electronics

MDPI

Review

Memristive Neural Networks-Assisted Training Algorithms: Current State and Future Challenges

Natasa M. Samardzic [1,*], Vuk Vranjkovic [1], Predrag Teodorovic [1], Jovana Maricic [1], Sergey Shchanikov [2], Susanna Gordleeva[3], Alexey Mikhaylov [2], Max Talanov [4,5,*], Victor Kazantsev [3], Rastislav Struharik [1] and Victor Erokhin [6]

1 Faculty of Technical Sciences, University of Novi Sad, 21000 Novi Sad, Serbia; bykbpa@uns.ac.rs (V.V.); t_pedja@uns.ac.rs (P.T.); jzoranovic@uns.ac.rs (J.M.); rasti@uns.ac.rs (R.S.)
2 Research and Education Center "Physics of Solid-State Nanostructures", National Research Lobachevsky State University of Nizhny Novgorod, 603022 Nizhny Novgorod, Russia; seach@inbox.ru (S.S.); mian@nifti.unn.ru (A.M.)
3 Neurotechnology Department, Lobachevsky State University of Nizhny Novgorod, 603950 Nizhny Novgorod, Russia; gordleeva@neuro.nnov.ru (S.G.); kazantsev@neuro.nnov.ru (V.K.)
4 The Institute for Artificial Intelligence Research and Development of Serbia, 21000 Novi Sad, Serbia
5 Engineering Department, University of Messina, 98122 Messina, Italy
6 Consiglio Nazionale delle Ricerche, Institute of Materials for Electronics and Magnetism (CNR-IMEM), 43124 Parma, Italy; victor.erokhin@imem.cnr.it
* Correspondence: nsamardzic@uns.ac.rs (N.M.S.); max.talanov@gmail.com (M.T.)

Abstract

Existing challenges in modern AI architectures, including the von Neumann bottleneck and high energy demand, have led to increased research efforts towards the development of novel computing paradigms based on memristive technologies. This comprehensive review summarizes algorithmic approaches for training memristive neural networks (MNNs), including Memristive Spiking Neural Networks (MSNNs), integrating solutions implemented on hardware platforms with those developed in software. In contrast to existing reviews that consider algorithmic aspects within the broader context of hardware implementation, this review provides an algorithm-centric perspective, starting from training algorithms and analyzing different approaches to their adaptation to memristive hardware. The survey covers traditional machine learning algorithms such as backpropagation and its enhancements toward hardware-friendly solutions, as well as gradient-free methods, with particular emphasis on algorithms that address device non-idealities. We additionally provide an overview of bio-inspired learning rules, including Spike-Timing Dependent Plasticity (STDP) and reward-modulated STDP, which can be mimicked using memristive devices, along with reinforcement learning algorithms. Special attention is given to biologically plausible models of spiking neuron-astrocyte networks, and their direct implementation in memristive hardware, which significantly increase the efficiency of self-learning algorithms. Finally, this review presents an algorithm-centric perspective on open questions and challenges toward the large-scale implementation of memristive devices in AI systems.

Keywords: memristors; memristive neural networks (MNN); training algorithms; neuromorphic computing; bio-inspired algorithms; astrocyte networks; memristive crossbar arrays

Check for updates

Academic Editor: Ferdinando Di Martino

Received: 2 September 2026
Revised: 9 October 2026
Accepted: 13 October 2026
Published: 22 October 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The tremendous development of artificial intelligence (AI) systems is reshaping nearly all aspects of human activity, ranging from everyday consumer electronics [1,2] to sophisticated technological solutions, making a significant influence on education, science, art, healthcare, politics, economy, and security [3–7]. However, recent technology trends highlight open issues including environmental impact [8,9], limited computational throughput [10], latency, scalability [11], the fundamental physical limits of CMOS scaling [12], as well as resources and data intensive computing [11].

A promising approach to address these challenges is the integration of resistive switching devices, memristors, originally postulated by Leon Chua in 1971 [13], and later experimentally verified in a well-known work by HP labs [14]. Memristors are two-terminal passive devices that link charge and magnetic flux, completing the fundamental set of circuit elements [13]. They are characterized by a pinched hysteresis loop in their current-voltage characteristics, a defining fingerprint of memristive behavior [15]. Research in memristive domain includes several overlapping categories ranging from theoretical studies [16–24], systems based on Compute-in-memory (CIM) [11,25–29], algorithmic and architecture approaches [30–38], as well as emerging paradigms such as neuromorphic computing [8,39–43] and Spiking Neural Networks (SNN) [44–50], studies on quantum effect in memristive devices [12,51–54], sensing applications [55–58] and materials and device-level studies [59–63].

Particularly interesting research approaches are those that address the von Neumann bottleneck by enabling computational operations to be performed directly within memory units, i.e., through CIM paradigms. Such functionality can be realized using memristive devices arranged in crossbar architectures, where Matrix-vector Multiplication (MVM) is physically executed according to Ohm's and Kirchhoff's laws [37,64]. This methodology enables a high degree of parallelism, resulting in substantial improvements in computational throughput [25,31,65], energy efficiency [64–66], and scalability [67].

Memristor crossbar arrays (MCAs) can effectively implement artificial neural networks (ANNs) of various architectures [66,68–71] while natively supporting both ex situ and in situ training approaches [30,72]. The ex situ method computes network weights in software and then maps them onto memristive hardware, whereas the in situ approach directly updates the weights on the hardware using dedicated schemes that implement specific learning rules [30,72]. The first experimental realization of a Single Layer Perceptron (SLP) based on memristors was demonstrated by Alibart et al. in 2013, demonstrating both ex situ and in situ training approaches [72]. A major milestone was achieved by Prezioso et al. [32], who introduced an integrated memristive ANN, capable of performing analog MVM in hardware and supporting delta rule-based training. In the same year, 2015, Burr et al. reported the first large-scale memristive neural network with 164,855 synapses, trained using a gradient-based approach adapted to device nonlinearity and intrinsic asymmetry [73]. A notable advance in the parallelization of training using MCA was reported by the work of Gokmen and Vlasov [31], who theoretically mapped all three training phases—forward pass, backward pass, and weight update—onto hardware, demonstrating potential for up to $30,000\times$ acceleration compared to conventional microprocessors available at the time.

In terms of energy efficiency of MCA-based systems, an early breakthrough was presented by Yao et al. [33], demonstrating online, in situ training of a single-layer perceptron implemented with 1T1R synapses, with approximately $1000\times$ lower energy consumption per iteration compared to the Intel Xeon Phi platform. A significant step toward improving the performance of memristive neural networks was reported in reference [34], demonstrating software-level accuracy using a mixed hardware-software training approach, which will be discussed in more detail in the following sections. Generalization of combined hardware-software training was proposed by Nandakumar et al. in 2020 [64] through a

mixed-precision architecture, enabling its application to various neural network architectures (CNNs, LSTMs, GANs) beyond multilayer perceptrons. Recently, hardware-aware training approaches for memristive neural networks have gained increasing attention in the scientific community, including methods that explicitly incorporate device stochasticity into the learning rule [37], as well as approaches such as Mosaic [46], which address the low precision of memristive devices through a hardware-algorithm co-design strategy.

The inherent properties of memristive devices, including nonvolatility, gradual conductance modulation, multilevel resistive states, low power operation, and scaling potential [74], together with their compatibility with CMOS technology, make them highly suitable for bio-inspired systems [75]. Furthermore, memristors enable the implementation of local learning rules, including STDP [76] and neuromodulated STDP [77]. Consequently, they serve as versatile electronic building blocks for realizing neurons and synapses of varying complexity in neuromorphic architectures [78–82], as well as the bio-inspired and bio-mimetic circuitry of the mammalian nervous system [70,83].

A number of review papers can be found that address the integration of memristors in ANNs and SNNs; however, they are predominantly focused on hardware-level aspects [10,12,84–86]. These include circuit-level reviews [87], studies targeting specific applications of memristors in synapses [88], or more generally in SNNs [89], as well as materials, technology, and device physics-oriented roadmaps and review papers [8,11,12].

Although some papers consider algorithmic aspects, these are typically treated as part of the broader picture of hardware implementation rather than as a primary focus [10,12,89,90]. Previous reviews focus on the synaptic properties of memristors and associated plasticity mechanisms [88], hardware implementation of neural networks [10], memristive hardware building blocks and corresponding learning mechanisms [85], or circuit architectures for edge computing [84]. Other works provide a roadmap emphasizing materials engineering and emerging technologies [8], examine memristors for IMC and SNN hardware and the learning mechanisms they support [11], or review broader advances in memristive materials, devices, algorithms, and systems [12]. While these works include algorithmic discussions, the current study is organized around different categories of training algorithms for MNNs and MSNNs and analyzes their adaptations to memristive hardware, rather than using device properties and hardware architectures as the main organizing framework. Additionally, the existing literature lacks a dedicated algorithm-focused perspective on memristive neural networks and memristive spiking neural networks, and this aspect remains insufficiently explored in a structured manner. The main motivation for this work is therefore to provide an algorithm-centric perspective on memristor-based neural networks with an emphasis on comparing the strengths, limitations, and hardware requirements of different training approaches and identifying challenges and opportunities. Considering the strong interdependence between hardware and software in modern AI solutions, this review aims to critically analyze the algorithmic aspects and training methodologies applicable to memristive architectures, which play a specific role in enabling efficient implementations and have the potential to drive further improvements in the field of energy-efficient AI systems and neuromorphic engineering.

The rest of the review paper is organized as follows. Gradient-based algorithmic approaches for memristive neural networks are surveyed in Section 2, while methods that do not rely on gradient computation, i.e., gradient-free algorithms, are summarized in Section 3. Specific bio-inspired algorithms that utilize biomimetic properties of memristive devices are presented in Section 4, and the following Section 5 is devoted to reinforcement learning algorithms using memristors. The overall organization of the sections in this review is illustrated in Figure 1. In the final Section 6, we discuss future directions and applications, the main challenges and limitations, and potential approaches to addressing them, and provide concluding remarks.

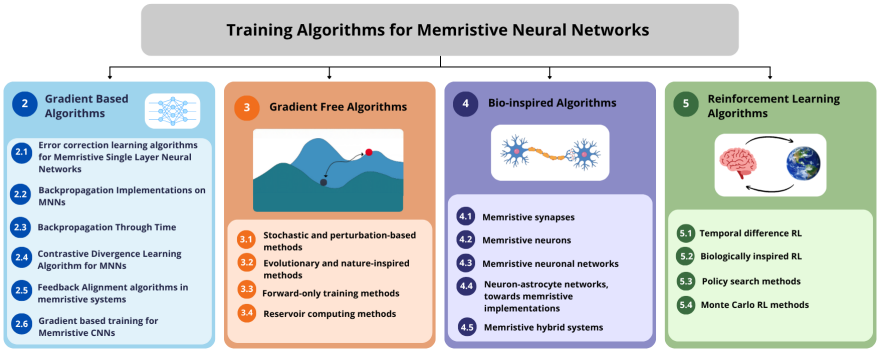


Figure 1. Overview of the training algorithms for memristive neural networks covered in this review.

2. Gradient Based Algorithms

Gradient-based learning relies on the modification of weights to reduce the error between actual and desired outputs by computing gradients of a loss function and updating the weights accordingly [91,92] represents the dominant training paradigm in modern machine learning. Memristive interpretation of gradient-based algorithms requires mapping synaptic weights to conductance values, while overcoming various sources of device non-idealities. Furthermore, efficient realization involves implementation of forward and backward passes, as well as parallelized weight updates through crossbar architecture. The following section provides a structural overview of gradient-based algorithms proposed for memristive neural networks. The topic is covered through six subsections, addressing error-correction learning algorithms for memristive single-layer neural networks, backpropagation implementations in memristive NNs, backpropagation through time, contrastive divergence learning algorithms, feedback alignment algorithms, and gradient-based training for memristive convolutional NNs. The classification is mainly based on the underlying learning mechanism, while memristive CNN architectures and their training are considered separately in the final subsection due to their broad applicability. This architecture-specific discussion remains within the gradient-based category and does not treat CNNs as a distinct class of learning algorithms

2.1. Error Correction Learning Algorithms for Memristive Single Layer Neural Networks

The error-correction learning algorithm belongs to supervised learning rules, where adjustment of network parameters is performed using an error signal defined as the difference between the target and actual output of the network [93]. Here, we summarize existing reports on error-correction learning rules for Memristive Single Layer Neural Networks architectures: the perceptron learning rule [94] and Widrow–Hoff rule [95] (Least Mean Square (LMS)). Although perceptron learning does not involve differentiation through the hard-threshold activation, its update can be interpreted as a subgradient update; therefore, we discuss it in the section on gradient-based algorithms. It is commonly discussed in the literature alongside the Widrow–Hoff rule, since both use an error signal to update the weights and rely on a linear combiner [93]. This convention is also adopted in the present overview of their memristive implementations.

