki, 2002). In Bayesian inference, hidden causes (such as presence of an object of a particular category) are inferred using both prior knowledge and the likelihood of new observations to obtain a posterior probability of the possible cause. Researchers have considered the possible role of probabilistic (Bayesian) computation as a primary information processing step in the brain in terms of STDP.

Nessler, Pfeiffer, and Maass (2009) showed that a form of STDP, when used with Poisson spiking input neurons coupled with the appropriate stochastic winner-take-all (WTA) circuit, is able to approximate a stochastic online expectation maximization (EM) algorithm to learn the parameters for a multinomial mixture distribution. The model was intended to have some biological plausibility. The STDP rule used in their network is shown in Eq. (2). LTP occurs if the presynaptic neuron fires briefly (e.g., within  $\epsilon = 10$  ms) before the postsynaptic neuron. Otherwise LTD occurs. Generating a spike by an output neuron creates a sample from the coded posterior distribution of hidden variables which can be considered as the E-step in the EM algorithm. The application of STDP to the synapses of fired output neurons specifies the M-step in EM. Nessler, Pfeiffer, Buesing, and Maass (2013) extended their network by using an inhibitory neuron to implement the WTA in

order to improve the compatibility of the model for embedding in a cortical microcircuit.

$$\Delta w_{ki} = \begin{cases} e^{-w_{ki}} - 1, & 0 < t_k^f - t_i^f < \epsilon \\ -1, & \text{otherwise} \end{cases} \quad (2)$$

Building on the stochastic WTA circuits described above, Klampfl and Maass (2013) developed a liquid state machine (LSM, Section 3.4.2) containing input neurons, a reservoir of the WTA circuits, and a linear output readout. Further extension showed that STDP, applied on both the lateral excitatory synapses and synapses from afferent neurons, can represent the underlying statistical structure of such spatio-temporal input patterns (Kappel, Nessler, & Maass, 2014). In this framework, each spike train generated by the WTA circuits can be viewed as a sample from the state space of a hidden Markov model (HMM).

One drawback of the STDP model introduced in Nessler et al. (2013, 2009) is that its excitatory synaptic weights are negative. This, however, can be solved by shifting the weights to a positive value by using a constant parameter in the LTP rule. Based on this idea, Tavanaei and Maida (2015b, 2018b) proposed an unsupervised rule for spatio-temporal pattern recognition and spoken word classification. It has been shown that the EM acquired in an SNN is able to approximately implement the EM algorithm in a Gaussian mixture model (GMM) embedded in the HMM states (Tavanaei & Maida, 2018b).

Using probabilistic learning in spiking neurons for modeling hidden causes has recently attracted attention. Rezende, Wierstra, and Gerstner (2011) developed a bio-plausible learning rule based on the joint distribution of perceptions and hidden causes to adapt spontaneous spike sequences to match the empirical distribution of actual spike sequences (Rezende et al., 2011). The learning strategy involved minimizing the Kullback–Leibler divergence (Kullback & Leibler, 1951) as a non-commutative distance measure between the distribution representing the model (SNN) and a target distribution (observation). The EM algorithm in recurrent SNNs (Brea, Senn, & Pfister, 2011) and probabilistic association between neurons generated by STDP in combination with intrinsic plasticity (Pecceviski & Maass, 2016) are two other instances of probabilistic learning in SNNs. The probabilistic rules also have been employed in sequential data processing (Zemel, Natarajan, Dayan, & Huys, 2004) and Markov chain Monte Carlo sampling interpreted by stochastic firing activity of spiking neurons (Buesing, Bill, Nessler, & Maass, 2011).

### 2.2.3. Supervised learning

All supervised learning uses labels of some kind. Most commonly, supervised learning adjusts weights via gradient descent on a cost function comparing observed and desired network outputs. In the context of SNNs, supervised learning tries to minimize the error between desired and output spike trains, sometimes called readout error, in response to inputs.

*SNN issues in relation to backpropagation.* From a biological vantage point, there has been considerable skepticism about whether the backpropagation training procedure can be directly implemented in the brain. With respect to SNNs, there are two prominent issues which can be seen from the formula below. Shown below is a core formula, obtained from the chain rule, that occurs in all variants of backpropagation (Bishop, 1995).

$$\delta_j^\mu = g'(a_j^\mu) \sum_k w_{kj} \delta_k^\mu \quad (3)$$

In the above,  $\delta_j^\mu$  and  $\delta_k^\mu$  denote the partial derivative of the cost function for input pattern  $\mu$  with respect to the net input to some arbitrary unit  $j$  or  $k$ . Unit  $j$  projects direct feedforward connections to the set of units indexed by  $k$ .  $g(\cdot)$  is the activation function

applied to the net input of unit  $j$ , where that net input is denoted  $a_j^\mu$ .  $w_{kj}$  are the feedforward weights projecting from unit  $j$  to the set of units indexed by  $k$ .

Both parts of the RHS of Eq. (3) present complications for bio-plausible spiking versions of backpropagation. First, the expression  $g'(\cdot)$  requires  $g(\cdot)$  with respect to  $w_{kj}$ . Since  $g(\cdot)$  applies to a spiking neuron, it is likely represented by a sum of Dirac delta functions, which means the derivative does not exist. The second, and more serious complication, applies to both spiking and non-spiking networks and was apparently first pointed out by Grossberg (1987, p. 49) and termed the “weight transport” problem. The problem is the following. The expression  $\sum_k w_{kj} \delta_k^\mu$  uses the feedforward weights  $w_{kj}$  in a feedback fashion. This means that matching symmetric feedback weights must exist and project accurately to the correct neurons (point-to-point feedback) in order for Eq. (3) to be usable.

In the literature, the first issue has generally been addressed by using substitute or approximate derivatives. One must be aware that some of these solutions are not bio-plausible. For example, using the membrane potential of the presynaptic neuron as a surrogate becomes problematic because its value is not local to the synapse (cf. Lee et al., 2016, Section 3.1 of this review). These approaches, however, are still useful from both engineering and scientific standpoints.

Progress on the second issue has recently been made by Lillicrap, Cownden, Tweed, and Akerman (2016) and Zenke and Ganguli (2017). It was shown in Lillicrap et al. (2016) that for some tasks, backpropagation could still perform well if random feedback weights were used. The authors in Zenke and Ganguli (2017) explored this further, examining three kinds of feedback (uniform, random, and symmetric). They found that simpler problems could be solved by any kind of feedback whereas complex problems needed symmetric feedback.

Another study that appears to make progress on the second issue is the introduction of mirrored STDP learning by Burbank (2015). This research built an STDP trained auto-encoder with matched-weight, point-to-point connections across layers. This could be an approach to implementing the infrastructure to support brain-plausible backpropagation. However, other complicating issues arise. One is stability of a large-scale brain architecture having strong feedback connections (no strong loops hypothesis (Crick & Koch, 1998)).

*Some supervised learning methods for SNNs.* SpikeProp (Bohte, Kok, & La Poutre, 2002a) appears to be the first algorithm to train SNNs by backpropagating errors. Their cost function took into account spike timing and SpikeProp was able to classify non-linearly separable data for a temporally encoded XOR problem using a 3-layer architecture. One of their key design choices was to use Gerstner’s (Gerstner & Kistler, 2002) spike-response model (SRM) for the spiking neurons. Using the SRM model, the issue of taking derivatives on the output spikes of the hidden units was avoided because those units’ responses could be directly modeled as continuous-valued PSPs applying to the output synapses that they projected to. One limitation of this work is that each output unit was constrained to discharge exactly one spike. Also, continuous variable values, such as in the temporally extended XOR problem, had to be encoded as spike-time delays which could be quite long.

Later advanced versions of SpikeProp, Multi-SpikeProp, were applicable in multiple spike coding (Booij & Nguyen, 2005; Ghosh-Dastidar & Adeli, 2009a). Using the same neural architecture of SpikeProp, new formulations of temporal spike coding and spike time errors have recently improved the spiking backpropagation algorithm (Liu et al., 2017; Mostafa, 2017). A recent implementation of backpropagation in SNNs was proposed by Wu, Deng,

Li, Zhu, and Shi (2017) who developed spatio-temporal gradient descent in multi-layer SNNs.

More recent approaches to supervised training of SNNs include ReSuMe (remote supervised learning) (Kasinski & Ponulak, 2006; Ponulak & Kasinski, 2010), Chronotron (Florian, 2012), and SPAN (spike pattern association neuron) (Mohammed, Schliebs, Matsuda, & Kasabov, 2012, 2013), among others. All of the above models consist of a single spiking neuron receiving inputs from many spiking presynaptic neurons. The goal is to train the synapses to cause the post-synaptic neuron to generate a spike train with desired spike times.

ReSuMe adapts the Widrow–Hoff (Delta) rule, originally used for non-spiking linear units, to SNNs. The Widrow–Hoff rule weight changes are proportional to the desired output minus the observed output, as shown below.

$$\Delta w = (y^d - y^o)x = y^d x - y^o x \quad (4)$$

where  $x$  is the presynaptic input and  $y^d$  and  $y^o$  are the desired and observed outputs, respectively. When expanded as shown on the RHS and reformulated for SNNs, the rule can be expressed as a sum of STDP and anti-STDP. That is, the rule for training excitatory synapses takes the form

$$\Delta w = \Delta w^{\text{STDP}}(S^{\text{in}}, S^d) + \Delta w^{\text{aSTDP}}(S^{\text{in}}, S^o). \quad (5)$$

In the above,  $\Delta w^{\text{STDP}}$  is a function of the correlation of the presynaptic and desired spike trains, whereas  $\Delta w^{\text{aSTDP}}$  depends on the presynaptic and observed spike trains. Because the learning rule uses the correlation between the teacher neuron (desired output) and the input neuron, there is not a direct physical connection. This is why the word “remote” is used in the phrase “remote supervised learning.” Although it is not apparent in the above equation, the learning is constrained to fall with typical STDP eligibility windows.

The Chronotron was developed to improve on the Tempotron (Gütig & Sompolinsky, 2006) which had the ability to train single neurons to recognize encodings by the precise timing of incoming spikes. The limitation of the Tempotron was that was restricted to outputting 0 or 1 spikes during a predetermined interval. Because of this, the output did not encode spike timing information. This precluded the ability of a Tempotron to meaningfully send its output to another Tempotron. The motivation of the Chronotron was similar to that of SpikeProp and its successors. The innovation of the Chronotron was to base the supervised training on a more sophisticated distance measure, namely the Victor–Purpura (VP) distance metric (Victor & Purpura, 1997) between two spike trains. This metric is “the minimum cost of transforming one spike train into the other by creating, removing, or moving spikes”. (Florian, 2012, p. 3) They adapted the VP distance so that it would be piecewise differentiable and admissible as a cost function to perform gradient descent with respect to the weights.

Similar to ReSuMe, the SPAN model develops its learning algorithm from the Widrow–Hoff rule. However, instead of adapting the rule to an SNN, SPAN makes the SNN compatible with Widrow–Hoff by digital-to-analog conversion of spike trains using alpha kernels of the form  $te^{-\frac{t}{\tau}}$ . As this is a common formula for modeling a postsynaptic potential, this step in effect converts all spikes to a linear summation of PSPs. Note that this is similar to the technique used in SpikeProp described at the beginning of this subsection. The learning rule can then be written as

$$\Delta w \propto \int \tilde{x}_i(\tilde{y}_d(t) - \tilde{y}_o(t))dt \quad (6)$$

where the tilde symbol indicates the analog version of the spike train and the bounds of integration cover the relevant local time interval.

In Huh and Sejnowski (2017), it was observed that the previous gradient-based learning methods all still had the constraint that the number and times of output spikes must be prespecified which placed limits on their applicability. They replaced the hard spike threshold with a narrow support gate function,  $g(\cdot)$ , such that  $g(v) \geq 0$  and  $\int g dv = 1$ , and  $v$  is the membrane potential. Intuitively, this allows modeled postsynaptic currents to be released when the membrane potential approaches threshold leading to continuity in the spike generation mechanism. Experimentally, it was found that weight updates occurred near spike times “bearing close resemblance to reward-modulated STDP” (Huh & Sejnowski, 2017, p. 8), which enhances the biological relevance of the model.

In Tavanaei and Maida (2018a), a supervised learning method was proposed (BP-STDP) where the backpropagation update rules were converted to temporally local STDP rules for multi-layer SNNs. This model achieved accuracies comparable to equal-sized conventional and spiking networks for the MNIST benchmark (see Section 3.1).

