




Review

# Neuromorphic Spiking Neural Networks and Their Memristor-CMOS Hardware Implementations

Luis A. Camuñas-Mesa <sup>\*</sup>, Bernabé Linares-Barranco  and Teresa Serrano-Gotarredona 

Instituto de Microelectrónica de Sevilla (IMSE-CNM), CSIC and Universidad de Sevilla, 41092 Sevilla, Spain

\* Correspondence: [camunas@imse-cnm.csic.es](mailto:camunas@imse-cnm.csic.es)

Received: 5 July 2019; Accepted: 10 August 2019; Published: 27 August 2019



**Abstract:** Inspired by biology, neuromorphic systems have been trying to emulate the human brain for decades, taking advantage of its massive parallelism and sparse information coding. Recently, several large-scale hardware projects have demonstrated the outstanding capabilities of this paradigm for applications related to sensory information processing. These systems allow for the implementation of massive neural networks with millions of neurons and billions of synapses. However, the realization of learning strategies in these systems consumes an important proportion of resources in terms of area and power. The recent development of nanoscale memristors that can be integrated with Complementary Metal–Oxide–Semiconductor (CMOS) technology opens a very promising solution to emulate the behavior of biological synapses. Therefore, hybrid memristor-CMOS approaches have been proposed to implement large-scale neural networks with learning capabilities, offering a scalable and lower-cost alternative to existing CMOS systems.

**Keywords:** neuromorphic systems; spiking neural networks; memristors; spike-timing-dependent plasticity

## 1. Introduction

The outstanding evolution of computers during the last 50 years has been based on the architecture proposed by Von Neumann in the 1940s [1]. In this model of stored-programme computer, data storage and processing are two independent tasks performed in separated areas with a high need of data communication between them. With the development of integrated circuits, Gordon Moore predicted in the 1960s that the number of transistors in an integrated circuit would double every 18 to 24 months [2]. This exponential evolution allowed for the development of more efficient computing systems, with increasing processing speed and decreasing power consumption. However, even the current technologies for semiconductor manufacturing are reaching the limits of Moore's law [3], so different solutions have been proposed to keep the future evolution of processing systems [4]. Two different strategies suggest the development of new processing paradigms and novel devices beyond conventional Complementary Metal–Oxide–Semiconductor (CMOS) technologies.

In parallel with the development of computing platforms, in the 1960s some researchers used the emerging electronic technologies as a mechanism for modeling neural systems, from individual neurons [5–10] to more complex networks [11]. The increasing understanding of the structure and fundamental principles of behavior of the human brain revealed a very different processing paradigm from the traditional computer architecture with a much better performance. Even when comparing with current supercomputers which excel at speed and precision, the human brain is still much more powerful when dealing with novelty, complexity and ambiguity for practical tasks like visual recognition and motion control, while presenting a negligible power consumption around 20W [12]. This comparison between conventional computers and the brain led to the emergence of neuromorphic computing. The term neuromorphic engineering was first coined by Carver Mead

to refer to developing microelectronic information processing systems mimicking the operation of their biological counterparts [13,14]. During the 1980s, Carver Mead highlighted the analogy between the physics in biological neurons and the behavior of transistors in sub-threshold regime [13,14], developing neural networks based on analog circuits; leading to the implementation of the first silicon retinas [15] and proposing a new computing paradigm where data and processing tasks are performed by indivisible entities, taking inspiration from biological neural systems. Along the years, the neuromorphic engineering field has broadened its inspiration. Today's neuromorphic computing engineers not only try to mimic the highly parallel architecture of biological brains and the use of in-memory computing architectures as a way of improving the speed and energy performance, but also have deeply studied the signal information encoding, computational principles and learning paradigms that enable even simple biological brains with admiring performance in the interaction and adaptation to complex and unexpected environments with high reaction speeds and minimal power consumption despite relying on very simple and highly unreliable computation units [16].

Alternatively, many novel beyond-CMOS technologies have been proposed to overcome the limits of Moore's law. One of the most promising available devices is the nanoscale memristor. The memristor was first described theoretically by Chua in the 1970s as the fourth passive element establishing a relationship between electric charge and magnetic flux [17]. Much later in 2008, a team at HP Labs claimed to have found Chua's memristor experimentally based on a thin film of titanium oxide [18]. This 2-terminal device behaves as a variable resistor whose value can be modified by applying certain voltages or currents. The most common structure for this device is a union metal-dielectric(s)-metal, where the dielectric layer can be as thin as a few nanometers. The application of electric fields and controlled currents across the dielectric produces an alteration of its resistance by growing a filament or other mechanisms like barrier modulation. Currently available memristors are mostly binary devices, as they can switch between two resistance values: HRS (High-Resistance State) and LRS (Low-Resistance State) [19]. Since the appearance of the memristors, many logic families based on memristors for digital computation have been proposed [20,21], their potential as digital long-term non-volatile memory technology has also been demonstrated [22–25], and their use as biosensing devices looks also promising [26]. In the field of neuromorphic engineering, the memristors have attracted a special interest due to its particular plasticity behaviour which resembles the adaptation rules observed in biological synapses. Memristors can adapt and change its behaviour over time in response to different stimulation patterns as it happens in the human brain. In particular, it has been demonstrated that if stimulated with pulse-trains simulating the input from spiking neurons, memristors may exhibit a biologically inspired learning rule [27–30] resembling the spike-timing-dependent plasticity (STDP) observed in biological neurons [31–36]. Hence, memristors have been considered as artificial inorganic synapses.

In this paper, we analyze the current trend towards using memristors over CMOS platforms to implement neuromorphic systems, demonstrating a new paradigm which overcomes current limitations in conventional processing systems. In Section 2, we give a general overview of the basis of neuromorphic computing, while in Section 3 we review the main large-scale CMOS hardware implementations of neuromorphic systems. In Section 4, we describe proposed hybrid Memristor-CMOS approaches, while in Section 5 we emphasize the suitability of this strategy to implement learning algorithms in neural systems. Finally, in Section 6 we give our future perspective for this field.

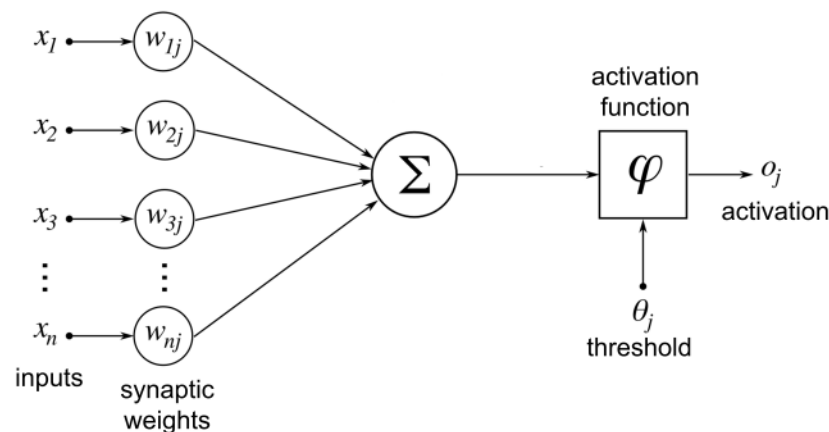
## 2. Neuromorphic Computing

As already stated, neuromorphic computing systems take inspiration on the architecture, the technology and the computational principles of biological brains. Morphologically, the human brain is composed of approximately  $10^{11}$  elementary processing units called neurons, massively interconnected by plastic adaptable interconnections called synapses. Each neuron connects approximately to  $10^3$ – $10^4$  other neurons through synaptic connections. The neurons are known

to be distributed in layers, and most of the synaptic interconnections are devoted to interconnect neurons belonging to successive layers.

The first computing systems inspired by this structure of biological brains were published in the 1940s–1950s and were called Artificial Neural Networks (ANNs) [37,38]. They appeared as powerful computational tools that proved to solve, by iteratively training algorithms that adapted the strength of the interconnection weights, complex pattern recognition, classification or function estimation problems not amenable to be solved by analytic tools. The first generations of neural networks did not involve any notion of time nor any temporal aspect in the computation.

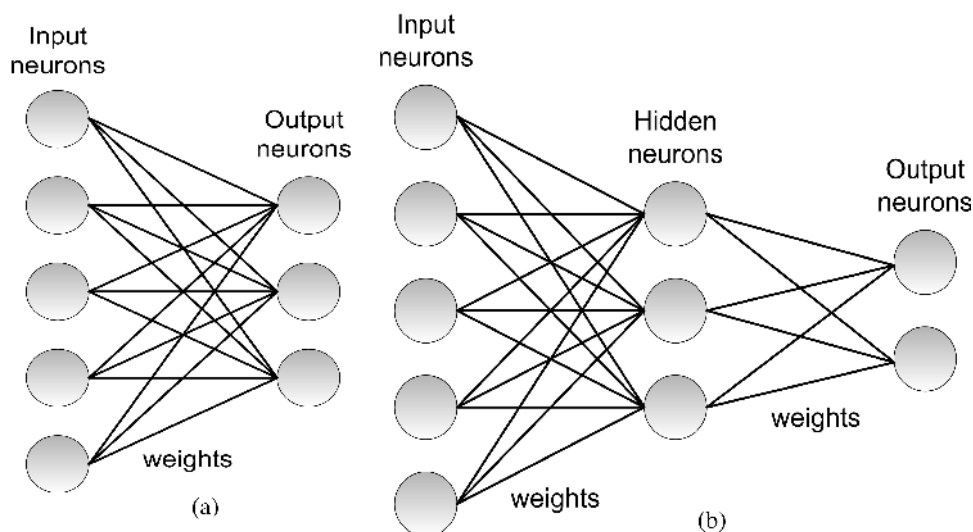
McCulloch and Pitts, proposed in 1943, one of the first computational models of the biological neurons. Figure 1 illustrates the operation of each proposed neural computational unit. As illustrated in Figure 1, a neuron  $N_j$  receives inputs from  $n$  other previous neurons  $x_1, x_2, \dots, x_n$ . The output of each neuron  $x_1, x_2, \dots, x_n$  in the previous layer is multiplied by the corresponding synaptic weight  $w_{1j}, w_{2j}, \dots, w_{nj}$ , also known as synaptic efficacy. The combined weighted input is transformed mathematically using a certain non-linear transfer function or an activation function  $\varphi$ , generating an output  $o_j$ . In the original McCulloch and Pitts' neural model the activation function was a thresholding gate, giving as neural output a digital signal [37]. This digital output neuron was the core of the first generation of neural networks.



**Figure 1.** Diagram of an artificial neuron with  $n$  inputs with their corresponding synaptic weights. All weighted inputs are added and an activation function controls the generation of the output signal.

In 1958, Rosenblatt proposed the perceptron. The architecture of the perceptron is shown in Figure 2a. In Figure 2, the computational units or neurons are represented by circles, interconnected through trainable weights representing the synaptic connections. The original perceptron consisted of a single layer of input neurons fully interconnected in a feedforward way to a layer of output neurons. A learning hebbian rule [39] to adapt the weights was proposed [38]. This single layer perceptron was able to solve only linearly separable problems [40].

In the 1950–60s, a second generation of computational units arose where the thresholding activation function was replaced by a continuous analog valued output like a smooth sigmoid, radial basis function or a continuous piece-wise linear function [41,42]. Recently, the rectifying non-linear activation function, also known as ReLU has become very popular for its better training convergence and its hardware friendly implementation [43]. Furthermore, gradient descent based learning algorithms could be now applied to optimize the network weights. Alternative learning rules were proposed as the delta rule based on the Least Mean Squares (LSM) algorithm published by Widrow [44,45]. This second generation proved to be universal approximators for any analog continuous function, that is, any analog continuous function could be approximated by a network of this type with a single hidden unit [41].



**Figure 2.** (a) Architecture of a single layer perceptron. The architecture consists of a layer on input neurons fully connected to a single layer of output neurons. (b) Extension to a multi-layer perceptron including more than one layer of trainable weights. In this example, the network includes 3 layers: input, hidden and output layer. Each connection between two neurons is given by a certain weight.

The backpropagation algorithm extended the application of the gradient descent techniques to networks with any number of hidden layers, popularly known as Deep Neural Networks (DNNs) [46–48]. Figure 2b illustrates a case with 3 layers: a first layer of input neurons, a second layer of hidden neurons, and a third layer of output neurons, although a general architecture can contain any given number of hidden layers.

The ANN architectures shown in Figure 2a,b are pure feedforward architectures as the signal propagates from input to output in an unidirectional way. Other architectures, known as recurrent neural networks, including feedback connections from upper layers in the architecture to lower layers, have been proposed. The Adaptive Resonance Theory (ART) architectures by Grossberg [49], the Kohonen self-organizing maps [50] or the Hopfield models [51] can be cited among the pioneering ones.

The presented ANNs have been typically developed in software, and trained offline. The training of DNNs requires a vast amount of annotated data to correctly generalize the problem without overfitting [52] and intensive computation resources. However, in recent years, the increase in the computation capabilities of modern computers and the availability of vast amounts of information have made DNN very popular allowing the development of many DNN-based applications [53,54] that use complex architectures like LeNet for handwritten digit recognition [55], Microsoft’s speech recognition system [56] or AlexNet for image recognition [43]. As a consequence we have witnessed the explosion of DNNs and machine learning.

Despite the impressive advances that DNNs have demonstrated in recent years, their performance in terms of efficiency (speed and power consumption) compared with the human brain is still low as it is low their resemblance to the human brain in terms of information coding. In the biological brain, the information is processed in a continuous way in time, not just as a sequence of static frames as DNNs recognition systems do. Furthermore, in conventional DNNs, the output of the different neural layers are computed in a sequential way. Each layer has to wait until the output of the previous layer has been computed to perform its computation, thus introducing a significant recognition delay in the network. On the contrary, biological neurons transmit their information to the next neuronal layers in the form of spikes. Whenever a neuron emits a spike, the spike is transmitted to its afferent connected neurons and processed with just the delay of the synaptic connection. In 1996, Thorpe demonstrated that the human brain was able to recognize a visual familiar object in the time that just one spike propagates through all the layers of the visual cortex [57]. Similar visual processing

speeds have been measured in the macaque monkeys by Rolls [58]. These experiments reveal an extremely efficient information coding in the biological brains. In this context, the 3rd generation of neural networks, spiking neural networks (SNNs), aims to bridge the gap between neuroscience and machine learning, using biologically-realistic models of neurons to carry out information coding and computation trying to fully exploit the efficiency in the spatio-temporal signal coding and processing and the corresponding power efficiency observed in the biological brains. SNNs operate using spikes in a similar way as biological neurons do. That way, in addition to the state of the neuron and the synaptic weight, SNNs also incorporate the concept of time into their model of operation. In these neurons, there is no propagation cycle, so each neuron fires an output spike only when its state reaches a certain threshold. Therefore, the information flows in these networks are spike trains which propagate between neurons asynchronously, and temporal correlation between spikes is crucial [41]. Spike trains offer the possibility of exploiting the richness of the temporal information contained in real-world sensory data. This allows SNNs to be applied to solve tasks which dynamically changing information like visual gesture recognition or speech recognition in a more natural way than current conventional (non spiking) artificial intelligent systems do. When dealing with dynamic information (as video sequences), conventional artificial systems perform computations using sequences of static images sampled at a constant periodic time (photogram time in the case of vision). Recognition of dynamic sequences may involve the use of recurrent neural network architectures or the resolution of continuous time differential equations. These computations are quite intensive using conventional framed ANN. However, the use of SNN where computation is driven in a continuous time way naturally and driven only by the occurrence of spikes detecting certain spatio-temporal correlations can be much more advantageous.