2.1.1. Perceptron Learning Algorithm

The perceptron introduced by Rosenblatt in 1958 [94], represents one of the earliest neural network models, initially used for pattern classification tasks [93], while still remaining relevant in contemporary AI systems. In its simplest form it is composed of a single neuron with learnable synaptic weights, and is trained using the perceptron learning rule, based on the following weight update:

$$w_{n+1} = w_n + \eta[t_n - y_n]x_n \quad (1)$$

, where x_n is the input vector, w_n and w_{n+1} represent weight vectors before and after the update, respectively, t_n is the target output while y_n is the actual output (thresholded output $y_n = \text{sgn}(w_n^T x_n)$), η stands for the learning rate and n refers to the time step in the algorithm [93].

Early work by Alibart et al. [72] demonstrated the feasibility of implementing the perceptron learning rule in memristive crossbar systems using a single-layer perceptron for the pattern classification task. The study compared ex situ and in situ training and showed that acceptable classification accuracy could be maintained despite significant device variability, establishing a foundation for hardware-based learning in MNNs.

2.1.2. Widrow–Hoff Algorithm

LMS algorithm, also known as the Widrow-Hoff Algorithm (WH) algorithm, developed in 1960 [95], represents an early implementation of stochastic Stochastic Gradient Descent (SGD) that minimizes the Mean Square Error (MSE) between predicted and target outputs. An update is defined as follows:

$$w_{n+1} = w_n + \eta e_n x_n \quad (2)$$

, while error signal e_n is calculated based on difference between target output t_n and continuous linear output $y_n = w_n^T x_n$,

$$e_n = t_n - w_n^T x_n. \quad (3)$$

Lin et al. [96] implemented the Widrow–Hoff algorithm using a memristive Neural Logic Block (NLB) architecture [97], highlighting its potential robustness against sneak-path currents, device variability, and defect tolerance. The NLB consists of rows, containing memristors and a digital neuron, that together form a Single Neuron Unit (SNU). These units are connected to a Finite State Machine (FSM), which digitally controls in situ learning (Figure 2). Each synapse is implemented using two organic memristive devices (multi-threshold), enabling reduced fabrication cost, improved scalability, and compatibility with flexible applications. The neuron circuits are CMOS-based, and they are programmed using FPGA [96]. Within the proposed training framework, the WH is simplified by relying on the sign of the actual and target outputs, with weight updates implemented through Set-Only (SO) and Set–Reset (SR) programming modes. This approach enables real-time weight scaling and was successfully demonstrated on three-input logic functions [96].

A more recent application of the WH was reported in reference [98], using a novel complex synaptic element composed of two memristors connected in reverse series, together with two PMOS, and two NMOS transistors. Controlled by a single voltage signal, the circuit can represent positive, negative, and zero synaptic weights. Although the authors describe the approach as a combined software–hardware realization of WH rule, the implementation is primarily software-based, with the hardware aspect limited to collecting neuron outputs from circuit-level LTspice simulations, while the MSE, gradient computation, and weight updates calculations are performed in software.

A similar approach was presented by Cai et al. [99], who demonstrated online learning on a hybrid memristive-CMOS chip using a batch gradient descent rule for training SLP. While the applied rule has a similar update form as Widrow–Hoff, the weights updates are accumulated over an entire batch and derived using cross entropy loss. The system can realize forward/backward vector–matrix operations by charge-domain processing with pulse-width modulation, thus mitigating nonlinearity in I–V characteristics. Besides the feed-forward SLP, the chip was experimentally

demonstrated using different models and algorithms (sparse coding and bilayer PCA network), illustrating flexibility and re-programmability.

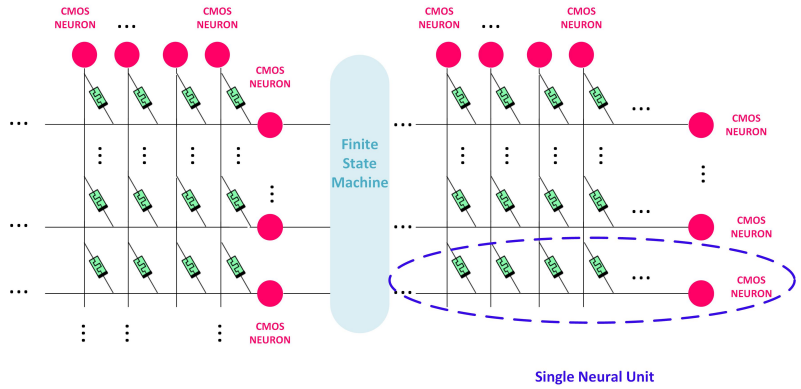


Figure 2. Neural Learning Block schematic [96].

Mapping the desired weight updates onto memristor conductance changes can be performed using variations of the delta learning rule, particularly the Manhattan rule, as presented by Prezioso et al. [32] and Yao et al. [33]. In their very well-known *Nature* paper [32], Prezioso et al. from the Strukov Research Group at UC Santa Barbara reported the first transistor-free memristor crossbar integrated in a fully connected Single Layer Perceptron, with 10 inputs and 3 outputs, using two memristors per synapse. The network was trained in situ using the Manhattan update rule optimization algorithm [100] for BP, with batch-mode learning and sign-related synaptic updates:

$$\Delta W_{ij} = \eta \operatorname{sgn} \sum_{n=1}^{N} \Delta_{ij}(n), \tag{4}$$

where η stands for the learning rate and $\Delta_{ij}(n)$ is the delta increment rule accumulated on all patterns (N) of the training set. Vector-matrix multiplication, which is the most hardware-critical operation, was computed by memristive crossbar, together with weight storage, while activation function and calculation of weight updates are realized using external electronics [32]. The results represented the first operational neural network (NN) based on a transistor-free memristive crossbar whose performance was tested on a simple pattern classification task.

Another significant contribution was reported by Yao et al. [33], who demonstrated parallel online training on a CMOS-compatible 1T1R memristor MCA consisting of 1024 cells (128 × 8) (Figure 3) and optimized for bidirectional analog switching. The one-layer perceptron architecture (Figure 3) achieved a high recognition rate during evaluation in grayscale face classification using the Yale Face Database [101]. Two programming methods were applied: (1) update using the write-verify scheme and (2) update without the write-verify scheme. Programming scheme (1) was implemented with the Delta learning rule [102], which preserves both sign and magnitude components:

$$\Delta W_{ij} = \eta \sum_{n=1}^{N} \Delta_{ij}(n), \tag{5}$$

whereas Scheme (2) employed the Manhattan learning rule (Equation (4)), which contains only the sign-based update. The realized neuromorphic system exhibits a substantial energy-efficiency advantage, achieving approximately 1000× lower synaptic energy consumption compared to an implementation based on an Intel Xeon Phi processor with off-chip memory, while also demonstrating strong scalability potential [33].

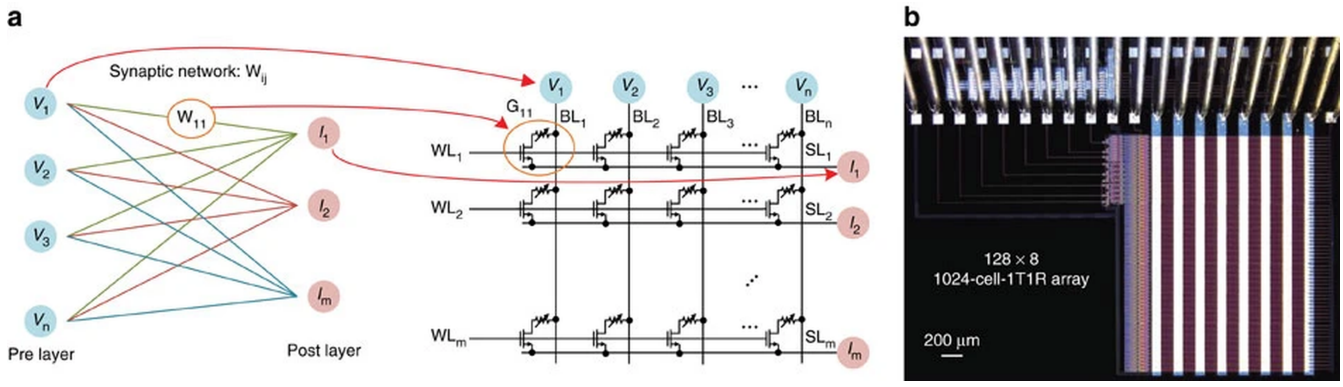


Figure 3. The 1T1R architecture and the 1024-cell-1T1R array. (**a**) Hardware realization of neural network layer. (**b**) Micrograph of 1T1R array, adapted with permission under the Creative Commons Attribution (CC BY) license from reference [33].

2.2. Backpropagation Implementations on Memristive Neural Networks

Classical approach of training artificial neural networks using Backpropagation (BP) algorithm originally introduced in 1986 [92] remains widely applied in memristive neural networks. Generally, BP training can be implemented using two learning paradigms: online (stochastic) and offline (batch) learning. In the online approach, weights are updated after each individual training sample, whereas in the offline approach the training dataset is divided into subsets (batches), and weight updates are performed after processing each batch [30]. With respect to the training location and the role of memristive hardware, a distinction is made between ex situ and in situ learning, as discussed previously. The following subsections provide an overview of conventional BP implementations in memristive neural networks, with particular emphasis on strategies for addressing device-related limitations.

2.2.1. Early Implementations of Backpropagation in Memristive Neural Networks

Addressing the hardware implementation of BP, Adhikari et al. proposed a modified chip-in-the-loop (CIL) training for an MNN employing four memristive bridge synapse circuits [103], representing nonvolatile memory. The reported modified CIL approach includes (1) initial training of the network on the host computer, (2) mapping of obtained weights on hardware, (3) forward pass realized on HW and (4) layer-wise weight update calculation on the host computer [103]. In reference [103], the authors proposed decomposition of the multilayer network into a single-layer network, which enables layer-wise updates (step 4) in the aforementioned procedure, reducing communication between the host and MNN.

Improved solution enabling on-chip training of memristive neural networks was proposed in reference [104], where a 2M crossbar was implemented in SPICE to account for finite wire resistance and sneak path currents. A hybrid MATLAB –SPICE simulation framework was utilized to implement the backpropagation training algorithm, where error and weight updates were computed in MATLAB, while forward computation and weight programming were performed in SPICE. Results were demonstrated on three input parity functions and the Wisconsin Breast Cancer Dataset [105], achieving around 8% and less than 3% recognition error, respectively. Zhang et al. [68] applied the same BP training strategy from reference [104] to a 1M crossbar array, achieving higher recognition accuracy in pattern recognition tasks with fewer training cycles. First, the authors simplified the previously proposed memristor-based AND logic switch (MS) [106] and integrated it into a crossbar circuit for weight updating, providing a robust on-line training solution for both SLNN and MLNN architectures.

Early research works on backpropagation-based algorithms for training Multi-Layer Perceptron (MLP)s implemented with memristive crossbars by Kataeva et al. [30] demonstrated the use of batch (offline) and stochastic (online) BP for in situ training of crossbar arrays based on differential pairs of Pt/TiO2-x/Pt devices, as well as stochastic ex situ training of the same architecture. In situ training was implemented using two schemes: (1) fixed-amplitude updates and (2) variable-amplitude updates. The variable-amplitude approach achieved higher classification accuracy on the MNIST dataset [107], reaching 98.53% and 95.94% for batch and stochastic training, respectively, while also compensating for device defects and variability through tuning of the activation function parameters [30]. In the same year, 2015, Soudry et al. [108] proposed a synaptic circuit architecture composed of single memristors and two CMOS transistors enabling supervised training of MLP and SLP networks using the backpropagation algorithm with online SGD. Both networks were first implemented in circuit-based software (SPICE and SimElectronics) and demonstrated comparable performance to their MATLAB-based implementation, while ensuring robustness to thermal noise and parameter variability of memristive devices, as gradient-based optimization is inherently resilient to noise. In a follow-up study by the same research group [109], a fully analog MNN was proposed and trained using BP with SGD optimization, while incorporating the influence of realistic peripheral circuits on network performance. The design integrates synaptic grid blocks enabling non-destructive readout, signal amplifications, capacitors for suppressing high voltage spikes (shock capacitors), and current-to-voltage converters together. Comparable performance to software-based models was achieved, alongside strong noise robustness and an execution speed approximately one order of magnitude higher than that of the MATLAB implementation [109].