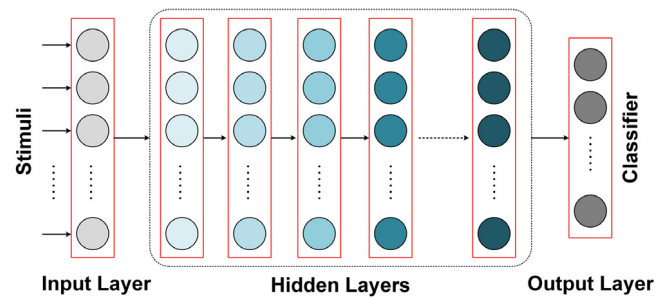
Another implementation of supervised learning in SNNs is based on optimizing the likelihood and probability of the post-synaptic spikes to match the desired ones. Pfister, Toyozumi, Barber, and Gerstner (2006) developed a model to optimize the likelihood of postsynaptic firing at one or several desired times. They proposed a modified version of the SRM neuronal model such that it uses a stochastic threshold on the membrane potential.

In another approach to supervised learning, each output neuron represents a class of data (pattern). Output neurons are in competition to be selected and responsive to the input patterns. In this approach, firing the target neuron causes STDP over the incoming synapses and firing the non-target neurons causes anti-STDP. This approach has successfully been used in SNNs for numerical data classification (Wang, Belatreche, Maguire, & McGinnity, 2014), handwritten digit recognition (Tavanaei & Maida, 2015a), spoken digit classification (Tavanaei & Maida, 2017b), and reinforcement learning in SNNs (Mozafari, Kheradpisheh, Masquelier, Nowzari-Dalini, & Ganjtabesh, 2018b). The sharp synaptic weight adaptation based on immediate STDP and anti-STDP results in fast learning.

### 3. Deep learning in SNNs

Deep learning uses an architecture with many layers of trainable parameters and has demonstrated outstanding performance in machine learning and AI applications (LeCun et al., 2015a; Schmidhuber, 2015). Deep neural networks (DNNs) are trained end-to-end by using optimization algorithms usually based on backpropagation. The multi-layer neural architecture in the primate's brain has inspired researchers to concentrate on the depth of non-linear neural layers instead of using shallow networks with many neurons. Also, theoretical and experimental results show better performance of deep rather than wide structures (Goodfellow, Bengio, & Courville, 2016; Kheradpisheh, Ghodrati, Ganjtabesh, & Masquelier, 2016b,c). Deep neural networks extract complex features through sequential layers of neurons equipped with non-linear, differentiable activation functions to provide an appropriate platform for the backpropagation algorithm. Fig. 1 depicts a deep NN architecture with several hidden layers.

For most classification problems, the output layer of a deep network uses a softmax module. The training vectors use a one-hot encoding. In a one-hot encoding each vector component corresponds to one of the possible classes. This vector is binary with exactly one component set to 1 that corresponds to the desired target class. The softmax module for the output layer guarantees that the values of each of the output units falls within the range (0, 1) and also sum to 1. This gives a set of mutually exclusive and



**Fig. 1.** Simplest deep neural architecture, usually fully connected, with input, hidden, and output layers. The input layer learns to perform pre-processing on the input. The information is then sent to a series of hidden layers, the number of which can vary. As the information propagates through hidden layers, more complex features are extracted and learned. The output layer performs classification and determines the label of the input stimulus, usually by softmax (see text).

exhaustive probability values. The softmax formula, sometimes called the normalized exponential, is given below

$$y_i = \frac{\exp(a_i)}{\sum_j \exp(a_j)} \quad (7)$$

where,  $a_i$ , is the net input to a particular output unit,  $j$  indexes the set of output units, and  $y_i$  is the value of output unit  $i$ , which falls in the range (0, 1).

In addition to the fully connected architecture in Fig. 1 and discussed in Section 3.1, there are also deep convolutional neural networks (DCNNs) discussed in Section 3.2, deep belief networks (DBNs) discussed in Section 3.3, and recurrent neural networks (RNNs) discussed in Section 3.4.

SNNs have also shown promising performance in a number of pattern recognition tasks (Ghosh-Dastidar & Adeli, 2009b; Kasabov, 2014). However, the performance of directly trained spiking deep networks is not as good as traditional DNNs represented in the literature. Therefore, a spiking deep network (spiking DNN, spiking CNN, spiking RNN, or spiking DBN) with good performance comparable with traditional deep learning methods, is a challenging topic because of its importance in DNN hardware implementations.

Masquelier and Thorpe (2007, 2010) developed one of the earliest feedforward hierarchical convolutional networks of spiking neurons for unsupervised learning of visual features (Masquelier & Thorpe, 2007, 2010). This network was extended for larger problems, such as Kheradpisheh et al. (2016a). Using an STDP rule with a probabilistic interpretation, the performance of the model was later improved in different object recognition tasks (Tavanaei et al., 2016b). Further attempts led to several multi-layer SNNs, with STDP learning, that performed greatly in adaptive multi-view pattern recognition (Wysoski, Benuskova, & Kasabov, 2008) and handwritten digit recognition (Beyeler et al., 2013). These models mostly used one or more layers for pre-processing, one learning layer, and one classifier (output neuron) layer. Although these networks are known as multi-layer SNNs, they do not offer multi-layer learning. Specifically, these SNNs are limited by using only one trainable layer, even though they have many layers of processing.

Encouraged by the power-efficiency and biological plausibility of neuromorphic platforms, a number of recent studies have concentrated on developing deep SNNs for these platforms (Querlioz, Bichler, Dollfus, & Gamrat, 2013). Previous studies exploring supervised and unsupervised learning rules in spiking architectures can be employed to develop hierarchies of feature extraction and classification modules. SNNs enable power-efficient platforms mimicking brain functionality for solving complex problems, such as in new trends of autonomous vehicles. Developing a neural



network that is as efficient and biologically plausible as SNNs but as powerful as DNNs in performing different tasks is an opportunity in the field of artificial intelligence and computational neuroscience.

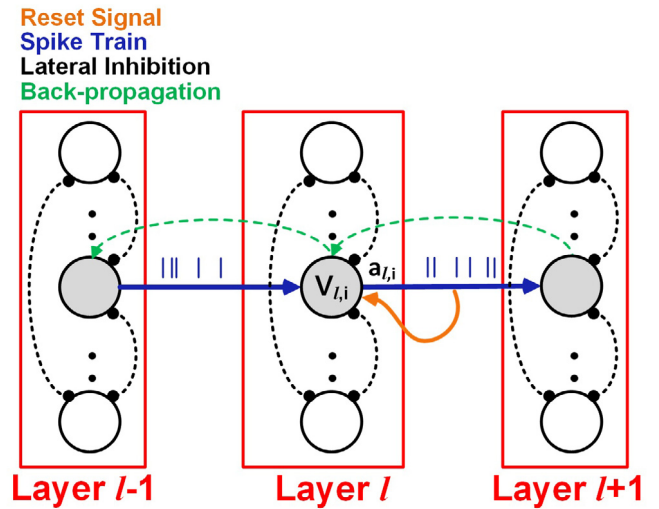
The remaining subsections review spiking deep learning approaches covering deep fully connected SNNs, spiking CNNs, spiking DBNs, and spiking RNNs.

### 3.1. Deep, fully connected SNNs

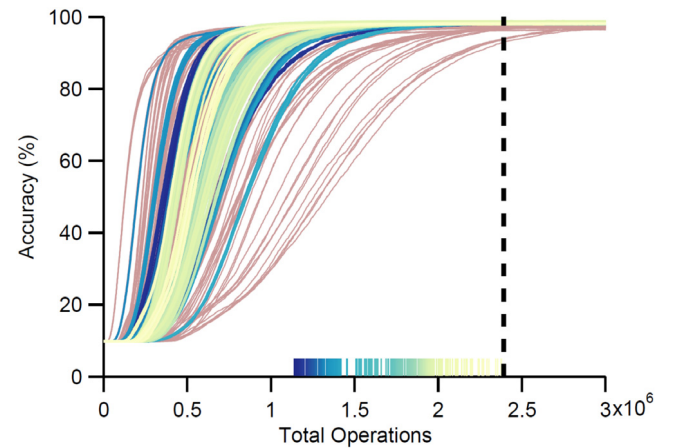
Recent studies have developed a number of deep SNNs using STDP and stochastic gradient descent. Spiking networks consisting of many LIF neurons equipped by spike-based synaptic plasticity rules have shown success in different pattern recognition tasks (Brader, Senn, & Fusi, 2007; Eliasmith et al., 2012). Diehl and Cook (2015) showed that STDP in a two-layer SNN is able to extract discriminative features and patterns from stimuli. They used unsupervised learning rules introduced by Morrison, Aertsen, and Diesmann (2007), Nessler et al. (2013) and Pfister and Gerstner (2006) to train the SNN for the Modified National Institute of Standards and Technology (MNIST) dataset (LeCun, Cortes, & Burges, 1998b) digit recognition with the best performance of 95%.

Towards linking biologically plausible learning methods and conventional learning algorithms in neural networks, a number of deep SNNs have recently been developed. Bengio, Lee, Bornschein, Mesnard, and Lin (2015) proposed a deep learning method using forward and backward neural activity propagation. The learning rule was based on the idea that STDP implements the gradient descent learning rule (Bengio, Mesnard, Fischer, Zhang, & Wu, 2017; Hinton, 2007). Using pre- and postsynaptic spike trains, O'Connor and Welling (2016) developed a backpropagation algorithm in deep SNNs using the outer product of pre- and postsynaptic spike counts. They showed high performance of the spiking multi-layer perceptron on the MNIST benchmark (97.93%) which is comparable to the performance of the conventional deep neural networks equipped with rectified linear units (ReLUs) of 98.37%. Recently, Lee et al. (2016) proposed a backpropagation algorithm by treating the neuron's membrane potential as the differentiable signal to act analogous to the non-linear activation functions in traditional neural networks (Fig. 2). Performance of 98.88% on the MNIST dataset was reported in this study while the number of computational operations was five times fewer than traditional DNNs, in their experiments. To further reduce the computational cost of learning in these deep SNNs, Neftci, Augustine, Paul, and Detorakis (2017) proposed an event-driven random backpropagation (eRBP) algorithm simplifying the backpropagation chain path. The eRBP rule used error-modulated synaptic plasticity in which all the information used for learning was locally available at the neuron and synapse (Neftci et al., 2017).

A more hardware-oriented approach to leveraging power-efficient SNNs is to convert an offline trained DNN to a neuro-morphic spiking platform (ANN-to-SNN conversion), specifically for hardware implementation (Perez-Carrasco et al., 2013). To substitute for the floating-point activation values in DNNs, rate-based coding is generally used in which higher activations are replaced by higher spike rates. Using this approach, several models have been developed that obtained excellent accuracy performance (Diehl et al., 2015; Esser, Appuswamy, Merolla, Arthur, & Modha, 2015; Rueckauer, Hu, Lungu, Pfeiffer, & Liu, 2017; Stromatias, Soto, Serano-Gotarredona, & Linares-Barranco, 2017). In another effort to assess the power consumption of deep SNNs, Neil, Pfeiffer, and Liu (2016b) studied many different models, all of which achieved the same accuracy rate of 98% on the MNIST digit recognition task. They all used the same 784–1200–10 three-layer architecture but applied optimized parameters and SNN architecture settings, in ANN-to-SNN conversion, to reduce power consumption and latency of the model. The performance of a DNN and different



**Fig. 2.** Deep SNN equipped with backpropagation proposed by Lee et al. (2016). The neuron's activation value,  $a_{l,i}$ , is given by the neuron's membrane potential. The differentiable activation function, which is calculated by the neuron's excitatory input, lateral inhibition, and threshold, is used for developing backpropagation using the chain rule. The output activation value of the current layer (layer  $l$ ) is used as input for the next layer in the backpropagation algorithm.