Many different coding methods for these spike trains have been proposed. Many authors have proposed to code the activity level of the neurons as the frequency of the firing rate. However, this type of coding does not benefit from the spike sparsity that should characterize SNN processing and thus, it does not enable the corresponding low power communication and computation due to the sparsity of the spike coding. Regarding the fast computation capability expected from SNN, this firing rate coding introduces a latency in the computation of the output firing rate. Furthermore, it is not biologically plausible as evidenced by the experiments of Thorpe [57] and Rolls [58] which demonstrated that the computation of a single cortical area is completed in 10–20 ms while the firing rate of the neurons involved in the computation is below 100 Hz, which does not make possible the computation based on the coding of analog variables in firing rates. However, as discussed by Thorpe et al. [59], there are many other biologically plausible and more efficient coding strategies. Other coding schemes that have been considered are in the timing between spikes [60], in the delay relative to a given synchronization time also known as time to first spike (TFS) [59] encoding, just coding the values in the order of spikes which is known as rank order coding [61], or synchronous detection coding [59].

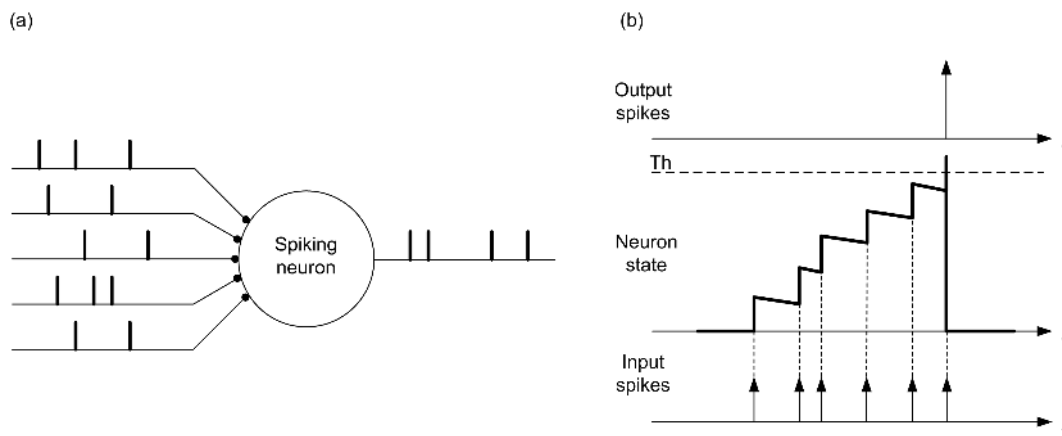
Regarding the SNN neuron models, there are many neuron models that describe the behaviour of biological neurons with different levels of complexity [5–10]. The classic Hodgkin-Huxley model [5] is a 4-th order biophysical model that describes the behaviour of the currents flowing into the neuron ion channels in a biologically realistic way. However, due to its complexity, different 2nd order simplified models have been proposed like the one proposed by FitzHugh and Nagumo [6,7] and the Morris-Lecar model [8], among others. In the last years, the Izhikevich model [10] and the Adaptive Exponential Integrate and Fire (AdEx) model [9] have become very popular for their ability to reproduce a large variety of spiking regimes observed in the biological neurons just by varying a reduced number of model parameters. However, while detailed biophysical models can reproduce electrophysiological activity of biological neurons with great accuracy, they are difficult to analyze computationally and not friendly for hardware implementations. Because of these reasons, for computational purposes simple first-order phenomenological models like the Integrate and Fire model are frequently used.

The behavior of a single integrate-and-fire spiking neuron is illustrated in Figure 3. A spiking neuron receives input spikes from several dendrites and sends out spikes from its output axon,

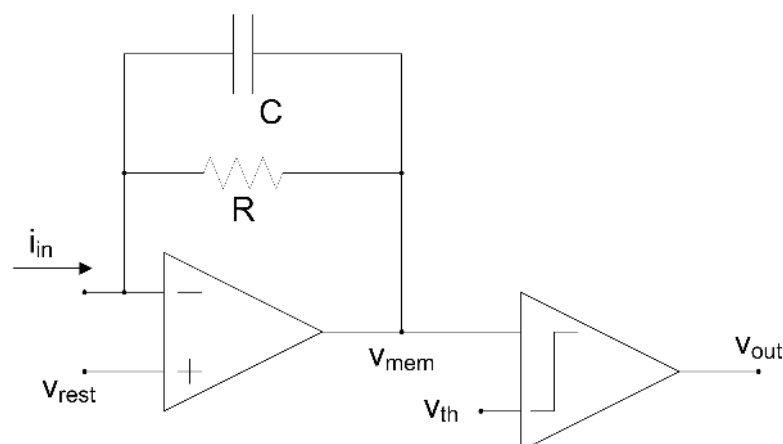
as shown in Figure 3a. Every time an input spike arrives, the state of the neuron is updated, and when it reaches the threshold, it generates an output spike and reset its state, as seen in Figure 3b. In this case, spikes are fully characterized by their firing time. In Figure 3, it can be observed that there is a constant slope decay of the membrane potential between two arriving spikes as it is the case of a leaky integrate and fire neuron. Mathematically, a leaky integrate-and-fire neuron can be described as:

$$i_{in}(t) = \frac{v_{mem}(t) - v_{rest}}{R} + C \frac{dv_{mem}(t)}{dt} \tag{1}$$

where  $v_{mem}(t)$  represents the membrane potential,  $i_{in}(t)$  the injected current,  $v_{rest}$  the resting value of the membrane potential,  $C$  the equivalent capacitance of the membrane, and  $R$  the leak resistance. A leaky integrate-and-fire neuron can be easily implemented in hardware following the resistance-capacitance (RC) "text book" concept scheme presented in Figure 4, where an input current  $i_{in}$  is integrated in capacitor  $C$  with leak resistance  $R$ . The integrated voltage  $v_{mem}$  is compared with a reference  $v_{th}$ , generating an output given by  $v_{out}$ . Additionally, integrate-and-fire neurons may consider a refractory period that forces a minimum time interval between two consecutive spikes of a neuron. A comprehensive overview of circuit realizations of spiking neurons with different levels of complexity can be found in [62].



**Figure 3.** Illustration of the behavior of a leaky integrate-and-fire spiking neuron. (a) A spiking neuron receives spikes from several inputs, processes them, and generates output spikes from its output node. (b) Temporal evolution of the neuron state while it receives input spikes. When the threshold is reached, it generates an output spike.



**Figure 4.** Example of a hardware implementation of an RC leaky integrate-and-fire neuron.

In terms of connectivity, the most general type of neural network is fully connected, meaning that each single neuron in layer  $i$  is connected to all neurons in layer  $i + 1$ . This scheme applies no limitation to the learning capabilities of the network; however, it presents some difficulties for practical implementations. A very popular way of reducing the amount of interconnections is represented by Convolutional Neural Networks (ConvNets), where each neuron in layer  $i$  is connected to a subset of neurons in layer  $i + 1$  representing a projective field. This receptive field can be represented as a convolutional kernel, with shared weights for each layer [63]. This scheme is inspired by biology, as it has been observed in the visual cortex [64]. In a similar way to the biological visual cortex, this convolutional neural network architecture is commonly used for image processing applications in the earlier more massive parallel feature extraction layers, as it implies an important reduction of the number of connections.

Table 1 (adapted from [65]) contains a comparison of the main distinctive features between ANNs and SNNs. As previously stated, the latency in each computation stage in an ANN is high as the whole computation in each stage has to be completed on the input image to generate the corresponding output. On the contrary, in an SNN processor the computation is performed spike by spike so that, output spikes in a computational layer are generated as soon as enough spikes evidencing the existence of a certain feature has been collected. In that way, the output of a computation stage is a flow of spikes that is almost simultaneous with its input spike flow. This property of SNN systems has been called “pseudo-simultaneity” [65,66]. The latency between the input and output spike flows of a processing SNN convolution layer has been measured to be as low as 155 ns [67]. Regarding the recognition speed, whereas in an ANN the recognition speed is strongly dependent on the computation capabilities of the hardware and the number of total operations to be computed (which is dependent on the system complexity), in an SNN, each input spike is processed in almost real time by the processing hardware and the recognition is performed as soon as there are enough input events that allow the system to take a decision. This recognition speed strongly depends on the input statistics and signal coding schemes as previously discussed. In terms of power consumption, the ANNs power depends on the consumption of the processor and the memory reading and writing operations but for a given input sampling frequency and size does not depend on the particular visual stimulus. However, in an SNN, the power consumption depends also strongly on the statistics of the stimulus and coding strategies. If efficient coding strategies are used, the system should benefit from the power efficiency of sparse spike representations.

On the negative side, as it has been already pointed out, the addition of the time variable makes SNN neuron models more complex than ANN ones. Also, as the computation of ANN is time-sampled, in each sampling time the algorithmic computation is performed using the available hardware resources that can be time multiplexed by fetching data and storing intermediate variables. However, in true SNN the spikes should be processed as they are generated in real time, requiring parallel hardware resources which cannot be multiplexed. The scaling up of the system can be done by modular expansion of the hardware resources.

However, where SNN should have major advantage is in applications requiring recurrent neural architectures, such as, in recognition of dynamic stimulus. The computation of recurrent connections in ANN requires computationally intensive iterations until convergence is reached, while the convergence of recurrent connections in SNN is almost instantaneous due to their pseudo-simultaneity property.

In terms of accuracy, as it will be discussed in Section 5, the learning methods that have been developed for ANN are not directly applicable to SNN. Although the learning theory of SNN still lacks behind its equivalent methods for ANN, some recent work reports for the same architecture an error increment of only 0.15% for the ImageNet dataset and 0.38% for the CIFAR10 dataset [68]. However, the temporal dependence introduces complexity so that once a SNN has been trained, its accuracy drops if the input temporal coding changes. But it also introduces the potential to recognize dynamic sequences in a more efficient way.

**Table 1.** Table comparing different features of ANNs and SNNs.

Feature	ANN	SNN
Data processing	Frame-based	Spike-based
Latency	High	Low
Time resolution	Low	Pseudo-simultaneity High
Time processing	Sampled	Continuous
Neuron model complexity	Low	High
Recognition accuracy	Higher	Lower
Hardware multiplexing	Possible	Not possible
System scale-up	Ad hoc	Adding modules
Recognition speed	Low	High
	Independent on input stimulus	Dependent on input statistics
	Dependent on hardware resources	
	Dependent on system complexity	Not dependent on system complexity
Power consumption	Determined by processor power and memory fetching	Determined by power-per-event processing in modules
	Independent on input stimulus	Dependent on stimulus statistics
Recurrent topologies	Need to iterate until converge	Instantaneous

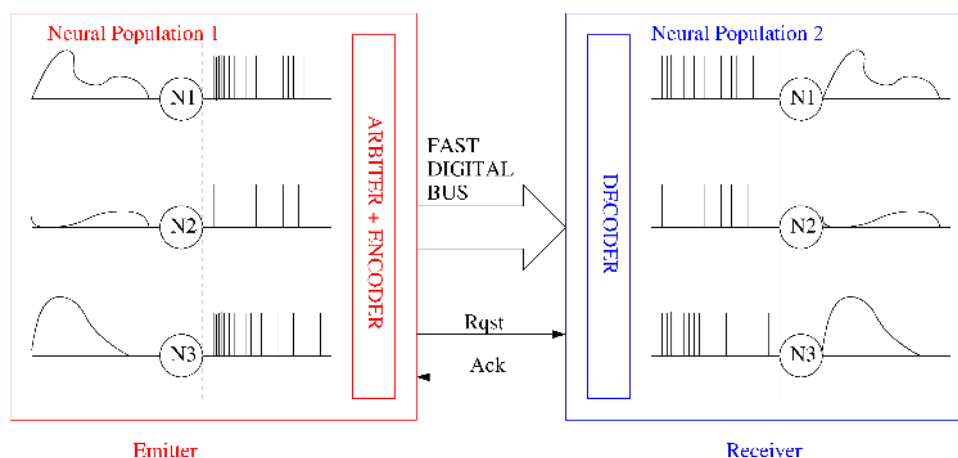
### 3. CMOS Neuromorphic Systems

Simulating SNNs on normal hardware is very computationally-intensive since it requires simulating coupled differential equations of large neuron populations running in parallel. Fully exploiting the coding and computation capabilities of biological brains requires the adequacy of the corresponding hardware platform to the peculiarities of the algorithm at different levels: from signal coding up to high level architectures. At the architectural level, the intrinsic parallelism of neural networks leads to the development of neuromorphic custom parallel hardware resembling the architecture of the biological brain to emulate its computing capabilities [62,69,70]. Furthermore, at the signal level, SNNs are better suited than ANNs for hardware implementation, as neurons are active only when they receive an input spike, reducing power consumption and simplifying computation.