The first large-scale experimental implementation of memristive neural networks with approximately 165,000 synapses in a three-layer MLP, trained using the backpropagation algorithm, was presented by Burr et al. in 2015 [73]. Each synaptic weight is mapped onto a differential pair of two nonvolatile Phase Change Memory (PCM) devices in a 1T1R configuration fabricated in a 180 nm technology node, while neurons are software-based. The conventional BP approach was improved using a crossbar-compatible weight-update rule, with the "G-diamond" concept (to address nonlinear and asymmetry issues of memristors) allowing parallel weight update and competitive accuracy.

A notable advancement toward fully hardware-based training of MNN was reported by Krestinskaya et al. [110], who developed an analog learning circuit for backpropagation and validated it through simulations using a TSMC 180 nm CMOS technology. The learning circuit, based on a memristive crossbar array with NMOS transistors placed at the end of each column, consists of dedicated blocks for forward propagation, separate blocks for backpropagation to the output and hidden layers, and additional circuitry for weight updates. The initial simulation results were compared with a MATLAB-based backpropagation implementation by analyzing output voltages and timing diagrams [110]. The application of these results, along with a detailed circuit design, was further presented by the same group [111] through the realization of several neuromorphic memristive architectures, including three-layer Deep Neural Networks (DNNs), Binary Neural Networks (BNNs), Multilayer Neural Networks (MLTNNs), Hierarchical Temporal Memory (HTM), and Long Short-Term Memory (LSTM) networks. Power and area estimations for the main circuit blocks were provided, together with system-level simulations performed on XOR, handwritten digit recognition (MNIST), and face recognition (Yale) tasks. The analysis additionally accounted for sneak-path effects, memristor device variability, and finite wire resistance [111].

2.2.2. Ex Situ Implementations of Backpropagation

As outlined previously, ex situ BP training involves first computation of network weights in software and afterwards their deployment onto memristive hardware circuits, whereas the mismatch between utilized models and characteristics of real devices, as well as hardware constraints, remains an important challenge. Below, we briefly review the main results in ex situ BP training methodologies.

Early work on ex situ BP implementation was presented in reference [67], using an MCA composed of 0T1M cells, together with novel training and reading circuits that achieved significant area saving of up to 93% compared to the 1T1M cell structure. However, the authors highlight that ex situ training offers limited robustness to device variability and stuck faults within the crossbar. In order to examine these limitations, Mehonic et al. [112] analyzed in detail the influence of silicon-oxide RRAM device non-idealities on the classification accuracy of an MLP during the inference phase. The network was trained in software using a standard BP approach, while during inference experimental parameters of real SiO_x devices were incorporated into the simulations to model weight mapping, HRS/LRS ratio, fault device influence, I/V nonlinearity and device-to-device variability. The obtained results indicate that the HRS/LRS ratio had the most significant impact on the MNIST classification accuracy, which reached 96.8% using the proportional mapping scheme (device conductances proportional to the synaptic weights) [112]. Inference using a 4 kbit RRAM array composed of HfAlO 1T1R cells with five resistive levels was presented in reference [113], where a 2-layer NN was previously trained in software using BP. Authors exploited low device-to-device variability of HfAlO devices to introduce a multilevel programming scheme, providing one HRS and four LRS states. A purely software-implemented network was compared to hardware inference implementation, which achieves more than 80% accuracy in the MNIST dataset, highlighting the importance of joint efforts in material science and algorithmic design.

Physical characteristics aware ex situ training framework (PHAX) was introduced in reference [114] for hybrid CMOS-memristive circuits aiming to bridge the gap between high-level models and circuit behavior. The authors develop an accurate SPICE-based modeling flow for memristive crossbars and inverter neurons and modify backpropagation by embedding hardware constraints. This enables effective offline optimization, achieving training runtime reductions of approximately 1805× compared to in situ methods, alongside significant energy and speed benefits, including up to 1478× power savings over ASIC implementations. The work in reference [115] further improved offline training robustness by addressing sensitivity to process variations. A circuit-level technique is proposed that reduces crossbar loading effects on the activation function by lowering the inverter voltage transfer characteristic slope via grounded resistors. The resistance is tuned to balance the nominal accuracy and the tolerance to variation. As a result, offline trained networks achieve up to 96% improvement in μMSE and 79% higher accuracy under 20% memristor variation compared to PHAX.

2.2.3. Parallel and In Situ Realizations of Backpropagation

Memristive crossbar arrays implemented in neural networks trained with BP offer the potential for three forms of parallelization: during the forward pass, backward pass and weight update [12,31]. Forward and backward vector-matrix multiplications are inherently performed in MCA through Ohm's and Kirchhoff's laws. The implementation of in situ training could enable full exploitation of hardware parallelism during weight update with an appropriate update scheme.

Gokmen et al., in one of the foundational papers on memristive neural networks training [31], introduced a theoretical approach for implementing local and parallel weight

updates in 2D crossbar arrays based on a concept of Resistive Processing Unit (RPU) devices, which can be realized using memristive technology and support both storage and computation. The proposed framework is a simulation-based study that shows that all stages of backpropagation—forward, backward, and weight update—can be mapped onto crossbar arrays. While the forward and backward passes are realized through Vector-Matrix Multiplication (VMM) operations, the weight update is based on the outer product between the input and error vector and is implemented using a pulse-based stochastic scheme, enabling fully parallel updates with $O(1)$ time complexity. The authors further provide a detailed analysis of strict device-level requirements that the RPU device must satisfy to realize the proposed training concept. Later work by Gokmen et al. proposed replacing the conventional SGD weight update within BP-based training with the Tiki-Taka (TT) algorithm [116], Figure 4, which allowed relaxing device symmetry requirements. In the TT training procedure, the weight matrix is represented as a linear combination of matrices A and C:

$$W = A + \gamma C, \tag{6}$$

where γ stands for scaling factor, while A and C are part of a coupled dynamical system, with gradients accumulated in matrix A and weight updates performed in both A and C. The improved version of TT, Tiki-Taka v2 (TTv2) [117], enabled the use of a much smaller number of conductance states (only 10) compared to the 1000 required by Gokmen and Vlasov's work from 2016 [31], while enhancing the noise tolerance with minimal implementation cost. This was achieved by adding matrix H in between A and C (Figure 4), performing low-pass filtering and accumulation, [117,118]. A simulation-level study of TT and TT2 algorithms using real device properties of $TiN/HfO_x/TiN$ ReRAM was done in reference [118], illustrating a significant decrease in test error of the LSTM network when TTv2 was employed, while emphasizing that the large-scale HW implementation remains an open issue.

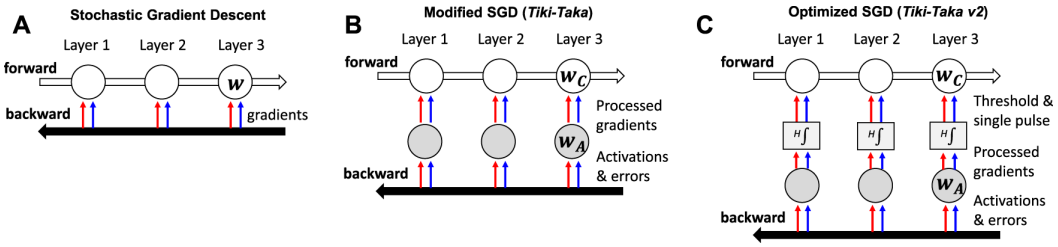


Figure 4. Algorithmic diagrams of (**A**) SGD, (**B**) Tiki-Taka, (**C**) Tiki-Taka v2 adapted with permission under the Creative Commons Attribution (CC BY) license from reference [118].

Moving towards hardware realization, backpropagation with SGD was used for training a two-layer MNN implemented on a 128 × 64 memristor array with a 1T1R configuration (Figure 5), where each weight was represented as a differential synaptic pair, and updated using in situ online training [119]. During the feedforward stage, the crossbar architecture was utilized to compute the weighted sums of voltages in parallel within each layer, while peripheral circuits were used to convert output currents to voltage, while the nonlinear activation was realized in software. In the feedback phase, error backpropagation and weight changes were computed in software, while the updates were applied to the crossbar using a partially parallel update scheme in which one memristor line was updated at a time [119,120]. A fully hardware implementation of a two-layer perceptron using 128 × 64 crossbar arrays and trained through a combined hardware and software approach was reported by Kiani et al. [121]. The training combined in situ approach with software support, and weight updates were performed in the form of one-shot blind programming, while inference was

realized completely in hardware. Authors also included non-ideality influence analysis of network performance covering nonlinearity in conductance tuning and noise originating from hardware realization of ReLU. The presented perceptron built using analog neurons achieves substantially lower hidden layer latency by eliminating the need for additional peripheral circuitry, thereby improving energy efficiency and computational throughput [121].

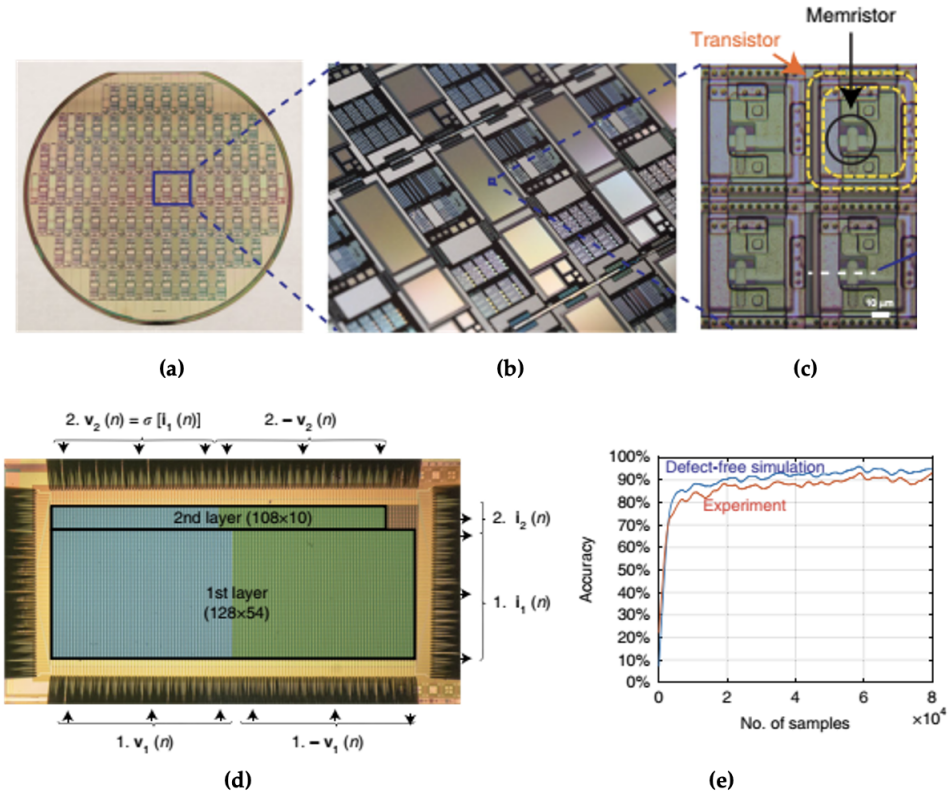


Figure 5. In situ learning with the memristive 1T1R chip. (**a**) Wafer level view of 1T1R arrays. (**b**) Chip view featuring arrays of different size. (**c**) Micrograph of 1T1R cell. (**d**) Integrated 128 × 64 array divided into first and second network layers. (**e**) Accuracy versus the number of training samples for experimental results and defect-free simulations., adapted with permission under the Creative Commons Attribution (CC BY 4.0.) license from reference [119].