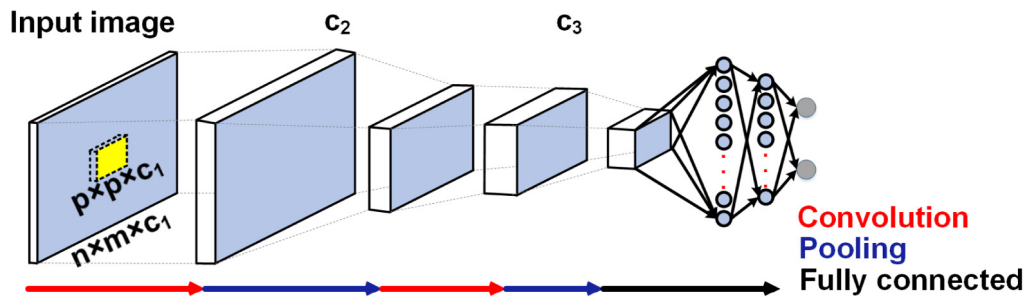


**Fig. 3.** The total number of operations needed to achieve a given accuracy for MNIST classification by deep SNNs converted from an offline trained deep neural network in comparison with the traditional (non-spiking) deep neural network (Neil et al., 2016b). The vertical dashed line shows the number of operations required for the non-spiking deep neural network to achieve accuracy of 98%. The other curves show the accuracy of 522 deep SNNs (with different network setups) versus the number of operations. The pink curves show the networks that achieve less than 98% accuracy within the computing constraint. The colored vertical lines on the horizontal axis indicate the number of operations at which the corresponding SNNs reached 98% accuracy. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

converted deep SNN versus the total required operations is shown in Fig. 3. For the SNNs, a computational operation is “the addition of a synaptic weight to its neuron's membrane potential” (Neil et al., 2016b). Some of the SNN variations explore a computation versus accuracy trade-off where fewer spikes incur less power consumption while preserving acceptable accuracy.

### 3.2. Spiking CNNs

Deep convolutional neural networks (DCNNs) are mostly used in applications involving images. They consist of a sequence of



**Fig. 4.** LeNet: Early CNN proposed by LeCun et al. (1998a, 2015b). The network consists of two convolutional/pooling layers followed by fully connected layers for image classification.

convolution and pooling (sub-sampling) layers followed by a feed-forward classifier like that in Fig. 1. This type of network has shown outstanding performance in image recognition (Krizhevsky et al., 2012; Oquab, Bottou, Laptev, & Sivic, 2014; Rawat & Wang, 2017; Simonyan & Zisserman, 2014), speech recognition (Abdel-Hamid, Deng, & Yu, 2013; Abdel-Hamid, Mohamed, Jiang, & Penn, 2012; Sainath, Mohamed, Kingsbury, & Ramabhadran, 2013), bioinformatics (Quang & Xie, 2016; Tavanaei, Maida, Kaniyammattam, & Loganathanaraj, 2016a; Zeng, Edwards, Liu, & Gifford, 2016), object detection and segmentation (Long et al., 2015; Ronneberger, Fischer, & Brox, 2015), and so forth. Fig. 4 shows the LeNet architecture, an early deep CNN, for image classification (LeCun, Bottou, Bengio, & Haffner, 1998a; LeCun et al., 2015b; Szegedy et al., 2015). The question is how a spiking CNN with such an architecture can be trained while incorporating traditional CNN properties. In the case of vision, the first layer of convolution is interpreted as extracting primary visual features (sometimes resembling oriented-edge detectors modeled by the outputs of Gabor filters (Marčelja, 1980)). Subsequent layers extract increasingly more complex features for classification purposes. The pooling layer performs subsampling and reduces the size of the previous layer using an arithmetic operation such as maximum or average over a square neighborhood of neurons in the relevant feature map. Later in the hierarchy, these layers develop invariance to changes in orientation, scale, and local translation.

The representational properties of early layers in the CNN mentioned above are similar to the response properties of neurons in primary visual cortex (V1), which is the first cortical area in the visual hierarchy of the primate's brain. For example, neurons in area V1 detect primary visual features, such as oriented edges, from input images (Hubel & Wiesel, 1959, 1962). Each V1 neuron is selective to a particular orientation, meaning that when a stimulus with this orientation is presented, only selective neurons to this orientation respond maximally. Representation learning methods, which use neural networks such as autoencoders and sparse coding schemes, learn to discover visual features similar to the receptive field properties found in V1 (Bell & Sejnowski, 1997; Földiák, 1990; Olshausen et al., 1996; Rehn & Sommer, 2007). Bio-inspired SNNs also have obvious footprints in representation learning using sparse coding (King, Zylberberg, & DeWeese, 2013; Zylberberg, Murphy, & DeWeese, 2011), independent component analysis (ICA) (Savin, Joshi, & Triesch, 2010), and an STDP-based autoencoder (Burbank, 2015).

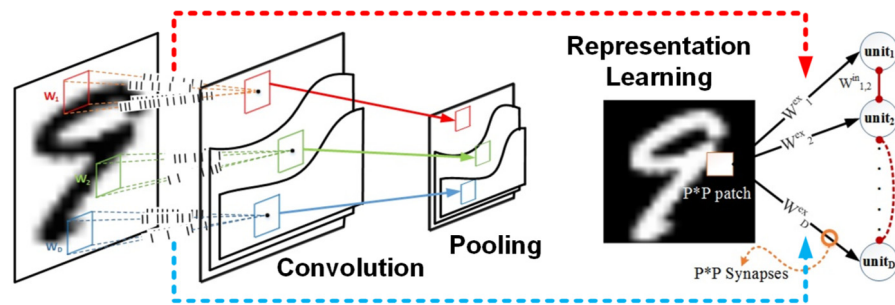
As mentioned earlier, CNNs commonly use V1-like receptive field kernels in early layers to extract features from stimuli by convolving the kernels over the input (e.g. image). Subsequent layers combine the previous layer's kernels to learn increasingly complex and abstract stimulus features. Representation filters (trained or hand-crafted) and STDP learning rules can be used to develop spiking CNNs. A convolutional/pooling layer trained by a local spike-based representation learning algorithm is shown in Fig. 5. Hand-crafted convolutional kernels have been used in the first

layer of a number of spiking CNNs that have obtained high classification performance (Kheradpisheh et al., 2016a; Kheradpisheh, Ganjtabesh, Thorpe, & Masquelier, 2018; Masquelier & Thorpe, 2007; Tavanaei et al., 2016b; Zhao, Ding, Chen, Linares-Barranco, & Tang, 2015). Difference-of-Gaussian (DoG) is a common hand-crafted filter that is used to extract features in the early layers of SNNs. This choice is bio-motivated to mimic inputs to the mammalian primary visual cortex. The primate visual system is known to have spatio-temporal (space-time) receptive fields where the RF's spatial characteristics evolve in over a brief time after the onset of a presented stimulus (Dayan & Abbott, 2001). This is the case for instance for direction selective cells. Lagorce, Orchard, Galluppi, Shi, and Benosman (2017) present a spiking neuromorphic model that may bear some computational relation to this phenomenon. They use so-called time surfaces acquired by hierarchical incremental clustering to create a hierarchical model for pattern recognition. An elementary time surface recognizes a local spatio-temporal feature provided by an event-based sensor.

One of the deepest (to date) STDP-trained convolutional architectures with multi-layer learning has used a layer of DoG filters as input layer of an SNN that is followed by more convolutional/pooling layers trained by STDP (Kheradpisheh et al., 2018). This network architecture extracted visual features that were sent to an SVM classifier, yielding high accuracy of 98.4% on MNIST. Later, Thiele, Bichler, and Dupret (2018) introduced dual accumulator neurons to address unfavorable interactions of multi-layer training and lateral inhibition. This produced a fully event-driven system inspired by the above design (Kheradpisheh et al., 2018) which used unsupervised STDP to train all layers simultaneously. To train convolutional filters, layer-wise spiking representation learning approaches have been implemented in recent spiking CNNs (Panda & Roy, 2016; Tavanaei, Kirby, & Maida, 2018a; Tavanaei & Maida, 2016, 2017c). Tavanaei and Maida (2016, 2017c) used SAILnet (Zylberberg et al., 2011) to train orientation selective kernels used in the initial layer of a spiking CNN. The convolutional layer in this network is followed by a feature discovery layer equipped with an STDP variant (Tavanaei et al., 2016b) to extract visual features for classification. Tavanaei, Masquelier, and Maida (2018b) created a new version of SAILnet to improve its locality in space and time. Implementing stacked convolutional autoencoders Panda and Roy (2016) showed further improvement in performance on MNIST (99.05%), which is comparable to the traditional CNNs.

Non-spiking CNNs are trained using the backpropagation algorithm. Recently, backpropagation has also been employed for training spiking CNNs (Lee et al., 2016; Panda & Roy, 2016). Panda and Roy (2016), using approximations developed in Anwani and Rajendran (2015), showed how to build a hierarchical spiking convolutional autoencoder (AE) using backpropagation. The spiking convolutional autoencoder is an important module for enabling the construction of deep spiking CNNs. Their proof-of-concept implementation (SpikeCNN) used two learning layers on the MNIST





**Fig. 5.** Representation learning (SAILnet (Zylberberg et al., 2011)) for layer-wise unsupervised learning of a spiking CNN (Tavanaei & Maida, 2017c). The excitatory synaptic weights connected to neurons in the representation layer specify convolutional filters. This architecture determines that representation learning in single-layer SNNs can be utilized to train layer-wise spiking CNNs.

dataset (handwritten digits) (LeCun et al., 1998a) and three learning layers on the CIFAR-10 dataset (10-category tiny images) (Krizhevsky & Hinton, 2009). They used local, layer-wise learning of convolutional layers while Lee et al. (2016) developed an end-to-end gradient descent learning method. Both methods used the neural membrane potential as replacements for differentiable activation functions to apply the backpropagation algorithm. Lee et al.'s approach (for the spiking CNN) (Lee et al., 2016) showed better performance than the layer-wise convolutional autoencoders (Panda & Roy, 2016). In these models, higher membrane potential correlates with higher spike probability.

As mentioned at the end of Section 3.1, a popular approach to leverage neuromorphic platforms while avoiding the training of spiking CNNs is to convert an already trained CNN to a spiking architecture by using a rule to map the trained weights to the spiking architecture, similar to the ANN-to-SNN conversion method. Many studies have shown high performance of converted spiking CNNs (close to conventional CNNs) while using fewer operations and consuming less energy (Esser et al., 2015; Hunsberger & Eliasmith, 2015, 2016; Neil & Liu, 2016), which enable deep CNNs to be implemented on hardware (Esser et al., 2016; Garbin et al., 2014; Indiveri, Corradi, & Qiao, 2015). One of the initial successful CNN-to-SNN conversion methods for energy efficient pattern recognition is the architecture shown in Fig. 6 (Cao, Chen, & Khosla, 2015). Later, Diehl et al. (2015) improved this architecture using weight normalization to reduce performance loss. Recent work by Rueckauer et al. (2017), Rueckauer, Lungu, Hu, and Pfeiffer (2016) proposed several conversion criteria such that the new spiking CNN recognizes more difficult objects than MNIST (e.g. CIFAR-10 and ImageNet (Deng et al., 2009)). Building on the strategy of Diehl et al. (2015), very recent work (Sengupta, Ye, Wang, Liu, & Roy, 2018) has shown that conversion techniques can be extended to much deeper SNNs extending up to 34 layers and effectively applied to complex data sets such as ImageNet (Deng et al., 2009).

### 3.3. Spiking deep belief networks

Deep belief networks (DBNs) (Bengio, 2009) are a type of multi-layer network initially developed by Hinton, Osindero, and Teh (2006). They efficiently use greedy layer-wise unsupervised learning and are made of stochastic binary units, meaning that the binary state of the unit is updated using a probability function. The layer-wise method stacks pre-trained, single-layer learning modules known as restricted Boltzmann machines (RBMs). The representation layer in an RBM is restricted from having lateral connections. This enables the learning algorithm to optimize the representation by making use of independence assumptions among the representation units, given a particular input state. The original DBN architecture was successfully trained on the MNIST dataset and is shown in Fig. 7(a). The RBMs are trained in a layerwise fashion by contrastive divergence (CD), which approximates a

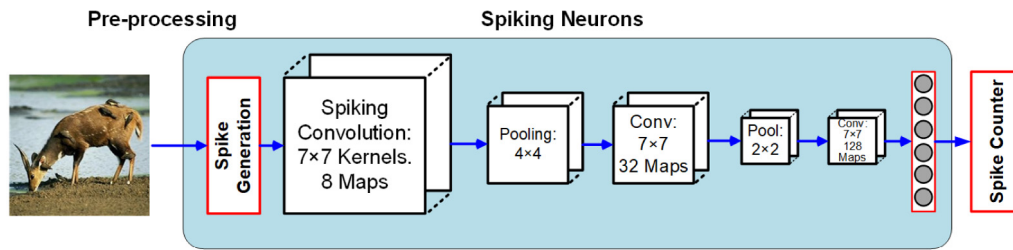
maximum-likelihood learning algorithm. Unlike backpropagation, the CD update equations do not use derivatives. The pre-trained hierarchy is fine-tuned by backpropagation if labeled data are available. DBNs provide a layer-wise structure for feature extraction, representation, and universal approximation (Le Roux & Bengio, 2008, 2010; Salama, Hassanien, & Fahmy, 2010).

