Deep learning in spiking neural networks

Saeed Reza Kheradpisheh, Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, Anthony Maida

▶ **To cite this version:**

Saeed Reza Kheradpisheh, Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, et al.. Deep learning in spiking neural networks. Neural Networks, 2019, 111, pp.47–63. 10.1016/j.neunet.2018.12.002 . hal-02341924

HAL Id: hal-02341924
https://hal.science/hal-02341924

Submitted on 8 Feb 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Deep Learning in Spiking Neural Networks

Amirhossein Tavanaei*, Masoud Ghodrati†, Saeed Reza Kheradpisheh‡,
Timothée Masquelier§ and Anthony Maida*

*Center for Advanced Computer Studies, University of Louisiana at Lafayette
Lafayette, Louisiana, LA 70504, USA
†Department of Physiology, Monash University, Clayton, VIC, Australia
‡Department of Computer Science, Faculty of Mathematical Sciences and Computer,
Kharazmi University, Tehran, Iran
§CERCO UMR 5549, CNRS-Université de Toulouse 3, F-31300, France

tavanaei@louisiana.edu, masoud.ghodrati@monash.edu, kheradpisheh@ut.ac.ir,
timothee.masquelier@cnrs.fr, maida@louisiana.edu

arXiv:1804.08150v4 [cs.NE] 20 Jan 2019

***Abstract*—[1] 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 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 arguably the only viable option if one wants to understand how the brain computes. SNNs are also more hardware friendly and energy-efficient than ANNs, and are thus appealing for technology, especially for portable devices. 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, but also computational cost and hardware friendliness. 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.**

Keywords: ***Deep learning, Spiking neural network, Biological plausibility, Machine learning, Power-efficient architecture.***

[1]**The final/complete version of this paper has been published in the Neural Networks journal. Please cite as:** *Tavanaei, A., Ghodrati, M., Kheradpisheh, S. R., Masquelier, T., and Maida, A. (2018). Deep learning in spiking neural networks. Neural Networks.*

I. 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 to unseen data and there have been huge advances in the performance of such networks.

A distinct historical landmark is the 2012 success of AlexNet [1] in the ILSVRC image classification challenge [2]. 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 [3], [4]. DNNs have been remarkably successful in many applications including image recognition [1], [5], [6], object detection [7], [8], speech recognition [9], biomedicine and bioinformatics [10], [11], temporal data processing [12], and many other applications [4], [13], [14]. These recent advances in artificial intelligence (AI) have opened up new avenues for developing different

engineering applications and understanding of how biological brains work [13], [14].

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 [15]. SNNs almost universally use idealized spike generation mechanisms in contrast to the actual biophysical mechanisms [16].

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 [17], [18], [19]. This precise temporal pattern in spiking activity is considered as a crucial coding strategy in sensory information processing areas [20], [21], [22], [23], [24] and neural motor control areas in the brain [25], [26]. SNNs have become the focus of a number of recent applications in many areas of pattern recognition such as visual processing [27], [28], [29], [30], speech recognition [31], [32], [33], [34], [35], [36], [37], and medical diagnosis [38], [39]. 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 identifying auditory targets in a noisy environment is a result of several processing stages and multiple learning mechanisms embedded in deep spiking networks [40], [41], [42]. 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. 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 [43]. This same advantage is maintained in hardware [44], [45], [46], [47]. Thus, it is possible to create low energy spiking hardware based on the property that spikes are sparse in time.

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 [48]. 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 [49]. Bio-inspired SNNs, in principle, have higher representation power and capacity than traditional rate-coded networks [50]. 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 [51], [52], [53], [54], [55].

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 [56], [57]. 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 [58], 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 [59], [60], [61]. 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 [23], [24], [25].

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.

II. Spiking Neural Network: A Biologically Inspired Approach to Information Processing

The introduction of SNNs in the last few decades, as a powerful third generation neural network [50], has encouraged many studies with the focus on biologically motivated approaches for pattern recognition [62], [63]. 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 [64], [65] 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.

A. 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 [64], [15], some form of temporal coding [66], [67], or population coding [68]. As stated earlier, a biological neuron in the brain (and similarly in a simulated spiking neuron) receives synaptic inputs form 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 [16]. 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 [16], [64], [69], diverse neuron models have been proposed such as the spike response model (SRM) [70], the Izhikevich neuron model [71], and the leaky integrated-and-fire (LIF) neuron [72]. 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 [73]. 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 back-propagation algorithm.

B. 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 [74]. 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.

1) Unsupervised Learning via STDP: As stated above, unsupervised learning in SNNs often involves STDP as part of the learning mechanism [74], [75]. 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 below idealizes the most common experimentally observed STDP rule for a single pair of spikes obtained by fitting to experimental data [76].

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

w is the synaptic weight. $A > 0$ and $B < 0$ are usually constant parameters indicating learning rates. τ 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) [77], [78]. Spike trains in many areas of the brain are highly reproducible. Guyonneau et al. [78] 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 [79], [80]. 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 [79], [81]. Using this approach, more complex networks with multiple output neurons have been developed [59], [82], [83], [84].

2) Probabilistic Characterization of Unsupervised STDP: Many studies have provided evidence that at least an approximate Bayesian analysis of sensory stimuli occurs in the brain [85], [86], [87], [88]. 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 et al. (2009) [89] 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 et al. (2013) [90] 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.

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

Building on the stochastic WTA circuits described above, Klampfl and Maass (2013) [91] developed a liquid state machine (LSM) 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, is able to represent the underlying statistical structure of such spatio-temporal input patterns [92]. 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 [89], [90] 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 [93], [94] 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 [94].

Using probabilistic learning in spiking neurons for modeling hidden causes has recently attracted attention. Rezende et al. (2011) [95] 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 [95]. The learning strategy involved minimizing the Kullback-Leibler divergence [96] as a non-commutative distance measure between the distribution representing the model (SNN) and a target distribution (observation). The EM algorithm in recurrent SNNs [97] and probabilistic association between neurons generated by STDP in combination with intrinsic plasticity [98] are two other instances of probabilistic learning in SNNs. The probabilistic rules also have been employed in sequential data processing [99] and Markov chain Monte Carlo sampling interpreted by stochastic firing activity of spiking neurons [100].

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.

a) 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 [101].

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

In the above, δ_j^μ and δ_k^μ denote the partial derivative of the cost function for input pattern μ 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^μ. 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 [102, p. 49] and termed the "weight transport" problem. The problem is the following. The expression $\sum_k w_{kj}\delta_k^\mu$ is using 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 useable.

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. [57], Section III-A 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 [103] and [104]. It was shown in [103] that for some tasks, backpropagation could still perform well if random feedback weights were used. The authors in [104] 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.

b) Some Supervised Learning Methods for SNNs: SpikeProp [105] 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 [64] 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 [106], [107]. Using the same neural architecture of SpikeProp, new formulations of temporal spike coding and spike time errors have recently improved the spiking backpropagation algorithm [108], [109]. The most recent implementation of backpropagation in SNNs has been proposed by Wu et. al. (2017) [110] who developed spatio-temporal gradient descent in multi-layer SNNs.

More recent approaches to supervised training of SNNs include ReSuMe (remote supervised learning) [111], [112], Chronotron [113], and SPAN (spike pattern association neuron) [114], [115], 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 is proportional to the desired output minus the observed output, as shown below.

$$\Delta w = (y^{\mathrm{d}} - y^{\mathrm{o}})x = y^{\mathrm{d}}x - y^{\mathrm{o}}x \tag{4}$$

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

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

In the above, Δw^{STDP} is a function of the correlation of the presynaptic and desired spike trains, whereas $\Delta w^{\mathrm{aSTDP}}$ 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 [116] 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-Purpora (VP) distance metric [117] between two spike trains. This metric is "the minimum cost of transforming one spike train into the other by creating, removing, or moving spikes." [113, 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 that was used in

SpikeProp described at the beginning of this subsection. The learning rule can then be written as

$$\Delta w \propto \int \tilde{x}_i(\tilde{y}_{\mathrm{d}}(t) - \tilde{y}_o(t))dt \tag{6}$$

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

In [56], 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 occured near spike times "bearing close resemblence to reward-modulated STDP" [56, p. 8], which enhances the biological relevance of the model.

In [118], 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 III-A).

Another implementation of supervised learning in SNNs can be based on optimizing the likelihood and probability of the postsynaptic spikes to match the desired ones. Pfister et al. (2006) [119] 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 [120], handwritten digit recognition [121], spoken digit classification [83], and reinforcement learning in SNNs [122]. The sharp synaptic weight adaptation based on immediate STDP and anti-STDP results in fast learning.

III. 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 [3], [4]. 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 [123], [124], [125]. Deep neural networks extract complex features through sequential layers of neurons equipped by 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 exhaustive probability values. The softmax formula, sometimes called the normalized exponential, is given below

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

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 III-A, there are also deep convolutional neural networks (DCNNs) discussed in Section III-B, deep belief networks (DBNs) discussed in Section III-C, and recurrent neural networks (RNNs) discussed in Section III-D.