Circuit-level analysis that introduces row parallel weight update within an in situ training scheme for both memristive Multi-Layer Neural Network (MLNN) and memristive convolutional neural networks (MCNNs) was reported by Liu et al. [122]. In this work, MCAs were utilized to implement fully connected layers as well as convolutional and average pooling operations. Training of the memristive MLNN and MCNN architectures was performed using the BP algorithm through a six-step procedure: (1) global reset of all memristors, (2) forward propagation and error computation, (3) backward propagation of error signals, (4) determination of the required weight updates, (5) calculation of the corresponding programming voltages, and (6) application of these voltages to update the memductance values via row-parallel programming. The procedure was iteratively repeated until the loss function reached a predefined convergence threshold. All input preprocessing, intermediate convolution operations, update computations, and programming-voltage generation were executed by dedicated peripheral circuits. The weight update is realized through the row-parallel updating method in two phases per each row: the first increasing the conductance, and the second decreasing the conductance of selected memristors, as illustrated in Figure 6, with color mapping (red memristors have increased conductance, green decreased and blue devices remain unchanged). The realized MCNN performed

well on the MNIST dataset; however, its performance deteriorated significantly on the more complex CIFAR-10 dataset.

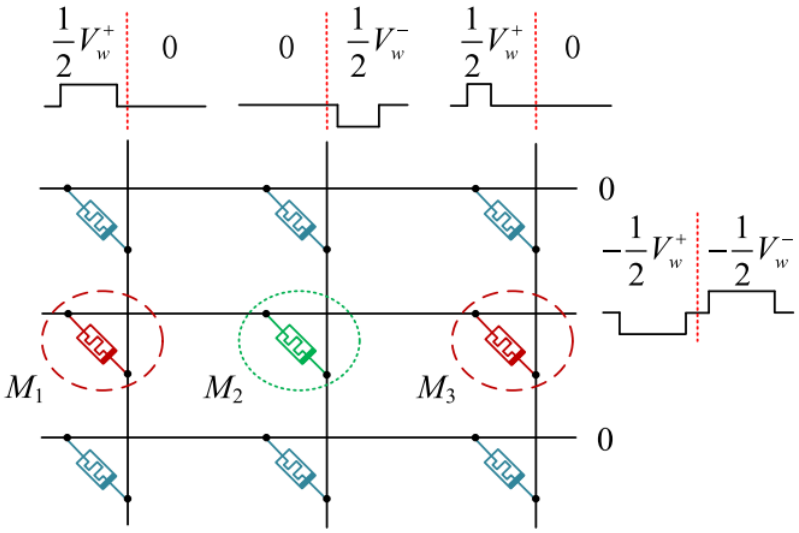


Figure 6. Two phase row parallel updating, conductance of memristors M1 and M3 was increased while conductance of M2 was decreased after update row, adapted with permission under the Creative Commons Attribution (CC BY) license from reference [122].

Focusing on parallelization of the weight update process, Feng et al. introduced a two-step method for implementing weight update using BP with the SGD algorithm, where weights in the crossbar are updated in parallel, independently of array size [123]. The method relies on the modulation of the duration and polarity of training pulses according to the amplitude and sign of the input and error product (for details, see [123]). Validation is performed through circuit-level PSPICE simulations and VTEAM memristive model [124]. The method is applicable to general 1T1M crossbar architectures employing voltage-controlled, time-invariant memristive devices [125], provided that the on-resistance of the transistor in the linear regime must be significantly lower than the corresponding memristance [108,123]. Further parallelization of all stages of BP was recently proposed by Prajapati et al. [120] with a 1M synaptic design for both MLNN and single-layer neural networks (SLNN), as in situ supervised training. The MLNN and SLNN architectures, realized in LTspice using the Yakopcic memristive device SPICE model [126], fitted to parameters of two realistic memristors [127,128]. The systems achieved classification accuracy comparable to state-of-the-art results on the NASA Asteroid [129], Breast Cancer Wisconsin [105], and IRIS datasets [130]. A fully parallel weight-update mechanism, allowing all memristive synapses to be programmed concurrently, significantly reduces training time and improves tolerance to device variations, nonlinearity and fault conditions. The presented results represent a follow-up of the authors' previous work [131], including nonlinearity and fault analysis, while reducing the synaptic circuit from a 2M to 1M structure. Although the weight updating operation in $O(1)$ time is reported, the study lacks a hardware demonstration and validation across a variety of different network architectures.

2.2.4. Hardware-Aware Backpropagation

Hardware-aware (HWA) backpropagation focuses on addressing issues associated with constraints of memristive hardware, including weight update, various device non-idealities, limited number of conductance states and related limitations, by modifying the training algorithm accordingly. One possible direction of HWA training is mixed-precision training, in which computational and storage are distributed between different

hardware components. An important breakthrough in the field was achieved by Ambrogio et al. [34] in 2018, demonstrating for the first time software-level accuracy using a combined hardware–software neural network approach. The synaptic cell was composed of the differential pair of non-volatile PCM devices, implemented in hardware, with a SPICE simulation model of a volatile 3T1C storage element also in the form of a differential pair. During training, weight updates occurred in the 3T1C units, and were periodically transferred to nonvolatile PCM devices, leading to improved computational energy efficiency. Gradual and bidirectional weight updates were enabled by adding or removing charge in the 3T1C capacitor, which mitigates the impact of PCM non-idealities on network performance. Following this concept, an on-chip mixed-precision training method using a PCM hardware platform was presented in reference [64]. The proposed approach in training MNNs using backpropagation with SGD is enabled by combining computational memory units composed of multiple crossbar arrays with high-precision digital units as illustrated in Figure 7. Analog memory units perform weighted summations in forward and backward training passes according to Kirchhoff's and Ohm's laws, while a high-precision digital unit computes and accumulates gradient-based weight updates. When the accumulated weight update exceeds the threshold, blind single-shot voltage pulses are applied for programming of PCM devices. Simulation-level results show further that this approach can be extended beyond MLP to more complex architectures such as CNNs, LSTMs and GANs while effectively suppressing the impact of the stochastic behavior of nonlinear devices [64]. Wu et al. [25] showed that the mixed-precision training scheme introduced in reference [64] can be applied to an analog–digital heterogeneous system employing 1T1R crossbar arrays based on non-filamentary bulk RRAM (b-RRAM) as analog CIM units. Convolutional kernels were mapped to memristors, while high-precision weight values W_{FP} were maintained in the digital unit. The weight update was accumulated in the weight matrix ΔW_{FP}^{k} as

$$\Delta W_{FP}^{k} += \Delta W^{k}, \tag{7}$$

where k denotes the layer index. When ΔW_{FP}^{k} exceeded the threshold value, the memristance was modified (write-and-verify mode), after which ΔW_{FP}^{k} was reinitialized to zero. The described system was used to experimentally implement LeNet, while VGG-8, and ResNet-18 were evaluated at the simulation level. The method enabled sparse weight updates, supported various optimization techniques, and mitigated different device- and system-level non-idealities [25].

Another HWA methodology relies on incorporating real device characteristics into the training algorithm. In line with this, Mikhaylov et al. [132] presented one of the first hardware implementations of an MLP with practical memristive synapses, using an array of 32 metal-oxide memristive devices in a fully integrated one-board analog–digital system that accounted for device-to-device and pulse-to-pulse variations. The authors demonstrate a complete training workflow based on error backpropagation adapted to memristor constraints, validate robustness to resistance variability, and showcase nonlinear classification on simulated neuronal activity patterns. Moreover, the impact of randomly distributed stack OFF devices, as selected non-ideality, was investigated as part of the non-ideality-aware training approach reported in reference [133], demonstrating improved robustness to a Fast Gradient Sign Method (FGSM) attack. Another problem of achieving linear control over weight updates, which is important for efficient implementation of the BP algorithm in memristive hardware, while preserving network accuracy, was investigated by Gunakin et al. [134]. The authors proposed the usage of the artificial bee colony (ABC) optimization algorithm to determine circuit parameters of memristive synaptic elements and their corresponding linear ranges, resulting in approximately linear control of weight updates. An additional circuit-level improvement of memristive MLNN, aimed at enabling

on-chip training, was presented in reference [135] by introducing a memristive circuit for storing the product of two analog voltages. Consequently, ADC and DAC units were not required, leading to approximately 2× improvement in energy efficiency and 1.5× speed up compared to systems relying on signal conversion.

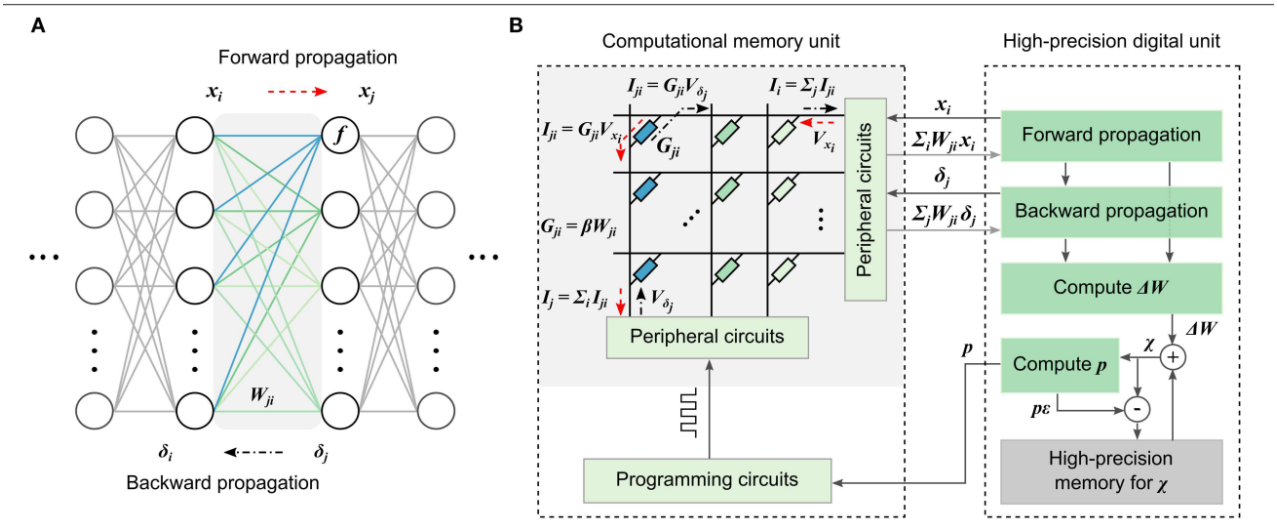


Figure 7. Mixed precision deep learning architecture using memristors: (**A**) neural-network forward and backward propagation; (**B**) in-memory computation combined with high-precision digital calculation of weight updates, adapted with permission under the Creative Commons Attribution (CC BY) license from reference [64].

Adaptations of BP from an algorithmic perspective, in the narrow sense, can also be classified as hardware-aware BP approaches. A representative example is the sign backpropagation (SBP) algorithm, an on-chip learning algorithm, based on the sign of the backpropagation error, demonstrated for training of Resistive Random Access Memory (RRAM)-based two-layer 744-300-100 MLP [136]. Within the presented work, the forward pass is implemented using binary interfaces with current comparators placed at the output of crossbar arrays, while the backward pass uses the sign of error to determine the update operation (SET or RESET) applied to RRAM devices [136]. The SBP algorithm achieves 94.5% accuracy on the MNIST dataset [107], whereas operation time and consequently energy consumption can be controlled with proper selection of the quantization threshold value [136]. Furthermore, non-idealities, including weight update nonlinearity and SET/RESET asymmetry, can be partially compensated. On the other hand, Dong et al. [36] proposed a new weight update rule for BP and implemented it on memristive neural networks (MNN), in which gradients are computed from current and previous iterations, i.e., accumulated during the gradient descent phase. The resulting algorithm named Backpropagation with gradient accumulation (BP-GA) was applied for training of single-layer perceptron (SLP), MLP and Convolutional Neural Networks (CNN) with memristive crossbars and compared with conventional BP [36]. The results obtained indicate insensitivity of the BP-GA algorithm in a wide range of learning rate values, as well as in device non-idealities (noise, variations in the ratio of ON-to-OFF states and low number of conductance states), while maintaining relatively high accuracy [36]. Recently, Liu et al. [37] proposed the Error-aware Probabilistic Update (EaPU) method as an enhanced backpropagation-based training approach for MNNs, addressing the algorithm–device mismatch arising from stochastic conductance updates that can affect training stability. The underlying idea is to enlarge the weight update magnitude (see Equation (4) in reference [37]) to suppress the influence of update noise, while reducing the overall number of weight updates to below 1%.