Lee, Ekanadham, and Ng (2008) used interleaving CD with gradient descent on the sparsity term to implement sparse DBNs which were used to model cortical visual areas V1 and V2. Further extensions of the model led to a convolutional sparse DBNs (Lee, Grosse, Ranganath, & Ng, 2011). This was accomplished by redefining the energy function to be consistent with the tied weights of a convolutional network and then using Gibbs sampling to realize the appropriate weight update rules. DBNs and convolutional DBNs have successfully been employed in many areas such as visual processing (Krizhevsky & Hinton, 2010; Liu, Han, Meng, & Tong, 2014; Mleczo, Kapuściński, & Nowicki, 2015; Susskind, Hinton, Movellan, & Anderson, 2008), audio processing (Hamel & Eck, 2010; Kang, Qian, & Meng, 2013; Lee, Pham, Largman, & Ng, 2009; Mohamed, Dahl, & Hinton, 2009, 2012), time series forecasting (Kuremoto, Kimura, Kobayashi, & Obayashi, 2014), and protein folding (Jo, Hou, Eickholt, & Cheng, 2015).

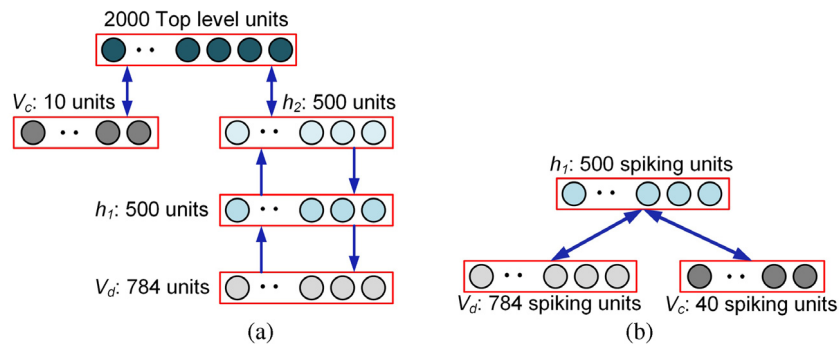
The first step in developing a spiking DBN is to start with a spiking RBM. Fig. 7(b) shows the architecture of a spiking RBM introduced by Neftci et al. (2014). A spiking RBM uses stochastic integrate-and-fire neurons instead of the memoryless stochastic units in a standard RBM. Neftci et al. (2014) showed that, in the context of a particular type of spiking network, a variant of STDP can approximate CD. That is, the learned distributions for spiking networks capture the same statistical properties as the learned distributions for the equivalent non-spiking networks, establishing an important foundational result.

One approach toward developing functional spiking DBNs is to convert previously trained DBNs to spiking platforms similar to the conversion method explained for SNNs or spiking CNNs. The first spiking DBN was introduced by O'Connor, Neil, Liu, Delbruck, and Pfeiffer (2013) in which a DBN is converted to a network of LIF spiking neurons for MNIST image classification. This work was then extended (Stromatias et al., 2015b) to develop a noise robust spiking DBN and as well as conforming to hardware constraints. The spiking DBNs and RBMs are power-efficient and this enables them to be implemented on low latency hardware with high accuracy close to that of traditional DBNs (Merolla et al., 2011; Neil & Liu, 2014; Stromatias et al., 2015a).

Recently, it has been shown there is an equivalence between so called 'hybrid' Boltzmann machines (HBMs) (Barra, Bernacchia, Santucci, & Contucci, 2012) and Hopfield networks (Hertz, Krogh, & Palmer, 1991; Hopfield, 1982). An HBM is a restricted Boltzmann machine in which the hidden (representation) units can take on continuous values (while the visible units still have binary values) (Barra et al., 2012). When the functions within the HBM are



**Fig. 6.** Spiking CNN architecture developed by [Cao et al. \(2015\)](#). The input image, after pre-processing, is converted to spike trains based on pixel intensity. The spiking layers use weights trained by a non-spiking CNN. The last component selects the neuron with maximum activity (spike frequency) as the image's class.



**Fig. 7.** (a): The DBN proposed by [Hinton et al. \(2006\)](#) for MNIST image classification. This network consists of three stacked RBMs with 500, 500, and 2000 representation neurons. The input and output include 784 (as the number of pixels,  $28 \times 28$ ) and 10 (as the number of classes, 0, ..., 9) neurons, respectively. (b): The spiking RBM architecture introduced by [Neftci et al. \(2014\)](#) consisting of 500 hidden neurons, 784 input neurons, and 40 class neurons (824 visible neurons).

marginalized over the hidden units, both the Hopfield and the HBM systems have been shown to be thermodynamically equivalent. Although Hopfield networks are primarily used as models of pattern association, the thermodynamic equivalence allows them to be simulated by HBMs. In the HBM, the  $N$  binary visible units correspond to binary stochastic neurons in the Hopfield network and the  $P$  hidden units in the HBM correspond to stored patterns in the Hopfield network. HBMs offer a new way to simulate Hopfield networks while using fewer synapses. Specifically, a Hopfield network requires updating  $N$  neurons and  $N(N-1)/2$  synapses, whereas the HBM requires  $H+P$  neurons and updating  $HP$  synapses, where  $P$  is the number of stored patterns. Further developments of this theory are found in [Barra, Genovese, Sollich, and Tantari \(2017, 2018\)](#) and [Tubiana and Monasson \(2017\)](#).

### 3.4. Recurrent SNNs

A neural network is recurrent if its directed graph representation has a cycle. Any network that has a winner(s)-take-all (WTA) module or a softmax module is at least implicitly recurrent because equivalent function is implemented in the brain by mutually recurrent inhibitory connections. Any network trained by backpropagation is also implicitly recurrent because the training algorithm (as explained in Section 2.2.3) presupposes the existence of recurrent connections. Sections 3.4.1 and 3.4.2 discuss gated SNNs and reservoir models, respectively. Both types of models are intended to process sequential data. The former processes spatiotemporal data and is usually intended as a model of cortical microcircuits. The latter focuses more on sequential data only.

#### 3.4.1. Gated SNNs

Recurrent neural networks (RNNs) are used for processing temporal information. The most common method to train RNNs is backpropagation through time (BPTT), which unrolls the recurrent network for some number of steps into the past, and then trains the unrolled network as if it was a feedforward network. Since the same recurrent weights are shared through all unrolled layers

for the resulting feedforward network, there are issues in training for long sequences, specifically the emergence of vanishing and exploding gradients. In the former case, the network stops learning and in the latter, the training becomes unstable ([Bengio, Simard, & Frasconi, 1994](#)).

Because of these problems, the research in [Hochreiter and Schmidhuber \(1997\)](#) introduced an innovation into recurrent networks called a constant error carousel (CEC) that avoided repeated multiplication of derivatives. These have become known as gated recurrent networks (in addition to the CEC, they have trainable gates) and have virtually replaced traditional RNNs. The first gated recurrent network was the long short-term memory (LSTM) ([Hochreiter & Schmidhuber, 1997](#)). LSTMs and other gated recurrent networks (GRUs) ([Chung, Gulcehre, Cho, & Bengio, 2014](#)) are conventional ANNs in the sense that they do not use spiking neurons, but they are also unconventional in the sense that they replace units having recurrent connections with 'cells' that contain state as well as gates and, because of this, can readily adapt to the structure of input sequences. The gates control information flow into, out of, and within the cells. The gates are controlled by trainable weights. The topic we consider in this subsection is the current status of the field with regard to creating spiking LSTMs or gated recurrent networks.

There are only a few recurrent SNNs, in which conventional (non-spiking) RNNs are converted to spiking frameworks. Perhaps the earliest was [Diehl, Zarella, Cassidy, Pedroni, and Neftci \(2016\)](#) which applied the previously discussed conversion method to a (non-gated) Elman network ([Elman, 1990](#)) for implementation on neuromorphic hardware. [Shrestha et al. \(2017\)](#) implemented an energy-efficient spiking LSTM onto the IBM TrueNorth neurosynaptic system platform ([Akopyan et al., 2015](#)). To do this, they had to solve two problems. The first was to build a spiking LSTM and the second was to implement it on a neuromorphic chip. We focus on the former problem. One of their design choices was to represent positive and negative values using two channels of spike trains. This is bio-plausible and the DoG filters discussed in Section 3.2 commonly come in two forms for exactly this purpose.

The inputs, outputs, and most of the internal LSTM variables used rate coding. There was one exception to this where the value of the variable representing cell state needed higher precision and was represented by a spike burst code. The main thrust of the paper was overcoming the complications in mapping the LSTM to a neuromorphic chip and accuracy results on standard benchmarks were not reported.

The phased LSTM (Neil, Pfeiffer, & Liu, 2016a), although not a spiking LSTM, is well suited to process event-driven, asynchronously sampled data, which is a common task for SNNs. This makes it useful to process inputs from (possibly several) biomorphic input sensors that sample inputs at multiple time scales. It is also potentially useful for processing outputs from SNNs. The innovation of the phased LSTM is the addition of a time gate to the usual set of gates within a memory cell. The time gate is synchronized to a rhythmic oscillation. When ‘open’, the time gate allows the usual updates to the hidden, output, and cell state vectors. When ‘closed’ it prevents the updates forcing the hidden and cell vectors to retain their values. The units within these vectors can have separate updates with their own oscillation periods and phases. This allows the phased LSTM to quantize its input at different time scales. In a number of experiments that involved event-driven sampling at varied time scales, the phased LSTM has trained more quickly than a regular LSTM while performing as accurately as the regular LSTM.

#### 3.4.2. Liquid state machines and reservoirs

The neocortex, unique to mammals, has the ability to drastically scale its surface area from about 1 square cm in the mouse to about 2500 square cm in the human while keeping its thickness fairly constant ( $\leq 3$  mm). To support this expansion, one hypothesis is that mammals discovered a structural motif that may be replicated and functionally adapted to new tasks. Initially this was called a minicolumn consisting of about 300 excitatory and inhibitory recurrently connected neurons that span the six layers of the three-millimeter-thick neocortex. More recently the term canonical or cortical microcircuit has been used. There has been great interest in modeling this hypothesized module because of the potential insights it may offer to universal computation.

In an effort to model the computations that might be taking place within the canonical neocortical microcircuit, the liquid state machine (LSM) was introduced (Maass, Natschläger, & Markram, 2002) which has since been partly absorbed by the field of reservoir computing (Lukoševičius & Jaeger, 2009; Lukoševičius, Jaeger, & Schrauwen, 2012). In an LSM context, a neural reservoir is a sparsely connected recurrent SNN composed of excitatory and inhibitory neurons designed to have enough structure to create a universal analog fading memory. This module is constructed so that it can transform a set of possibly multi-modal spike trains into a spatiotemporal representation whose instantaneous state can be recognized and readout by a layer of linear units.

A reservoir model has three parts:

1. It needs one or more sensory-based, time-varying input streams of spikes or continuous inputs that can be transduced into spikes.
2. It needs a recurrent SNN known as a reservoir or liquid whose synapses may (or may not) be able to learn. The neurons are given physical locations in space as a means to establish connection probabilities, which generally decrease exponentially with distance. Some ideas on why this might be useful are given in Pyle and Rosenbaum (2017). Connections tend to be sparse to avoid chaotic dynamics. The ratio of excitatory to inhibitory neurons is usually about 80% to 20% to reflect the ratio found in the neocortex.
3. It needs (usually) linear readout units which can be trained to recognize instantaneous patterns within the liquid. Part of the motivation for the linearity is to prove that the information in the reservoir’s dynamically evolving state can be easily read out.

The neuromorphic cube (NewCube) (Kasabov, 2014; Paulun, Wendt, & Kasabov, 2018; Schliebs, Hamed, & Kasabov, 2011) is a broad and futuristic model proposed as a unifying computational architecture for modeling multi-modality spatiotemporal data, most especially data related to the brain, such as EEG analysis (Doborjeh, Kasabov, Doborjeh, & Sumich, 2018). The core of the NeuCube architecture is a 3D reservoir of spiking neurons trained by an STDP-like mechanism. The reservoir’s structure is intended to directly reflect the inter-area connectivity of the human neocortex as revealed by structural connectivity based on anatomy such as diffusion tensor imaging (DTI) and functional connectivity based on measures like functional magnetic resonance imaging (fMRI). This is in contrast to the use of a reservoir as a model of a neocortical microcircuit. The claim of NeuCube is that the connectivity of the brain at the macro scale obeys the properties of a reservoir.