One of the major issues when trying to implement in a parallel hardware large arrays of neural populations is the implementation of the synaptic interconnections. In a parallel 2D hardware, the physical wiring does allow to implement connections between just neighbouring neurons, while the biological neurons are distributed in 3D and massively interconnected among populations. Address-Event-Representation (AER) [71] is an asynchronous communication protocol that was conceived to massively interconnect neuron populations that can be located in the same or different chips as a ‘virtual wiring’ system. Figure 5 illustrates two neural populations communicated through an AER bus. In the particular case of this figure, neurons in the emitter population code their activity as a density of output pulses which is proportional to their activation level. However, the AER communication scheme can be applied to any type of pulse signal encoding [59]. Whenever a neuron in the emitter population generates a spike, it codes its physical coordinates  $(x, y)$  or address in a digital word in a fast digital bus and activates an asynchronous request (Rqst) signal. The coded address is sent through the fast digital bus to the receiver population. Upon reception of an active request, the receiver decodes the arriving neuron address and activates the acknowledge (Ack) signal. The received pulse can be sent to the corresponding neuron where the original activity of the sending neurons can be reproduced (as illustrated in Figure 5) or to a group of virtually connected neurons in the receiving population implementing a projection field [72]. The high-speed of the inter-population digital bus (in the order of nanoseconds) compared to the inter spike interval of biological neurons (in the order of milliseconds) allows to multiplex the connections of a million neurons in a shared time-multiplexed digital bus. Most of the developed large-scale CMOS neuromorphic computing



platforms make use of this AER communication protocol. As neuromorphic systems have scaled up in size and architectural complexity, many variations of the original point-to-point AER communication scheme [71,73,74] have been proposed trying to improve the overall system communication bandwidth. The broadcast-mesh-AER [75–77] proposes a generic approach to interconnect a mesh of AER devices using a global mapper and interconnecting the devices in a chain architecture. The pre-structured hierarchical AER approach [78] uses the knowledge of the network topology to interconnect AER devices through different AER links. Mappers can be used in every link, however, once the AER devices have been physically interconnected the changes in the configuration are limited. The Hierarchical-Fractal AER [79] proposes different levels of interconnection by adding address bits at higher level based on the idea that the traffic of spikes is going to be more intense at a local level. The router-mesh AER [80] proposes to avoid an external mapper by placing a router with a mapping table inside every AER module taking ideas from traditional NoC topologies [81]. The multicasting-mesh AER approach [82] proposes a simplification of the router-mesh AER by employing routing tables that contain only information of the connectivity between modules instead of allowing full neuron to neuron connectivity programming. Another approach developed to allow programmable interconnections inside the same chip or at wafer scale has been to implement massive programmable cross-point interconnects to configure the network topology [83] and including off-wafer rerouting for longer range interconnects [84]. Recently, the Hierarchical Routing AER has been proposed that establishes different hierarchical levels of nested AER links where each link has a dynamically reconfigurable synaptic routing table which allows programmable connectivity of the neurons without restriction on the spatial range of connectivity [85]. Moradi et al. have proposed a mixed-mode hierarchical-mesh routing scheme that exploits a clustered connectivity structure to reduce memory requirements and get a balance among memory overhead and reconfigurability [86].

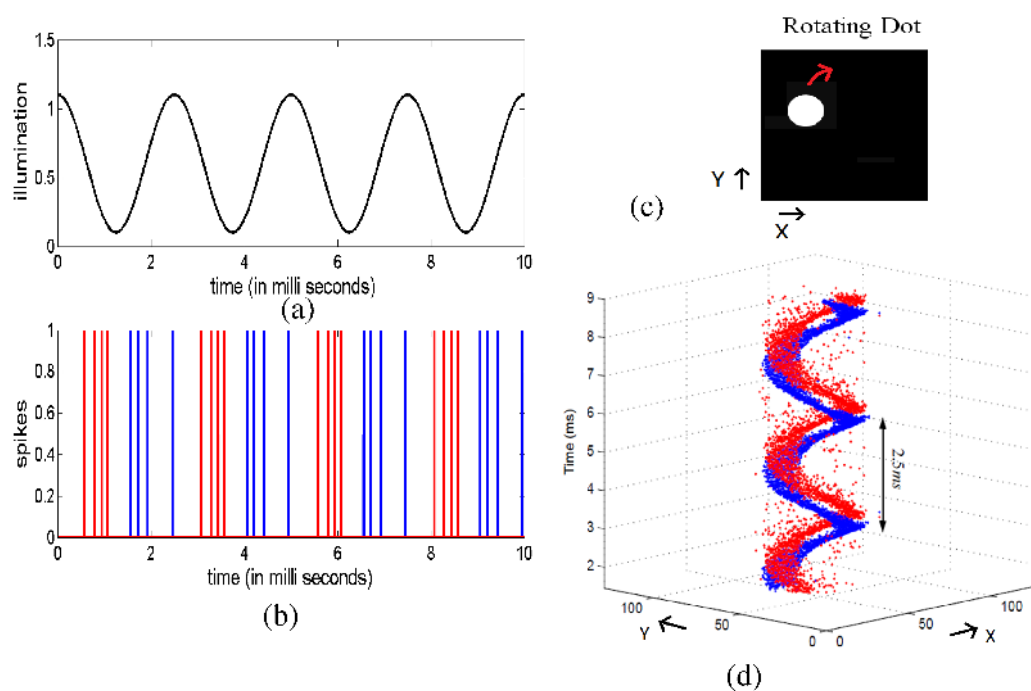


**Figure 5.** Illustration of two neural populations communicated through a point-to-point AER bus. Each neuron in the emitter population can be virtually connected to every neuron in the receiver population.

The above mentioned spike routing schemes have allowed the implementation of highly parallel massively interconnected spiking neural networks and the multichip integration of SNN hardware devoted to realize different specific parts of the cognitive function including integration of spike-based sensors and neural processors.

CMOS spike-based vision sensors have been developed since the very beginning of the neuromorphic engineering field [15]. Since then, a variety of AER visual sensors can be found in the literature that use different approaches to encode the luminance such as simple luminance to frequency transformation sensors [87], Time-to-First-Spike (TFS) coding sensors [88–91], foveated sensors [92,93], sensors encoding the spatial contrast [94,95], spatial and temporal filtering sensors that adapt to illumination and spatio-temporal contrast [96] and temporal transient detectors [97–104]. Among them, the temporal transient detectors also known as Dynamic Vision Sensors (DVSs) have recently become

very popular. They produce as output a stream of asynchronous events where each pixel codes the temporal variation of the illumination impinging on the pixel. Figure 6 illustrates the operation of a DVS sensor. One of the advantages of this sensor is that it codes the information in a compressive way sending only spikes when there is a relevant change in the illumination and thus removing the static background features of the scene from the moving object. Another advantage is that all the exact spatio-temporal information of the object is preserved with a reported precision in the spiking times of the order of 10  $\mu$ s. This converts these sensors in ideal candidates for high-speed processing and recognition systems. Several companies are nowadays making an effort to develop commercial prototypes of high-resolution DVS cameras: iniVation, Insightness, Samsung [105], CelePixel [106], and Prophesee, aiming to develop high-speed autonomous intelligent vision systems. Other types of spiking sensors have been developed such as cochleas [107–109] and tactile sensors [110,111] following similar principles of encoding the sensed signal relative changes as a flow of neural spikes, thus, generating a compressed information.



**Figure 6.** Illustration of the operation of a Dynamic Vision Sensor. (a,b) illustrate the operation a DVS pixel. (a) plots the illumination impinging on a pixel that varies as a sinusoidal waveform along time with period 2.5 ms, and (b) illustrates the output spikes generated by the corresponding DVS pixel. The blue traces correspond to positive output spikes which are generated when the illumination increases, while the red traces illustrate the negative signed spikes generated by an illumination decreasing over time. (d) illustrates real measurements of the response of a DVS when observing a white rotating dot on a black background rotating with a 2.5 ms period, as shown in (c).

Regarding the neuromorphic hardware for processing, it should be distinguished between the hardware implementing specific functionalities of the cognitive function and general purpose SNN hardware platforms intended for emulating massive neural arrays. Among the specific functional neuromorphic circuits, researchers have developed SNN neuromorphic chips implementing computational primitives and operations performed in the brain such as:

- Winner-Take-All (WTA) is a brain inspired mechanism implemented by inhibitory interactions between neurons in a population that compete to inhibit each other. The result is that the neuron in the population receiving the highest input remains active while silencing the output of the rest of the neurons. Hardware modules of spiking Winner-take-all networks have been reported [112].

- Spiking Convolutional Networks (ConvNets): neural networks implementing in real time the behaviour of the feature extraction layers of the cortex region have been implemented in hardware [113–115].
- Hardware implementations of spiking neural networks for saliency maps detection have been proposed as emulators of brain attention mechanisms [116].
- Spiking Liquid State Machines have also been implemented for recognition of sequential patterns such as speech recognition tasks [117,118].

The specific SNN neuromorphic chips can be combined in a modular and scalable way [78] to achieve optimum performance in terms of complexity, speed, and power consumption depending on the specific application. However, the current evolution of hardware neuromorphic platforms tends to large-scale modular computing systems with increasing numbers of neurons and synapses [62,119] that are meant to be easily reconfigurable for different applications. Some of the most remarkable large-scale neuromorphic systems developed until the present are:

- The IBM TrueNorth chip is based upon distributed digital neural models aimed at real-time cognitive applications [120].
- The Stanford NeuroGrid uses real-time sub-threshold analogue neural circuits [121]. It has been recently reversioned with the Braindrop chip prototype [122] which is a single core planned to be part of the 1-million-neuron Brain Storm System [123].
- The Heidelberg BrainScaleS system uses wafer-scale above threshold analogue neural circuits running 10,000 times faster than biological real time aimed at understanding biological systems, and in particular, long-term learning [124].
- The Manchester SpiNNaker is a real-time digital many-core system that implements neural and synapse models in software running on small embedded processors, again primarily aimed at modelling biological nervous systems [125].
- The Intel Loihi chip consists of a mesh of 128 neuromorphic cores with an integrated learning engine on-chip [126].
- The Darwin Neural Processing Unit is a hardware co-processor with digital logic specifically designed for resource-constrained embedded applications [127].
- The ROLLS chip was developed at ETHZ-INI including 256 neurons and 128 k on-line learning synapses [128]. Recently, it has been updated to the Dynamic Neuromorphic Asynchronous Processor (DYNAPs) with 1 K neurons and 64 k on-line learning synapses [86].
- A digital realization of a neuromorphic chip (ODIN) containing 256 neurons and 64 K 4-bit synapses exhibiting a spike-driven synaptic plasticity in FDSOI 28 nm technology has recently been developed in the University of Leuven [129].

A comparison of the main features of these generic neuromorphic systems and the human brain is shown in Table 2. In general, these systems are based on a processing chip which is part of a multi-chip board (or wafer for BrainScaleS), and in some cases these boards can be assembled in multi-board racks, scaling up more and more the size of the implemented network. Some of the most recent approaches have not reported yet such multi-chip platforms.

**Table 2.** Comparison of the major features of the human brain and the large-scale neuromorphic systems described in this work.

Platform	Human Brain	Neurogrid	BrainScaleS	Truenorth	SpiNNaker	Loihi	Darwin	ROLLS	DYNAPs	ODIN
Technology	Biology	Analog, sub-threshold	Analog, over threshold	Digital, fixed	Digital, programmable	Digital, programmable	Digital, programmable	Mixed-signal, sub-threshold	Mixed-signal, subthreshold	Digital, programamable
Feature size	10 $\mu\text{m}$	180 nm	180 nm	28 nm	130 nm	14 nm	180 nm	180 nm	180 nm	28 nm
# transistors		23 M	15 M	5.4 B	100 M	2.07 B	$\approx$ M	12.2 M	-	-
Chip size		1.7 $\text{cm}^2$	0.5 $\text{cm}^2$	4.3 $\text{cm}^2$	1 $\text{cm}^2$	60 $\text{mm}^2$	25 $\text{mm}^2$	51.4 $\text{mm}^2$	43.79 $\text{mm}^2$	0.086 $\text{mm}^2$
# neurons (chip)		65 k	512	1 M	16 k	131 k	$\approx$ M	256	1 k	256
# synapses (chip)		100 M	100 k	256 M	16 M	126 M	Programmable	128 k	64 k	64 k
# chips per board		16	352	16	48	-	-	-	-	-
# neurons (board)	$10^{11}$	1 M	200 k	16 M	768 k	-	-	-	-	-
# synapses (board)	$10^{15}$	4 B	40 M	4 B	768 M	-	-	-	-	-
Energy per connection	10 fJ	100 pJ	100 pJ	25 pJ	10 nJ	81 pJ	10 nJ	>77 fJ	30 pJ	12.7 pJ
On-chip learning	Yes	No	Yes	No	Yes	Yes	Yes	Yes	No	Yes

#### 4. Hybrid Memristor-CMOS Systems

As was mentioned in Section 1, progress in silicon technologies is reaching physical limitations which are causing the end of Moore's law, and traditional Von Neumann computing architectures are reaching scalability limits in terms of computing speed and power consumption. Novel brain inspired architectures have emerged as alternative computing platforms specially suitable for cognitive tasks that require the processing of massively parallel data. As already stated in Section 3, one of the main bottlenecks of the CMOS implementation of these neuromorphic parallel architectures is the physical implementation of the massive synaptic interconnections among neurons and the synaptic adaptability. The implementation of adaptable synaptic connections in CMOS technology requires the use of large amount of circuitry for analog memory or digital memory blocks that are costly in terms of area and energy requirements. Furthermore, learning rules to update these synaptic memory devices have to be implemented. The interest in developing a compact adaptable device obeying biological learning rules to implement the synaptic connections has motivated the investigation on alternative nanotechnologies to complement the CMOS technology in this regard. Memristive devices are novel two terminal devices able to change their conductance as a function of the voltage/current applied to their terminals that were predicted in 1971 by Chua based on circuit theory reasoning [17] and whose existence was experimentally demonstrated in nanomaterials devices much later in 2008 [18]. Different materials with different conductance switching mechanisms have been proposed [130] such as Phase-Change-Memory (PCM) [131], Conductive Bridge Memory (CBRAM) [132], Ferroelectric Memories (FeRAM) [133], Redox-based resistive switching Memories (ReRAM) [134], or organic memristive devices (OMD) [135–139]. Each of them presents different characteristics in terms of compactness, reliability, endurance, memory retention term, programmable states, and energy efficiency [69,140].

These devices present some properties specially valuable as electronic synaptic elements [141]:

- Memristors can be scaled down to feature sizes below 10 nm.
- They can retain memory states for years.
- They can switch with nanosecond timescales.
- They undergo spike-based learning in real time under biologically inspired learning rules as Spike-Time-Dependent Plasticity (STDP) [31,32,34–36].

The characteristic  $i/v$  equations of a memristive element can be approximated by:

$$\begin{aligned} i_{MR} &= G(w, v_{MR})v_{MR} \\ dw/dt &= f_{MR}(w, v_{MR}) \end{aligned} \quad (2)$$

where  $i_{MR}$ ,  $v_{MR}$  are the current and the voltage drop at the terminal devices, respectively (as shown in Figure 7a,  $G(w, v_{MR})$  is the conductance of the device that changes as function of the applied voltage (supposing a voltage or flux controlled device model [142]), and  $w$  is some physical parametric characteristic whose change is typically governed by a nonlinear function  $f_{MR}$  of the applied voltage including a threshold barrier. A typical  $f_{MR}$  observed in memristive devices [142] can be mathematically approximated by [28–30,143]

$$f_{MR} = \begin{cases} I_0 * \text{sign}(v_{MR})(e^{|v_{MR}|/v_0} - e^{v_{TH}/v_0}) & \text{if } |v_{MR}| > v_{Th} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Figure 7b depicts the typical non-linear memristive adaptation curve  $f_{MR}$ .