As a result, the training energy is reduced by 50× compared to the reported BP applied on MNNs. He et al. introduced a model-centric HWA training approach based on the discrete memristor model [137] that satisfies the three-fingerprint definition of memristors [15]. The model was afterwards used as a building element of a Discrete Memristor Backpropagation Neural Network (DM-BPNN) [138]. Within the discrete memristive neural module, the input signal is implemented as a discrete current, while the output is represented as a discrete voltage. Additionally, a control signal determines whether the memristance of the synaptic element is updated, depending on whether the network is operating in the forward or backward phase. The application of the DM-BPNN was demonstrated on the MNIST character recognition task using a three-layer neural network, achieving 97.16% classification accuracy with a preprocessing stage [138], although the study was limited to numerical simulations.

2.3. Backpropagation Through Time

Backpropagation Through Time (BPTT) represents an extension of the conventional BP algorithm, specifically designed for dynamic systems, including neural networks with memory (Recurrent Neural Networks, RNNs), control systems, fuzzy logic structures, and temporal pattern recognition systems [139,140]. The ability of BPTT to capture spatio-temporal dynamics makes it particularly suitable for SNNs, including Memristive SNNs (MSNNs), in which neuronal internal states vary over both space and time [38,141] (Figure 8).

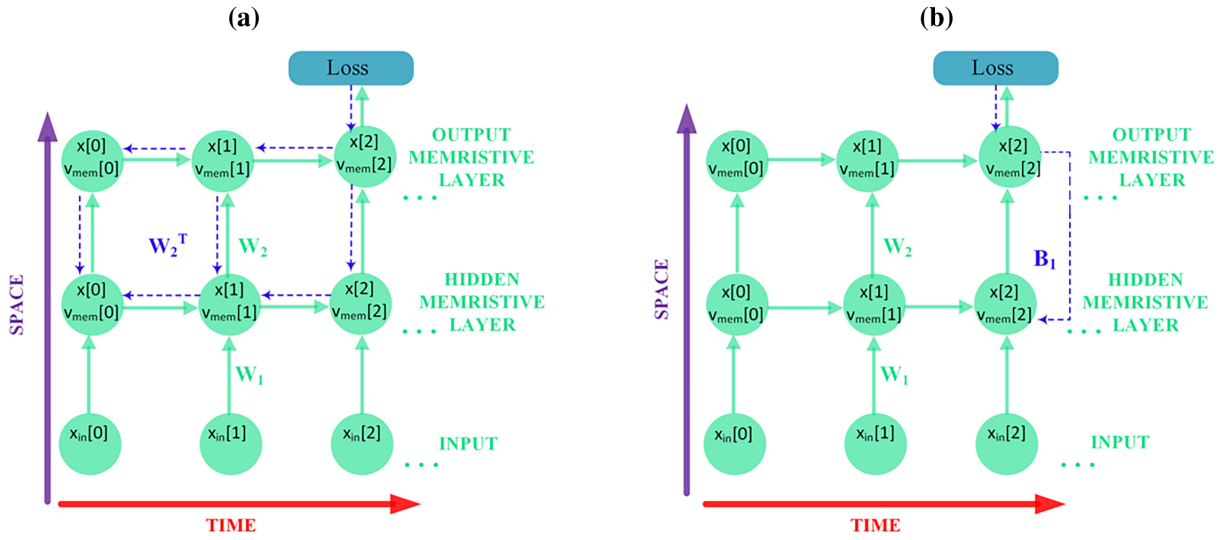


Figure 8. BPTT (**a**) and DFATT (**b**) training algorithms—forward pass annotated with green lines, while backward pass is represented as blue dashed lines. Reproduces with permission from reference [38], © John Wiley & Sons, 2025.

A direct implementation of BPTT on fully memristive SNNs was presented by Zhou et al. [142,143], exploiting the analog nature of memristive neurons and synapses, allowing intrinsic differentiation without the need for approximate approaches such as surrogate gradients [144]. Kang et al. first introduced the memristive integrate-and-fire (MIF) neuron model in reference [44], composed of two memristors, two DC voltage sources, and a capacitor, enabling the reproduction of depolarization, hyperpolarization, and repolarization phases. The model was later employed in reference [142,143], where network-level learning was validated through SPICE simulations together with memristive synapses. The reported accuracies of the lightweight MSNN are comparable to those in the existing literature on fully memristive SNNs, while retaining low peripheral circuit complexity and achieving significant improvements in area, latency, and power consumption (due to sparse

activity of only 2% of neurons) compared to mixed-signal architectures [143]. A particularly simple realization of the leaky integrate-and-fire (LIF) neuron based on memristors can be achieved using only a single volatile (diffusive) memristor and one capacitor [145,146], while preserving spatial and temporal integration, spike generation, and the decay of the local gradient potential (LGP). This hardware-efficient structure facilitates shallow MSNN designs that support direct BPTT training by leveraging the inherent physical properties of the volatile memristor, thus reducing the requirement for auxiliary peripheral circuitry [38].

Another application of the BPTT algorithm is the training of a distinct class of recurrent neural networks with memory blocks, known as LSTM networks [147]. Introduced in 1997 to address the vanishing and exploding gradient problems [148], this architecture is composed of LSTM cells that initially included input and output gates, while a forget gate was added later to enable the removal of unnecessary information [147,149]. The computational structure of LSTM can be interpreted in a memristive framework by implementing its vector–matrix multiplications within a MCA formed from memristive synaptic circuits [147], offering reduced area footprint and improved energy efficiency [150]. Gokmen et al. [151] mapped the LSTM computational block onto a same-size Resistive Processing Unit (RPU) array to enable parallel execution of vector–matrix multiplications in large-scale LSTM networks. The remaining nonlinear operations are handled by a digital module referred to as the nonlinear function (NLF) unit. The reported results indicate that the asymmetry of weight updates has a crucial impact on network performance, alongside the reduced signal resolution through the use of a stochastic rounding scheme [151]. Notable contributions to memristive LSTM architectures were reported by Smagulova et al., who first proposed a current-based LSTM design [152] and later demonstrated its applicability across several tasks [147]. The proposed memristive LSTM framework was applied to a range of sequence-processing problems, including time-series forecasting [153], sequential data processing and storage [154], and wafer quality inspection [155].

The first hardware implementation of a memristive LSTM was reported by Li et al. [156], who integrated a 128 × 64 1T1R MCA into the LSTM layer and used it for in situ VMM and parameter storage. The resulting LSTM-based RNN was trained using the BPTT algorithm with a two-phase weight update scheme [119], with software implementation of nonlinear units of the FC layers and gated units in the LSTM layers, and evaluated through regression and classification experiments. The delta rule was used for the calculation of weight gradients, while the optimization method for airline passenger prediction (regression) was stochastic gradient descent with momentum (SGDM), and the root mean square propagation (RMSProp) optimizer was used for the classification task (human gait identification). The authors claim that the proposed in situ training approach can adapt to hardware defects and has the potential to reduce latency and power consumption compared to classical LSTM implementations [156].

Improvements in the system architecture were presented through the memristor-based LSTM (MbLSTM), which is composed of a memristor-based LSTM cell, a dense layer, and a Winner-take-all (WTA) module [150]. The authors presented and simulated a circuit design enabling in situ training, including implementations of activation functions and a solution for identifying the maximum output in classification tasks using a WTA block following the dense layer. Robustness analysis with respect to update inaccuracies and conductance errors demonstrates promising performance, while requiring a reduced number of components within the LSTM cell [150].

Recent work by Dalgaty et al. [46] employs BPTT with surrogate gradient for training of SNNs within the Mosaic neuromorphic architecture, which is based on a small-world topology characterized by dense local and sparse long-range connections between neurons. ReRAM arrays are integrated into computing and routing blocks of the architecture, allow-

ing in-memory routing, while maintaining competitive accuracy and offering significant potential for energy savings. The authors emphasize the need for novel training procedures to fully exploit the advantage of small-world topology.

2.4. Contrastive Divergence Learning Algorithm for Memristive Neural Networks

The Contrastive Divergence (CD) Learning Algorithm was proposed in 2002 by Geoffrey E. Hinton [157] as a stochastic gradient approximation method relying on Gibbs sampling, designed for training of Restricted Boltzmann Machines (RBM), an energy-based model used as a constituent layer of Deep Belief Networks (DBNs). The two-memristor model utilized for storing synaptic Long-term Potentiation (LTP) and long-term depression (LTD) weights in the RBM model was presented in reference [158] in 2015. An altered CD algorithm adjusted for HW implementation is employed, where weight updates are performed in two write cycles, which reduces the need for storing of weight value within a neural node but potentially increases training latency.

More recently, Kvatinsky et al. [159] reported hardware-level simulation of memristive DBN with online Contrastive Divergence training, which enables stochastic binary activations and eliminates the need for ADC or DAC peripherals, external circuits for activation functions and memory units for intermediate states of hidden layers. Authors claim immunity of memristive synapses from non-idealities, including asymmetry in potentiation and depression, limited conductance levels (even two conductance levels accuracy reaches above 97%), nonlinear and asymmetric weight updates, write variations, device yield and nonlinear I-V curves. Scaling up requires including convolutional layers, which was not considered in this study.

2.5. Feedback Alignment Algorithms in Memristive Systems

Unlike the classical BP method, which calculates the gradient by sequentially propagating the error backward from the output layer through all the hidden layers, the feedback alignment demonstrates that the weights used for error propagation do not need to be symmetric with the feedforward weights [160]. Instead, the network reduces the error by learning to align with fixed, random feedback weights, through which the error signal propagates backward. Nokland [160] introduces two variations in this approach, which differ mainly in how the error signal is routed. In Direct Feedback Alignment (DFA), the error signal is transmitted directly and simultaneously from the output layer to each hidden layer. In contrast, in Indirect Feedback Alignment (IFA), the error is projected directly to the first hidden layer via random weights, and then propagates forward to subsequent layers using the original feedforward weights. A comparison of these methods is illustrated in Figure 9.

The motivation for utilizing the DFA algorithm arises from the localized processing observed in biological neural networks [161]. The primary objective is to perform computations directly within memory, thereby minimizing global data movement and avoiding the von Neumann bottleneck. While DFA aims to approximate the performance of standard BP, it is specifically designed to accommodate non-von Neumann architectures by significantly lowering computational and routing requirements [162]. However, while standard DFA successfully bypasses the weight transport problem inherent in BP, it still requires dense feedback connections. Consequently, as the network scales, routing the entire global error vector to every hidden neuron becomes a hardware bottleneck. To address this limitation, Crafton et al. [161,162] propose extensions of the DFA algorithm for Deep Neural Network (DNN), known as Sparse Direct Feedback Alignment (Sparse DFA) and Single-connection Sparse Direct Feedback Alignment (SSparse DFA). In these algorithms, the amount of information required to compute the error gradient at any individual neuron remains fixed,

regardless of network size. Unlike BP, which strictly requires sequential error propagation through all interconnected layers, DFA directly projects the network's output error to the hidden layers. Sparse DFA further optimizes this by applying a highly sparse feedback matrix, effectively eliminating the vast majority of error routing connections in the hardware realization. In SSparse DFA, this sparsity is maximized such that each neuron requires only a single scalar error signal to update its weights. Regarding network performance, Crafton et al. note that training deep CNN entirely with feedback alignment can result in accuracy degradation [161]. However, the authors demonstrate that by utilizing pre-trained convolutional layers as feature extractors, Sparse DFA and SSparse DFA can be used to train the fully connected layers while maintaining near-BP performance. Empirical results indicate that drastically increasing the sparsity of the feedback matrix has a minimal impact on accuracy across tested networks. Conversely, increasing the rank of the feedback matrix generally sustains or improves accuracy. As highlighted in reference [162], the transition away from standard von Neumann architectures requires technologies optimized for localized, in-memory computing, such as Resistive Random Access Memory (RRAM). These memristive crossbar arrays are highly compatible with the localized update rules of DFA and Sparse DFA, offering a highly energy-efficient and scalable alternative to the computationally expensive and memory-intensive BP algorithm.