The previous subsection discussed the status of attempts to create spiking versions of LSTMs. Rather than pursuing a direct approach to structurally translating an LSTM to a spiking version, the work of Bellec, Salaj, Subramoney, Legenstein, and Maass (2018) took a reservoir inspired approach. Their LSNN architecture (long short-term memory SNNs) consisted of four modules labeled: X, R, A, and Y. X provided multiple streams of input spikes, R was the reservoir consisting of excitatory and inhibitory neurons, A was a module of excitatory neurons (connected to X, R, and Y) with adaptive thresholds whose purpose was in part to maintain a tight excitatory–inhibitory balance in R, and module Y consisted of the readout neurons. The LSNN was trained using BPTT (Section 3.4.1) with pseudo derivatives using the membrane potential (as explained in Section 2.2.3). The network achieved comparable accuracy performance to LSTMs on the sequential MNIST benchmark and also on the TIMIT Acoustic-Phonetic Continuous Speech Corpus. Sequential MNIST is a sequence learning benchmark for assessing recurrent network performance first described in Lamb et al. (2016). The task is to recognize MNIST digits but now the input is the set of  $784 = 28^2$  input pixels delivered sequentially over consecutive time steps. Although the LSNN architecture does not have a direct mapping to the architecture of an LSTM, it has learning abilities that are apparently unique to LSTMs. It has been shown that LSTMs “can learn nonlinear functions from a teacher without modifying their weights, and using their short-term memory instead”. In Hochreiter, Younger, and Conwell (2001), it was shown that LSTMs had this property.

Another recent study, (Costa, Assael, Shillingford, de Freitas, & Vogels, 2017), has attempted to adapt the architecture of a conventional LSTM to be a plausible model of a cortical microcircuit. There were two innovations in this model. First, to facilitate mapping to a microcircuit, the multiplicative gating operations within a standard LSTM were replaced with subtractive operations that could be implemented by lateral inhibitory circuits in the neocortex. This leads to the name subLSTM (subtractive LSTM). Second, to facilitate learning and study using readily available deep learning frameworks, the neural coding scheme was assumed to use rate-coded LIF neurons. This allowed them to model spiking neurons using a continuous approximation that was compatible with deep learning environments. Their bio-plausible subLSTM achieved comparable performance to a standard LSTM on the sequential MNIST task. Similar results were obtained in a language processing benchmark. The subLSTMs did not perform better than standard LSTMs but they have opened a venue for interdisciplinary



**Table 1**

Summary of recent deep learning models developed in SNN platforms and their accuracy on MNIST (LeCun et al., 1998a,b), N-MNIST (Orchard et al., 2015), CIFAR-10, and CIFAR-100 (Krizhevsky & Hinton, 2009). MNIST: handwritten digits. N-MNIST: neuromorphic-MNIST representing a spiking version of MNIST. CIFAR: tiny colored images.

Model	Architecture	Learning method	Dataset	Acc
Feedforward, fully connected, multi-layer SNNs				
O'Connor and Welling (2016)	Deep SNN	Stochastic gradient descent	MNIST	96.40
O'Connor and Welling (2016)	Deep SNN	Fractional stochastic gradient descent	MNIST	97.93
Lee et al. (2016)	Deep SNN	Backpropagation	MNIST	98.88
Lee et al. (2016)	Deep SNN	Backpropagation	N-MNIST	98.74
Neftci et al. (2017)	Deep SNN	Event-driven random backpropagation	MNIST	97.98
Liu et al. (2017)	SNN	Temporal backpropagation (3-layer)	MNIST	99.10
Eliasmith et al. (2012)	SNN	Spaun brain model	MNIST	94.00
Diehl and Cook (2015)	SNN	STDP (2-layer)	MNIST	95.00
Tavanaei and Maida (2018a)	SNN	STDP-based backpropagation (3-layer)	MNIST	97.20
Mostafa (2017)	SNN	Temporal backpropagation (3-layer)	MNIST	97.14
Querlioz et al. (2013)	SNN	STDP, Hardware implementation	MNIST	93.50
Brader et al. (2007)	SNN	Spike-driven synaptic plasticity	MNIST	96.50
Diehl et al. (2015)	Deep SNN	Offline learning, Conversion	MNIST	98.60
Neil et al. (2016b)	Deep SNN	Offline learning, Conversion	MNIST	98.00
Hunsberger and Eliasmith (2015, 2016)	Deep SNN	Offline learning, Conversion	MNIST	98.37
Esser et al. (2015)	Deep SNN	Offline learning, Conversion	MNIST	99.42
Spiking CNNs				
Lee et al. (2016)	Spiking CNN	Backpropagation	MNIST	99.31
Lee et al. (2016)	Spiking CNN	Backpropagation	N-MNIST	98.30
Panda and Roy (2016)	Spiking CNN	Convolutional autoencoder	MNIST	99.05
Panda and Roy (2016)	Spiking CNN	Convolutional autoencoder	CIFAR-10	75.42
Tavanaei and Maida (2016, 2017c)	Spiking CNN	Layer wise sparse coding and STDP	MNIST	98.36
Tavanaei et al. (2018a)	Spiking CNN	Layer-wise and end-to-end STDP rules	MNIST	98.60
Mozafari, Ganjtabesh, Nowzari -Dalini, Thorpe, and Masquelier (2018a)	Spiking CNN	STDP and Reward-modulated STDP	MNIST	97.20
Kheradpisheh et al. (2018)	Spiking CNN	Layer wise STDP	MNIST	98.40
Zhao et al. (2015)	Spiking CNN	Tempotron	MNIST	91.29
Cao et al. (2015)	Spiking CNN	Offline learning, Conversion	CIFAR-10	77.43
Neil and Liu (2016)	Spiking CNN	Offline learning, Conversion	N-MNIST	95.72
Diehl et al. (2015)	Spiking CNN	Offline learning, Conversion	MNIST	99.10
Rueckauer et al. (2017)	Spiking CNN	Offline learning, Conversion	MNIST	99.44
Rueckauer et al. (2017)	Spiking CNN	Offline learning, Conversion	CIFAR-10	90.85
Hunsberger and Eliasmith (2015)	Spiking CNN	Offline learning, Conversion	CIFAR-10	82.95
Garbin et al. (2014)	Spiking CNN	Offline learning, Hardware	MNIST	94.00
Esser et al. (2016)	Spiking CNN	Offline learning, Hardware	CIFAR-10	87.50
Esser et al. (2016)	Spiking CNN	Offline learning, Hardware	CIFAR-100	63.05
Spiking RBMs and DBNs				
Neftci et al. (2014)	Spiking RBM	Contrastive divergence in LIF neurons	MNIST	91.90
O'Connor et al. (2013)	Spiking DBN	Offline learning, Conversion	MNIST	94.09
Stromatias et al. (2015b)	Spiking DBN	Offline learning, Conversion	MNIST	94.94
Stromatias et al. (2015a)	Spiking DBN	Offline learning, Hardware	MNIST	95.00
Merolla et al. (2011)	Spiking RBM	Offline learning, Hardware	MNIST	94.00
Neil and Liu (2014)	Spiking DBN	Offline learning, Hardware	MNIST	92.00

dialog between questions of brain function and theoretical insights from deep learning.

These studies show a bright future of recurrent SNNs which take advantages of the conventional RNNs and the bio-inspired recurrent neural framework of reservoir computing.

### 3.5. Performance comparisons of contemporary models

Table 1 shows previous models for developing deep SNNs and their architectures along with their accuracy rates on different datasets. This table shows two tracks of spiking models: (1) using online learning and (2) using offline learning (deployment). The latter method has reported higher performance but it avoids training the multi-layer SNNs by converting the offline trained neural networks to the relevant spiking platform. On the other hand, online learning offers multi-layer learning in SNNs but reports lower accuracy rates. Additionally, as expected, the spiking CNNs have achieved higher accuracy rates than the spiking DBNs and the fully connected SNNs on image classification. This comparison provides insight into different SNN architectures and learning mechanisms to guide future investigations. Additionally, the maximum accuracy rate of MNIST classification reported in this table is 99.44% while the non-spiking (traditional) CNN has reported 99.77% accuracy (Cireşan, Meier, & Schmidhuber, 2012).

## 4. Summary

There has been much progress in the multi-layer training of deep spiking networks, both in the use of supervised and unsupervised methods. In regard to performance accuracy on complex data sets, the conversion technique that starts with backpropagation-trained ANNs (Sections 3.1 and 3.2) has to date been the most successful. A good judge of success is the choice of benchmark data sets used in testing deep SNNs. They are much less complex than those used in deep ANN applications, and heavily use the simple MNIST and CIFAR data sets shown in Table 1. However, there has been recent success in using SNNs to train ImageNet (Sengupta et al., 2018), which is a great advance.

It has been pointed out in Lagorce et al. (2017) and Pfeiffer and Pfeil (2018) that the static frame-based benchmarks used to assess convolutional ANNs may not be the most appropriate for assessing spiking and event-based models. Spiking models perform best when responding to high information value stimulus changes thereby supporting sparse spike rates, and they can also support a latency-accuracy trade-off. This trade-off allows SNNs to react as soon as evidence arrives (Kheradpisheh et al., 2018; Masquelier & Thorpe, 2007) while continuing to improve their accuracy as more evidence accumulates.

Despite the performance successes of recent conversion methods, they do not appear to directly shed light on how spiking neurons in the brain may be trained. Using local learning methods, such as STDP which is bioplausible, to train large-scale networks remains a deep challenge. Meanwhile, backpropagation, the near universally used optimization algorithm for supervised training of both SNNs and ANNs, still suffers from the point-to-point feedback constraint (Section 2.2.3). There are still no well accepted approaches to get around this.

Work on training recurrent SNNs appears to be in its infancy. The most established work on recurrent SNNs are the liquid state machines. So far, there are few attempts to convert mainstream deep learning recurrent networks to comparable spiking variants (Section 3.4).

Although SNNs were originally developed in direct response to the spiking characteristics of biological neurons, the emergence of energy-efficient, neuromorphic computing hardware has given the field, both from scientific and engineering vantage points, a great boost.