SNNs have also shown promising performance in a number of pattern recognition tasks [62], [126]. However, the performance of directly trained spiking deep networks are 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

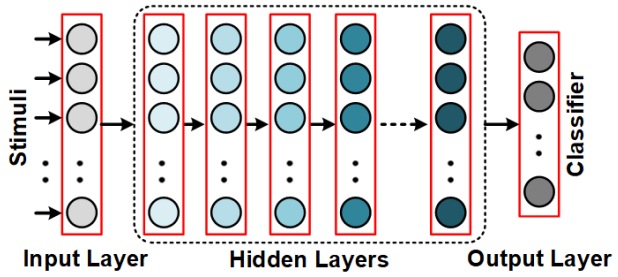


Fig. 1: Simplest deep neural architecture, usually fully connected, with input, hidden, and output layers. The input layer learns to perform pre-processing on the input. The information is then sent to a series of hidden layers, the number of which can vary. As the information propagates through hidden 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).

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 network of spiking neurons for unsupervised learning of visual features [59], [82]. This network was extended for larger problems, such as [84]. Using an STDP rule with a probabilistic interpretation, the performance of the model was later improved in different object recognition tasks [60]. Further attempts led to several multi-layer SNNs, with STDP learning, that performed greatly in adaptive multi-view pattern recognition [127] and handwritten digit recognition [61]. 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. Previous studies exploring supervised and unsupervised learning rules in spiking architectures can be employed to develop hierarchies of feature extraction and classification modules. Existing deep SNNs do not perform as accurately as the traditional deep learning models. However, SNNs enable power-efficient platforms mimicking the brain functionality for solving complex problems, especially in new trends of autonomous objects. Therefore, developing a neural network that is as efficient and biologically plausible as SNNs but as powerful as DNNs in performing different tasks can be the next challenging topic in the field of artificial intelligence and computational neuroscience. The initial steps for implementing a deep spiking framework can be fulfilled by either converting the trained neural networks to a spiking platform or using spike-based, online learning methods.

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

A. 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 [128], [129]. Diehl et al. [130] 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 [131], [90], [132] to train the SNN for the Modified National Institute of Standards and Technology (MNIST) dataset [133] 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. For example, Bengio et al. [134] proposed a deep learning method using forward and backward neural activity propagation. The learning rule in this network is based on the idea that STDP implements the gradient descent learning rule [135], [136]. Using pre- and postsynaptic spike trains, O'Connor and Welling [137] 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) [57] proposed a backpropagation algorithm by treating the neuron's membrane potential as the differentiable signal to act analogous to the nonlinear activation functions in traditional neural networks (Fig. 2). The performance of 98.88% on the MNIST dataset was reported in this study while the number of computational operations were five times fewer than traditional DNNs, in their experiments. To further reduce

the computational cost of learning in these deep SNNs, Neftci et. al. (2017) [138] proposed an event-driven random backpropagation (eRBP) algorithm simplifying the backpropagation chain path. The eRBP rule used an error-modulated synaptic plasticity in which all the information used for learning was locally available at the neuron and synapse [138].

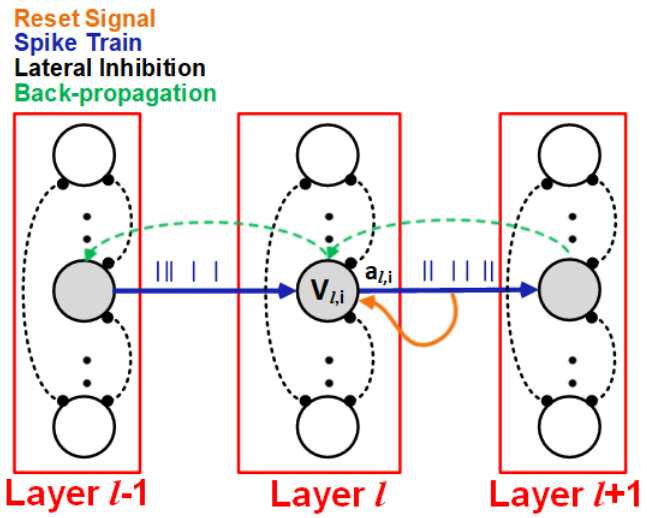


Fig. 2: Deep SNN equipped with backpropagation proposed by Lee et. al. [57]. The neuron's activation value, $a_{l,i}$, is given by the neuron's membrane potential. The differentiable activation function, which is calculated by the neuron's excitatory input, lateral inhibition, and threshold, is used 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.

A more direct approach to taking advantage of power-efficient SNNs is to convert an offline trained DNN to a neuromorphic spiking platform (ANN-to-SNN), specifically for hardware implementation [139]. 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 [140], [141], [142], [143]. In a separate effort to assess the power consumption of deep SNNs, Neil et al [144] 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 784-1200-1200-10 three-layer architecture but applied optimized parameters and SNN architecture settings, in ANN-to-SNN conversion, to reduce the power and latency of the model. The performance of a DNN and its converted deep SNN versus the total required operations is shown in Fig. 3.

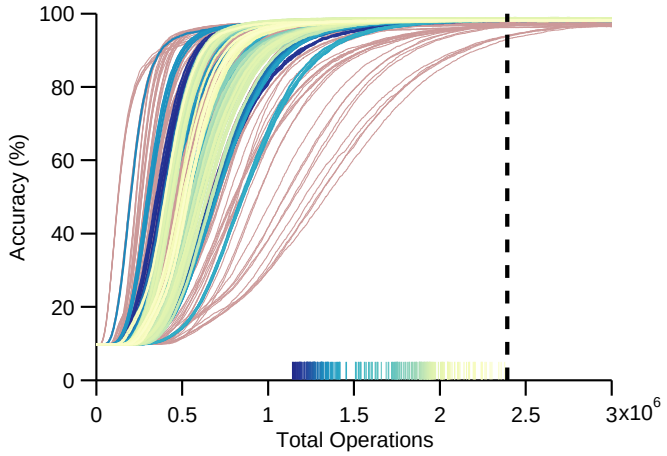


Fig. 3: The total number of operations needed to achieve a given accuracy for MNIST classification by deep SNNs converted from an offline trained deep neural network in comparison with the traditional (non-spiking) deep neural network [144]. The vertical dashed line shows the number of operations required for the non-spiking deep neural network to achieve the 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.

B. Spiking CNNs

Deep convolutional neural networks (DCNNs) are mostly used in applications involving images. They consist of a sequence of convolution and pooling (sub-sampling) layers followed by a feedforward classifier like that in Fig. 1. This type of network has shown outstanding performance in image recognition [1], [145], [146], [147], speech recognition [148], [149], [150], bioinformatics [151], [152], [153], object detection and segmentation [154], [7], and so forth. Fig. 4 shows the LeNet architecture, an early deep CNN, for image classification [155], [156], [157]. The question is how an 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 [158]). 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 [159], [160]. 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 [161], [162], [163], [164]. Bio-inspired SNNs also have obvious footprints in representation learning using sparse coding [165], [166], independent component analysis (ICA) [167], and an STDP-based autoencoder [168].

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 [59], [84], [60], [169], [170]. 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. A recent study has used a layer of DoG filters as input layer of an SNN that is followed by more convolutional/pooling layers trained by STDP [170]. This network architecture extracted visual features that were sent to an SVM classifier, yielding accuracy of 98.4% on MNIST. To train convolutional filters, layer-wise spiking representation learning approaches have been implemented in recent spiking CNNs [171], [172], [173], [174]. Tavanaei et al. (2017) [171], [172] used SAILnet [165] 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 [60] to extract visual features for classification. Implementing the stacked convolutional autoencoders [173] 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 [57], [173]. Panda and Roy (2016) [173], using approximations developed in [175], 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 dataset (handwritten digits) [155] and three learning layers on the CIFAR-10 dataset (10-category tiny images) [176]. They used local, layer-wise learning of convolutional layers while Lee et. al. (2016) [57] 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) [57] showed better performance than the layer-wise convolutional autoencoders [173]. In these models, higher membrane potential correlates with higher spike probability.

The main approach to take advantage of spiking platforms while avoiding the training process of spiking CNNs is to convert an already trained CNN to a spiking architecture by using the trained synaptic weights, 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 [177], [178], [141], [179], which enable the deep CNNs to be implemented on hardware [180], [181], [182]. One of the initial successful CNN-to-SNN conversion methods for energy efficient pattern recognition is the architecture shown in Fig. 6 [183]. Later, Diehl et al. (2015) [140] improved this architecture using weight normalization to reduce performance loss. Recent work by Rueckauer et al. [184], [142] proposed several conversion criteria such that the new spiking CNN recognize's more difficult objects than MNIST (e.g. CIFAR-10 and ImageNet [185]).