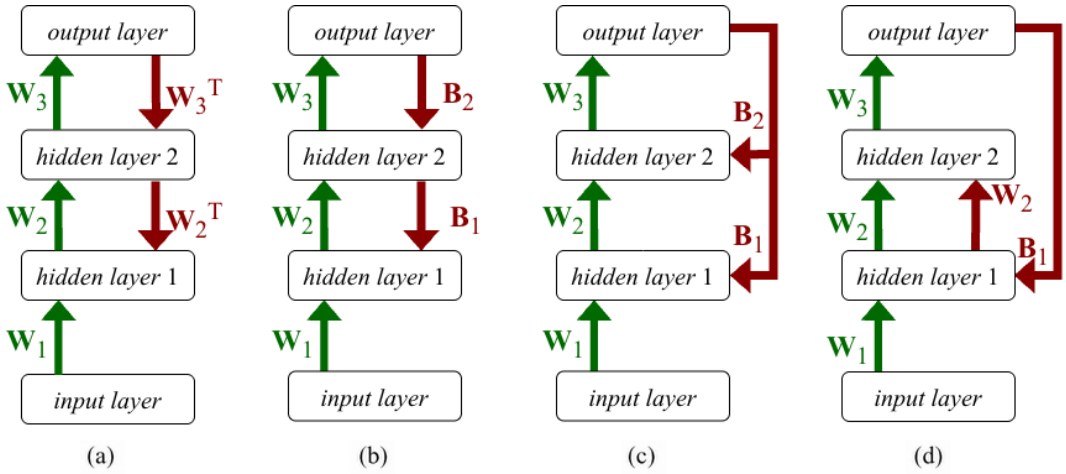


Figure 9. Comparison of different error transportation methods. The green arrows represent the activation paths, while the red arrows illustrate the error propagation. Trainable weights updated during the learning process are labeled as W_i, whereas B_i denotes the fixed and random weights. (**a**) Backpropagation, (**b**) Feedback Alignment, (**c**) Direct Feedback Alignment, (**d**) Indirect Feedback Alignment [160].

To establish baseline models, the authors in reference [163] initially train SNN offline employing methods such as Backpropagation Through Time (BPTT) and Surrogate Gradient Learning. However, implementing these networks on In-Memory Computing (IMC) architectures such as RRAM exposes them to unavoidable hardware variations, primarily because IMC relies on continuous, analog dot-product operations. To solve this problem, online adaptation of pre-trained models is necessary. The use of BPTT introduces additional delays through layer-wise gradient propagation across multiple timesteps. Furthermore, implementing BPTT structurally requires transposable memory crossbars, an addition that substantially inflates both the physical footprint and power consumption of the peripheral circuitry. For this reason Bhattacharjee et al. [163] propose using DFA for online adaptation on resource-constrained and non-ideal edge devices, which has shown better non-ideal accuracy compared to BPTT. The results show a 64.1% reduction in energy consumption,

a 10.1% decrease in area overhead, and a 2.1 times reduction in latency relative to BP, all while achieving higher accuracy.

Using an RRAM-based IMC architecture, a software-hardware co-design featuring a hardware-friendly training algorithm, Spiking Direct Feedback Alignment (SDFA), is proposed in reference [164], with the goal of accelerating SNN training. On the software side, sequential error propagation is eliminated, and on the hardware side, a three-level pipelined dataflow is implemented to parallelize the training process. The reported training accelerator demonstrates significant improvements in both performance and efficiency compared to traditional BP, specifically achieving up to a 10.5 times speedup and a 2.1 times reduction in energy consumption with less than 2% accuracy loss [164].

The first implementation of Direct Feedback Alignment Through Time (DFATT) for training shallow memristive spiking neural networks was recently presented by Samardzic and Dautovic [38]. The proposed architecture employs SPICE-based models of LIF neurons alongside memristive synapses, achieving average reduction in training time per epoch of approximately 50% compared to BPTT, while maintaining comparable accuracy. DFATT enables simultaneous training of all layers (Figure 8), reduces memory requirements by eliminating the need for weight transpose matrices, and allows direct training without surrogate gradient methods due to the availability of analog voltage signals. Furthermore, although the presented MSNNs were shallow, DFATT facilitates also layer-wise parallelization, providing additional acceleration in deeper neural networks.

2.6. Gradient Based Training for Memristive Convolutional Neural Networks

CNNs, as a widely used deep learning architecture for image-related tasks, can be significantly advanced through hardware implementations based on memristive devices. In such systems, memristive crossbar arrays enable highly parallel execution of convolution operations, while serving as tunable synaptic weights [26,165]. These architectures have been demonstrated to support both ex situ and in situ training under device constraints, exploit three-dimensional integration for scalable network design, and provide local storage of input data and intermediate results, improving computational efficiency and energy consumption [66,166]. Due to their widespread adoption and broad applicability, we dedicate a separate subsection to memristive implementations of CNN architectures trained using gradient-based methods.

Yakopcic et al. [165] first introduced the possibility of performing convolution operations using memristor circuits for CNNs trained ex situ, where the trained weights are programmed into memristive devices using a feedback write scheme [167]. The demonstrated CNN recognition system, trained using a gradient-based algorithm, can perform $N \times M$ convolution operations in parallel, where N denotes the number of input feature maps and M the number of output feature maps, using memristive devices with at least 16 conductance states. The same group later proposed a method for obtaining the entire output feature map in a single processing cycle, eliminating the need for intermediate digital storage between layers [168]. The approach stores convolutional kernels in memristive crossbars and expands them into large sparse matrices, enabling fully parallel dataflow. Training is performed ex situ, allowing arbitrary learning algorithms to be used for weight determination. The first implementation of deconvolutional layer forming complete Memristive Fully Convolutional Network (MFCN) architecture trained ex situ and validated through Simulink, was presented by Wen et al. [169]. The deconvolution kernel was represented through a sparse matrix, improving memristive crossbar utilization, whereas max-pooling layers were realized using voltage selector circuits, while the large number of weight parameters affecting computation speed remains untackled. Later on, Wen et al. [170] introduced the Convolution Kernel First Operated (CKFO) algorithm and applied it to the newly designed

simulated memristive CNN, as an improvement of the Caffe algorithm [171], aiming to reduce calculation after weight pruning. The CKFO algorithm replaces multiple dot products of matrices with a multiplicative operation of a matrix and a single kernel [170]. Reported advantages include reduction in sliding time with increased input size and straightforward application of pruned weight results without required modification of memristive convolution layers [170].

Extending the application of memristive CNNs, the pioneering use of memristors for video recognition was demonstrated in reference [166] through a simulation-based proof of concept employing 3D CNNs, whose additional temporal dimension captures correlations between consecutive frames. A voltage pulse–based training scheme was proposed to adjust network weights implemented using 3×3 memristor arrays organized into functional layers, including convolutional–pooling and Fully Connected (FC) layers, whereas memristor read and write error were not taken into account.

Moving toward hardware-efficient realization of memristive CNN based on MCA, Tan et al. [172] introduced a novel full-size convolutional algorithm (FSCA) with mini-batch SGD optimization. According to this method, the number of convolution operations is independent of the input matrix size, enabling a larger number of convolutions to be performed within a single computational flow. The input convolution operation in the FSCA applied to an input matrix X of size $m \times m$ can be described by the following equation:

$$Y(k,l) = \sum_{i=1}^{\lceil d/s \rceil} \sum_{j=1}^{\lceil d/s \rceil} \sum_{p=i+(i-1)s}^{m-d+1} \sum_{q=j+(j-1)s}^{m-d+1} X_{d\times d}(p : p+d-1, q : q+d-1) * K_{d\times d} \quad (8)$$

where K is the convolution kernel of size $d \times d$, s is the stride, indexes k and l in increments of step $d + s - 1$, $*$ is the convolution operation and $\lceil . \rceil$ is the ceiling function. The resulting approach requires $\lceil d/s \rceil^2$ operations to complete the computational flow while achieving directional $d \times d$ kernel convolution of size $\lceil (m-d+1)/(d+s-1) \rceil^2$ for an $m \times m$ input matrix with a $d \times d$ kernel, outperforming other algorithmic approaches such as Caffe [171] and CKFO [170]. A memristive crossbar composed of 1T3M cells for significant suppression of sneak-path currents achieves a computational performance of 1.23 TOPS, representing the most energy-efficient MCNN architecture with a power consumption of 2.41 mV for individual convolution operations.

A significant result in the hardware realization and training of a fully hardware-implemented memristive Convolutional Neural Network (mCNN) was demonstrated by Yao et al. [66], using backpropagation with SGD and a hybrid in situ–ex situ methodology to compensate for device imperfections. The network architecture consists of 2048 memristor cell arrays (128×16 1T1R) with 32 programmable conductance states, grouped into eight processing elements that together realize a five-layer mCNN. Ex situ training was first performed in Python, after which the trained weights of the convolutional and FC layers were quantized to 15-level fixed-point precision and mapped to corresponding memductance values. Afterwards, in situ training of the final FC layer was performed for fine-tuning, improving the network robustness to device non-idealities. This methodology is applicable to various neural network models and types of memristive devices, while exploiting parallelization through multiple MCA blocks to achieve low latency, and up to a 110× improvement in energy efficiency compared with the Tesla V100 GPU [66]. As another hardware approach, Lin et al. [173] constructed a hardware/software hybrid 3D CNN using eight layers of monolithic integrated memristor circuits, allowing pixel-wise parallel convolutional kernel operations. The extension of classical CNN architectures to a third dimension is motivated by the need to reduce the area overhead of parallel kernel implementations in conventional 2D arrays and to support processing of dynamic data, such as edge detection in video frames. Replicated kernels are placed within a single array

organized into row banks, composed of memristors whose conductance values define the weights of the convolutional kernels. Following offline training, the obtained 3 × 3 kernels were programmed into the 3D arrays, enabling parallel kernel operations in hardware, while the remaining network operations were performed in software. The system achieved performance comparable to a purely software implementation on the MNIST dataset [107] and demonstrated the capability for parallel video processing [173].

Hardware-aware training was also applied to gradient-based memristive CNNs, as presented in reference [27], where HWA methodology was adopted to ensure the reliable deployment of neural networks on analog PCM in-memory computing cores. Prior to weight mapping, networks were trained in software using floating-point number representation while incorporating device-aware constraints such as weight clipping and controlled noise injection to emulate PCM variability and non-idealities. For both a two-layer MLP on MNIST and a ResNet-9 on CIFAR-10, the models were first conventionally trained and subsequently retrained with injected weight noise and progressive clipping, thereby improving robustness against analog imperfections. This custom retraining strategy enables accurate transfer of digital weights to hardware, achieving 98.6% accuracy for the MLP and 88.4% for ResNet-9 after mapping.

A notable paper on hardware algorithm co-optimization for CNNs presented the NeuRRAM CIM chip [26], which integrates 48 cores and 3 million RRAM cells Transposable Neurosynaptic Array (TNSA), Figure 10. The presented system enables reconfigurable hardware for different network architectures and supports parallel inference through various weight-mapping methodologies. Reported CNNs were initially trained using Keras, after which obtained parameters were mapped on NeurRAM arrays, where mapping strategies differ depending on the size of the convolutional layers, followed by the fine-tuning chip-in-the-loop technique, one layer at a time for improved classification accuracy. The NeuRRAM chip also supports LSTM and Restricted Boltzmann Machine (RBM) architectures, achieving inference accuracies comparable to software implementations while attaining the lowest energy–delay product (EDP) reported at the time [26].

A comparative summary of the gradient-based algorithms reviewed in this section is provided in Table 1, covering both memristive NNs and SNNs, together with their synaptic circuits, architectures, hardware implementations, datasets and corresponding accuracies. Within this table, hardware implementation refers to the use of actual physical hardware, such as memristive crossbar arrays, chips, and peripheral circuitry, to implement part or all of the neural network, rather than hardware-level simulations.

Table 1. Gradient-based algorithms in MNNs and SNNs.