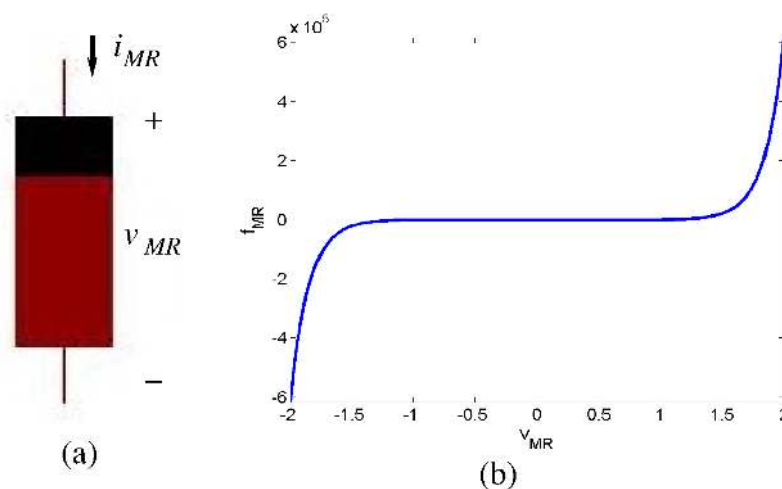
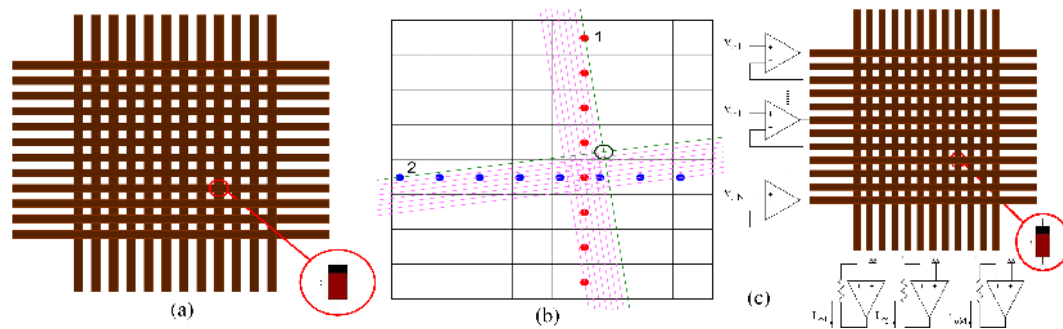


Figure 7. (a) Memristor symbol and (b) typical thresholded memristive adaptation curve

According to Equations (2) and (3), when a voltage higher than  $v_{TH}$  is applied between the terminals of a voltage-controlled memristor, its resistance changes. This property has been used to adapt supervisory the weights of simple perceptron networks [38] by applying voltage pulses controlled by some error function to memristive devices. The performance of correct categorization has been experimentally demonstrated [144–146]. Although these novel memristive devices open very promising alternatives for electronic technologies, they are still far from the maturity reached by CMOS systems during the last decades. Instead, they are very promising technologies for being integrated in 3D with CMOS technology providing a high-density memory closely tight to computational units, thus overcoming the limitations of Von Neumann's architecture. Very dense architectures for 3D-integration of CMOS computing units with crossbar arrays of nanodevices like the semiconductor/nanowire/molecular integrated circuits (CMOL) [147] architecture have been proposed. A CMOL system combines the advantages of CMOS technology (flexibility and high fabrication yield) with the high density of crossbar arrays of nanoscale devices. This structure consists of a dense nanowire crossbar fabric on top of the CMOS substrate with memristor devices assembled in the crossings between nanowires as shown in Figure 8. Figure 8a shows a crossbar nanoarray where nanowires run in orthogonal directions. A memristive device is located at each cross point of a vertical and horizontal nanowire. Figure 8b shows the proposed CMOL structure. The nanowire crossbar is tilted with respect to the orientation of the 2D array of CMOS neurons. Each CMOS neuron has an output pin (red dots in Figure 8b) and an input pin (blue dots in Figure 8b). Each neuron output is connected to just one nanowire and each neuron input is connected to another nanowire in the perpendicular direction. The crosspoint memristive devices implement the synaptic connections between neurons. In the illustration of Figure 8b, the output of neuron 2 is connected to the input of neuron 1 through the synaptic memristive device located at the intersection point (marked as a black circle) of the two perpendicular nanowires (plotted as green lines) connected to neuron 2 output and neuron 1 input, respectively. Other alternative architectures for neuromorphic structures based on 3D integration of CMOS neurons and memristive synapses have been proposed as CrossNets [148]. A functional digital FPGA-like implementation of a small CMOL prototype where the memristors were used as digital switches to re-configure the digital hardware implemented in the CMOS cells has been demonstrated [149].



**Figure 8.** Illustration of the proposed hybrid CMOS/memristive CMOL architecture. (a) Memristive devices fabricated in the cross-points of a crossbar array and (b) proposed CMOL architecture. (c) Neuromorphic architecture composed of CMOS neurons connected to a crossbar array of memristors.

Neuromorphic architectures composed of CMOS neurons connected to a crossbar array of memristors as shown in Figure 8c have also been proposed as accelerators to perform the intensive matrix multiplications needed in deep machine learning architectures. In the memristive crossbar shown in Figure 8c, the input vector  $[V_{in1}, V_{in2}, \dots, V_{inN}]$  is applied as input voltages to the rows, each memristor in an  $(i,j)$  crossbar position is programmed with an analog value  $w_{ij}$  so that the currents flowing through the vertical columns are the result of the vector-matrix multiplication

$$I_j = \sum w_{ij} V_{ini}. \quad (4)$$

Many works have proposed including ReRAM memristive memory crossbars to implement Matrix-Vector-Multiplication Units in computer architectures to accelerate Neural Network applications [150–155] demonstrating great benefits in power consumption levels. PRIME [151] and RESPARC [150] report simulations of energy savings compared to fully CMOS Neural Processors Units in the order of  $10^3$  depending on the particular neural network architecture. Energy savings in the order of  $10^3$ – $10^5$  respect to baseline CPU implementations have been reported [153,155]. However, in these works the memristor crossbars are included at a simulation level. A real hardware implementation of a hybrid CMOS system including an array of ReRAM crossbar as vector matrix multiplication elements for neural network computing acceleration at low energy consumption has been reported [22]. However, in this work the memristors are used in digital flip-flops as non-volatile digital devices. The real integration of CMOS neurons with a crossbar of CBRAM memristors is also demonstrated [156] for functional programming of a crossbar array of memristors in a digital way. More advanced fabrication techniques have been proposed to integrate up to 5 layers of 100 nm memristors in 3D crossbar arrays [157]. Some works have demonstrated the feasibility of integrating both carbon nanotube field-effect transistors (CNFETs) and RRAM on vertically stacked layers in a single chip on top of silicon logic circuitry, reporting 1952 CNFETs integrated with 224 RRAM cells for brain-inspired computing [158], or a prototype with more than 1 million RRAM cells with more than 2 million CNFETs in a single chip [25]. A recent work reported some circuit-level techniques for the design of a 65 nm 1 Mb pseudo-binary nonvolatile computing-in-memory RRAM macro which is capable of storing 512 k weights for Deep Neural Networks (DNN) [159].

However, so far experimental demonstrations of classification and training of memristive based analogue-memory learning systems have been on reduced systems and without achieving monolithic integration of the CMOS and memristive part [160], and suffered from classification inaccuracies due to device imperfections as control of the weight update, the programming of multilevel values, or variation in the device conductance range, limiting their application and severely degrading the performance of the network [161,162]. Another important shortcoming that limits the density of the implemented crossbars, as well as the practical hardware implementation of CMOL neuromorphic memristive systems, is the necessity of implementing a MOSFET in series with each memristive device (the so-called 1T1R devices) to limit the currents flowing through each memristor avoiding

damage due to transient high-currents. When the transistor device is omitted, the current limitation is done in the peripheral CMOS circuitry, limiting the size of the array to reduce the risk of local high parasitic transient currents. In the 1T1R structures, the transistor also acts as a selection device to update individually each memristor avoiding alteration of the nearby devices. As a summary, although memristors are a very promising technology to implement high-density analog memories close to the computing system that could potentially implement high-speed low power learning cognitive system, there are still some technological limitations that are currently being investigated that have not allowed to implement such large scale systems.

## 5. Learning with Memristors (STDP)

Given that these SNNs are more powerful, in theory, than 2nd generation networks, it is natural to wonder why we do not see widespread use of them. One main issue that currently lies in practical use of SNNs is that of training. Learning mechanisms are crucial for the ability of neuromorphic systems to adapt to specific applications. In general, the goal of a learning algorithm is to modify the weights of the synaptic connections between neurons in order to improve the response of the network to a certain stimulus. Two main categories can be considered: supervised or unsupervised learning. In supervised learning, the dataset samples are labeled with the identification of the expected 'correct' network output. The measured deviation between the desired output and the real one is used to modify the synaptic weights. In unsupervised learning, there is no labeled information, so the own characteristics of the input data are analyzed by the network in order to self-organize.

As explained in Section 2, in the ANN field, the powerful computational capabilities of modern GPUs and CPUs and the availability of large amount of annotated data have made possible to train complex deep learning architectures using the supervised backpropagation learning algorithm [48] to solve complex cognitive problems in some cases with better accuracy than humans. However, there are no known effective supervised training methods for SNNs that offer higher performance than 2nd generation networks. The popular backpropagation learning strategies are not directly usable in SNN networks. On the one hand, if spikes are represented computationally as the occurrence of an output event at a particular time (as represented in Figure 3) they are not differentiable; on the other hand, differentiating the error back across the spatial layers (as it is done in the backpropagation algorithm) loses the precise temporal information contained in the spike timings. Therefore, in order to properly use SNNs for real-world tasks, we would need to develop an effective supervised learning method that takes space and time simultaneously into account [163]. Several approaches for SNN training have been adopted:

**Training an ANN and conversion to SNN [66,164–167].** Some authors have proposed ANN to SNN direct conversion methods which are based on the training of ANN using static input images and directly mapping the network to an SNN converting the input stimulus to spikes using frequency rate encoding [164,165,167]. Bodo et al. implemented several optimizations achieving for a rate coded input similar performance than equivalent ANN implementations [165]. However, such encoding reduces the power efficiency of SNN. Other authors have proposed to train SNN with sensory data coming directly from a spike-based sensor (as a DVS recording). For that purpose, an equivalent ANN using static images generated from histograms of the input recordings of spiking stimulus is trained. Afterwards, a method to convert the weights of the ANN to the corresponding SNN is devised [66]. The additional timing parameters as leakage time or refractory period characteristics of SNN are optimized as hyper-parameters in the SNN resulting on different optimized parameter values for different input dynamics. Bodo et al. recently proposed an ANN to SNN conversion method based on time-to-first-spike input conversion code [166]. In all of these methods, training is done on static images and thus they do not fully exploit directly all the spatio-temporal information contained in the events.

**Supervised training in the spiking domain.** For the above mentioned reason, some methods for direct supervised learning in the spiking domain have been proposed [168–179]. Some of the



earlier SNN training methods were based on an adaptation of the Delta Learning Rule [44] and were appropriate to train single layer architectures [169,171,172]. More recent SNN learning methods have been reported that try to apply the backpropagation learning rules to SNN with several learning layers. They include coding the spike times to have a differentiable relationship with a subset of previous spikes and hence compatible with the gradient descent back-propagation rule in the temporal domain [180], or approximating the spike shape response activity to be differentiable across neural layers [174,175,177]. Wu et al. introduced an SNN Spatio-Temporal BackPropagation algorithm [177]. Not only do they approximate the spike shape as a continuous differentiable function, but also they use a back-propagation-through-time (BTT) [163] which backpropagates the error in the space as well as the time dimension reporting the best recognition accuracy achieved by previously reported SNN on the MNIST and N-MNIST datasets and equivalent to the state-of-the-art of ANNs. Similarly, the SLAYER method [178] considers back-propagation in space and time and trains both weights and delays of the synaptic connections.

**Unsupervised training in the spiking domain.** The unsupervised SNN training methods are mostly based on the well known Spike-Timing-Dependent Plasticity (STDP) learning rule [31,32]. STDP is a Hebbian learning rule. The traditional Hebbian synaptic plasticity rule was formulated in 1940 suggesting that synapses increase their efficiency if they persistently take part in firing the post-synaptic neuron [39]. Much later in 1993, STDP learning algorithms were reported [31,32] as a refinement of this rule taking into account the precise relative timing of individual pre- and post-synaptic spikes, and not their average rates over time. In comparison with traditional Hebbian correlation-based plasticity, STDP proved to be better suited for explaining brain cortical phenomena [181,182], and demonstrated to be successful in learning hidden spiking patterns [183] or performing competitive spike pattern learning [184]. Interestingly, shortly after that, in 1997, STDP learning was experimentally observed in biological neurons [33–35]. Figure 9a,b illustrate the STDP learning rule as observed in biological synapses. Figure 9a plots a presynaptic neuron with a membrane potential  $V_{pre}$  which is connected through a synapse with synaptic strength  $w$  to a postsynaptic neuron with membrane potential  $V_{post}$ . The presynaptic neuron emits a spike at time  $t_{pre}$  which contributes to the generation of a postsynaptic spike at time  $t_{post}$ . The biological learning rule observed by Bi and Poo is illustrated in Figure 9b. When the two connected neurons generate spikes close in time, if  $\Delta T = t_{post} - t_{pre}$  is positive, meaning that the presynaptic pulse contributed causally to generate the postsynaptic pulse, there is a positive variation in the efficacy of the synaptic connection  $\zeta(\Delta T) > 0$ ; on the contrary, if  $\Delta T = t_{post} - t_{pre}$  is negative, the variation in the efficacy of the synaptic connection  $\zeta(\Delta T) < 0$  is negative. Being STDP a local learning rule, and memristors two-terminal devices exhibiting plasticity controlled by the local applied voltage/current to their terminals converts memristors as ideal candidates to implement high-density on-line STDP-based neuromorphic learning systems [27]. Linares et al. [28] showed that by combining the memristance model formulated in Equation (2) with the electrical wave signals of neural impulses (spikes) as shaped in Figure 9c applied to the pre- and post-synaptic terminals of the memristive synaptic-like device, the STDP behavior shown in Figure 9d emerges naturally. Considering the mathematical equation describing the spike shape shown in Figure 9c versus time

$$spk(t) = \begin{cases} A_{mp}^+ \frac{e^{t/\tau^+} - e^{t_{tail}^+/\tau^+}}{1 - e^{t_{tail}^+/\tau^+}} & \text{if } -t_{tail}^+ < t < 0 \\ A_{mp}^- \frac{e^{-t/\tau^-} - e^{-t_{tail}^-/\tau^-}}{1 - e^{-t_{tail}^-/\tau^-}} & \text{if } 0 < t < t_{tail}^- \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