## References

- Abdel-Hamid, O., Deng, L., & Yu, D. (2013). Exploring convolutional neural network structures and optimization techniques for speech recognition. In *Interspeech* (pp. 3366–3370).
- Abdel-Hamid, O., Mohamed, A. -r., Jiang, H., & Penn, G. (2012). Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition. In *Acoustics, speech and signal processing, 2012 IEEE international conference on* (pp. 4277–4280). IEEE.
- Akopyan, F., Sawada, J., Cassidy, A., Alvarez-Icaza, R., Arthur, J., Merolla, P., et al. (2015). Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(10), 1537–1557.
- Anwani, N., & Rajendran, B. (2015). Normad-normalized approximate descent based supervised learning rule for spiking neurons. In *2015 international joint conference on neural networks* (pp. 1–8). IEEE.
- Bair, W., & Koch, C. (1996). Temporal precision of spike trains in extrastriate cortex of the behaving macaque monkey. *Neural Computation*, 8(6), 1185–1202.
- Barra, A., Bernacchia, A., Santucci, E., & Contucci, P. (2012). On the equivalence of Hopfield networks and Boltzmann machines. *Neural Networks*, 34, 1–9.
- Barra, A., Genovesi, G., Sollich, P., & Tantari, D. (2017). Phase transitions in restricted Boltzmann machines with generic priors. *Physical Review E*, 96(4), 042156.
- Barra, A., Genovesi, G., Sollich, P., & Tantari, D. (2018). Phase diagram of restricted Boltzmann machines and generalized Hopfield networks with arbitrary priors. *Physical Review E*, 97(2), 022310.
- Bell, A. J., & Sejnowski, T. J. (1997). The “independent components” of natural scenes are edge filters. *Vision Research*, 37(23), 3327–3338.
- Bellec, G., Salaj, D., Subramoney, A., Legenstein, R., & Maass, W. (2018). Long short-term memory and learning-to-learn in networks of spiking neurons, arXiv preprint arXiv:1803.09574.
- Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1), 1–127.
- Bengio, Y., Lee, D. -H., Bornschein, J., Mesnard, T., & Lin, Z. (2015). Towards biologically plausible deep learning, arXiv preprint arXiv:1502.04156, 1–10.
- Bengio, Y., Mesnard, T., Fischer, A., Zhang, S., & Wu, Y. (2017). STDP-compatible approximation of backpropagation in an energy-based model. *Neural Computation*, 555–577.
- Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2), 157–166.
- Beyeler, M., Dutt, N. D., & Krichmar, J. L. (2013). Categorization and decision-making in a neurobiologically plausible spiking network using a STDP-like learning rule. *Neural Networks*, 48, 109–124.
- Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford University Press.
- Bohte, S. M. (2004). The evidence for neural information processing with precise spike-times: A survey. *Natural Computing*, 3(2), 195–206.
- Bohte, S. M., Kok, J. N., & La Poutre, H. (2002a). Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1), 17–37.
- Bohte, S. M., La Poutre, H., & Kok, J. N. (2002b). Unsupervised clustering with spiking neurons by sparse temporal coding and multilayer RBF networks. *IEEE Transactions on Neural Networks*, 13(2), 426–435.
- Booi, O., & tat Nguyen, H. (2005). A gradient descent rule for spiking neurons emitting multiple spikes. *Information Processing Letters*, 95(6), 552–558.
- Brader, J. M., Senn, W., & Fusi, S. (2007). Learning real-world stimuli in a neural network with spike-driven synaptic dynamics. *Neural Computation*, 19(11), 2881–2912.
- Brea, J., Senn, W., & Pfister, J. -P. (2011). Sequence learning with hidden units in spiking neural networks. In *Advances in neural information processing systems* (pp. 1422–1430).
- Buesing, L., Bill, J., Nessler, B., & Maass, W. (2011). Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons. *PLoS Computational Biology*, 7(11), e1002211.
- Burbank, K. S. (2015). Mirrored STDP implements autoencoder learning in a network of spiking neurons. *PLoS Computational Biology*, 11(12), e1004566.
- Butts, D. A., Weng, C., Jin, J., Yeh, C. -I., Lesica, N. A., Alonso, J. -M., et al. (2007). Temporal precision in the neural code and the timescales of natural vision. *Nature*, 449(7158), 92–95.
- 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.
- Caporale, N., & Dan, Y. (2008). Spike timing-dependent plasticity: A Hebbian learning rule. *Annual Review of Neuroscience*, 31, 25–46.
- Carrillo, S., Harkin, J., McDaid, L. J., Morgan, F., Pande, S., Cawley, S., et al. (2013). Scalable hierarchical network-on-chip architecture for spiking neural network hardware implementations. *IEEE Transactions on Parallel and Distributed Systems*, 24(12), 2451–2461.
- Carrillo, S., Harkin, J., McDaid, L., Pande, S., Cawley, S., McGinley, B., et al. (2012). Advancing interconnect density for spiking neural network hardware implementations using traffic-aware adaptive network-on-chip routers. *Neural Networks*, 33, 42–57.
- Chavez-Noriega, L., Halliwell, J., & Bliss, T. (1990). A decrease in firing threshold observed after induction of the EPSP-spike (ES) component of long-term potentiation in rat hippocampal slices. *Experimental Brain Research*, 79(3), 633–641.
- Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling, arXiv preprint arXiv:1412.3555.
- Cireşan, D., Meier, U., & Schmidhuber, J. (2012). Multi-column deep neural networks for image classification, arXiv preprint arXiv:1202.2745.
- Costa, R., Assael, I. A., Shillington, B., de Freitas, N., & Vogels, T. (2017). Cortical microcircuits as gated-recurrent neural networks. In *Advances in neural information processing systems* (pp. 272–283).
- Crick, F., & Koch, C. (1998). Constraints on cortical and thalamic projections: The no-strong-loops hypothesis. *Nature*, 391, 245–250.
- Dan, Y., & Poo, M. -M. (2006). Spike timing-dependent plasticity: From synapse to perception. *Physiological Reviews*, 86(3), 1033–1048.
- Dayan, P., & Abbott, L. (2001). *Theoretical neuroscience: Computational and mathematical modeling of neural systems*. MIT Press.
- Delorme, A., Gautrais, J., Van Rullen, R., & Thorpe, S. (1999). SpikeNET: A simulator for modeling large networks of integrate and fire neurons. *Neurocomputing*, 26, 989–996.
- Deng, J., Dong, W., Socher, R., Li, L. -J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer vision and pattern recognition, 2009. CVPR 2009. IEEE conference on* (pp. 248–255). IEEE.
- Diehl, P. U., & Cook, M. (2015). Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in Computational Neuroscience*, 9, 1–9.
- 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 *Neural networks, 2015 international joint conference on* (pp. 1–8). IEEE.
- Diehl, P. U., Zarella, G., Cassidy, A., Pedroni, B. U., & Neftci, E. (2016). Conversion of artificial recurrent neural networks to spiking neural networks for low-power neuromorphic hardware. In *2016 IEEE international conference on rebooting computing, San Diego*.
- Doborjeh, Z. G., Kasabov, N., Doborjeh, M. G., & Sumich, A. (2018). Modelling perceptual brain processes in a deep learning spiking neural network architecture. *Scientific Reports*, 8(1), 8912.
- Doya, K. (2007). *Bayesian brain: Probabilistic approaches to neural coding*. MIT press.
- 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(6111), 1202–1205.
- Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, 14(2), 179–211.
- Escobar, M. -J., Masson, G. S., Vieville, T., & Kornprobst, P. (2009). Action recognition using a bio-inspired feedforward spiking network. *International Journal of Computer Vision*, 82(3), 284–301.
- Esser, S. K., Appuswamy, R., Merolla, P., Arthur, J. V., & Modha, D. S. (2015). Backpropagation for energy-efficient neuromorphic computing. In *Advances in neural information processing systems* (pp. 1117–1125).
- Esser, S. K., Merolla, P. A., Arthur, J. V., Cassidy, A. S., Appuswamy, R., Andreopoulos, A., et al. (2016). Convolutional networks for fast, energy-efficient neuromorphic computing. *Proceedings of the National Academy of Sciences*, 201604850.
- Felleman, D. J., & Van Essen, D. C. (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex*, 1(1), 1–47.
- Florian, R. V. (2012). The chronotron: A neuron that learns to fire temporally precise spike patterns. *PLoS One*, 7(8), e40233.
- Földiák, P. (1990). Forming sparse representations by local anti-hebbian learning. *Biological Cybernetics*, 64(2), 165–170.

- Freiwald, W. A., & Tsao, D. Y. (2010). Functional compartmentalization and viewpoint generalization within the macaque face-processing system. *Science*, 330(6005), 845–851.
- Garbin, D., Bichler, O., Vianello, E., Raffay, Q., Gamrat, C., Perniola, L., et al. (2014). Variability-tolerant convolutional neural network for pattern recognition applications based on oxram synapses. In *Electron devices meeting, 2014 IEEE international* (pp. 1–4). IEEE.
- 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.
- 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. (2009a). A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection. *Neural Networks*, 22(10), 1419–1431.
- Ghosh-Dastidar, S., & Adeli, H. (2009b). Spiking neural networks. *International Journal of Neural Systems*, 19(04), 295–308.
- Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580–587).
- Gollisch, T., & Meister, M. (2008). Rapid neural coding in the retina with relative spike latencies. *Science*, 319(5866), 1108–1111.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
- Grossberg, S. (1987). Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, 11(23–63).
- Gupta, A., & Long, L. N. (2007). Character recognition using spiking neural networks. In *Neural networks, 2007. IJCNN 2007. International joint conference on* (pp. 53–58). IEEE.
- Gütig, R., & Sompolinsky, H. (2006). The tempotron: A neuron that learns spike timing-based decisions. *Nature Neuroscience*, 9(3), 420–428.
- Guyonneau, R., VanRullen, R., & Thorpe, S. J. (2005). Neurons tune to the earliest spikes through STDP. *Neural Computation*, 17(4), 859–879.
- Hamel, P., & Eck, D. (2010). Learning features from music audio with deep belief networks. In *ISMIR: Vol. 10*, (pp. 339–344). Utrecht, The Netherlands.
- Hassabis, D., Kumaran, D., Summerfield, C., & Botvinick, M. (2017). Neuroscience-inspired artificial intelligence. *Neuron*, 95(2), 245–258.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).
- Herikstad, R., Baker, J., Lachaux, J., Gray, C., & Yen, S. (2011). Natural movies evoke spike trains with low spike time variability in cat primary visual cortex. *Journal of Neuroscience*, 31(44), 15844–15860.
- Hertz, J., Krogh, A., & Palmer, R. G. (1991). *Introduction to the theory of neural computation*. Addison-Wesley.
- Hinton, G. (2007). How to do backpropagation in a brain. In *Invited talk at the NIPS'2007 deep learning workshop: Vol. 656*.
- Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. -r., Jaitly, N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6), 82–97.
- Hinton, G. E., Osindero, S., & Teh, Y. -W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7), 1527–1554.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.
- Hochreiter, S., Younger, A. S., & Conwell, P. R. (2001). Learning to learn using gradient descent. In *Intl conf on artificial neural networks* (pp. 87–94). Springer.
- 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), 500–544.
- Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8), 2554–2558.
- Hopfield, J. J., et al. (1995). Pattern recognition computation using action potential timing for stimulus representation. *Nature*, 376(6535), 33–36.
- Hubel, D. H., & Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3), 574–591.
- Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160(1), 106–154.
- Huh, D., & Sejnowski, T. J. (2017). Gradient descent for spiking neural networks, arXiv preprint arXiv:1706.04698, 1–10.
- Hunsberger, E., & Eliasmith, C. (2015). Spiking deep networks with LIF neurons, arXiv preprint arXiv:1510.08829, 1–9.
- Hunsberger, E., & Eliasmith, C. (2016). Training spiking deep networks for neuromorphic hardware, arXiv preprint arXiv:1611.05141, 1–10.
- Indiveri, G., Corradi, F., & Qiao, N. (2015). Neuromorphic architectures for spiking deep neural networks. In *Electron devices meeting, 2015 IEEE international*. IEEE, (pp. 4–2).
- Izhikevich, E. M., et al. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6), 1569–1572.
- Jo, T., Hou, J., Eickholt, J., & Cheng, J. (2015). Improving protein fold recognition by deep learning networks. *Scientific Reports*, 5, 17573.
- Jolivet, R., Timothy, J., & Gerstner, W. (2003). The spike response model: A framework to predict neuronal spike trains. In *Artificial neural networks and neural information processing—ICANN/ICONIP 2003* (pp. 846–853). Springer.
- Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S. A., & Hudspeth, A. J. (2013). *Principles of neural science* (5th ed.). McGraw-Hill New York.
- Kang, S., Qian, X., & Meng, H. (2013). Multi-distribution deep belief network for speech synthesis. In *Acoustics, speech and signal processing, 2013 IEEE international conference on* (pp. 8012–8016). IEEE.
- Kappel, D., Nessler, B., & Maass, W. (2014). STDP installs in winner-take-all circuits an online approximation to hidden Markov model learning. *PLoS Computational Biology*, 10(3), e1003511.
- Kasabov, N. (2012). Evolving spiking neural networks and neurogenetic systems for spatio-and spectro-temporal data modelling and pattern recognition. In *Advances in computational intelligence* (pp. 234–260). Springer.
- Kasabov, N. K. (2014). NeuCube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data. *Neural Networks*, 52, 62–76.
- Kasabov, N. K. (2018). *Time-space, spiking neural networks and brain-inspired artificial intelligence*. Springer.
- 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, 188–201.
- Kasabov, N., Feigin, V., Hou, Z. -G., Chen, Y., Liang, L., Krishnamurthi, R., et al. (2014). Evolving spiking neural networks for personalised modelling, classification and prediction of spatio-temporal patterns with a case study on stroke. *Neurocomputing*, 134, 269–279.
- Kasinski, A., & Ponulak, F. (2006). Comparison of supervised learning platforms for spike time coding in spiking neural networks. *International Journal of Applied Mathematics and Computer Science*, 16(1), 101–113.
- Kheradpisheh, S. R., Ganjtabesh, M., & Masquelier, T. (2016a). Bio-inspired unsupervised learning of visual features leads to robust invariant object recognition. *Neurocomputing*, 205, 382–392.
- Kheradpisheh, S. R., Ganjtabesh, M., Thorpe, S. J., & Masquelier, T. (2018). STDP-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99, 56–67.
- Kheradpisheh, S. R., Ghodrati, M., Ganjtabesh, M., & Masquelier, T. (2016b). Humans and deep networks largely agree on which kinds of variation make object recognition harder. *Frontiers in Computational Neuroscience*, 10, 92.
- Kheradpisheh, S. R., Ghodrati, M., Ganjtabesh, M., & Masquelier, T. (2016c). Deep networks can resemble human feed-forward vision in invariant object recognition. *Scientific Reports*, 6, 32672.
- King, P. D., Zylberberg, J., & DeWeese, M. R. (2013). Inhibitory interneurons decorrelate excitatory cells to drive sparse code formation in a spiking model of V1. *Journal of Neuroscience*, 33(13), 5475–5485.
- Kistler, W. M., Gerstner, W., & van Hemmen, J. (1997). Reduction of the Hodgkin-Huxley equations to a single-variable threshold model. *Neural Computation*, 9(5), 1015–1045.
- Klampafl, S., & Maass, W. (2013). Emergence of dynamic memory traces in cortical microcircuit models through STDP. *The Journal of Neuroscience*, 33(28), 11515–11529.
- Körding, K. P., & Wolpert, D. M. (2004). Bayesian integration in sensorimotor learning. *Nature*, 427(6971), 244–247.
- Krizhevsky, A., & Hinton, G. (2009). *Learning multiple layers of features from tiny images* (pp. 1–60). Citeseer, <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>.
- Krizhevsky, A., & Hinton, G. (2010). Convolutional deep belief networks on CIFAR-10, Unpublished Manuscript 40.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).
- Kröger, B. J., Kannampuzha, J., & Neuschaefer-Rube, C. (2009). Towards a neurocomputational model of speech production and perception. *Speech Communication*, 51(9), 793–809.
- Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1), 79–86.
- Kuremoto, T., Kimura, S., Kobayashi, K., & Obayashi, M. (2014). Time series forecasting using a deep belief network with restricted Boltzmann machines. *Neurocomputing*, 137, 47–56.
- Lagorce, X., Orchard, G., Galluppi, F., Shi, B. E., & Benosman, R. B. (2017). HOTS: A hierarchy of event-based time-surfaces for pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(7), 1346–1359.
- Lamb, A., Goyal, A., Zhang, Y., Zhang, S., Courville, A., & Bengio, Y. (2016). Professor forcing: A new algorithm for training recurrent networks, arXiv preprint arXiv:1610.09038.
- Le Roux, N., & Bengio, Y. (2008). Representational power of restricted Boltzmann machines and deep belief networks. *Neural Computation*, 20(6), 1631–1649.