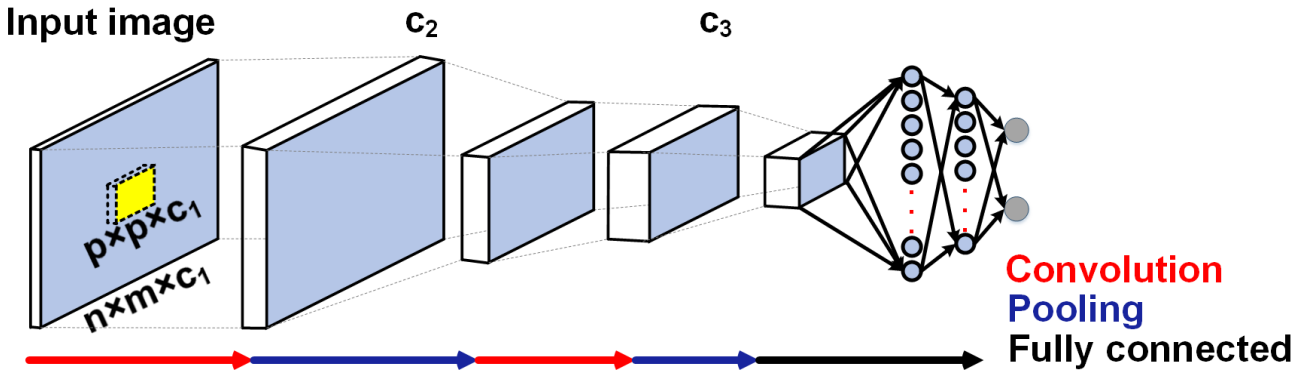


Fig. 4: LeNet: Early CNN proposed by LeCun et. al. [155], [156]. The network consists of two convolutional/pooling layers followed by fully connected layers for image classification.

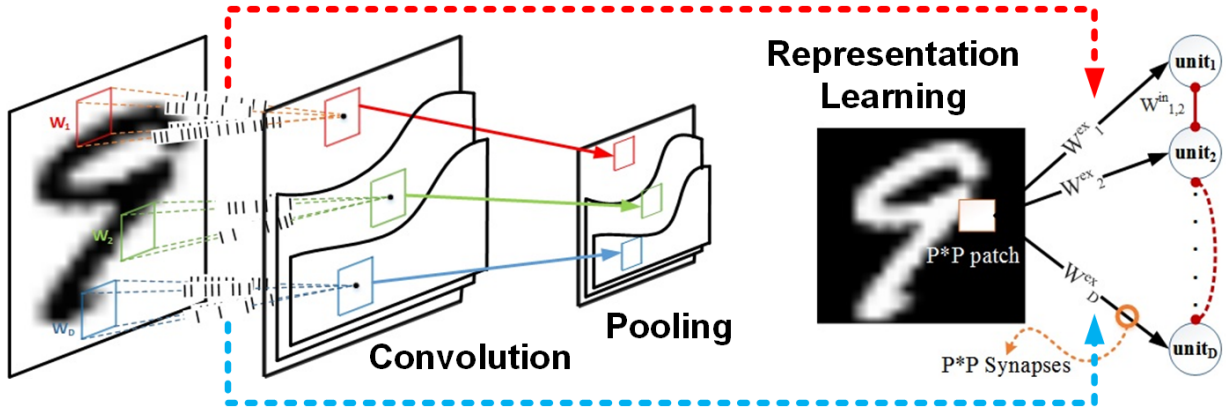


Fig. 5: Representation learning (SAILnet [165]) for layer-wise unsupervised learning of a spiking CNN [172]. The excitatory synaptic weights connected to neurons in the representation layer specify convolutional filters. This architecture determines that representation learning in single-layer SNNs can be utilized to train layer-wise spiking CNNs.

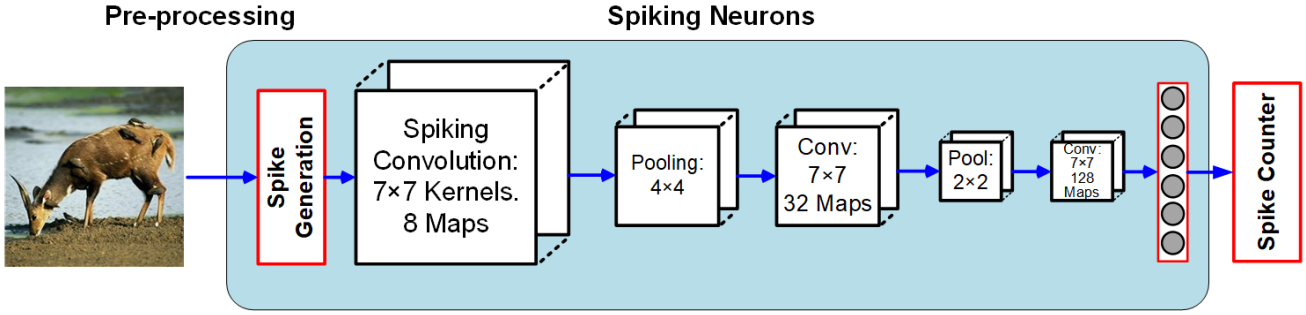


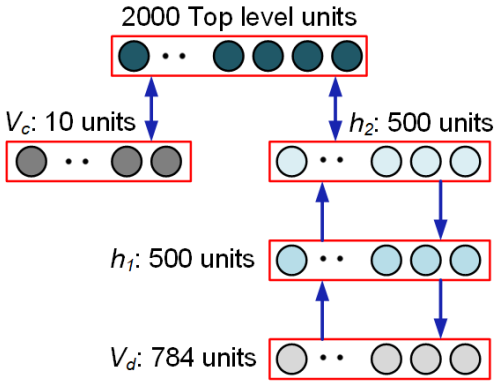
Fig. 6: Spiking CNN architecture developed by Cao et. al. [183]. The input image, after pre-processing, is converted to spike trains based on the pixel intensity. The spiking layers use the weights trained by a non-spiking CNN. The last component selects the neuron with maximum activity (spike frequency) as the image's class.

C. Spiking Deep Belief Networks

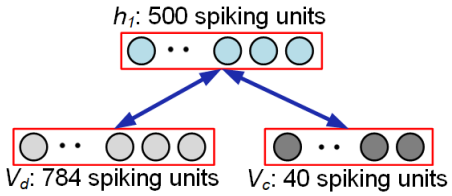
Deep belief networks (DBNs) [48] are a type of multi-layer network initially developed by Hinton et al. (2006) [186]. They efficiently use greedy layer-wise unsupervised learning and are made of stochastic binary units, meaning that the binary state of the unit is updated using a probability function. 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 Figure 7a. 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 [187], [188], [189].

Lee et. al. (2008) [190] 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 [191]. 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 [192], [193], [194], [195], audio processing [196], [197], [198], [199], [200], time series forecasting [201], and protein folding [202].

The first step in developing a spiking DBN is to start with a spiking RBM. Figure 7b shows the architecture of a spiking RBM introduced by Neftci et al. (2014) [203]. 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.



(a)



(b)

Fig. 7: (a): The DBN proposed by Hinton et. al. [186] 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, 28×28) and 10 (as the number of classes, 0,...,9) neurons, respectively. (b): The spiking RBM architecture introduced by Neftci et. al. [203] consisting of 500 hidden neurons, 784 input neurons, and 40 class neurons (824 visible neurons).

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 et al. (2013) [204] in which a DBN is converted to a network of LIF spiking neurons for MNIST image classification. This work was then extended [205] 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 [206], [207], [208].

Recently, it has been shown there is an equivalence between so called 'hybrid' Boltzmann machines (HBMs) [209] and Hopfield networks [210], [211]. 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) [209]. When

the functions within the HBM are 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 [212], [213], [214].

D. 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 II-B3) presupposes the existence of recurrent connections. Sections III-D1 and III-D2 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.

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 [215].

Because of these problems, the research in [216] 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 and have virtually replaced traditional RNNs. The first gated recurrent network was the long short-term memory (LSTM) [216]. LSTMs and other gated recurrent networks (GRUs) [217] 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.

Shrestha et al. (2017) [218] implemented an energy-efficient spiking LSTM onto the IBM TrueNorth neurosynaptic system platform [219]. 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 III-B 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 [220], 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 usal 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 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.

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 2,500 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 microcircuit has been used. There has been great interest in modeling this hypothesized module.

In an effort to model the computations that might be taking place within the canonical neocortical microcircuit, the liquid state machine (LSM) was introduced [221] which has since been partly absorbed by the field of reservoir computing [222], [223]. 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 [224]. 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) [126], [225], [226] is a broad and futuristic model proposed as a unifying computational architecture for modeling multimodality spatiotemporal data, most especially data related to the brain, such as EEG analysis [227]. 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 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 [228] 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 III-D1) with pseudo derivatives using the membrane potential (as explained in Section II-B3). The network achieved comparable accuracy performance to LSTMs on the sequential MNIST benchmark and also on the TIMIT Acoustic-Phonetic Continuous Speech Corpus. Sequencial MNIST is a sequence learning benchmark for assessing recurrent network performance first described in [229]. 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 [230], it was shown that LSTMs had this property.

Another recent study, [231], 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 lead 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 an avenue for interdisciplinary 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.

E. Performance Comparisons of Contemporary Models

Table I shows the 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 choose the right tool for the right purpose in future investigations.