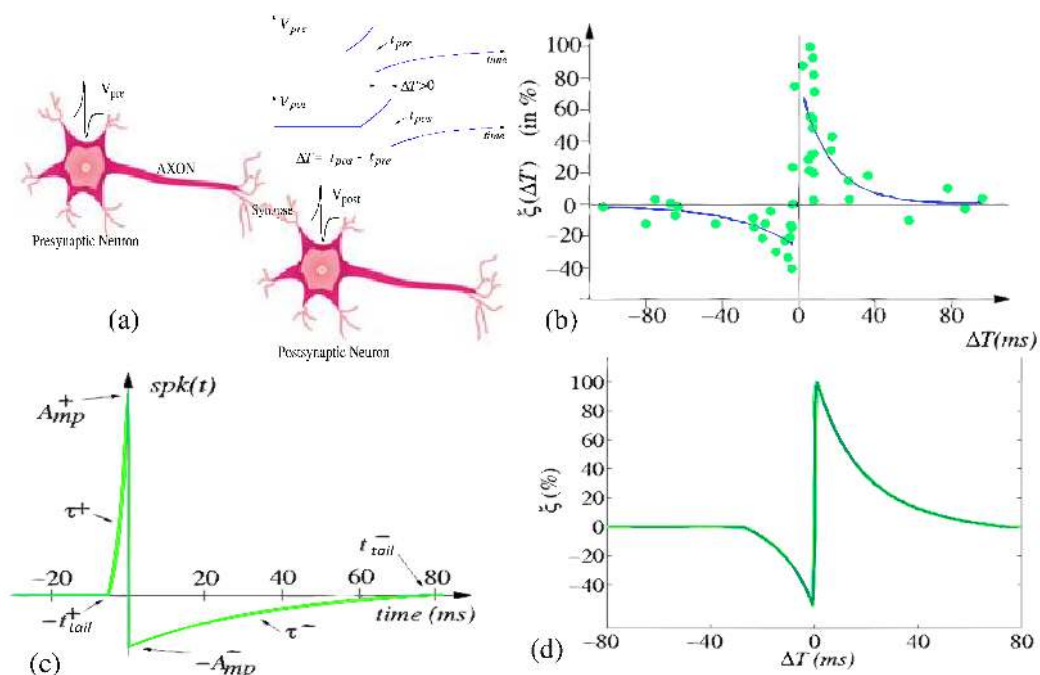
and a memristive synapse-like device where a presynaptic spike  $spk(t)$  with attenuation  $\alpha_{pre}$  arrives at time  $t$  to its negative terminal and a postsynaptic spike  $spk(t + \Delta T)$  with attenuation  $\alpha_{pos}$  arrives at time  $t + \Delta T$  to its positive terminal, a voltage difference

$$v_{MR}(t + \Delta T) = \alpha_{pos} spk(t + \Delta T) - \alpha_{pre} spk(t) \tag{6}$$

is generated among the device terminals. The total change in the memristance parameter  $w$  can thus be computed as,

$$\Delta w(\Delta T) = \int f_{MR}(v_{MR}(t + \Delta T)) dt = \zeta(\Delta T) \tag{7}$$

Interestingly, for the memristor model considered in Equation (2) and the spike shape considered in Equation (5), the memristance learning rule shown in Figure 9d  $\zeta(\Delta T)$  is obtained which resembles the STDP rule observed by Gerstner in biological neurons. By playing with the spike shapes, many other STDP update rules can be tuned as demonstrated by Zamarreño et al. [29,30].



**Figure 9.** Illustration of STDP learning rule. (a) Pre-synaptic neuron generating a spike  $V_{pre}$  at time  $t_{pre}$  that arrives to a post-synaptic neuron that generates a spike  $V_{post}$  at time  $t_{post}$ , being  $\Delta T = t_{post} - t_{pre}$ , and (b) illustrates the variation of the synaptic efficacy  $\zeta(\Delta T)$  Vs  $\Delta T$ , STDP learning rule, as the observed by Bi and Poo in biological synapses. (c) Illustrates the spike shape that applied to the memristive devices describes in Section 4 reproduces the STDP learning rule shown in (d).

In the last decade, many different works have demonstrated the emergence of STDP learning in memristive devices of different kinds of materials [137,180,185–189]. However, as already stated in Section 4, at a system level, the current limitations of the memristor technology in terms of control of the resolution of the weigh updating, have not made possible the implementation of working STDP memristive learning systems with analog synaptic elements. Precision in the weight update is difficult to control and most of the memristive devices operate changing between binary states. For that reason, stochastic STDP learning rules that operate with binary weights during inference and updating operation have been proposed. Seo et al. [190] applied this idea to simple classification problems, but they found that they could not learn to separate more than 5 patterns. Recently, Yousefzadeh et al. [191] were able to classify more elaborated databases (as MNIST) by introducing some other techniques that improved the performance.

**Combining unsupervised feature extraction methods with supervised categorization training.** While supervised learning methods like backpropagation are not energy efficient, are not appropriate

for on-line chip learning, and do not look like biologically plausible, unsupervised learning rules are appropriate to extract repetitive structures in the training data but not appropriate to take decisions [192,193]. For example, Mozafari et al. propose to combine unsupervised STDP layer with supervised Reinforcement Learning STDP layers [193]. The resulting network is more robust to overfitting compared to backpropagation training as it extracts common features and performs well with reduced number of training samples.

## 6. Future Perspective

It is well known that the human brain contains about  $10^{11}$  neurons interconnected through  $10^{15}$  synapses, and with a power consumption of around 20 W it is capable of performing complex sensing and cognitive processing, sophisticated motor control, learning and abstraction, and it can dynamically adapt to changing environments and unpredicted conditions. For this reason, neuromorphic engineers have been using the brain as a processing paradigm for several decades in order to fabricate artificial processing systems with similar capabilities. After the initial attempts of building the first spike-based processing systems demonstrated their feasibility and showed their promising potential [78], it became evident the need for scaling up these systems in terms of number of neurons and synapses [62]. Several works developed by both academic institutions [86,121–125,127–129] and industrial players like IBM [120] or Intel [126] fabricated neuromorphic chips with up to 1 M neurons and 256 M synapses, which could be ensembled in multi-chip boards and multi-board platforms, opening the way to implement large systems in the near future with numbers of neurons and synapses similar to the brain. However, these systems, based on different CMOS technologies, will be limited by their large room-scale size. Besides, the complexity of current implementations of learning algorithms in CMOS limits their scalability.

The emergence of memristors and their synaptic-like behavior opened the possibility to overcome the limitations of CMOS technologies. Memristors can be a few nanometers size and can be packed densely in a two-dimensional layer with nanometer-range pitch, potentially offering higher neuron and synaptic density. With a fabrication process much cheaper than CMOS, memristor layers can be stacked in 3D. Assuming a reasonable 30-nm pitch, the superposition of 10 memristive layers could theoretically provide a memory density of  $10^{11}$  non-volatile analog cells per  $\text{cm}^2$ . This approach could in principle reach the neuron and synaptic density of the human brain in a single board, including learning capabilities [194]. Furthermore, the close 3D dense packaging between the CMOS neural computation units and the memristive adaptive memory synaptic elements can significantly reduce the current consumption of the resulting systems.

Current available memristors are described as 1T1R devices, meaning that they are formed by the series connection of a MOS transistor and a memristive element. This transistor is used to limit the current flowing through the memristor during each operation (Forming, Writing, Erasing, Reading) to avoid damaging the device. However, this structure is limiting the density of memristors, as they are also consuming area in the CMOS substrate. An alternative to overcome this limitation is given by 1S1R devices (1-selector-1-resistor), where a volatile memristor (1S) is connected in series with a non-volatile memristor (1R), eluding any CMOS area consumption [195].

Hybrid systems with memristor layers fabricated on top of a CMOS substrate can provide highly parallel massive storage tightly coupled to CMOS computing circuitry. Therefore, computing and learning processes in the brain can be imitated by combining memristors with spiking processors and integrate-and-fire neurons in silicon. Using mesh techniques [82], grids of tens of chips can be assembled modularly on a Printed Circuit Board (PCB), allowing for scaling up the numbers of neurons and synapses in a neural system [65]. The combination of all these techniques together with the resolution of the multiple technical challenges currently associated to dense memristive layers (reliability, repeatability, reprogrammability) could provide an important step towards the hardware implementation of brain-scale low-power neuromorphic processing systems with online STDP learning.

**Author Contributions:** Writing—original draft preparation, L.A.C.-M. and T.S.-G.; writing—review and editing, L.A.C.-M., T.S.-G. and B.L.-B.; supervision, B.L.-B. and T.S.-G.; funding acquisition, T.S.-G. and L.A.C.-M.

**Funding:** This work was funded by EU H2020 grants 687299 “NEURAM3” and 824164 “HERMES”, and by Spanish grant from the Ministry of Economy and Competitiveness TEC2015-63884-C2-1-P (COGNET) (with support from the European Regional Development Fund). Luis A. Camuñas-Mesa was funded by the VI PPIT through the Universidad de Sevilla.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Von Neumann, J. First Draft of a Report on the EDVAC. *IEEE Ann. Hist. Comput.* **1945**, *15*, 27–75. [[CrossRef](#)]
2. Moore, G.E. Cramping more components onto integrated circuits. *Electronics* **1965**, *38*, 114–117. [[CrossRef](#)]
3. Waldrop, M.M. The chips are down for Moore’s law. *Nature* **2016**, *530*, 144–147. [[CrossRef](#)] [[PubMed](#)]
4. Kaur, J. Life Beyond Moore: More Moore or More than Moore—A Review. *Int. J. Comput. Sci. Mob. Comput.* **2016**, *5*, 233–237.
5. Hodgkin, A.L.; Huxley, A.F. Currents carried by sodium and potassium ions through the membrane of the giant squid axon of loligo. *J. Physiol.* **1952**, *116*, 449–472. [[CrossRef](#)] [[PubMed](#)]
6. FitzHugh, R. Impulses and physiological states in models of nerve membrane. *Biophys. J.* **1961**, *1*, 445–466. [[CrossRef](#)]
7. Nagumo, J.S.; Arimoto, S.; Yoshizawa, S. An active pulse transmission line simulating nerve axon. *Proc. IRE* **1962**, *50*, 2061–2070. [[CrossRef](#)]
8. Morris, C.; Lecar, H. Voltage oscillations in the barnacle giant muscle fiber. *Biophys. J.* **1981**, *35*, 193–213. [[CrossRef](#)]
9. Brette, R.; Gerstner, W. Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *J. Neurophysiol.* **2005**, *94*, 3637–3642. [[CrossRef](#)]
10. Izhikevich, E.M. Simple Model of Spiking Neurons. *IEEE Trans. Neural Netw.* **2003**, *14*, 1569–1572. [[CrossRef](#)]
11. Runge, R.G.; Uemura, M.; Viglione, S.S. Electronic synthesis of the avian retina. *IEEE Trans. Biomed. Eng.* **1968**, *15*, 138–151. [[CrossRef](#)] [[PubMed](#)]
12. Furber, S. Large-scale neuromorphic computing systems. *J. Neural Eng.* **2016**, *13*, 051001. [[CrossRef](#)] [[PubMed](#)]
13. Mead, C. *Analog VLSI and Neural Systems*; Addison-Wesley: Boston, MA, USA, 1989.
14. Mead, C. Neuromorphic Electronic Systems. *Proc. IEEE* **1990**, *78*, 1629–1636. [[CrossRef](#)]
15. Mahowald, M.A.; Mead, C. The silicon retina. *Sci. Am.* **1991**, *264*, 76–82. [[CrossRef](#)] [[PubMed](#)]
16. Smith, L.S. Neuromorphic Systems: Past, Present and Future. *Br. Inspir. Cognit. Syst.* **2008**, 167–182.
17. Chua, L.O. Memristor—The Missing Circuit Element. *IEEE Trans. Circuit Theory* **1971**, *18*, 507–519. [[CrossRef](#)]
18. Strukov, D.B.; Snider, G.S.; Stewart, D.R.; Williams, R.S. The missing memristor found. *Nature* **2008**, *453*, 80–83. [[CrossRef](#)]
19. Hashem, N.; Das, S. Switching-time analysis of binary-oxide memristors via a non-linear model. *Appl. Phys. Lett.* **2012**, *100*, 262106. [[CrossRef](#)]
20. Kvatinsky, S.; Belousov, D.; Liman, S.; Satat, G.; Wald, N.; Friedman, E.G.; Kolodny, A.; Weiser, U.C. MAGIC—Memristor-Aided Logic. *IEEE Trans. Circuits Syst. II Express Br.* **2014**, *11*, 895–899. [[CrossRef](#)]
21. Kvatinsky, S.; Friedman, E.G.; Kolodny, A.; Weiser, U.C. Memristor-based material implication (IMPLY) logic: Design principles and methodologies. *IEEE Trans. Very Large Scale Integr. (VLSI)* **2013**, *10*, 2054–2066. [[CrossRef](#)]
22. Su, F.; Chen, W.H.; Xia, L.; Lo, C.P.; Tang, T.; Wang, Z.; Hsu, K.H.; Cheng, M.; Li, J.Y.; Xie, Y.; et al. A 462 GOPs/J RRAM-based nonvolatile intelligent processor for energy harvesting IoE system featuring nonvolatile logics and processing-in-memory. In Proceedings of the 2017 Symposium on VLSI Technology, Kyoto, Japan, 5–8 June 2017.
23. Liu, Y.; Wang, Z.; Lee, A.; Su, F.; Lo, C.; Yuan, Z.; Lin, C.; Wei, Q.; Wang, Y.; King, Y.; et al. A 65 nm ReRAM-Enabled Nonvolatile Processor with 6× Reduction in Restore Time and 4× Higher Clock Frequency Using Adaptive Data Retention and Self-Write-Termination Nonvolatile Logic. *Int. Conf. Solid-State Circuits* **2016**, *59*, 84–86.