Reference	Algorithm	Synaptic Circuit	Architecture	HW	MNN/SNN	Dataset
[36]	BP with gradient accumulation (BP-GA)	1M	SLP, MLP, CNN	NO	MNN	MNIST (96.7%), CIFAR-10 (58.8%)
[119]	BP with SGD	1T1R	MLP	memristive crossbar (128 × 64)	MNN	MNIST (91.71%)
[134]	hardware aware BP with ABC optimization	1M	SLP, MLP	HW level simulations	MNN	IRIS (76%)
[121]	BP with RMSprop	1T1R	MLP	hardware-based neurons and memristive crossbar for inference	MNN	MNIST (93.63%)

Table 1. *Cont.*

Reference	Algorithm	Synaptic Circuit	Architecture	HW	MNN/SNN	Dataset
[136]	Sign-BP	1T1R	MLP	HW level simulations	MNN	MNIST (94.5%)
[112]	BP	1M	MLP	NO	MNN	MNIST (96.8%)
[30]	BP	2M	MLP	NO	MNN	MNIST (98.53% for batch in situ, 95.94% for stochastic in situ)
[73]	BP	1T1R	MLP	crossbar with PCM devices	MNN	subset of MNIST (82.9%)
[32]	Manhattan update/Delta rule	2M	SLP	transistor-free memristor crossbar (12 × 12)	MNN	3 × 3 binary image classification
[159]	Contrastive divergence (CD)	1M	Memristive Deep Belief Net (DBM) composed of restricted Boltzmann machines (RBM)	HW level simulations	MNN	MNIST (95–97%)
[158]	Contrastive divergence (CD)	2M	Restricted Boltzmann machines (RBM)	HW level simulations	MNN	MNIST (87.85%)
[108]	Online gradient descent	1M2T	MLP, SLP	HW level simulations	MNN and SNN	IRIS (97.2%), Wisconsin Breast Cancer Dataset (98.5%)
[103]	Modified chip-on-the-loop learning	4M memristor bridge synapse	MLP	HW level simulations	MNN	three-bit parity, car detection
[104]	BP with pulse width weight update	2M	MLP	HW level simulations	MNN	three-input odd parity (100%), WBCD (92–93%)
[120]	BP	1M	SLP, MLP	HW level simulations	MNN	NASA Asteroid (90.43%), WBCD (98.59%), IRIS (98.22%), MNIST (91.27%)
[113]	Online gradient descent	1T1R	MLP	1T1R array for inference	MNN	MNIST (82.82%)
[109]	Online gradient descent	2T1R	SLP, MLP	HW level simulations	MNN	WDBC (97.0%), Wine (96.25%), IRIS (84.33%)
[138]	BP	Discrete memristive synapse	three-layer DM-BPNN	Simulations only	MNN	MNIST (97.16%)
[123]	BP with SGD	1T1M	SLP, MLP	HW level simulations	MNN	OR/AND logic, MNIST (96.47%)
[131]	Online gradient descent	2M	SLP, MLNN	HW level simulations	MNN	WBCD (90.14%), IRIS (99.11%), NASA Asteroid (89.04%)
[68]	Adaptive BP	1M	SLP, MLP	HW level simulations	MNN	digit/pattern recognition (97.4% for 30% noise level)
[96]	Widrow—Hoff (LMS)	2M	SLP (SNU), MLP	Single Neuron Unit	MNN	3-input logic functions, MNIST (88% Set Reset mode)

Table 1. *Cont.*

Reference	Algorithm	Synaptic Circuit	Architecture	HW	MNN/SNN	Dataset
[98]	Widrow—Hoff (LMS)	4T2M	SLP	HW level simulations	MNN	three-character recognition
[111]	Analog BP	1M	MLP, DNN, BNN, HTM, LSTM	HW level simulations	MNN	XOR, MNIST (92–93%), Yale face recognition (73.3–78.9%)
[110]	Analog BP	1M	MLP	HW level simulations	MNN	N/A
[174]	BP gradient descent	1T1R	MLP	HW level simulations	MNN	MAGIC Gamma Telescope (86.57%), Page Blocks (96.35%), Ecoli (83.58%), Vehicle Silhouettes (83.43%), Wall-Following Robot Navigation (88.63%)
[122]	BP	1M	MLP, CNN	HW level simulations	MNN	XOR, MNIST (MLP: 96.23%, CNN: 98.02%), FMNIST (MLP: 84.07%, CNN: 84.43%), CIFAR-10 (CNN: 50.12%)
[33]	Delta rule, Manhattan rule	1T1R	SLP	1024 RRAM array	MNN	Yale Face Database (85.04%)
[142]	BPTT (SPICEprop)	1M	fully memristive SNN (MIF output)	Simulations only	SNN	MNIST (97.58%), Fashion-MNIST (75.26%)
[143]	BPTT (MEMprop)	1M	fully memristive SNN (MIF hidden/output)	NO	SNN	MNIST (93.18%), Fashion-MNIST (84.79%), DVS-128 Gesture (83.21%)
[38]	BPTT / DFATT	1M	fully memristive SNN (MIF hidden/output)	NO	SNN	MNIST (BPTT:98.7%, DFATT: 98%), Fashion-MNIST (BPTT: 92.8%, DFATT: 92.05%)
[150]	BPTT	1M	memristor-based LSTM (MbLSTM)	HW level simulations	MNN	MNIST (92.9%), IMDB (87.5%)
[147]	BPTT	1M / 1T1R	LSTM	HW level simulations	MNN	airline passengers, wafer quality, human gait
[151]	BPTT with SGD	RPU	LSTM	HW level simulations	MNN	War and Peace, Linux Kernel
[156]	BPTT with SGDM	1T1R	LSTM/RNN	1T1R memristor crossbar (128 × 64)	MNN	airline passengers, human gait
[154]	BPTT	1M	LSTM	HW level simulations	MNN	N/A
[153]	BPTT	1M	LSTM	HW level simulations	MNN	time-series forecasting
[155]	BPTT	1M	LSTM, MLP, DNN, HTM	HW level simulations	MNN	wafer quality inspection dataset (98.51%)

Table 1. *Cont.*

Reference	Algorithm	Synaptic Circuit	Architecture	HW	MNN/SNN	Dataset
[175]	BPTT	1M	LSTM	HW level simulations	MNN	IMDB (84.3%), SemEval Twitter dataset (65.8%)
[66]	Hybrid in situ/ex situ BP-SGD	1T1R	5-layer memristor CNN	fully HW implementation of memristive CNN	MNN	MNIST (96.19%)
[166]	BP	1M	3D CNN	NO	MNN	Weizmann video dataset (>70%)
[173]	BP	1T1R	CNN	3D memristor array for convolution	MNN	MNIST (98.10%)
[169]	BP	1M	fully convolutional network (FCN)	HW level simulations	MNN	Weizmann horse dataset (97.56%), LFW
[165]	BP	1M	CNN	HW level simulations	MNN	MNIST (91.8%)
[168]	BP	1M	CNN	HW level simulations	MNN	MNIST (98.92%)
[172]	Full size convolutional algorithm (FSCA) with mini-batch SGD	1T3M	MLP	HW level simulations	MNN	MNIST (98.25%)
[170]	CKFO algorithm	1M	CNN/LeNet-style	HW level simulation	MNN	MNIST (98.43%)
[26]	hardware–algorithm co-design	1T1R	CNN, LSTM	RRAM based CIM chip	MNN	MNIST (99%), CIFAR-10 (85.7%), Google Speech Commands (84.7%), image reconstruction (70% error reduction)
[25]	mixed-precision BP	1T1R	LeNet-5, VGG-8, ResNet-18	CIM SoC	MNN	MNIST (97.73%)
[64]	mixed-precision BP with SGD	PCM differential pair	MLP, CNN, LSTM, GAN	PCM chip (MLP)	MNN	MNIST (97.73%)
[31]	BP stochastic update rule	RPU device	MLP	HW level simulations	MNN	MNIST (≈98%)
[34]	BP	2PCM + 3T1C	MLP	PCM arrays	MNN	MNIST (97.95%), MNIST-backrand (82.13%), CIFAR-10 (88.29%), CIFAR-100 (67.96%)
[67]	BP	2M	MLP	HW level simulation	MNN	MNIST (92.4%)
[37]	hardware-aware BP/EaPU	1T1R	Autoencoder, ResNet, SRNet, Swin Transformer	1T1R memristor crossbar	MNN	MNIST denoising (SSIM 0.896), SRNet ×2 super-resolution (SSIM 0.933)
[135]	stochastic BP	2M	MLNN	HW level simulations	MNN	XOR, odd parity, IRIS

Table 1. *Cont.*

Reference	Algorithm	Synaptic Circuit	Architecture	HW	MNN/SNN	Dataset
[46]	BPTT with surrogate gradient	1T1R	recurrent SNN	memristor crossbar	SNN	ECG anomaly detection (91.8%), SHD keyword spotting (70.3%), motor control (gained reward: 723.86)
[99]	batch gradient descent	2M	SLP	fully integrated chip with passive 54 × 108 memristor crossbar, interface circuitry and OpenRISC processor	MNN	5 × 5 Greek-letter classification (100%)
[133]	nonideality-aware BP	2M	MLP	Simulations only	MNN	MNIST (98%), FMNIST (88%)
[115]	Offline BP with nonlinear conjugate-gradient optimization	2M	Inverter-based MNN	HW-level simulations	MNN	/
[114]	Physical Characteristics-Aware ex situ BP (PHAX)	2M	MLP	HW-level simulations	MNN	MNIST (100%), BCW (98%), IRIS (100%)
[132]	Device-aware conventional BP	2M	MLP	memristive chip, peripheral circuitry, simulation of ANN	MNN	nonlinear classification problem/neuronal activity simulated in cultured hippocampal network
[163]	DFA	1T1R (4-bit RRAM devices, NeuRRAM)	multilayer SNN	HW level simulations based on experimental data from NeuRRAM [26]	SNN	FMNIST (85.12%), UCI-HAR (85.61%), HHAR (82.55%)
[72]	Perceptron learning rule	2M	SLP	memristive crossbar	MNN	binary X/T pattern classification
[164]	SDFA	1T1R	PipeSFDA, MLP, CNN	HW level simulations based on experimental data from NeuRRAM [26]	SNN	N-MNIST (97.97%), SHD (73.29%), Braille letter (99.26%), DVS-Gesture (93.75%), N-Caltech101 (64.38%)
[116]	BP with Tiki-Taka	RPU	FCN, LeNet-5, LSTM	Simulations only	MNN	MNIST (≈98%), War and Peace dataset (test cross entropy loss ≈ 2)
[117]	BP with Tiki-Taka v2	RPU	FCN, LeNet-5, LSTM	Simulations only	MNN	War and Peace dataset (test cross entropy loss ≈ 1.5)
[118]	BP with TT and TTv2	1R/1T1R	LSTM	HW level simulations	MNN	War and Peace dataset (test cross entropy loss ≈ 1.55)

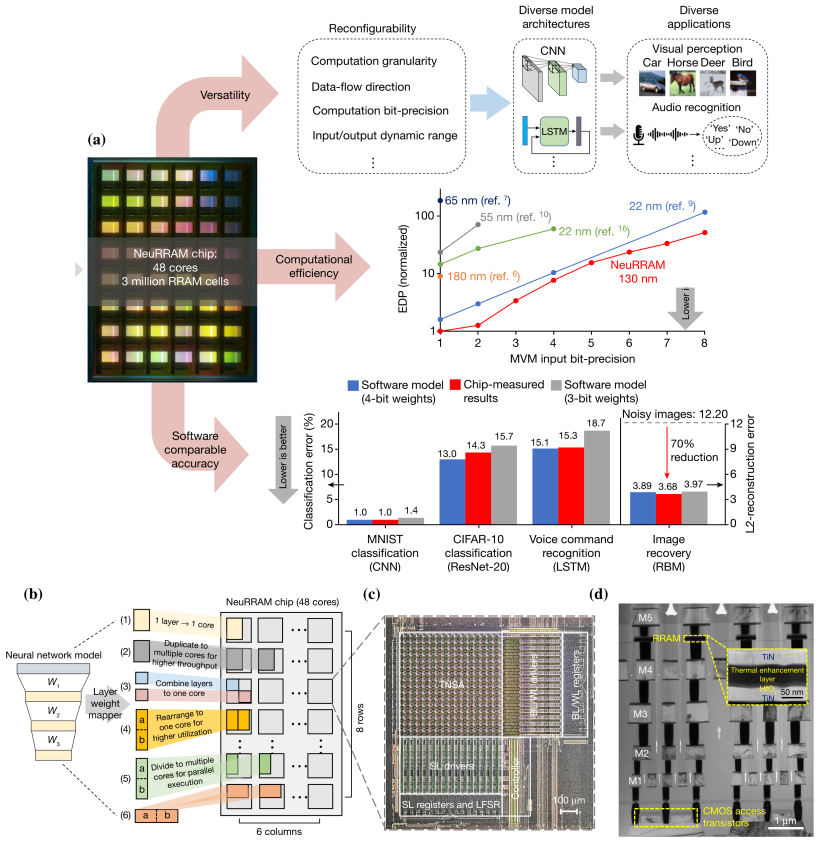


Figure 10. NeurRAM chip. (**a**) Chip overview including cross layer design optimization, reconfiguration, energy–delay benchmarking and hardware inference accuracy comparable to software models with 4-bit weights. (**b**) Multi-core architecture. (**c**) Micrograph of CIM core. (**d**) Cross-section TEM image of integrated RRAM CMOS layers adapted with permission under the Creative Commons Attribution (CC BY) license from reference [26].