IV. Summary

Deep learning approaches have recently shown breakthrough performance in many areas of pattern recognition recently. In spite of their effectiveness in hierarchical feature extraction and classification, these types of neural networks are computationally expensive and difficult to implement on hardware for portable devices. In another line of research of neural network architectures, SNNs have been described as power-efficient models because of their sparse, spike-based communication framework. Recent studies try to take advantages of both frameworks (deep learning and SNN) to develop a multi-layer SNN architecture to achieve high performance of recently proved deep networks while implementing bio-inspired, power-efficient platforms. Additionally, the literature has shown that the brain detects stimuli patterns through multi-layer SNNs communicating by spike trains via adaptive synapses. The biologically realistic spiking neurons communicate using spike trains which do not have obvious derivatives. This makes SNNs unable to directly use derivative-based optimization for training. This paper reviewed novel learning approaches for different layers of SNNs to address some of the open questions in this field. As biological neurons use sparse, stochastic, spike-based communication, a spiking network can be an appropriate starting point for modeling the brain's functionality.

SNNs with specific neural architectures demand new neuron models and learning techniques. Spiking neurons communicate through discrete spike trains via synapses adapting locally to distinguish the pattern of stimuli. The quest to meet these requirements can be accomplished by bio-inspired neural simulations for integrating the stimuli and releasing discriminative spike patterns according to the adaptive filters associated with synaptic weight sets. An important challenge in developing SNNs is to develop appropriate learning rules to detect spatio-temporally local patterns of spike trains. In this paper we reviewed state-of-the-art deep SNNs developed to reach the performance of conventional deep learning methods while providing a bio-inspired, power-efficient platform. Three popular deep learning methods as deep fully connected SNNs, spiking CNNs, and spiking DBNs were reviewed. The performances reported by recent approaches determine that the spike-based deep learning methods perform as well as traditional DNNs. Furthermore, SNNs are based on the human brain functionality and are able to perform much better than traditional ones in the future, as human brain does. This paper reviewed methods, network architectures, experiments, and results of recently proposed deep spiking networks to be useful for next studies and experiments.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

TABLE I: Summary of recent deep learning models developed in SNN platforms and their accuracy on MNIST [155], [133], N-MNIST [232], CIFAR-10, and CIFAR-100 [176]. 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 (2016) [137]	Deep SNN	Stochastic gradient descent	MNIST	96.40
O'Connor (2016) [137]	Deep SNN	Fractional stochastic gradient descent	MNIST	97.93
Lee (2016) [57]	Deep SNN	Backpropagation	MNIST	98.88
Lee (2016) [57]	Deep SNN	Backpropagation	N-MNIST	98.74
Neftci (2017) [138]	Deep SNN	Event-driven random backpropagation	MNIST	97.98
Liu (2017) [108]	SNN	Temporal backpropagation (3-layer)	MNIST	99.10
Eliasmith (2012) [129]	SNN	Spaun brain model	MNIST	94.00
Diehl (2015) [130]	SNN	STDP (2-layer)	MNIST	95.00
Tavanaei (2017) [118]	SNN	STDP-based backpropagation (3-layer)	MNIST	97.20
Mostafa (2017) [109]	SNN	Temporal backpropagation (3-layer)	MNIST	97.14
Querlioz (2013) [139]	SNN	STDP, Hardware implementation	MNIST	93.50
Brader (2007)[128]	SNN	Spike-driven synaptic plasticity	MNIST	96.50
Diehl (2015) [140]	Deep SNN	Offline learning, Conversion	MNIST	98.60
Neil (2016) [144]	Deep SNN	Offline learning, Conversion	MNIST	98.00
Hunsberger (2015) [177], [178]	Deep SNN	Offline learning, Conversion	MNIST	98.37
Esser (2015) [141]	Deep SNN	Offline learning, Conversion	MNIST	99.42
Spiking CNNs				
Lee (2016) [57]	Spiking CNN	Backpropagation	MNIST	99.31
Lee (2016) [57]	Spiking CNN	Backpropagation	N-MNIST	98.30
Panda (2016) [173]	Spiking CNN	Convolutional autoencoder	MNIST	99.05
Panda (2016) [173]	Spiking CNN	Convolutional autoencoder	CIFAR-10	75.42
Tavanaei (2017) [171], [172]	Spiking CNN	Layer wise sparse coding and STDP	MNIST	98.36
Tavanaei (2018) [174]	Spiking CNN	Layer-wise and end-to-end STDP rules	MNIST	98.60
Kheradpisheh (2016) [170]	Spiking CNN	Layer wise STDP	MNIST	98.40
Zhao (2015) [169]	Spiking CNN	Tempotron	MNIST	91.29
Cao (2015) [183]	Spiking CNN	Offline learning, Conversion	CIFAR-10	77.43
Neil (2016) [179]	Spiking CNN	Offline learning, Conversion	N-MNIST	95.72
Diehl (2015) [140]	Spiking CNN	Offline learning, Conversion	MNIST	99.10
Rueckauer (2017) [142]	Spiking CNN	Offline learning, Conversion	MNIST	99.44
Rueckauer (2017) [142]	Spiking CNN	Offline learning, Conversion	CIFAR-10	90.85
Hunsberger (2015) [177]	Spiking CNN	Offline learning, Conversion	CIFAR-10	82.95
Garbin (2014) [181]	Spiking CNN	Offline learning, Hardware	MNIST	94.00
Esser (2016) [182]	Spiking CNN	Offline learning, Hardware	CIFAR-10	87.50
Esser (2016) [182]	Spiking CNN	Offline learning, Hardware	CIFAR-100	63.05
Spiking RBMs and DBNs				
Neftci (2014) [203]	Spiking RBM	Contrastive divergence in LIF neurons	MNIST	91.90
O'Connor (2013) [204]	Spiking DBN	Offline learning, Conversion	MNIST	94.09
Stromatias (2015) [205]	Spiking DBN	Offline learning, Conversion	MNIST	94.94
Stromatias (2015) [206]	Spiking DBN	Offline learning, Hardware	MNIST	95.00
Merolla (2011) [207]	Spiking RBM	Offline learning, Hardware	MNIST	94.00
Neil (2014) [208]	Spiking DBN	Offline learning, Hardware	MNIST	92.00

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[5] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[7] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on*

computer vision and pattern recognition, 2014, pp. 580–587.

[9] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[10] P. Mamoshina, A. Vieira, E. Putin, and A. Zhavoronkov, "Applications of deep learning in biomedicine," *Molecular Pharmaceutics*, vol. 13, no. 5, pp. 1445–1454, 2016.

[11] S. Min, B. Lee, and S. Yoon, "Deep learning in bioinformatics," *Briefings in bioinformatics*, vol. 18, no. 5, pp. 851–869, 2017.

[12] S. R. Venna, A. Tavanaei, R. N. Gottumukkala, V. V. Raghavan, A. Maida, and S. Nichols, "A novel data-driven model for real-time influenza forecasting," *bioRxiv*, p. 185512, 2017.

[13] D. Hassabis, D. Kumaran, C. Summerfield, and M. Botvinick, "Neuroscience-inspired artificial intelligence," *Neuron*, vol. 95, no. 2, pp. 245–258, 2017.

[14] R. VanRullen, "Perception science in the age of deep neural networks," *Frontiers in Psychology*, vol. 8, p. 142, 2017.

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

[16] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, pp. 500–544, 1952.

[17] Z. F. Mainen and T. J. Sejnowski, "Reliability of spike timing in neocortical neurons," *Science*, vol. 268, pp. 1503–1506, 1995.

[18] W. Bair and C. Koch, "Temporal precision of spike trains in extrastriate cortex of the behaving macaque monkey," *Neural Computation*, vol. 8, no. 6, pp. 1185–1202, 1996.

[19] R. Herikstad, J. Baker, J. Lachaux, C. Gray, and S. Yen, "Natural movies evoke spike trains with low spike time variability in cat primary visual cortex," *Journal of Neuroscience*, vol. 31, no. 44, pp. 15 844–15 860, 2011.

[20] T. Gollisch and M. Meister, "Rapid neural coding in the retina with relative spike latencies," *Science*, vol. 319, no. 5866, pp. 1108–1111, 2008.

[21] R. Sinha, M. Hoon, J. Baudin, H. Okawa, R. O. Wong, and F. Rieke, "Cellular and circuit mechanisms shaping the perceptual properties of the primate fovea," *Cell*, vol. 168, no. 3, pp. 413–426, 2017.

[22] J. D. Victor, "Spike train metrics," *Current opinion in neurobiology*, vol. 15, no. 5, pp. 585–592, 2005.

[23] D. A. Butts, C. Weng, J. Jin, C.-I. Yeh, N. A. Lesica, J.-M. Alonso, and G. B. Stanley, "Temporal precision in the neural code and the timescales of natural vision," *Nature*, vol. 449, no. 7158, pp. 92–95, 2007.

[24] P. Reinagel and R. C. Reid, "Temporal coding of visual information in the thalamus," *Journal of Neuroscience*, vol. 20, no. 14, pp. 5392–5400, 2000.

[25] K. H. Srivastava, C. M. Holmes, M. Vellema, A. R. Pack, C. P. Elemans, I. Nemenman, and S. J. Sober, "Motor control by precisely timed spike patterns," *Proceedings of the National Academy of Sciences*, p. 201611734, 2017.