24. Onuki, T.; Uesugi, W.; Tamura, H.; Isobe, A.; Ando, Y.; Okamoto, S.; Kato, K.; Yew, T.; Lin, C.; Wu, J.; et al. Embedded memory and ARM Cortex-M0 core using 60-nm C-axis aligned crystalline indium-gallium-zinc oxide FET integrated with 65-nm Si CMOS. *IEEE Symp. VLSI Circuits* **2017**, *52*, 925–932.
25. Shulaker, M.M.; Hills, G.; Park, R.; Howe, R.; Saraswat, K.; Wong, H.; Mitra, S. Three-dimensional integration of nanotechnologies for computing and data storage on a single chip. *Nature* **2017**, *547*, 74–78. [[CrossRef](#)]
26. Carrara, S.; Sacchetto, D.; Doucey, M.A.; Baj-Rossi, C.; De Micheli, G.; Leblebici, Y. Memristive-biosensors: A new detection method by using nanofabricated memristors. *Sens. Actuators B Chem.* **2012**, *171–172*, 449–457. [[CrossRef](#)]
27. Snider, G.S. Spike-time-dependent Plasticity in Memristive Nanotechnologies. In Proceedings of the IEEE/ACM International Symposium on Nanoscale Architectures, Washington, DC, USA, 12–13 June 2008.
28. Linares-Barranco, B.; Serrano-Gotarredona, T. Memristance can explain spike-time-dependent-plasticity in neural synapses. *Nat. Preced.* **2009**. [[CrossRef](#)]
29. Zamarreno-Ramos, C.; Camuñas-Mesa, L.A.; Pérez-Carrasco, J.A.; Masquelier, T.; Serrano-Gotarredona, T.; Linares-Barranco, B. On spike-timing-dependent-plasticity, memristive devices, and building a self-learning visual cortex. *Front. Neurosci.* **2011**, *5*, 26. [[CrossRef](#)]
30. Serrano-Gotarredona, T.; Masquelier, T.; Prodromakis, T.; Indiveri, G.; Linares-Barranco, B. STDP and STDP variations with memristors for spiking neuromorphic learning systems. *Front. Neurosci.* **2013**, *7*, 2. [[CrossRef](#)]
31. Gerstner, W.; Ritz, R.; Hemmen, J.L. Why spikes? Hebbian learning and retrieval of time-resolved excitation patterns. *Biol. Cybern.* **1993**, *69*, 503–515. [[CrossRef](#)]
32. Gerstner, W.; Kempter, R.; Leo van Hemmen, J.; Wagner, H. A neuronal learning rule for sub-millisecond temporal coding. *Lett. Nat.* **1996**, *383*, 76–78. [[CrossRef](#)]
33. Markram, H.; Lübke, J.; Frotscher, M.; Sakmann, B. Regulation of synaptic efficacy by coincidence of postsynaptic APS and EPSPs. *Science* **1997**, *275*, 213–215. [[CrossRef](#)]
34. Bi, G.; Poo, M. Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type. *J. Neurosci.* **1998**, *18*, 10464–10472. [[CrossRef](#)]
35. Bi, G.; Poo, M. Synaptic modification by correlated activity: Hebb’s postulate revisited. *Ann. Rev. Neurosci.* **2001**, *24*, 139–166. [[CrossRef](#)]
36. Jacob, V.; Brasier, D.J.; Erchova, I.; Feldman, D.; Shulz, D.E. Spike timing-dependent synaptic depression in the in vivo barrel cortex of the rat. *J. Neurosci.* **2007**, *27*, 1271–1284. [[CrossRef](#)]
37. McCulloch, W.S.; Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.* **1943**, *5*, 115–133. [[CrossRef](#)]
38. Rosenblatt, F. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychol. Rev.* **1958**, *65*, 386–408. [[CrossRef](#)]
39. Hebb, D. *The Organization of Behavior*; Wiley: New York, NY, USA, 1949.
40. Minsky, M.L.; Papert, S.A. *Perceptrons*; MIT Press: Cambridge, MA, USA, 1969.
41. Maass, W. Networks of spiking neurons: The third generation of neural network models. *Neural Netw.* **1997**, *10*, 1659–1671. [[CrossRef](#)]
42. Ghosh-Dastidar S.; Adeli H. Third Generation Neural Networks: Spiking Neural Networks. In *Advances in Computational Intelligence. Advances in Intelligent and Soft Computing*; Yu, W., Sanchez, E.N., Eds.; Springer: Berlin, Germany, 2009.
43. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the ImageNet Classification with Deep Convolutional Neural Networks NIPS, Lake Tahoe, CA, USA, 3–6 December 2012.
44. Widrow, B. *Adaptive “Adaline” Neuron Using Chemical “Memistors”*; Number Technical Report 1553-2; Stanford Electron. Labs.: Stanford, CA, USA, 1960.
45. Widrow, B.; Lehr, M.A. 30 years of Adaptive Neural Networks: Peceptron, Madaline, and Backpropagation. *Proc. IEEE* **1990**, *78*, 1415–1442. [[CrossRef](#)]
46. Werbos, P. Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. Ph.D. Thesis, Harvard University, Cambridge, MA, USA, 1974.
47. Parker, D. *Learning-Logic*; Invention Report 581-64, File 1; Office of Technology Licensing, Stanford Univ.: Stanford, CA, USA, 1982.
48. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *1476–4687*. [[CrossRef](#)]

49. Carpenter, G.A.; Grossberg, S. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Comput. Vis. Gr. Image Process.* **1983**, *37*, 54–115. [[CrossRef](#)]
50. Kohonen, T. Self-organized formation of topologically correct feature maps. *Biolog. Cybern.* **1982**, *43*, 59–69. [[CrossRef](#)]
51. Hopfield, J.J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA* **1982**, *79*, 2554–2558. [[CrossRef](#)]
52. Bishop, C.M. *Neural Networks for Pattern Recognition*; Clarendon Press: Oxford, UK, 1995.
53. Bengio, Y. Learning Deep Architectures for AI. *Found. Trends Mach. Learn.* **2009**, *2*, 1–127. [[CrossRef](#)]
54. Schmidhuber, J. Deep Learning in Neural Networks: An Overview. *Neural Netw.* **2015**, *61*, 85–117. [[CrossRef](#)]
55. LeCun, Y.; Jackel, L.D.; Boser, B.; Denker, J.S.; Graf, H.P.; Guyon, I.; Henderson, D.; Howard, R.E.; Hubbard, W. Handwritten digit recognition: Applications of neural network chips and automatic learning. *IEEE Commun. Mag.* **1989**, *27*, 41–46. [[CrossRef](#)]
56. Deng, L.; Li, J.; Huang, J.; Yao, K.; Yu, D.; Seide, F.; Seltzer, M.; Sweig, G.; He, X.; Williams, J.; et al. Recent advances in deep learning for speech research at Microsoft. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013.
57. Thorpe, S.; Fize, D.; Marlot, C. Speed of processing in the human visual system. *Nature* **1996**, *381*, 520–522. [[CrossRef](#)]
58. Rolls, E.T.; Tovee, M.J. Processing speed in the cerebral cortex and the neurophysiology of visual masking. *Proc. R. Soc. London. Ser. B Biol. Sci.* **1994**, *257*, 9–15.
59. Thorpe, S.; Delorme, A.; Van Rullen, R. Spike-based strategies for rapid processing. *Neural Netw.* **2001**, *14*, 715–725. [[CrossRef](#)]
60. Huys, Q.; Zemel, R.; Natarajan, R.; Dayan, P. Fast population coding. *Neural Comput.* **2007**, *19*, 404–441. [[CrossRef](#)]
61. Rullen, R.V.; Thorpe, S.J. Rate coding versus temporal order coding: What the retinal ganglion cells tell the visual cortex. *Neural Comput.* **2001**, *13*, 1255–1283. [[CrossRef](#)]
62. Indiveri, G.; Linares-Barranco, B.; Hamilton, T.J.; Schaik, A.; Etienne-Cummings, R.; Delbrück, T.; Liu, S.; Dudek, P.; Häfliger, P.; Renaud, S.; et al. Neuromorphic silicon neuron circuits. *Front. Neurosci.* **2011**, *5*, 73. [[CrossRef](#)]
63. Fukushima, K. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Netw.* **1988**, *1*, 119–130. [[CrossRef](#)]
64. Hubel, D.H.; Wiesel, T.N. Receptive fields and functional architecture of monkey striate cortex. *J. Physiol.* **1968**, *195*, 215–243. [[CrossRef](#)]
65. Farabet, C.; Paz, R.; Pérez-Carrasco, J.; Zamarreño-Ramos, C.; Linares-Barranco, A.; Lecun, Y.; Culurciello, E.; Serrano-Gotarredona, T.; Linares-Barranco, B. Comparison between frame-constrained fix-pixel-value and frame-free spiking-dynamic-pixel ConvNets for visual processing. *Front. Neurosci.* **2012**, *6*, 32. [[CrossRef](#)]
66. Perez-Carrasco, J.A.; Zhao, B.; Serrano, C.; Acha, B.; Serrano-Gotarredona, T.; Chen, S.; Linares-Barranco, B. Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate coding and coincidence processing—application to feedforward ConvNets. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 2706–2719. [[CrossRef](#)]
67. Camunas-Mesa, L.; Acosta-Jiménez, A.; Zamarreño-Ramos, C.; Serrano-Gotarredona, T.; Linares-Barranco, B. A 32x32 Pixel Convolution Processor Chip for Address Event Vision Sensors With 155 ns Event Latency and 20 Meps Throughput. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2011**, *58*, 777–790. [[CrossRef](#)]
68. Sengupta, A.; Ye, Y.; Wang, R.; Liu, C.; Roy, K. Going Deeper in Spiking Neural Networks: VGG and Residual Architectures. *Front. Neurosci.* **2019**. [[CrossRef](#)]
69. Bouvier, M.; Valentian, A.; Mesquida, T.; Rummens, F.; Reybox, M.; Vianello, E.; Biegne, E. Spiking Neural Networks Hardware Implementations and Challenges: A Survey. *ACM J. Emerg. Technol. Comput. Syst.* **2019**, *15*, 1–35. [[CrossRef](#)]
70. Schmid, A. Neuromorphic microelectronics from devices to hardware systems and applications. *Nonlinear Theory Its Appl. IEICE* **2016**, *7*, 468–498. [[CrossRef](#)]
71. Sivilotti, M. Wiring Considerations in Analog VLSI Systems with Application to Field-Programmable Networks. Ph.D. Thesis, Computation and Neural Systems, California Inst. Technol., Pasadena, CA, USA, 1991.
72. Serrano-Gotarredona, T.; Andreou, A.G.; Linares-Barranco, B. AER image filtering architecture for vision-processing systems. *IEEE Trans. Circuits Syst. I* **1999**, *46*, 1064–1071. [[CrossRef](#)]

73. Boahen, K. Point-to-Point connectivity between neuromorphic chips using address events. *IEEE Trans. Circuits Syst. II* **2000**, *47*, 416–434. [[CrossRef](#)]
74. Boahen, K. A burst-mode word-serial address-event link-I,II,III. *IEEE Trans. Circuits Syst. I* **2004**, *51*, 1269–1280. [[CrossRef](#)]
75. Lin, J.; Merolla, P.; Arthur, J.; Boahen, K. Programmable connections in neuromorphic grids. In Proceedings of the 2006 49th IEEE International Midwest Symposium on Circuits and Systems, San Juan, Puerto Rico, 6–9 August 2006; pp. 80–84.
76. Merolla, P.; Arthur, J.; Shi, B.; Boahen, K. Expandable networks for neuromorphic chips. *IEEE Trans. Circuits Syst. I* **2007**, *54*, 301–311. [[CrossRef](#)]
77. Bamford, S.A.; Murray, A.F.; Willshaw, D.J. Large developing receptive fields using a distributed and locally reprogrammable address-event receiver. *IEEE Trans. Neural Netw.* **2010**, *21*, 286–304. [[CrossRef](#)]
78. Serrano-Gotarredona, R.; Oster, M.; Lichtsteiner, P.; Linares-Barranco, A.; Paz-Vicente, R.; Gomez-Rodriguez, F.; Camuñas-Mesa, L.; Berner, R.; Rivas-Perez, M.; Delbrück, T.; et al. CAVIAR: A 45k neuron, 5M synapse, 12G connects/s AER hardware sensory-processing-learning-actuating system for high-speed visual object recognition and tracking. *IEEE Trans. Neural Netw.* **2009**, *20*, 1417–1438. [[CrossRef](#)]
79. Joshi, S.; Deiss, S.; Arnold, M.; Park, J.; Yu, T.; Cauwenberghs, G. Scalable event routing in hierarchical neural array architecture with global synaptic connectivity. In Proceedings of the International Workshop Cellular Nanoscale Networks and Their Applications, Berkeley, CA, USA, 3–5 February 2010.
80. Khan, M.; Lester, D.; Plana, L.; Rast, A.; Jin, X.; Painkras, E.; Furber, S. SpiNNaker: Mapping neural networks onto a massively-parallel chip multiprocessor. In Proceedings of the 2008 IEEE International Joint Conference on Neural Networks, Hong Kong, China, 1–8 June 2008; pp. 2849–2856.
81. Benini, L.; Micheli, G.D. Networks on chips: A new SoC paradigm. *IEEE Comput.* **2002**, 70–78. [[CrossRef](#)]
82. Zamarrano-Ramos, C.; Linares-Barranco, A.; Serrano-Gotarredona, T.; Linares-Barranco, B. Multicasting mesh AER: A scalable assembly approach for reconfigurable neuromorphic structured AER systems. Application to ConvNets. *IEEE Trans. Biomed. Circuits Syst.* **2013**, *7*, 82–102. [[CrossRef](#)]
83. Fieres, J.; Schemmel, J.; Meier, K. Realizing biological spiking network models in a configurable wafer-scale hardware system. In Proceedings of the 2008 IEEE International Joint Conference on Neural Networks, Hong Kong, China, 1–8 June 2008; pp. 969–976.
84. Scholze, S.; Schiefer, S.; Partzsch, J.; Hartmann, S.; Mayr, C.; Höppner, S.; Eisenreich, H.; Henker, S.; Vogginger, B.; Schüffny, R. VLSI implementation of a 2.8 gevent/s packet based AER interface with routing and event sorting functionality. *Front. Neurosci.* **2011**, *5*, 117. [[CrossRef](#)]
85. Park, J.; Yu, T.; Joshi, S.; Maier, C.; Cauwenberghs, G. Hierarchical Address Event Routing for Reconfigurable Large-Scale Neuromorphic Systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *28*, 2408–2422. [[CrossRef](#)]
86. Moradi, S.; Qiao, N.; Stefanini, F.; Indiveri, G. A Scalable Multicore Architecture with Heterogeneous Memory Structures for Dynamic Neuromorphic Asynchronous Processors (DYNAPs). *IEEE Trans. Biomed. Circuits Syst.* **2018**, *12*, 106–122. [[CrossRef](#)]
87. Culurciello, E.; Etienne-Cummings, R.; Boahen, K.A. A biomorphic digital image sensor. *IEEE J. Solid-State Circuits* **2003**, *38*, 281–294. [[CrossRef](#)]
88. Ruedi, P.F.; Heim, P.; Kaess, F.; Grenet, E.; Heitger, F.; Burgi, P.; Gyger, S.; Nussbaum, P. A 128 × 128 pixel 120-dB dynamic-range vision-sensor chip for image contrast and orientation extraction. *IEEE J. Solid-State Circuits* **2003**, *1*, 2325–2333. [[CrossRef](#)]
89. Barbaro, M.; Burgi, P.; Mortara, R.; Nussbaum, P.; Heitger, F. A 100 × 100 pixel silicon retina for gradient extraction with steering filter capabilities and temporal output coding. *IEEE J. Solid-State Circuits* **2002**, *37*, 160–172. [[CrossRef](#)]
90. Chen, S.; Bermak, A. Arbitrated time-to-first spike CMOS image sensor with on-chip histogram equalization. *IEEE Trans. Very Large Scale Integr. Syst.* **2007**, *15*, 346–357.
91. Qi, X.G.; Harris, J. A time-to-first-spike CMOS imager. In Proceedings of the 2004 IEEE International Symposium on Circuits and Systems (IEEE Cat. No.04CH37512), Vancouver, BC, Canada, 23–26 May 2004; pp. 824–827.
92. Azadmehr, M.; Abrahamsen, J.; Häfliger, P. A foveated AER imager chip. In Proceedings of the IEEE International Symposium on Circuits and Systems, Kobe, Japan, 23–26 May 2005; pp. 2751–2754.