- Le Roux, N., & Bengio, Y. (2010). Deep belief networks are compact universal approximators. *Neural Computation*, 22(8), 2192–2207.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015a). Deep learning. *Nature*, 521(7553), 436–444.
- LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998a). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.
- LeCun, Y., Cortes, C., & Burges, C. J. (1998b). The MNIST database, URL <http://yann.lecun.com/exdb/mnist>.
- LeCun, Y., et al. (2015b). LeNet-5, convolutional neural networks, URL: <http://yann.lecun.com/exdb/lenet>.
- Lee, J. H., Delbruck, T., & Pfeiffer, M. (2016). Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 10, 508.
- Lee, H., Ekanadham, C., & Ng, A. Y. (2008). Sparse deep belief net model for visual area V2. In *Advances in neural information processing systems* (pp. 873–880).
- Lee, H., Grosse, R., Ranganath, R., & Ng, A. Y. (2011). Unsupervised learning of hierarchical representations with convolutional deep belief networks. *Communications of the ACM*, 54(10), 95–103.
- Lee, H., Pham, P., Largman, Y., & Ng, A. Y. (2009). Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in neural information processing systems* (pp. 1096–1104).
- Liaw, J., -S., & Berger, T. W. (1998). Robust speech recognition with dynamic synapses. In *Neural networks proceedings, 1998. IEEE world congress on computational intelligence. The 1998 IEEE international joint conference on: Vol. 3*, (pp. 2175–2179). IEEE.
- Lillicrap, T. P., Cownden, D., Tweed, D. B., & Akerman, C. J. (2016). Randonne synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 1–10.
- Liu, P., Han, S., Meng, Z., & Tong, Y. (2014). Facial expression recognition via a boosted deep belief network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1805–1812).
- Liu, T., Liu, Z., Lin, F., Jin, Y., Quan, G., & Wen, W. (2017). MT-spike: A multilayer time-based spiking neuromorphic architecture with temporal error backpropagation. In *Proceedings of the 36th international conference on computer-aided design* (pp. 450–457). IEEE Press.
- Liu, Q., -s., Pu, L., & Poo, M. -m. (2005). Repeated cocaine exposure in vivo facilitates LTP induction in midbrain dopamine neurons. *Nature*, 437(7061), 1027.
- Loiselle, S., Rouat, J., Pressnitzer, D., & Thorpe, S. (2005). Exploration of rank order coding with spiking neural networks for speech recognition. In *Neural networks, 2005. IJCNN'05. Proceedings. 2005 IEEE international joint conference on: Vol. 4*, (pp. 2076–2080). IEEE.
- Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3431–3440).
- Lukoševičius, M., & Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3), 127–149.
- Lukoševičius, M., Jaeger, H., & Schrauwen, B. (2012). Reservoir computing trends. *KI-Künstliche Intelligenz*, 26(4), 365–371.
- Maass, W. (1996). Lower bounds for the computational power of networks of spiking neurons. *Neural Computation*, 8(1), 1–40.
- Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. *Neural networks*, 10(9), 1659–1671.
- Maass, W. (2015). To spike or not to spike: That is the question. *Proceedings of the IEEE*, 103(12), 2219–2224.
- Maass, W., Natschläger, T., & Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11), 2531–2560.
- Mainen, Z. F., & Sejnowski, T. J. (1995). Reliability of spike timing in neocortical neurons. *Science*, 268, 1503–1506.
- Mamoshina, P., Vieira, A., Putin, E., & Zhavoronkov, A. (2016). Applications of deep learning in biomedicine. *Molecular Pharmaceutics*, 13(5), 1445–1454.
- Marčelja, S. (1980). Mathematical description of the responses of simple cortical cells. *Journal of the Optical Society of America*, 70(11), 1297–1300.
- Markram, H., Gerstner, W., & Sjöström, P. J. (2011). A history of spike-timing-dependent plasticity. *Frontiers in Synaptic Neuroscience*, 3, 1–24.
- Masquelier, T., Guyonneau, R., & Thorpe, S. J. (2008). Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains. *PLoS One*, 3(1), e1377.
- Masquelier, T., Guyonneau, R., & Thorpe, S. J. (2009). Competitive STDP-based spike pattern learning. *Neural computation*, 21(5), 1259–1276.
- Masquelier, T., & Kheradpisheh, S. R. (2018). Optimal localist and distributed coding of spatiotemporal spike patterns through STDP and coincidence detection, arXiv preprint arXiv:1803.00447, 99.
- Masquelier, T., & Thorpe, S. J. (2007). Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS Computational Biology*, 3(2), e31.
- Masquelier, T., & Thorpe, S. J. (2010). Learning to recognize objects using waves of spikes and spike timing-dependent plasticity. In *Neural Networks, The 2010 international joint conference on* (pp. 1–8). IEEE.
- Meftah, B., Lezoray, O., & Benyettou, A. (2010). Segmentation and edge detection based on spiking neural network model. *Neural Processing Letters*, 32(2), 131–146.
- Merolla, P., Arthur, J., Akopyan, F., Imam, N., Manohar, R., & Modha, D. S. (2011). A digital neurosynaptic core using embedded crossbar memory with 45pj per spike in 45nm. In *Custom integrated circuits conference, 2011 IEEE* (pp. 1–4). IEEE.
- Merolla, P. A., Arthur, J. V., Alvarez-Icaza, R., Cassidy, A. S., Sawada, J., Akopyan, F., et al. (2014). A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197), 668–673.
- Min, S., Lee, B., & Yoon, S. (2017). Deep learning in bioinformatics. *Briefings in Bioinformatics*, 18(5), 851–869.
- Mlecckzo, W. K., Kapuściński, T., & Nowicki, R. K. (2015). Rough deep belief network-application to incomplete handwritten digits pattern classification. In *International conference on information and software technologies* (pp. 400–411). Springer.
- Mohamed, A. -r., Dahl, G., & Hinton, G. (2009). Deep belief networks for phone recognition. In *NIPS workshop on deep learning for speech recognition and related applications: Vol. 1*, (9), (p. 39).
- Mohamed, A. -r., Dahl, G. E., & Hinton, G. (2012). Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1), 14–22.
- Mohammed, 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), 1250012.
- Mohammed, A., Schliebs, S., Matsuda, S., & Kasabov, N. (2013). Training spiking neural networks to associate spatio-temporal input-output spike patterns. *Neurocomputing*, 107, 3–10.
- Morrison, A., Aertsen, A., & Diesmann, M. (2007). Spike-timing-dependent plasticity in balanced random networks. *Neural Computation*, 19(6), 1437–1467.
- Mostafa, H. (2017). Supervised learning based on temporal coding in spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 1–9.
- Mozafari, M., Ganjtabesh, M., Nowzari-Dalini, A., Thorpe, S. J., & Masquelier, T. (2018a). Combining STDP and reward-modulated STDP in deep convolutional spiking neural networks for digit recognition, arXiv preprint arXiv:1804.00227.
- Mozafari, M., Kheradpisheh, S. R., Masquelier, T., Nowzari-Dalini, A., & Ganjtabesh, M. (2018b). First-spike based visual categorization using reward-modulated STDP. *IEEE Transactions on Neural Networks and Learning Systems*, 1–24, (in press).
- Moser, M. C., Pashler, H., & Homaei, H. (2008). Optimal predictions in everyday cognition: The wisdom of individuals or crowds? *Cognitive Science*, 32(7), 1133–1147.
- Näger, C., Storck, J., & Deco, G. (2002). Speech recognition with spiking neurons and dynamic synapses: A model motivated by the human auditory pathway. *Neurocomputing*, 44, 937–942.
- Neftci, E. O., Augustine, C., Paul, S., & Deterakis, G. (2017). Event-driven random back-propagation: Enabling neuromorphic deep learning machines. *Frontiers in Neuroscience*, 11, 324.
- Neftci, E., Das, S., Pedroni, B., Kreutz-Delgado, K., & Cauwenberghs, G. (2014). Event-driven contrastive divergence for spiking neuromorphic systems. *Frontiers in Neuroscience*, 8, 1–14.
- Neil, D., & Liu, S. -C. (2014). Minitaur, an event-driven FPGA-based spiking network accelerator. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 22(12), 2621–2628.
- Neil, D., & Liu, S. -C. (2016). Effective sensor fusion with event-based sensors and deep network architectures. In *Circuits and systems, 2016 IEEE international symposium on* (pp. 2282–2285). IEEE.
- Neil, D., Pfeiffer, M., & Liu, S. (2016a). Phased LSTM: Accelerating neural network training for long or event-based sequences. In *NIPS'16 Proceedings of the 30th international conference on neural information processing systems* (pp. 3889–3897).
- Neil, D., Pfeiffer, M., & Liu, S. -C. (2016b). Learning to be efficient: Algorithms for training low-latency, low-compute deep spiking neural networks. In *Proceedings of the 31st annual ACM symposium on applied computing* (pp. 293–298). ACM.
- Nessler, B., Pfeiffer, M., Buesing, L., & Maass, W. (2013). Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity. *PLoS Computational Biology*, 9(4), e1003037.
- Nessler, B., Pfeiffer, M., & Maass, W. (2009). STDP enables spiking neurons to detect hidden causes of their inputs. In *Advances in neural information processing systems* (pp. 1357–1365).
- O'Connor, P., Neil, D., Liu, S. -C., Delbruck, T., & Pfeiffer, M. (2013). Real-time classification and sensor fusion with a spiking deep belief network. *Frontiers in Neuroscience*, 7, 1–13.
- O'Connor, P., & Welling, M. (2016). Deep spiking networks, arXiv preprint arXiv:1602.08323, 1–16.
- Olshausen, B. A., et al. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583), 607–609.
- Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1717–1724).
- Orchard, G., Jayawant, A., Cohen, G. K., & Thakor, N. (2015). Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in Neuroscience*, 9, 437.
- Panchev, C., & Wermter, S. (2004). Spike-timing-dependent synaptic plasticity: From single spikes to spike trains. *Neurocomputing*, 58, 365–371.