[26] C. Tang, D. Chehayeb, K. Srivastava, I. Nemenman, and S. J. Sober, "Millisecond-scale motor encoding in a cortical vocal area," *PLoS biology*, vol. 12, no. 12, p. e1002018, 2014.

[27] S. G. Wysoski, L. Benuskova, and N. Kasabov, "Evolving spiking neural networks for audiovisual information processing," *Neural Networks*, vol. 23, no. 7, pp. 819–835, 2010.

[28] A. Gupta and L. N. Long, "Character recognition using spiking neural networks," in *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*. IEEE, 2007, pp. 53–58.

[29] B. Meftah, O. Lezoray, and A. Benyettou, "Segmentation and edge detection based on spiking neural network model," *Neural Processing Letters*, vol. 32, no. 2, pp. 131–146, 2010.

[30] M.-J. Escobar, G. S. Masson, T. Vieville, and P. Kornprobst, "Action recognition using a bio-inspired feedforward spiking network," *International Journal of Computer Vision*, vol. 82, no. 3, pp. 284–301, 2009.

[31] J.-S. Liaw and T. W. Berger, "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. IEEE, 1998, pp. 2175–2179.

[32] B. J. Kröger, J. Kannampuzha, and C. Neuschaefer-Rube, "Towards a neurocomputational model of speech production and perception," *Speech Communication*, vol. 51, no. 9, pp. 793–809, 2009.

[33] C. Panchev and S. Wermter, "Spike-timing-dependent synaptic plasticity: from single spikes to spike trains," *Neurocomputing*, vol. 58, pp. 365–371, 2004.

[34] A. Tavanaei and A. Maida, "Bio-inspired multi-layer spiking neural network extracts discriminative features from speech signals," in *International Conference on Neural Information Processing*. Springer, 2017, pp. 899–908.

[35] J. J. Wade, L. J. McDaid, J. A. Santos, and H. M. Sayers, "SWAT: a spiking neural network training algorithm for classification problems," *Neural Networks, IEEE Transactions on*, vol. 21, no. 11, pp. 1817–1830, 2010.

[36] C. Näger, J. Storck, and G. Deco, "Speech recognition with spiking neurons and dynamic synapses: a model motivated by the human auditory pathway," *Neurocomputing*, vol. 44, pp. 937–942, 2002.

[37] S. Loiselle, J. Rouat, D. Pressnitzer, and S. Thorpe, "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. IEEE, 2005, pp. 2076–2080.

[38] S. Ghosh-Dastidar and H. Adeli, "Improved spiking neural networks for EEG classification and epilepsy and seizure detection," *Integrated Computer-Aided Engineering*, vol. 14, no. 3, pp. 187–212, 2007.

[39] N. Kasabov, V. Feigin, Z.-G. Hou, Y. Chen, L. Liang, R. Krishnamurthi, M. Othman, and P. Parmar, "Evolving spiking neural networks for personalised modelling, classification and prediction of spatio-temporal patterns with a case study on stroke," *Neurocomputing*, vol. 134, pp. 269–279, 2014.

[40] D. J. Felleman and D. C. Van Essen, "Distributed hierarchical processing in the primate cerebral cortex." *Cerebral Cortex (New York, NY: 1991)*, vol. 1, no. 1, pp. 1–47, 1991.

[41] T. Serre, "Hierarchical models of the visual system," in *Encyclopedia of computational neuroscience*. Springer, 2014, pp. 1–12.

[42] W. A. Freiwald and D. Y. Tsao, "Functional compartmentalization and viewpoint generalization within the macaque face-processing system," *Science*, vol. 330, no. 6005, pp. 845–851, 2010.

[43] J. V. Stone, *Principles of Neural Information Theory: Computational Neuroscience and Metabolic Efficiency*. Sebtel Press, 2018.

[44] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo,

Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[45] J.-s. Seo, B. Brezzo, Y. Liu, B. D. Parker, S. K. Esser, R. K. Montoye, B. Rajendran, J. A. Tierno, L. Chang, D. S. Modha *et al.*, "A 45nm CMOS neuromorphic chip with a scalable architecture for learning in networks of spiking neurons," in *Custom Integrated Circuits Conference (CICC), 2011 IEEE*. IEEE, 2011, pp. 1–4.

[46] S. Carrillo, J. Harkin, L. McDaid, S. Pande, S. Cawley, B. McGinley, and F. Morgan, "Advancing interconnect density for spiking neural network hardware implementations using traffic-aware adaptive network-on-chip routers," *Neural networks*, vol. 33, pp. 42–57, 2012.

[47] S. Carrillo, J. Harkin, L. J. McDaid, F. Morgan, S. Pande, S. Cawley, and B. McGinley, "Scalable hierarchical network-on-chip architecture for spiking neural network hardware implementations," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 12, pp. 2451–2461, 2013.

[48] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[49] W. Maass, "To spike or not to spike: that is the question," *Proceedings of the IEEE*, vol. 103, no. 12, pp. 2219–2224, 2015.

[50] ——, "Networks of spiking neurons: the third generation of neural network models," *Neural networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[51] G. Rozenberg, T. Bck, and J. N. Kok, *Handbook of natural computing*. Springer Publishing Company, Incorporated, 2011.

[52] H. S. Seung, "Learning in spiking neural networks by reinforcement of stochastic synaptic transmission," *Neuron*, vol. 40, no. 6, pp. 1063–1073, 2003.

[53] Q.-s. Liu, L. Pu, and M.-m. Poo, "Repeated cocaine exposure in vivo facilitates LTP induction in midbrain dopamine neurons," *Nature*, vol. 437, no. 7061, p. 1027, 2005.

[54] T. Song, L. Pan, and G. Păun, "Asynchronous spiking neural p systems with local synchronization," *Information Sciences*, vol. 219, pp. 197–207, 2013.

[55] L. Chavez-Noriega, J. Halliwell, and T. Bliss, "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*, vol. 79, no. 3, pp. 633–641, 1990.

[56] D. Huh and T. J. Sejnowski, "Gradient descent for spiking neural networks," *arXiv preprint arXiv:1706.04698*, pp. 1–10, 2017.

[57] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Frontiers in Neuroscience*, vol. 10, p. 508, 2016.

[58] W. Maass, "Lower bounds for the computational power of networks of spiking neurons." *Neural Computation*, vol. 8, no. 1, pp. 1–40, 1996.

[59] T. Masquelier and S. J. Thorpe, "Unsupervised learning of visual features through spike timing dependent plasticity," *PLoS Comput Biol*, vol. 3, no. 2, p. e31, 2007.

[60] A. Tavanaei, T. Masquelier, and A. S. Maida, "Acquisition of visual features through probabilistic spike timing dependent plasticity," in *Neural Networks (IJCNN), The 2016 International Joint Conference on*. IEEE, 2016, pp. 1–8.

[61] M. Beyeler, N. D. Dutt, and J. L. Krichmar, "Categorization and decision-making in a neurobiologically plausible spiking network using a STDP-like learning rule," *Neural Networks*, vol. 48, pp. 109–124, 2013.

[62] S. Ghosh-Dastidar and H. Adeli, "Spiking neural networks," *International Journal of Neural Systems*, vol. 19, no. 04, pp. 295–308, 2009.

[63] N. Kasabov, K. Dhoble, N. Nuntalid, and G. Indiveri, "Dynamic evolving spiking neural networks for on-line spatio- and spectro-temporal pattern recognition," *Neural Networks*, vol. 41, pp. 188–201, 2013.

[64] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge University Press, 2002.

[65] F. Rieke, *Spikes: exploring the neural code*. MIT press, 1999.

[66] S. M. Bohte, "The evidence for neural information processing with precise spike-times: A survey," *Natural Computing*, vol. 3, no. 2, pp. 195–206, 2004.

[67] J. J. Hopfield *et al.*, "Pattern recognition computation using action potential timing for stimulus representation," *Nature*, vol. 376, no. 6535, pp. 33–36, 1995.

[68] S. M. Bohte, H. La Poutré, and J. N. Kok, "Unsupervised clustering with spiking neurons by sparse temporal coding and multilayer RBF networks," *IEEE Transactions on neural networks*, vol. 13, no. 2, pp. 426–435, 2002.

[69] W. M. Kistler, W. Gerstner, and J. L. van Hemmen, "Reduction of the Hodgkin-Huxley equations to a single-variable threshold model," *Neural Computation*, vol. 9, no. 5, pp. 1015–1045, 1997.

[70] R. Jolivet, J. Timothy, and W. Gerstner, "The spike response model: a framework to predict neuronal spike trains," in *Artificial Neural Networks and Neural Information Processing—ICANN/ICONIP 2003*. Springer, 2003, pp. 846–853.

[71] E. M. Izhikevich *et al.*, "Simple model of spiking neurons," *IEEE Transactions on neural networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[72] A. Delorme, J. Gautrais, R. Van Rullen, and S. Thorpe, "Spikenet: A simulator for modeling large networks of integrate and fire neurons," *Neurocomputing*, vol. 26, pp. 989–996, 1999.