93. Vogelstein, R.J.; Mallik, U.; Culurciello, E.; Etienne-Cummings, R.; Cauwenberghs, G. Spatial acuity modulation of an address-event imager. In Proceedings of the IEEE ICECS, Tel Aviv, Israel, Israel, 15 December 2004; pp. 207–210.
94. Costas-Santos, J.; Serrano-Gotarredona, T.; Serrano-Gotarredona, R.; Linares-Barranco, B. A Spatial Contrast Retina with On-chip Calibration for Neuromorphic Spike-Based AER Vision Systems. *IEEE Trans. Circuits Syst. I* **2007**, *54*, 1444–1458. [[CrossRef](#)]
95. Leñero-Bardallo, J.A.; Serrano-Gotarredona, T.; Linares-Barranco, B. A 5-Decade Dynamic Range Ambient-Light-Independent Calibrated Signed-Spatial-Contrast AER Retina with 0.1ms Latency and Optional Time-to-First-Spike Mode. *IEEE Trans. Circuits Syst I* **2010**, *57*, 2632–2643. [[CrossRef](#)]
96. Zaghoul, K.A.; Boahen, K. Optic nerve signals in a neuromorphic chip: Parts 1 and 2. *IEEE Trans. Biomed. Eng.* **2004**, *51*, 657–675. [[CrossRef](#)]
97. Leñero-Bardallo, J.A.; Serrano-Gotarredona, T.; Linares-Barranco, B. A 3.6us Asynchronous Frame-Free Event-Driven Dynamic-Vision-Sensor. *IEEE J. Solid-State Circuits* **2011**, *46*, 1443–1455. [[CrossRef](#)]
98. Kramer, J. An integrated optical transient sensor. *IEEE Trans. Circuits Syst. II Analog Digit. Signal Process* **2002**, *49*, 612–628. [[CrossRef](#)]
99. Lichtsteiner, P.; Posch, C.; Delbrück, T. A  $128 \times 128$  120 dB 15  $\mu$ s latency asynchronous temporal contrast vision sensor. *IEEE J. Solid-State Circuits* **2008**, *43*, 566–576. [[CrossRef](#)]
100. Serrano-Gotarredona, T.; Linares-Barranco, B. A  $128 \times 128$  1.5% Contrast Sensitivity 0.9% FPN 3us Latency 4mW Asynchronous Frame-Free Dynamic Vision Sensor Using Transimpedance Amplifiers. *IEEE J. Solid-State Circuits* **2013**, 827–838. [[CrossRef](#)]
101. Brandli, C.; Berner, R.; Yang, M.; Liu, S.; Delbrück, T. A  $240 \times 180$  130 dB 3  $\mu$ s Latency Global Shutter Spatiotemporal Vision Sensor. *IEEE J. Solid-State Circuits* **2014**, 2333–2341. [[CrossRef](#)]
102. Moeys, D.P.; Corradi, F.; Li, C.; Bamford, S.; Longinotti, L.; Voigt, F.; Berry, S.; Taverni, G.; Helmchen, F.; Delbrück, T. A Sensitive Dynamic and Active Pixel Vision Sensor for Color or Neural Imaging Applications. *IEEE Trans. Biomed. Circuits Syst.* **2018**, *12*, 123–136. [[CrossRef](#)]
103. Posch, C.; Matolin, D.; Wohlgenannt, R. A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS. *IEEE J. Solid-State Circuits* **2011**, *46*, 259–275. [[CrossRef](#)]
104. Posch, C.; Serrano-Gotarredona, T.; Linares-Barranco, B.; Delbrück, T. Retinomorphic Event-Based Vision Sensors: Bioinspired Cameras with Spiking Output. *Proc. IEEE* **2014**, *102*, 1470–1484. [[CrossRef](#)]
105. Son, B.; Suh, Y.; Kim, S.; Jung, H.; Kim, J.; Shin, C.; Park, K.; Lee, K.; Park, J.; Woo, J.; et al. A  $640 \times 480$  dynamic vision sensor with a 9um pixel and 300Meps address-event representation. *IEEE Intl. Solid-State Circuits Conf.* **2017**. [[CrossRef](#)]
106. Guo, M.; Huang, J.; Chen, S. Live demonstration: A  $768 \times 640$  pixels 200 Meps dynamic vision sensor. In Proceedings of the 2017 IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, MD, USA, 28–31 May 2017.
107. Lyon, R.F.; Mead, C. An analog electronic cochlea. *IEEE Trans. Acoust. Speech Signal Process.* **1988**, *36*, 1119–1134. [[CrossRef](#)]
108. Chan, V.; Liu, S.; van Schaik, A. AER EAR: A Matched Silicon Cochlea Pair with Address Event Representation Interface. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2007**, *54*, 48–59. [[CrossRef](#)]
109. Wen, B.; Boahen, K. A Silicon Cochlea With Active Coupling. *IEEE Trans. Biomed. Circuits Syst.* **2009**, *3*, 444–455. [[CrossRef](#)]
110. Caviglia, S.; Pinna, L.; Valle, M.; Bartolozzi, C. Spike-Based Readout of POSFET Tactile Sensors. *IEEE Trans. Circuits Syst. I* **2017**, *64*, 1421–1431. [[CrossRef](#)]
111. Ros, P.M.; Crepaldi, M.; Demarchi, D. A hybrid quasi-digital/neuromorphic architecture for tactile sensing in humanoid robots. In Proceedings of the International Workshop on Advances in Sensors and Interfaces, Gallipoli, Italy, 18–19 June 2015; pp. 126–130.
112. Oster, M.; Douglas, R.; Liu, S.C. Computation with Spikes in a Winner-Take-All Network. *Neural Comput.* **2009**, *21*, 2437–2465. [[CrossRef](#)]
113. Camuñas-Mesa, L.; Zamarreño-Ramos, C.; Linares-Barranco, A.; Acosta-Jiménez, A.; Serrano-Gotarredona, T.; Linares-Barranco, B. An event-driven multi-kernel convolution processor module for event-driven vision sensors. *IEEE J. Solid-State Circuits* **2012**, *47*, 504–517. [[CrossRef](#)]



114. Camuñas-Mesa, L.A.; Domínguez-Cordero, Y.L.; Linares-Barranco, A.; Serrano-Gotarredona, T.; Linares-Barranco, B. A Configurable Event-Driven Convolutional Node with Rate Saturation Mechanism for Modular ConvNet Systems Implementation. *Front. Neurosci.* **2018**, *12*, 63. [[CrossRef](#)]
115. Camuñas-Mesa, L.A.; Serrano-Gotarredona, T.; Linares-Barranco, B. Event-driven sensing and processing for high-speed robotic vision. In Proceedings of the IEEE Biomedical Circuits and Systems Conference (BioCAS) Proceedings, Lausanne, Switzerland, 22–24 October 2014; pp. 516–519.
116. Indiveri, G. Modeling Selective Attention Using a Neuromorphic Analog VLSI Device. *Neural Comput.* **2000**, *12*, 2857–2880. [[CrossRef](#)]
117. Schrauwen, B.; D’Haene, M.; Verstraeten, D.; Campenhout, J. Compact hardware liquid state machines on FPGA for real-time speech recognition. *Neural Netw.* **2008**, *21*, 511–523. [[CrossRef](#)]
118. Alomar, M.L.; Canals, V.; Morro, A.; Oliver, A.; Rossello, J.L. Stochastic hardware implementation of Liquid State Machines. In Proceedings of the International Joint Conference on Neural Networks, Vancouver, BC, Canada, 24–29 July 2016; pp. 1128–1133.
119. Liu, S.C.; Delbruck, T.; Indiveri, G.; Whatley, A.; Douglas, R. *Event-Based Neuromorphic Systems*; Wiley: Hoboken, NJ, USA, 2015.
120. Merolla, P.A.; Arthur, J.V.; Alvarez-Icaza, R.; Cassidy, A.S.; Sawada, J.; Akopyan, F.; Jackson, B.L.; Imam, N.; Guo, C.; Nakamura, Y.; et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science* **2014**, *345*, 668–673. [[CrossRef](#)]
121. Benjamin, B.V.; Gao, P.; McQuinn, E.; Choudhary, S.; Chandrasekaran, A.R.; Bussat, J.M.; Alvarez-Icaza, R.; Arthur, J.V.; Merolla, P.A.; Boahen, K. Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations. *Proc. IEEE* **2014**, *102*, 699–716. [[CrossRef](#)]
122. Neckar, A.S. Braindrop: A Mixed Signal Neuromorphic Architecture with a Dynamical Systems-Based Programming Model. Ph.D. Thesis, Stanford University, Stanford, CA, USA, 2018.
123. Neckar, A.; Fok, S.; Benjamin, B.; Stewart, T.; Oza, N.; Voelker, A.; Eliasmith, C.; Manohar, R.; Boahen, K. Braindrop: A Mixed-Signal Neuromorphic Architecture With a Dynamical Systems-Based Programming Model. *Proc. IEEE* **2019**, *107*, 144–164. [[CrossRef](#)]
124. Schemmel, J.; Briiderle, D.; Gribbl, A.; Hock, M.; Meier, K.; Millner, S. A wafer-scale neuromorphic hardware system for large-scale neural modeling. In Proceedings of the 2010 IEEE International Symposium on Circuits and Systems, Paris, France, 30 May–2 June 2010; pp. 1947–1950.
125. Furber, S.B.; Galluppi, F.; Temple, S.; Plana, L.A. The SpiNNaker project. *Proc. IEEE* **2014**, *102*, 652–65. [[CrossRef](#)]
126. Davies, L.; Srinivasa, N.; Lin, T.; Chinya, G.; Cao, Y.; Choday, S.; Dimou, G.; Joshi, P.; Imam, N.; Jain, S.; et al. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. *IEEE Micro* **2018**, *38*, 82–99. [[CrossRef](#)]
127. Ma, D.; Shen, J.C.; Gu, Z.H.; Zhang, M.; Zhu, X.; Xu, X.; Xu, Q.; Shen, Y.; Pan, G. Darwin: A neuromorphic hardware co-processor based on Spiking Neural Networks. *Sci. China Inf. Sci.* **2016**, *59*, 023401. [[CrossRef](#)]
128. Qiao, N.; Mostafa, H.; Corradi, F.; Osswald, M.; Stefanini, F.; Sumislawska, D.; Indiveri, G. A re-configurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128K synapses. *Front. Neurosci.* **2015**, *9*, 141. [[CrossRef](#)]
129. Frenkel, C.; Lefebvre, M.; Legat, J.; Bol, D. A 0.086-mm<sup>2</sup> 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking Neuromorphic Processor in 28-nm CMOS. *IEEE Trans. Biomed. Circuits Syst.* **2019**, *13*, 145–158.
130. Eryilmaz, S.B.; Joshi, S.; Neftci, E.; Wan, W.; Cauwenberghs, G.; Wong, H.P. Neuromorphic architectures with electronic synapses. In Proceedings of the 17th International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, USA, 15–16 March 2016; pp. 118–123.
131. Suri, M.; Bichler, O.; Querlioz, D.; Cueto, O.; Perniola, L.; Sousa, V.; Vuillaume, D.; Gamrat, C.; DeSalvo, B. Phase change memory as synapse for ultra-dense neuromorphic systems: Application to complex visual pattern extraction. In Proceedings of the IEEE International Electron Devices Meeting (IEDM), Washington, DC, USA, 5–7 December 2011.
132. Valov, I.; Waser, R.; Jameson, J.; Kozicki, M. Electrochemical metallization memories-fundamentals, applications, prospects. *Nanotechnology* **2011**, *22*, 254003. [[CrossRef](#)]
133. Chanthbouala, A.; Garcia, V.; Cherifi, R.; Bouzehouane, K.; Fusil, S.; Moya, X.; Xavier, S.; Yamada, H.; Deranlot, C.; Mathur, N.; et al. A ferroelectric memristor. *Nat. Mater.* **2012**, *11*, 860–864. [[CrossRef](#)]