- Panda, P., & Roy, K. (2016). Unsupervised regenerative learning of hierarchical features in spiking deep networks for object recognition. In *International conference on neural networks* (pp. 299–306). IEEE.
- Paulun, L., Wendt, A., & Kasabov, N. K. (2018). A retinotopic spiking neural network system for accurate recognition of moving objects using NeuCube and dynamic vision sensors. *Frontiers in Computational Neuroscience*, 12, 42.
- Pecevski, D., & Maass, W. (2016). Learning probabilistic inference through STDP. *neuro*, ENEURO–0048.
- Perez-Carrasco, J., Zhao, B., Serrano, C., Acha, B., Serrano-Gotarredona, T., Chen, S., et al. (2013). Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate-coding and coincidence processing. application to feed forward convnets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35, 2706–2719.
- Pfeiffer, M., & Pfeil, T. (2018). Deep learning with spiking neurons: Opportunities and challenges. *Frontiers in Neuroscience*, 12(774), 1–18.
- Pfister, J. -P., & Gerstner, W. (2006). Triplets of spikes in a model of spike timing-dependent plasticity. *Journal of Neuroscience*, 26(38), 9673–9682.
- Pfister, J. -P., Toyozumi, 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.
- Ponulak, F., & Kasinski, A. (2010). Supervised learning in spiking neural networks with ReSuMe: Sequence learning, classification, and spike shifting. *Neural Computation*, 22(2), 467–510.
- Pyle, R., & Rosenbaum, R. (2017). Spatiotemporal dynamics and reliable computations in recurrent spiking neural networks. *Physical Review Letters*, 118(1), 018103.
- Quang, D., & Xie, X. (2016). DanQ: A hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences. *Nucleic Acids Research*, 44(11), e107–e107.
- Querlioz, D., Bichler, O., Dollfus, P., & Gamrat, C. (2013). Immunity to device variations in a spiking neural network with memristive nanodevices. *IEEE Transactions on Nanotechnology*, 12(3), 288–295.
- Rao, R. P., Olshausen, B. A., & Lewicki, M. S. (2002). *Probabilistic models of the brain: Perception and neural function*. MIT press.
- Rawat, W., & Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, 2352–2449.
- Rehn, M., & Sommer, F. T. (2007). A network that uses few active neurones to code visual input predicts the diverse shapes of cortical receptive fields. *Journal of Computational Neuroscience*, 12(3), 135–146.
- Reinagel, P., & Reid, R. (2000). Temporal coding of visual information in the thalamus. *Journal of Neuroscience*, 20(14), 5392–5400.
- Rezende, D. J., Wierstra, D., & Gerstner, W. (2011). Variational learning for recurrent spiking networks. In *Advances in neural information processing systems* (pp. 136–144).
- Rieke, F. (1999). *Spikes: Exploring the neural code*. MIT press.
- Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention* (pp. 234–241). Springer.
- Rozenberg, G., Bck, T., & Kok, J. N. (2011). *Handbook of natural computing*. Springer Publishing.
- Rueckauer, B., Hu, Y., Lungu, I. -A., Pfeiffer, M., & Liu, S. -C. (2017). Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11, 682.
- Rueckauer, B., Lungu, I. -A., Hu, Y., & Pfeiffer, M. (2016). Theory and tools for the conversion of analog to spiking convolutional neural networks, arXiv preprint arXiv:1612.04052, 1–9.
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3), 211–252. <http://dx.doi.org/10.1007/s11263-015-0816-y>.
- Sainath, T. N., Mohamed, A. -r., Kingsbury, B., & Ramabhadran, B. (2013). Deep convolutional neural networks for LVCSR. In *Acoustics, speech and signal processing, 2013 IEEE international conference on* (pp. 8614–8618). IEEE.
- Salama, M. A., Hassani, A. E., & Fahmy, A. A. (2010). Deep belief network for clustering and classification of a continuous data. In *Signal processing and information technology, 2010 IEEE international symposium on* (pp. 473–477). IEEE.
- Savin, C., Joshi, P., & Triesch, J. (2010). Independent component analysis in spiking neurons. *PLoS Computational Biology*, 6(4), e1000757.
- Schliebs, S., Hamed, H. N. A., & Kasabov, N. (2011). Reservoir-based evolving spiking neural network for spatio-temporal pattern recognition. In *Neural information processing* (pp. 160–168). Springer.
- Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61, 85–117.
- Sengupta, A., Ye, Y., Wang, R., Liu, C., & Roy, K. (2018). Going deeper in spiking networks: VGG and residual architectures, arXiv [Preprint], arXiv:1802.02627v3.
- Seo, J. -s., Brezzo, B., Liu, Y., Parker, B. D., Esser, S. K., Montoye, R. K., et al. (2011). A 45nm cmos neuromorphic chip with a scalable architecture for learning in networks of spiking neurons. In *Custom integrated circuits conference, 2011 IEEE* (pp. 1–4). IEEE.
- Serre, T. (2014). Hierarchical models of the visual system. In *Encyclopedia of computational neuroscience* (pp. 1–12). Springer.
- Seung, H. (2003). Learning in spiking neural networks by reinforcement of stochastic synaptic transmission. *Neuron*, 40(6), 1063–1073.
- Shrestha, A., Ahmed, K., Wang, Y., Widemann, D. P., Moody, A. T., Van Essen, B. C., et al. (2017). A spike-based long short-term memory on a neurosynaptic processor. In *Computer-aided design, 2017 IEEE/ACM international conference on* (pp. 631–637). IEEE.
- Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556, 1–14.
- Sinha, R., Hoon, M., Baudin, J., Okawa, H., Wong, R. O., & Rieke, F. (2017). Cellular and circuit mechanisms shaping the perceptual properties of the primate fovea. *Cell*, 168(3), 413–426.
- Song, S., Miller, K. D., & Abbott, L. F. (2000). Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience*, 3(9), 919–926.
- Song, T., Pan, L., & Păun, G. (2013). Asynchronous spiking neural P systems with local synchronization. *Information Sciences*, 219, 197–207.
- Srivastava, K. H., Holmes, C. M., Vellema, M., Pack, A. R., Elemans, C. P., Nemenman, I., & Sober, S. J. (2017). Motor control by precisely timed spike patterns. *Proceedings of the National Academy of Sciences*, 201611734.
- Stone, J. V. (2018). *Principles of neural information theory: Computational neuroscience and metabolic efficiency*. Sebtel Press.
- Stromatias, E., Neil, D., Galluppi, F., Pfeiffer, M., Liu, S. -C., & Furber, S. (2015a). Scalable energy-efficient, low-latency implementations of trained spiking deep belief networks on spinnaker. In *Neural networks, 2015 international joint conference on* (pp. 1–8). IEEE.
- Stromatias, E., Neil, D., Pfeiffer, M., Galluppi, F., Furber, S. B., & Liu, S. -C. (2015b). Robustness of spiking deep belief networks to noise and reduced bit precision of neuro-inspired hardware platforms. *Frontiers in Neuroscience*, 9, 1–14.
- Stromatias, E., Soto, M., Serrano -Gotarredona, T., & Linares -Barranco, B. (2017). An event-driven classifier for spiking neural networks fed with synthetic or dynamic vision sensor data. *Frontiers in Neuroscience*, 11, 350.
- Susskind, J. M., Hinton, G. E., Movellan, J. R., & Anderson, A. K. (2008). Generating facial expressions with deep belief nets. In *Affective computing*. InTech.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., et al. (2015). *Going deeper with convolutions*. CVPR.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2818–2826).
- Tang, C., Chehayeb, D., Srivastava, K., Nemenman, I., & Sober, S. J. (2014). Millisecond-scale motor encoding in a cortical vocal area. *PLoS Biology*, 12(12), e1002018.
- Tavanaei, A., Kirby, Z., & Maida, A. S. (2018a). Training spiking convnets by STDP and Gradient Descent. In *Neural networks, The 2018 international joint conference on* (pp. 1–8). IEEE.
- Tavanaei, A., & Maida, A. S. (2015a). A minimal spiking neural network to rapidly train and classify handwritten digits in binary and 10-digit tasks. *International Journal of Advanced Research in Artificial Intelligence*, 4(7), 1–8.
- Tavanaei, A., & Maida, A. S. (2015b). Studying the interaction of a hidden Markov model with a Bayesian spiking neural network. In *Machine learning for signal processing, 2015 IEEE 25th international workshop on* (pp. 1–6). IEEE.
- Tavanaei, A., & Maida, A. S. (2016). Bio-inspired spiking convolutional neural network using layer-wise sparse coding and STDP learning, arXiv preprint arXiv:1611.03000, 1–16.
- Tavanaei, A., & Maida, A. (2017a). Bio-inspired multi-layer spiking neural network extracts discriminative features from speech signals. In *International conference on neural information processing* (pp. 899–908). Springer.
- Tavanaei, A., & Maida, A. S. (2017b). A spiking network that learns to extract spike signatures from speech signals. *Neurocomputing*, 240, 191–199.
- Tavanaei, A., & Maida, A. S. (2017c). Multi-layer unsupervised learning in a spiking convolutional neural network. In *Neural Networks, 2017 international joint conference on* (pp. 81–88). IEEE.
- Tavanaei, A., & Maida, A. S. (2018a). BP-STDP: Approximating Backpropagation using spike timing dependent plasticity. *Neurocomputing*, 1–12.
- Tavanaei, A., & Maida, A. S. (2018b). Training a hidden Markov model with a Bayesian spiking neural network. *Journal of Signal Processing Systems*, 90(2), 211–220.
- Tavanaei, A., Maida, A. S., Kaniyammattam, A., & Loganathanaraj, R. (2016a). Towards recognition of protein function based on its structure using deep convolutional networks. In *Bioinformatics and biomedicine, 2016 IEEE international conference on* (pp. 145–149). IEEE.
- Tavanaei, A., Masquelier, T., & Maida, A. S. (2016b). Acquisition of visual features through probabilistic spike timing dependent plasticity. In *Neural Networks, The 2016 international joint conference on* (pp. 1–8). IEEE.
- Tavanaei, A., Masquelier, T., & Maida, A. S. (2018b). Representation learning using event-based STDP. *Neural Networks*, 105, 294–303.
- Thiele, J. C., Bichler, O., & Dupret, A. (2018). Event-based, timescale invariant unsupervised online learning with STDP. *Frontiers in Computational Neuroscience*, 12, 1–13.
- Tubiana, J., & Monasson, R. (2017). Emergence of compositional representations in restricted Boltzmann machines. *Physical Review Letters*, 118(13), 138301.
- VanRullen, R. (2017). Perception science in the age of deep neural networks. *Frontiers in Psychology*, 8, 142.

- Venna, S. R., Tavanaei, A., Gottumukkala, R. N., Raghavan, V. V., Maida, A., & Nichols, S. (2017). A novel data-driven model for real-time influenza forecasting. *bioRxiv*, 185512.
- Victor, J. D. (2005). Spike train metrics. *Current Opinion in Neurobiology*, 15(5), 585–592.
- Victor, J. D., & Purpura, K. P. (1997). Metric-space analysis of spike trains: Theory, algorithms and application. *Network. Computation in Neural Systems*, 8(2), 127–164.
- Vreeken, J. (2003). *Spiking neural networks, an introduction*. Utrecht University: Information and Computing Sciences.
- 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., & McGinnity, T. M. (2014). An online supervised learning method for spiking neural networks with adaptive structure. *Neurocomputing*, 144, 526–536.
- Wu, Y., Deng, L., Li, G., Zhu, J., & Shi, L. (2017). Spatio-temporal backpropagation for training high-performance spiking neural networks, arXiv preprint arXiv: 1706.02609, 1–10.
- 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.
- Wysoski, S. G., Benuskova, L., & Kasabov, N. (2010). Evolving spiking neural networks for audiovisual information processing. *Neural Networks*, 23(7), 819–835.
- Yu, Q., Tang, H., Tan, K. C., & Yu, H. (2014). A brain-inspired spiking neural network model with temporal encoding and learning. *Neurocomputing*, 138, 3–13.
- Zemel, R. S., Natarajan, R., Dayan, P., & Huys, Q. J. (2004). Probabilistic computation in spiking populations. In *Advances in neural information processing systems* (pp. 1609–1616).
- Zeng, H., Edwards, M. D., Liu, G., & Gifford, D. K. (2016). Convolutional neural network architectures for predicting DNA–protein binding. *Bioinformatics*, 32(12), i121–i127.
- Zenke, F., & Ganguli, S. (2017). SuperSpike: Supervised learning in multi-layer spiking neural networks. *Neural Computation*, 30(6), 1514–1541.
- 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.
- Zylberberg, J., Murphy, J. T., & DeWeese, M. R. (2011). A sparse coding model with synaptically local plasticity and spiking neurons can account for the diverse shapes of V1 simple cell receptive fields. *PLoS Computational Biology*, 7(10), e1002250.