[73] E. R. Kandel, J. H. Schwartz, T. M. Jessell, S. A. Siegelbaum, A. J. Hudspeth *et al.*, *Principles of neural science*. McGraw-hill New York, 2000, vol. 4.

[74] N. Caporale and Y. Dan, "Spike timing-dependent plasticity: a Hebbian learning rule," *Annu. Rev. Neurosci.*, vol. 31, pp. 25–46, 2008.

[75] H. Markram, W. Gerstner, and P. J. Sjöström, "A history of spike-timing-dependent plasticity," *Spike-timing dependent plasticity*, p. 11, 2011.

[76] Y. Dan and M.-M. Poo, "Spike timing-dependent plasticity: from synapse to perception," *Physiological reviews*, vol. 86, no. 3, pp. 1033–1048, 2006.

[77] S. Song, K. D. Miller, and L. F. Abbott, "Competitive Hebbian learning through spike-timing-dependent synaptic plasticity," *Nature neuroscience*, vol. 3, no. 9, pp. 919–926, 2000.

[78] R. Guyonneau, R. VanRullen, and S. J. Thorpe, "Neurons tune to the earliest spikes through STDP," *Neural Computation*, vol. 17, no. 4, pp. 859–879, 2005.

[79] T. Masquelier, R. Guyonneau, and S. J. Thorpe, "Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains," *PloS one*, vol. 3, no. 1, p. e1377, 2008.

[80] T. Masquelier and S. R. Kheradpisheh, "Optimal localist and distributed coding of spatiotemporal spike patterns

through stdp and coincidence detection," *arXiv preprint arXiv:1803.00447*, p. 99, 2018.

[81] T. Masquelier, R. Guyonneau, and S. J. Thorpe, "Competitive STDP-based spike pattern learning," *Neural computation*, vol. 21, no. 5, pp. 1259–1276, 2009.

[82] T. Masquelier and S. J. Thorpe, "Learning to recognize objects using waves of spikes and spike timing-dependent plasticity," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*. IEEE, 2010, pp. 1–8.

[83] A. Tavanaei and A. S. Maida, "A spiking network that learns to extract spike signatures from speech signals," *Neurocomputing*, vol. 240, pp. 191–199, 2017.

[84] S. R. Kheradpisheh, M. Ganjtabesh, and T. Masquelier, "Bio-inspired unsupervised learning of visual features leads to robust invariant object recognition," *Neurocomputing*, vol. 205, pp. 382–392, 2016.

[85] R. P. Rao, B. A. Olshausen, and M. S. Lewicki, *Probabilistic models of the brain: Perception and neural function*. MIT press, 2002.

[86] K. Doya, *Bayesian brain: Probabilistic approaches to neural coding*. MIT press, 2007.

[87] M. C. Mozer, H. Pashler, and H. Homaei, "Optimal predictions in everyday cognition: The wisdom of individuals or crowds?" *Cognitive Science*, vol. 32, no. 7, pp. 1133–1147, 2008.

[88] K. P. Körding and D. M. Wolpert, "Bayesian integration in sensorimotor learning," *Nature*, vol. 427, no. 6971, pp. 244–247, 2004.

[89] B. Nessler, M. Pfeiffer, and W. Maass, "STDP enables spiking neurons to detect hidden causes of their inputs," in *Advances in neural information processing systems*, 2009, pp. 1357–1365.

[90] B. Nessler, M. Pfeiffer, L. Buesing, and W. Maass, "Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity," *PLoS Comput Biol*, vol. 9, no. 4, p. e1003037, 2013.

[91] S. Klampfl and W. Maass, "Emergence of dynamic memory traces in cortical microcircuit models through STDP," *The Journal of Neuroscience*, vol. 33, no. 28, pp. 11 515–11 529, 2013.

[92] D. Kappel, B. Nessler, and W. Maass, "STDP installs in winner-take-all circuits an online approximation to hidden Markov model learning," *PLoS Comput Biol*, vol. 10, no. 3, p. e1003511, 2014.

[93] A. Tavanaei and A. S. Maida, "Studying the interaction of a hidden Markov model with a Bayesian spiking neural network," in *Machine Learning for Signal Processing (MLSP), 2015 IEEE 25th International Workshop on*. IEEE, 2015, pp. 1–6.

[94] ——, "Training a hidden Markov model with a Bayesian spiking neural network," *Journal of Signal Processing Systems*, pp. 1–10, 2016.

[95] D. J. Rezende, D. Wierstra, and W. Gerstner, "Variational learning for recurrent spiking networks," in *Advances in Neural Information Processing Systems*, 2011, pp. 136–144.

[96] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[97] J. Brea, W. Senn, and J.-P. Pfister, "Sequence learning with hidden units in spiking neural networks," in *Advances in neural information processing systems*, 2011, pp. 1422–1430.

[98] D. Pecevski and W. Maass, "Learning probabilistic inference through STDP," *eneuro*, pp. ENEURO–0048, 2016.

[99] R. S. Zemel, R. Natarajan, P. Dayan, and Q. J. Huys, "Probabilistic computation in spiking populations," in *Advances in neural information processing systems*, 2004, pp. 1609–1616.

[100] L. Buesing, J. Bill, B. Nessler, and W. Maass, "Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons," *PLoS Comput Biol*, vol. 7, no. 11, p. e1002211, 2011.

[101] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[102] S. Grossberg, "Competitive learning: From interactive activation to adaptive resonance," *Cognitive Science*, vol. 11, no. 23-63, 1987.

[103] T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman, "Randome synaptic feedback weights support error backpropagation for deep learning," *Nature Communications*, pp. 1–10, 2016.

[104] F. Zenke and S. Ganguli, "Superspike: Supervised learning in multi-layer spiking neural networks," *Neural Computation*, vol. 30, no. 6, pp. 1514–1541, 2017.

[105] S. M. Bohte, J. N. Kok, and H. La Poutre, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, no. 1, pp. 17–37, 2002.

[106] O. Booij and H. tat Nguyen, "A gradient descent rule for spiking neurons emitting multiple spikes," *Information Processing Letters*, vol. 95, no. 6, pp. 552–558, 2005.

[107] S. Ghosh-Dastidar and H. Adeli, "A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection," *Neural Networks*, vol. 22, no. 10, pp. 1419–1431, 2009.

[108] T. Liu, Z. Liu, F. Lin, Y. Jin, G. Quan, and W. Wen, "Mt-spike: a multilayer time-based spiking neuromorphic architecture with temporal error backpropagation," in *Proceedings of the 36th International Conference on Computer-Aided Design*. IEEE Press, 2017, pp. 450–457.

[109] H. Mostafa, "Supervised learning based on temporal coding in spiking neural networks," *IEEE transactions on neural networks and learning systems*, pp. 1–9, 2017.

[110] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *arXiv preprint arXiv:1706.02609*, pp. 1–10, 2017.

[111] F. Ponulak and A. Kasinski, "Supervised learning in spiking neural networks with ReSuMe: sequence learning, classification, and spike shifting," *Neural Computation*, vol. 22, no. 2, pp. 467–510, 2010.

[112] A. Kasinski and F. Ponulak, "Comparison of supervised learning platforms for spike time coding in spiking neural networks," *Int. J. Appl. Math. Comput. Sci*, vol. 16, no. 1, pp. 101–113, 2006.

[113] R. V. Florian, "The chronotron: A neuron that learns to fire temporally precise spike patterns," *PLOS One*, vol. 7, no. 8, p. e40233, 2012.

[114] A. Mohemmed, S. Schliebs, S. Matsuda, and N. Kasabov, "Span: Spike pattern association neuron for learning spatio-temporal spike patterns," *International Journal of Neural Systems*, vol. 22, no. 04, p. 1250012, 2012.

[115] ——, "Training spiking neural networks to associate spatio-temporal input-output spike patterns," *Neurocomputing*, vol. 107, pp. 3–10, 2013.

[116] R. Gütig and H. Sompolinsky, "The tempotron: a neuron that learns spike timing-based decisions," *Nature neuroscience*, vol. 9, no. 3, pp. 420–428, 2006.

[117] J. D. Victor and K. P. Purpura, "Metric-space analysis of spike trains: theory, algorithms and application," *Network:*

computation in neural systems, vol. 8, no. 2, pp. 127–164, 1997.

[118] A. Tavanaei and A. S. Maida, "BP-STDP: Approximating backpropagation using spike timing dependent plasticity," *arXiv preprint arXiv:1711.04214*, pp. 1–20, 2017.

[119] J.-P. Pfister, T. Toyoizumi, D. Barber, and W. Gerstner, "Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning," *Neural computation*, vol. 18, no. 6, pp. 1318–1348, 2006.

[120] J. Wang, A. Belatreche, L. Maguire, and T. M. McGinnity, "An online supervised learning method for spiking neural networks with adaptive structure," *Neurocomputing*, vol. 144, pp. 526–536, 2014.

[121] A. Tavanaei and A. S. Maida, "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*, vol. 4, no. 7, pp. 1–8, 2015.