134. Wei, Z.; Kanzawa, Y.; Arita, K.; Katoh, Y.; Kawai, K.; Muraoka, S.; Mitani, S.; Fujii, S.; Katayama, K.; Iijima, M.; et al. Highly reliable TaOx ReRAM and direct evidence of redox reaction mechanism. In Proceedings of the IEEE International Electron Devices Meeting, San Francisco, CA, USA, 15–17 December 2008; pp. 1–4.
135. Kaneto, K.; Asano, T.; Takashima, W. Memory device using a conducting polymer and solid polymer electrolyte. *Jpn J. Appl. Phys.* **1991**, *30*, L215. [[CrossRef](#)]
136. Battistoni, S.; Erokhin, V.; Iannotta, S. Frequency driven organic memristive devices for neuromorphic short term and long term plasticity. *Org. Electron.* **2019**, *65*, 434–438. [[CrossRef](#)]
137. Liu, G.; Wang, C.; Zhang, W.; Liang, P.; Zhang, C.; Yang, X.; Fan, F.; Chen, Y.; Li, R. Organic biomimicking memristor for information storage and processing applications. *Adv. Electron. Mater.* **2016**, *2*, 1500298. [[CrossRef](#)]
138. Alibart, F.; Pleutin, S.; Bichler, O.; Gamrat, C.; Serrano-Gotarredona, T.; Linares-Barranco, B.; Vuillaume, D. A memristive nanoparticle/organic hybrid synapstor for neuroinspired computing. *Adv. Funct. Mater.* **2012**, *22*, 609–616. [[CrossRef](#)]
139. Song, S.; Cho, B.; Kim, T.W.; Ji, Y.; Jo, M.; Wang, G.; Choe, M.; Kahng, Y.H.; Hwang, H.; Lee, T. Three-dimensional integration of organic resistive memory devices. *Adv. Mater.* **2010**, *22*, 5048–5052. [[CrossRef](#)]
140. Kuzum, D. Synaptic electronics: Materials, devices and applications. *Nanotechnology* **2013**, *24*, 382001. [[CrossRef](#)]
141. Zidan, M.A.; Strachan, J.P.; Lu, W.D. The future of electronics based on memristive systems. *Nat. Electron.* **2018**, *1*, 22–29. [[CrossRef](#)]
142. Jo, S.H.; Chang, T.; Ebong, I.; Bhadviya, B.B.; Mazumder, P.; Lu, W. Nanoscale Memristor Device as Synapse in Neuromorphic Systems. *Nano Lett.* **2010**, *10*, 1297–1301. [[CrossRef](#)] [[PubMed](#)]
143. Serrano-Gotarredona, T.; Prodromakis, T.; Linares-Barranco, B. A Proposal for Hybrid Memristor-CMOS Spiking Neuromorphic Learning Systems. *IEEE Circuits Syst. Mag.* **2013**, *13*, 74–88. [[CrossRef](#)]
144. Demin, V.A.; Erokhin, V.V.; Emelyanov, A.V.; Battistoni, S.; Baldi, G.; Iannotta, S.; Kashkarov, P.K.; Kovalchuk, M.V. Hardware elementary perceptron based on polyaniline memristive devices. *Org. Electron.* **2015**, *25*, 16–20. [[CrossRef](#)]
145. Lin, Y.P.; Bennett, C.H.; Cabaret, T.; Vodenicarevic, D.; Chabi, D.; Querlioz, D.; Joussetme, B.; Derycke, V.; Klein, J.O. Physical realization of a supervised learning system built with organic memristive synapses. *Sci. Rep.* **2016**, *6*, 31932 [[CrossRef](#)] [[PubMed](#)]
146. Emelyanov, A.V.; Lapkin, D.A.; Demin, V.A.; Erokhin, V.V.; Battistoni, S.; Baldi, G.; Dimonte, A.; Korovin, A.N.; Iannotta, S.; Kashkarov, P.K.; et al. First steps towards the realization of a double layer perceptron based on organic memristive devices. *Aip Adv.* **2016**, *6*, 111301. [[CrossRef](#)]
147. Likharev, K.; Strukov, D. CMOL: Devices, Circuits, and Architectures. In *Introducing Molecular Electronics*; Cuniberti, G., Fagas, G., Richter, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; pp. 447–477.
148. Likharev, K. CrossNets: Neuromorphic Hybrid CMOS/Nanoelectronic Networks. *Sci. Adv. Mater.* **2011**, *3*, 322–331. [[CrossRef](#)]
149. Xia, Q.; Robinett, W.; Cumbie, M.W.; Banerjee, N.; Cardinali, T.J.; Yang, J.J.; Wu, W.; Li, X.; Tong, W.M.; Strukov, D.B.; et al. Memristor-CMOS Hybrid Integrated Circuits for Reconfigurable Logic. *Nano Lett.* **2009**, *9*, 3640–3645. [[CrossRef](#)]
150. Ankit, A.; Sengupta, A.; Panda, P.; Roy, K. RESPARC: A Reconfigurable and Energy-Efficient Architecture with Memristive Crossbars for Deep Spiking Neural Networks. In Proceedings of the Design Automation Conference 2017, Austin, TX, USA, 18–22 June 2017.
151. Chi, P.; Li, S.; Xu, C.; Zhang, T.; Zhao, J.; Liu, Y.; Wang, Y.; Xie, Y. PRIME: A novel processing-in-memory architecture for neural network computation in ReRAM-based main memory. *Int. Symp. Comp. Arch.* **2016**, *44*, 27–39. [[CrossRef](#)]
152. Cheng, M.; Xia, L.; Zhu, Z.; Cai, Y.; Xie, Y.; Wang, Y.; Yang, H. TIME: A Training-in-memory Architecture for Memristor-based Deep Neural Networks. In Proceedings of the Annual Design Automation Conference, Austin, TX, USA, 18–22 June 2017.
153. Ankit, A.; El Hajj, I.; Chalalasetti, S.R.; Ndu, G.; Foltin, M.; Williams, R.S.; Faraboschi, P.; Hwu, W.M.; Strachan, J.P.; Roy, K.; et al. PUMA: A Programmable Ultra-efficient Memristor-based Accelerator for Machine Learning Inference. In Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems, Providence, RI, USA, 13–17 April 2019.

154. Huang, H.; Ni, L.; Wang, K.; Wang, Y.; Yu, H. A highly parallel and energy efficient three-dimensional multilayer CMOS-RRAM accelerator for tensorized neural network. *IEEE Trans. Nanotechnol.* **2017**, *17*, 645–656. [[CrossRef](#)]
155. Ni, L.; Wang, Y.; Yu, H.; Yang, W.; Weng, C.; Zhao, J. An energy-efficient matrix multiplication accelerator by distributed in-memory computing on binary RRAM crossbar. In Proceedings of the Asia and South Pacific Design Automation Conference, Macau, China, 25–28 January 2016.
156. Kim, K.H.; Gaba, S.; Wheeler, D.; Cruz-Albrecht, J.M.; Hussain, T.; Srinivasa, N.; Lu, W. A functional hybrid memristor crossbar-array/CMOS system for data storage and neuromorphic. *Appl. Nano Lett.* **2011**, 389–395. [[CrossRef](#)]
157. Li, C.; Han, L.; Jiang, H.; Jang, M.H.; Lin, P.; Wu, Q.; Barnell, M.; Yang, J.J.; Xin, H.L.; Xia, Q. Three-dimensional crossbar arrays of self-rectifying Si/SiO<sub>2</sub>/Si memristors. *Nat. Commun.* **2017**, *8*, 15666. [[CrossRef](#)]
158. Wu, T.F.; Li, H.; Huang, P.C.; Rahimi, A.; Rabaey, J.M.; Wong, H.S.P.; Shulaker, M.M.; Mitra, S. Brain-inspired computing exploiting Carbon Nanotube FETs and Resistive RAM. Hyperdimensional computing case study. In Proceedings of the International Solid-State Circuits Conference (ISSCC), San Francisco, CA, USA, 11–15 February 2018; pp. 492–493.
159. Chen, W.H.; Li, K.X.; Lin, W.Y.; Hsu, K.H.; Li, P.Y.; Yang, C.H.; Xue, C.X.; Yang, E.Y.; Chen, Y.K.; Chang, Y.S.; et al. A 65 nm 1 Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors. In Proceedings of the International Solid-State Circuits Conference, San Francisco, CA, USA, 11–15 February 2018; pp. 494–495.
160. Bayat, F.M.; Prezioso, M.; Chakrabarti, B.; Nili, H.; Kataeva, I.; Strukov, S. Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits. *Nat. Commun.* **2018**, *9*, 2331. [[CrossRef](#)]
161. Kim, S.; Lim, M.; Kim, Y.; Kim, H.D.; Choi, S.J. Impact of Synaptic Device Variations on Pattern Recognition Accuracy in a Hardware Neural Network. *Sci. Rep.* **2018**, *8*, 2638. [[CrossRef](#)]
162. Ambrogio, S.; Narayanan, P.; Tsai, H.; Shelby, R.M.; Boybat, I.; di Nolfo, C.; Sidler, S.; Giordano, M.; Bordini, M.; Farinha, N.C.; et al. Equivalent-accuracy accelerated neural-network training using analogue memory. *Nature* **2018**, *558*, 60–67. [[CrossRef](#)]
163. Werbos, P.J. Backpropagation through time: What it does and how to do it. *Proc. IEEE* **1990**, *78*, 1550–1560. [[CrossRef](#)]
164. Diehl, P.U. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In Proceedings of the International Joint Conference on Neural Networks, Killarney, Ireland, 12–17 July 2015; pp. 1–8.
165. Rueckauer, B.; Lungu, I.A.; Hu, Y.; Pfeiffer, M.; Liu, S.C. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Front. Neurosci.* **2017**, *682*. [[CrossRef](#)]
166. Rueckauer, B.; Liu, S.C. Conversion of analog to spiking neural networks using sparse temporal coding. In Proceedings of the IEEE International Symposium on Circuits and Systems, Florence, Italy, 27–30 May 2018; pp. 1–5.
167. Cao, Y.; Chen, Y.; Khosla, D. Spiking deep convolutional neural networks for energy-efficient object recognition. *Int. J. Comput. Vis.* **2015**, *113*, 54–66. [[CrossRef](#)]
168. Bohte, S.M.; Kok, J.N.; Poutrā, H.L. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing* **2002**, *48*, 17–37. [[CrossRef](#)]
169. Ponulak, F. *ReSuMe—New Supervised Learning Method for Spiking Neural Networks*; Technical Report; Institute of Control and Information Engineering, Poznan University of Technology: Poznań, Poland, 2005.
170. Gutig, R.; Sompolinsky, H. The tempotron: A neuron that learns spike timing-based decisions. *Nat Neurosci* **2006**, *9*, 420–428. [[CrossRef](#)]
171. Mohammed, A.; Schliebs, S.; Matsuda, S.; Kasabov, N. SPAN: Spike pattern association neuron for learning spatio-temporal spike patterns. *Int. J. Neural Syst.* **2012**, *9*, 1250012. [[CrossRef](#)]
172. Florian, R.V. The chronotron: A neuron that learns to fire temporally precise spike patterns. *PLoS ONE* **2012**, *7*, e40233. [[CrossRef](#)]
173. Florian, R.V. Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural Comput.* **2007**, *19*, 1468–1502. [[CrossRef](#)]

174. Yu, Q.; Tang, H.; Tan, K.C.; Li, H. Precise-spike-driven synaptic plasticity: Learning hetero-association of spatiotemporal spike patterns. *PLoS ONE* **2013**, *8*, e78318 [[CrossRef](#)]
175. Lee, J.H.; Delbruck, T.; Pfeiffer, M. Training Deep Spiking Neural Networks Using Backpropagation. *Front. Neurosci.* **2016**, *10*, 508. [[CrossRef](#)]
176. Mostafa, H. Supervised Learning Based on Temporal Coding in Spiking Neural Networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 3227–3235. [[CrossRef](#)]
177. Wu, Y.; Deng, L.; Li, G.; Zhu, J.; Shi, L. Spatio-Temporal Backpropagation for Training High-Performance Spiking Neural Networks. *Front. Neurosci.* **2018**, *12*, 331. [[CrossRef](#)]
178. Shrestha, S.B.; Orchard, G. SLAYER: Spike Layer Error Reassignment in Time. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 1412–1421.
179. Zheng, N.; Mazumder, P. Online Supervised Learning for Hardware-Based Multilayer Spiking Neural Networks Through the Modulation of Weight-Dependent Spike-Timing-Dependent Plasticity. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 4287–4302. [[CrossRef](#)]
180. Mostafa, H.; Khiat, A.; Serb, A.; Mayr, C.G.; Indiveri, G.; Prodromakis, T. Implementation of a spike-based perceptron learning rule using  $TiO_{2-x}$  memristors. *Front. Neurosci.* **2015**, *9*, 357. [[CrossRef](#)]
181. Young, J.M. Cortical reorganization consistent with spike timing-but not correlation-dependent plasticity. *Nat. Neurosci.* **2007**, *10*, 887–895. [[CrossRef](#)]
182. Finelli, L.A.; Haney, S.; Bazhenov, M.; Stopfer, M.; Sejnowski, T.J. Synaptic learning rules and sparse coding in a model sensory system. *PLoS Comput. Biol.* **2008**, *4*, e1000062. [[CrossRef](#)]
183. Masquelier, T.; Guyonneau, R.; Thorpe, S.J. Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains. *PLoS ONE* **2008**, *3*, e1377. [[CrossRef](#)]
184. Masquelier, T.; Guyonneau, R.; Thorpe, S.J. Competitive STDP-based spike pattern learning. *Neural Comput.* **2009**, *21*, 1259–1276. [[CrossRef](#)]
185. Tan, Z.H.; Yang, R.; Terabe, K.; Yin, X.B.; Zhang, X.D.; Guo, X. Synaptic metaplasticity realized in oxide memristive devices. *Adv. Mater.* **2016**, *28*, 377–384. [[CrossRef](#)]
186. Prezioso, M.; Bayat, F.M.; Hoskins, B.D.; Adam, G.C.; Likharev, K.K.; Strukov, D.B. Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **2015**, *521*, 61–64. [[CrossRef](#)]
187. Matveyev, Y.; Kirtaev, R.; Fetisova, A.; Zakharchenko, S.; Negrov, D.; Zenkevich, A. Crossbar nanoscale HfO<sub>2</sub>-based electronic synapses. *Nanoscale Res. Lett.* **2016**, *11*, 147. [[CrossRef](#)]
188. Du, N.; Kiani, M.; Mayr, C.G.; You, T.; Bürger, D.; Skorupa, I.; Schmidt, O.G.; Schmidt, H. Single pairing spike-timing dependent plasticity in  $BiFeO_3$  memristors with a time window of 25 ms to 125  $\mu$ s. *Front. Neurosci.* **2015**, *9*, 227. [[CrossRef](#)]
189. Xiao, Z.; Huang, J. Energy-efficient hybrid perovskite memristors and synaptic devices. *Adv. Electron. Mater.* **2016**, *2*, 1600100. [[CrossRef](#)]
190. Seo, J.; Seok, M. Digital CMOS neuromorphic processor design featuring unsupervised online learning. In Proceedings of the IFIP/IEEE International Conference on Very Large Scale Integration, Daejeon, Korea, 5–7 October 2015; pp. 49–51.
191. Yousefzadeh, A.; Stomatias, E.; Soto, M.; Serrano-Gotarredona, T.; Linares-Barranco, B. On Practical Issues for Stochastic STDP Hardware With 1-bit Synaptic Weights. *Front. Neurosci.* **2018**. [[CrossRef](#)]
192. Mozafari, M.; Kheradpisheh, S.R.; Masquelier, T.; Nowzari-Dalini, A.; Ganjtabesh, M. First-spike-based visual categorization using reward-modulated STDP. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 6178–6190. [[CrossRef](#)]
193. Mozafari, M.; Ganjtabesh, M.; Nowzari-Dalini, A.; Thorpe, S.J.; Masquelier, T. Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks. *Pattern Recognit.* **2019**, *94*, 87–95. [[CrossRef](#)]
194. Linares-Barranco, B. Memristors fire away. *Nat. Electron.* **2018**, *1*, 100. [[CrossRef](#)]
195. Chen, P.Y.; Yu, S. Compact Modeling of RRAM Devices and Its Applications in 1T1R and 1S1R Array Design. *IEEE Trans. Electron Devices* **2015**, *62*, 4022–4028. [[CrossRef](#)]