[122] M. Mozafari, S. R. Kheradpisheh, T. Masquelier, A. Nowzari-Dalini, and M. Ganjtabesh, "First-spike based visual categorization using reward-modulated stdp," *IEEE Transactions on Neural Networks and Learning Systems, In Press*, pp. 1–24, 2018.

[123] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT Press, 2016.

[124] S. R. Kheradpisheh, M. Ghodrati, M. Ganjtabesh, and T. Masquelier, "Deep networks can resemble human feed-forward vision in invariant object recognition," *Scientific reports*, vol. 6, p. 32672, 2016.

[125] ——, "Humans and deep networks largely agree on which kinds of variation make object recognition harder," *Frontiers in Computational Neuroscience*, vol. 10, p. 92, 2016.

[126] N. K. Kasabov, "Neucube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data," *Neural Networks*, vol. 52, pp. 62–76, 2014.

[127] S. G. Wysoski, L. Benuskova, and N. Kasabov, "Fast and adaptive network of spiking neurons for multi-view visual pattern recognition," *Neurocomputing*, vol. 71, no. 13, pp. 2563–2575, 2008.

[128] J. M. Brader, W. Senn, and S. Fusi, "Learning real-world stimuli in a neural network with spike-driven synaptic dynamics," *Neural computation*, vol. 19, no. 11, pp. 2881–2912, 2007.

[129] C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, and D. Rasmussen, "A large-scale model of the functioning brain," *Science*, vol. 338, no. 6111, pp. 1202–1205, 2012.

[130] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in Computational Neuroscience*, vol. 9, pp. 1–9, 2015.

[131] A. Morrison, A. Aertsen, and M. Diesmann, "Spike-timing-dependent plasticity in balanced random networks," *Neural Computation*, vol. 19, no. 6, pp. 1437–1467, 2007.

[132] J.-P. Pfister and W. Gerstner, "Triplets of spikes in a model of spike timing-dependent plasticity," *Journal of Neuroscience*, vol. 26, no. 38, pp. 9673–9682, 2006.

[133] Y. LeCun, C. Cortes, and C. J. Burges, "The MNIST database," *URL http://yann. lecun. com/exdb/mnist*, 1998.

[134] Y. Bengio, D.-H. Lee, J. Bornschein, T. Mesnard, and Z. Lin, "Towards biologically plausible deep learning," *arXiv preprint arXiv:1502.04156*, pp. 1–10, 2015.

[135] G. Hinton, "How to do backpropagation in a brain," in *Invited talk at the NIPS'2007 Deep Learning Workshop*, vol. 656, 2007.

[136] Y. Bengio, T. Mesnard, A. Fischer, S. Zhang, and Y. Wu, "STDP-compatible approximation of backpropagation in an energy-based model," *Neural Computation*, pp. 555–577, 2017.

[137] P. O'Connor and M. Welling, "Deep spiking networks," *arXiv preprint arXiv:1602.08323*, pp. 1–16, 2016.

[138] E. O. Neftci, C. Augustine, S. Paul, and G. Detorakis, "Event-driven random back-propagation: Enabling neuromorphic deep learning machines," *Frontiers in Neuroscience*, vol. 11, p. 324, 2017.

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

[140] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *Neural Networks (IJCNN), 2015 International Joint Conference on*. IEEE, 2015, pp. 1–8.

[141] S. K. Esser, R. Appuswamy, P. Merolla, J. V. Arthur, and D. S. Modha, "Backpropagation for energy-efficient neuromorphic computing," in *Advances in Neural Information Processing Systems*, 2015, pp. 1117–1125.

[142] B. Rueckauer, Y. Hu, I.-A. Lungu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in Neuroscience*, vol. 11, p. 682, 2017.

[143] E. Stromatias, M. Soto, T. Serrano-Gotarredona, and B. Linares-Barranco, "An event-driven classifier for spiking neural networks fed with synthetic or dynamic vision sensor data," *Frontiers in Neuroscience*, vol. 11, p. 350, 2017.

[144] D. Neil, M. Pfeiffer, and S.-C. Liu, "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*. ACM, 2016, pp. 293–298.

[145] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computation*, pp. 2352–2449, 2017.

[146] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1717–1724.

[147] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, pp. 1–14, 2014.

[148] T. N. Sainath, A.-r. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Acoustics, speech and signal processing (ICASSP), 2013 IEEE international conference on*. IEEE, 2013, pp. 8614–8618.

[149] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4277–4280.

[150] O. Abdel-Hamid, L. Deng, and D. Yu, "Exploring convolutional neural network structures and optimization techniques for speech recognition." in *Interspeech*, 2013, pp. 3366–3370.

[151] H. Zeng, M. D. Edwards, G. Liu, and D. K. Gifford, "Convolutional neural network architectures for predicting DNA–protein binding," *Bioinformatics*, vol. 32, no. 12, pp. i121–i127, 2016.

[152] D. Quang and X. Xie, "Danq: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences," *Nucleic acids research*, vol. 44, no. 11, pp. e107–e107, 2016.

[153] A. Tavanaei, A. S. Maida, A. Kaniymattam, and R. Loganantharaj, "Towards recognition of protein function based on its structure using deep convolutional networks," in *Bioinformatics and Biomedicine (BIBM), 2016 IEEE International Conference on*. IEEE, 2016, pp. 145–149.

[154] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[155] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[156] Y. LeCun *et al.*, "Lenet-5, convolutional neural networks," *URL: http://yann. lecun. com/exdb/lenet*, 2015.

[157] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich *et al.*, "Going deeper with convolutions." CVPR, 2015.

[158] S. Marĉelja, "Mathematical description of the responses of simple cortical cells," *JOSA*, vol. 70, no. 11, pp. 1297–1300, 1980.

[159] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of physiology*, vol. 148, no. 3, pp. 574–591, 1959.

[160] ——, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of physiology*, vol. 160, no. 1, pp. 106–154, 1962.

[161] P. Földiak, "Forming sparse representations by local anti-hebbian learning," *Biological cybernetics*, vol. 64, no. 2, pp. 165–170, 1990.

[162] B. A. Olshausen *et al.*, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.

[163] A. J. Bell and T. J. Sejnowski, "The "independent components" of natural scenes are edge filters," *Vision research*, vol. 37, no. 23, pp. 3327–3338, 1997.

[164] M. Rehn and F. T. Sommer, "A network that uses few active neurones to code visual input predicts the diverse shapes of cortical receptive fields," *Journal of computational neuroscience*, vol. 22, no. 2, pp. 135–146, 2007.

[165] J. Zylberberg, J. T. Murphy, and M. R. DeWeese, "A sparse coding model with synaptically local plasticity and spiking neurons can account for the diverse shapes of V1 simple cell receptive fields," *PLoS Comput Biol*, vol. 7, no. 10, p. e1002250, 2011.

[166] P. D. King, J. Zylberberg, and M. R. DeWeese, "Inhibitory interneurons decorrelate excitatory cells to drive sparse code formation in a spiking model of V1," *Journal of Neuroscience*, vol. 33, no. 13, pp. 5475–5485, 2013.

[167] C. Savin, P. Joshi, and J. Triesch, "Independent component analysis in spiking neurons," *PLoS Comput Biol*, vol. 6, no. 4, p. e1000757, 2010.

[168] K. S. Burbank, "Mirrored STDP implements autoencoder learning in a network of spiking neurons," *PLoS Comput Biol*, vol. 11, no. 12, p. e1004566, 2015.

[169] B. Zhao, R. Ding, S. Chen, B. Linares-Barranco, and H. Tang, "Feedforward categorization on AER motion events using cortex-like features in a spiking neural network," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 9, pp. 1963–1978, 2015.

[170] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "Stdp-based spiking deep convolutional neural networks for object recognition," *Neural Networks*, vol. 99, pp. 56–67, 2017.

[171] A. Tavanaei and A. S. Maida, "Bio-inspired spiking convolutional neural network using layer-wise sparse coding and STDP learning," *arXiv preprint arXiv:1611.03000*, pp. 1–16, 2016.

[172] ——, "Multi-layer unsupervised learning in a spiking convolutional neural network," in *Neural Networks (IJCNN), 2017 International Joint Conference on*. IEEE, 2017, pp. 81–88.

[173] P. Panda and K. Roy, "Unsupervised regenerative learning of hierarchical features in spiking deep networks for object recognition," in *International Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 299–306.

[174] A. Tavanaei, Z. Kirby, and A. S. Maida, "Training spiking ConvNets by STDP and gradient descent," in *Neural Networks (IJCNN), The 2018 International Joint Conference on*. IEEE, 2018, pp. 1–8.

[175] N. Anwani and B. Rajendran, "Normad-normalized approximate descent based supervised learning rule for spiking neurons," in *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2015, pp. 1–8.

[176] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," pp. 1–60, 2009.

[177] E. Hunsberger and C. Eliasmith, "Spiking deep networks with LIF neurons," *arXiv preprint arXiv:1510.08829*, pp. 1–9, 2015.

[178] ——, "Training spiking deep networks for neuromorphic hardware," *arXiv preprint arXiv:1611.05141*, 2016.

[179] D. Neil and S.-C. Liu, "Effective sensor fusion with event-based sensors and deep network architectures," in *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 2282–2285.

[180] G. Indiveri, F. Corradi, and N. Qiao, "Neuromorphic architectures for spiking deep neural networks," in *Electron Devices Meeting (IEDM), 2015 IEEE International*. IEEE, 2015, pp. 4–2.

[181] D. Garbin, O. Bichler, E. Vianello, Q. Rafhay, C. Gamrat, L. Perniola, G. Ghibaudo, and B. DeSalvo, "Variability-tolerant convolutional neural network for pattern recognition applications based on oxram synapses," in *Electron Devices Meeting (IEDM), 2014 IEEE International*. IEEE, 2014, pp. 28–4.

[182] S. K. Esser, P. A. Merolla, J. V. Arthur, A. S. Cassidy, R. Appuswamy, A. Andreopoulos, D. J. Berg, J. L. McKinstry, T. Melano, D. R. Barch *et al.*, "Convolutional networks for fast, energy-efficient neuromorphic computing," *Proceedings of the National Academy of Sciences*, p. 201604850, 2016.

[183] Y. Cao, Y. Chen, and D. Khosla, "Spiking deep convolutional neural networks for energy-efficient object recognition," *International Journal of Computer Vision*, vol. 113, no. 1, pp. 54–66, 2015.

[184] B. Rueckauer, I.-A. Lungu, Y. Hu, and M. Pfeiffer, "Theory and tools for the conversion of analog to spiking convolutional neural networks," *arXiv preprint arXiv:1612.04052*, pp. 1–9, 2016.

[185] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255.

[186] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[187] M. A. Salama, A. E. Hassanien, and A. A. Fahmy, "Deep belief network for clustering and classification of a continuous data," in *Signal Processing and Information Technology (ISSPIT), 2010 IEEE International Symposium on*. IEEE, 2010, pp. 473–477.

[188] N. Le Roux and Y. Bengio, "Representational power of restricted boltzmann machines and deep belief networks," *Neural computation*, vol. 20, no. 6, pp. 1631–1649, 2008.

[189] ——, "Deep belief networks are compact universal approximators," *Neural computation*, vol. 22, no. 8, pp. 2192–2207, 2010.

[190] H. Lee, C. Ekanadham, and A. Y. Ng, "Sparse deep belief net model for visual area V2," in *Advances in neural information processing systems*, 2008, pp. 873–880.

[191] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Unsupervised learning of hierarchical representations with convolutional deep belief networks," *Communications of the ACM*, vol. 54, no. 10, pp. 95–103, 2011.

[192] A. Krizhevsky and G. Hinton, "Convolutional deep belief networks on CIFAR-10," *Unpublished manuscript*, vol. 40, 2010.

[193] J. M. Susskind, G. E. Hinton, J. R. Movellan, and A. K. Anderson, "Generating facial expressions with deep belief nets," in *Affective Computing*. InTech, 2008.

[194] W. K. Mleczko, T. Kapuściński, and R. K. Nowicki, "Rough deep belief network-application to incomplete handwritten digits pattern classification," in *International Conference on Information and Software Technologies*. Springer, 2015, pp. 400–411.

[195] P. Liu, S. Han, Z. Meng, and Y. Tong, "Facial expression recognition via a boosted deep belief network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1805–1812.

[196] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in neural information processing systems*, 2009, pp. 1096–1104.

[197] S. Kang, X. Qian, and H. Meng, "Multi-distribution deep belief network for speech synthesis," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 8012–8016.

[198] A.-r. Mohamed, G. Dahl, and G. Hinton, "Deep belief networks for phone recognition," in *Nips workshop on deep learning for speech recognition and related applications*, vol. 1, no. 9, 2009, p. 39.

[199] P. Hamel and D. Eck, "Learning features from music audio with deep belief networks." in *ISMIR*, vol. 10. Utrecht, The Netherlands, 2010, pp. 339–344.

[200] A.-r. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.

[201] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, "Time series forecasting using a deep belief network with restricted boltzmann machines," *Neurocomputing*, vol. 137, pp. 47–56, 2014.

[202] T. Jo, J. Hou, J. Eickholt, and J. Cheng, "Improving protein fold recognition by deep learning networks," *Scientific Reports*, vol. 5, p. 17573, 2015.

[203] E. Neftci, S. Das, B. Pedroni, K. Kreutz-Delgado, and G. Cauwenberghs, "Event-driven contrastive divergence for spiking neuromorphic systems," *Frontiers in Neuroscience*, vol. 8, pp. 1–14, 2014.

[204] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer, "Real-time classification and sensor fusion with a spiking deep belief network," *Frontiers in Neuroscience*, vol. 7, pp. 1–13, 2013.

[205] E. Stromatias, D. Neil, M. Pfeiffer, F. Galluppi, S. B. Furber, and S.-C. Liu, "Robustness of spiking deep belief networks to noise and reduced bit precision of neuro-inspired hardware platforms," *Frontiers in Neuroscience*, vol. 9, pp. 1–14, 2015.

[206] E. Stromatias, D. Neil, F. Galluppi, M. Pfeiffer, S.-C. Liu, and S. Furber, "Scalable energy-efficient, low-latency implementations of trained spiking deep belief networks on spinnaker," in *Neural Networks (IJCNN), 2015 International Joint Conference on*. IEEE, 2015, pp. 1–8.

[207] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, and D. S. Modha, "A digital neurosynaptic core using embedded crossbar memory with 45pj per spike in 45nm," in *Custom Integrated Circuits Conference (CICC), 2011 IEEE*. IEEE, 2011, pp. 1–4.

[208] D. Neil and S.-C. Liu, "Minitaur, an event-driven FPGA-based spiking network accelerator," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 12, pp. 2621–2628, 2014.

[209] A. Barra, A. Bernacchia, E. Santucci, and P. Contucci, "On the equivalence of hopfield networks and boltzmann machines," *Neural Networks*, vol. 34, pp. 1–9, 2012.

[210] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[211] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.

[212] A. Barra, G. Genovese, P. Sollich, and D. Tantari, "Phase transitions in restricted boltzmann machines with generic priors," *Physical Review E*, vol. 96, no. 4, p. 042156, 2017.

[213] ——, "Phase diagram of restricted boltzmann machines and generalized hopfield networks with arbitrary priors," *Physical Review E*, vol. 97, no. 2, p. 022310, 2018.

[214] J. Tubiana and R. Monasson, "Emergence of compositional representations in restricted boltzmann machines," *Physical Review Letters*, vol. 118, no. 13, p. 138301, 2017.

[215] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[216] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[217] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[218] A. Shrestha, K. Ahmed, Y. Wang, D. P. Widemann, A. T. Moody, B. C. Van Essen, and Q. Qiu, "A spike-based long short-term memory on a neurosynaptic processor," in *Computer-Aided Design (ICCAD), 2017 IEEE/ACM International Conference on*. IEEE, 2017, pp. 631–637.

[219] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam *et al.*, "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*, vol. 34, no. 10, pp. 1537–1557, 2015.

[220] D. Neil, M. Pfeiffer, and S. Liu, "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*, 2016, pp. 3889–3897.

[221] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural Computation*, vol. 14, no. 11, pp. 2531–2560, 2002.
[222] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.
[223] M. Lukoševičius, H. Jaeger, and B. Schrauwen, "Reservoir computing trends," *KI-Künstliche Intelligenz*, vol. 26, no. 4, pp. 365–371, 2012.
[224] R. Pyle and R. Rosenbaum, "Spatiotemporal dynamics and reliable computations in recurrent spiking neural networks," *Physical Review Letters*, vol. 118, no. 1, p. 018103, 2017.
[225] L. Paulun, A. Wendt, and N. K. Kasabov, "A retinotopic spiking neural network system for accurate recognition of moving objects using NeuCube and dynamic vision sensors," *Frontiers in Computational Neuroscience*, vol. 12, p. 42, 2018.
[226] S. Schliebs, H. N. A. Hamed, and N. Kasabov, "Reservoir-based evolving spiking neural network for spatio-temporal pattern recognition," in *Neural Information Processing*. Springer, 2011, pp. 160–168.
[227] Z. G. Doborjeh, N. Kasabov, M. G. Doborjeh, and A. Sumich, "Modelling peri-perceptual brain processes in a deep learning spiking neural network architecture," *Scientific Reports*, vol. 8, no. 1, p. 8912, 2018.
[228] G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass, "Long short-term memory and learning-to-learn in networks of spiking neurons," *arXiv preprint arXiv:1803.09574*, 2018.
[229] A. Lamb, A. Goyal, Y. Zhang, S. Zhang, A. Courville, and Y. Bengio, "Professor forcing: A new algorithm for training recurrent networks," *arXiv preprint arXiv:1610.09038*, 2016.
[230] S. Hochreiter, A. S. Younger, and P. R. Conwell, "Learning to learn using gradient descent," in *Intl Conf on Artificial Neural Networks*. Springer, 2001, pp. 87–94.
[231] R. Costa, I. A. Assael, B. Shillingford, N. de Freitas, and T. Vogels, "Cortical microcircuits as gated-recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 272–283.
[232] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers in Neuroscience*, vol. 9, p. 437, 2015.