3. Gradient-Free Algorithms

Gradient-free learning methods in memristive neural networks can be broadly categorized into stochastic perturbation-based approaches, evolutionary algorithms, forward-only training paradigms, and reservoir computing methods. These categories reflect distinct strategies for avoiding explicit gradient computation while enabling efficient in situ learning under the non-ideal and often stochastic characteristics of memristive devices. In contrast to conventional backpropagation, these methods aim to reduce computational overhead, relax precision requirements, and better align with the physical properties of emerging hardware. Reservoir computing follows a similar motivation, relying on fixed dynamical systems combined with trained readout layers, and has also been explored in the context of memristive implementations. The topic is covered through four subsections, addressing stochastic and perturbation-based methods, evolutionary and nature-inspired algorithms, forward-only training methods, and reservoir computing approaches. The classification is mainly based on the underlying gradient-free learning mechanism, while reservoir computing is considered separately in the final subsection due to its distinct training paradigm, in which the reservoir dynamics are typically fixed, and learning is primarily restricted to the readout layer.

3.1. Stochastic and Perturbation-Based Methods

These methods approximate the update direction using stochastic sampling or small perturbations of conductance values, thereby avoiding explicit gradient evaluation. Learn-

ing is typically guided by probabilistic rules or performance-based feedback, which simplifies hardware implementation and reduces sensitivity to device non-idealities.

Zhang et al. [176] introduced a probabilistic update scheme for memristor-based CNNs, in which multiple operating modes balance accuracy and hardware cost, while eliminating the need for precise pulse control. Similarly, [177] employed a model-free perturbation strategy in a robotic control setting, where system performance is evaluated under small conductance variations to iteratively guide learning. These approaches significantly reduce programming complexity and enable efficient, low-overhead hardware implementations while maintaining competitive performance.

3.2. Evolutionary and Nature-Inspired Methods

Evolutionary approaches rely on population-based search mechanisms, such as mutation, crossover, and selection, which can be naturally mapped to stochastic switching behavior in memristive devices. These methods completely avoid gradient information and instead explore the solution space through iterative refinement of candidate solutions.

Several works have demonstrated the suitability of memristive hardware for implementing such algorithms. In reference [178], a Memristive Network-based Genetic Algorithm (MNGA) was proposed, where the structure of the network is directly mapped to the population and evolutionary operations, enabling efficient image edge detection. In reference [179], stochastic stateful logic was used to realize mutation and crossover operations through inherent switching variability, significantly reducing circuit complexity and enabling compact binary-weight implementations. Furthermore, [180] presented a brain-inspired architecture that integrates evolutionary processes, including parent selection and offspring generation, directly within memristive arrays, enabling continuous online adaptation and fault tolerance. Simulated annealing-based optimization was also explored in reference [181], where the method mitigates the impact of hardware imperfections and improves accuracy across multiple neural network architectures. Despite these advantages, comparative analysis in reference [182] indicates that evolutionary approaches do not universally outperform alternative methods, as simpler local update rules may exhibit greater robustness in the presence of permanent device degradation.

3.3. Forward-Only Training Methods

Forward-only methods eliminate the need for backward signal propagation by using layer-wise or local objectives, enabling training with strictly forward passes. This paradigm is particularly well suited to memristive hardware due to its compatibility with in-memory computation, reduced data movement, and simplified control logic.

Recent work has adapted the Forward-Forward algorithm to memristive systems, demonstrating the feasibility of fully in-memory learning under realistic hardware constraints. In reference [183], the algorithm was modified to operate on RRAM devices, including co-optimization of programming conditions to achieve noise characteristics compatible with forward-only updates. The approach was validated on real-world biomedical data, showing robustness to device variability. Extending this line of work, [184] demonstrated energy-efficient on-chip learning using low-voltage, single-pulse updates, significantly reducing power consumption while maintaining stable analog states. The method achieved performance comparable to backpropagation on complex tasks, indicating that forward-only training represents a promising direction for scalable and hardware-aligned learning in memristive systems.

Table 2 summarizes representative gradient-free, local, and reservoir-based learning approaches reported for memristive systems, covering local update rules, stochastic-adaptive learning, evolutionary strategies, forward-only learning, and reservoir computing. For

each work, it compares the learning algorithm, training scope, gradient use, memristive implementation, network architecture, implementation level, and evaluation dataset to highlight the main design trade-offs.

3.4. Reservoir Computing Methods

Reservoir computing methods employ a fixed dynamical system to transform inputs into a high-dimensional representation, while only a readout layer is trained. In memristive implementations, device dynamics and variability can be utilized to realize such reservoirs efficiently, enabling low-cost training and reduced hardware complexity.

The first experimental memristor-based Reservoir Computing (RC) was demonstrated in [185]. This work also illustrates the operation of a typical RC system and will therefore be described in greater detail. In that study, reservoir computing with memristors is organized as shown in Figure 11, where a dynamic memristor reservoir transforms temporal inputs into internal states that are subsequently processed by a readout layer. The architecture comprises a reservoir with intrinsic dynamics and a readout function, in which only the weight matrix Θ, connecting the reservoir state $x(t)$ to the output $y(t)$, is trained. Temporal input signals are applied as pulse streams to the memristor array, whose intrinsic ionic dynamics give rise to distinct resistance states that encode the reservoir states. A readout network then performs the final computation based on these states.

The study experimentally demonstrates that dynamic memristors can directly process temporal information due to their inherent short-term memory effects. During training, only the readout layer is optimized, which significantly reduces the training complexity compared to conventional recurrent neural networks. The reservoir states are obtained from the resistance values of the memristors after applying specific pulse sequences corresponding to the input data. These states are then used to train the readout weights for classification or prediction tasks. The proposed system is evaluated on handwritten digit recognition using 5 × 4 pixel images and on a second-order nonlinear dynamic task. The results show that even a relatively small reservoir composed of memristor devices can achieve effective learning and temporal signal processing.

Memristor-based RC exploits the inherent nonlinear dynamics of the devices to enable efficient temporal processing with reduced training overhead. In this paradigm, the reservoir is generally left untrained, while learning is restricted to the readout layer. The training procedure is most commonly implemented using linear or ridge regression, including the Moore–Penrose pseudo-inverse or L_2-regularized methods, to optimize the readout weights. Such regression-based training forms the foundation of most practical implementations, and has been demonstrated across a wide range of device technologies, architectures, and application domains.

From a device and material perspective, several works explore how intrinsic physical properties shape reservoir behavior. For instance, [185] employs Tungsten Oxide (WO_x) volatile memristors where ionic dynamics constitute the reservoir states, while [186] uses diffusive silver memristors combined with a nonvolatile crossbar readout. Other material systems include $Pt/HfO_2/TaO_x/TiN$ RRAM devices with 4-bit input encoding in reference [187], molecular-scale organic memristors in reference [188], halide perovskite memristors in reference [189], ZnO-based architectures in reference [190], solution-processed perovskites with large dynamic range in reference [191], $Ag@TiO_2$ nanowires in reference [192], $Pt/Ta_2O_5/HfO_2/TiN$ bilayer devices in reference [193], Al-doped HfO_2 ferroelectric memristors in reference [194], Bi_2Se_3 networks in reference [195], and discrete memristor models with chaotic maps in reference [196].

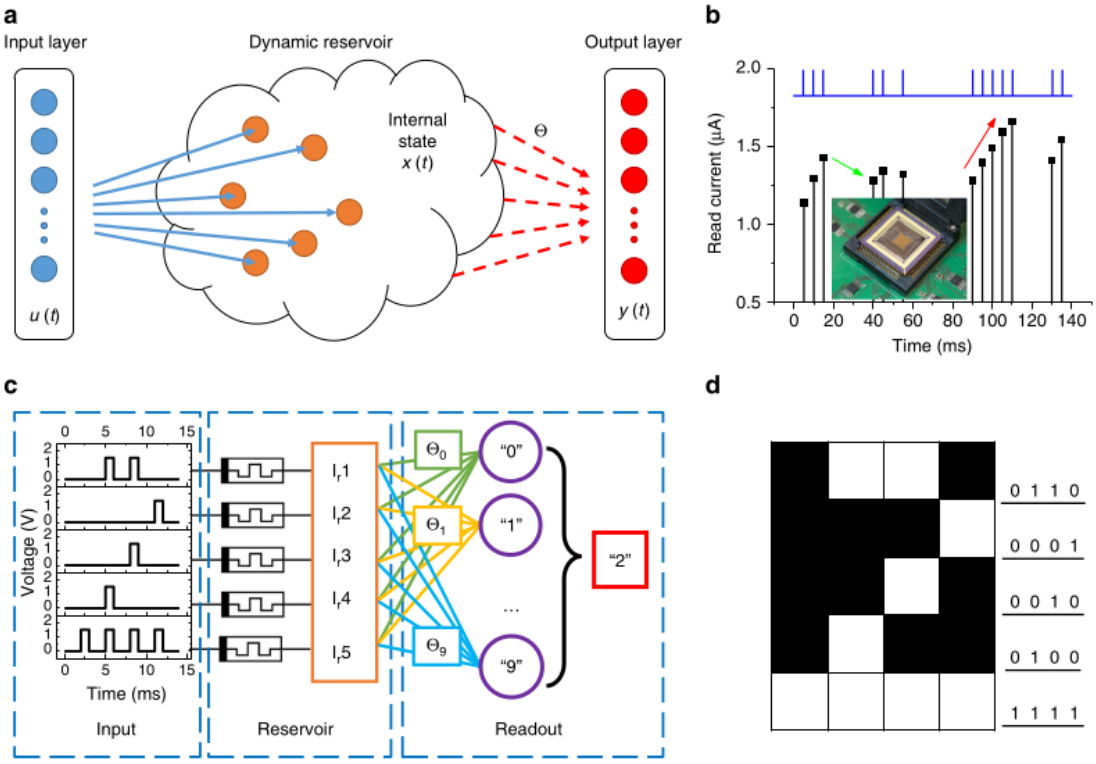


Figure 11. Reservoir computing system based on a memristor array. (**a**) Schematic of a dynamical reservoir and its trainable readout layer. (**b**) Response of a tungsten-oxide memristor to pulse trains with different time intervals; the inset shows the memristor array mounted on a test board. (**c**) Memristor-based reservoir and readout network for classifying five-by-four-pixel digit images using five memristors. (**d**) Example of the digit 2 and its binary input encoding. The ellipses in panels (**a**,**c**) indicate omitted output neurons rather than missing figure content; in panel (**c**), the omitted neurons correspond to digit classes 2–8. Adapted from reference [185] under the Creative Commons Attribution (CC BY) license.

From an architectural and system design perspective, multiple approaches focus on structuring the reservoir to enhance computational richness. Examples include the doubly twisted toroidal topology exploiting transistor mismatch for ultra-low-power biosignal processing in reference [197], virtual-node time-multiplexing in reference [198], parallel arrays with tunable mask lengths in reference [199], quasi-3D stacked nanowire networks in reference [200], hierarchical processing using α-In_2Se_3 Field-effect Transistor (FET) devices in reference [201], nanoparticle random networks in reference [202], and single-device implementations enabling Histogram of Oriented Gradients (HOG)-based image recognition in reference [203]. Additionally, fully integrated realizations are demonstrated in reference [204], where both reservoir and readout are implemented within a single device structure.

From an application-driven perspective, several works highlight domain-specific use cases enabled by regression-based training. These include in-sensor computing with tin monosulfide memristors in reference [205], chaotic time-series prediction across multiple architectures in reference [206], parallel fingerprint recognition using photoelectronic memristors in reference [207], ECG diagnosis via light-modulated reservoirs in reference [208], and robotic navigation using memristive networks in reference [195]. Furthermore, in-materia computing is explored in reference [209], where a self-organized silver nanowire network directly serves as the reservoir.

Beyond regression-based methods, a variety of iterative, online, and adaptive training strategies have been proposed to address hardware constraints and improve convergence.

From an optimization and evolutionary perspective, reference [210] utilizes a Genetic Algorithm (GA) for readout optimization.

From an online and incremental learning perspective, reference [211] employs a LMS algorithm, while reference [212] adopts recursive least squares (RLS) for faster convergence. Similarly, reference [213] and reference [214] implement on-chip training via supervised STDP and online learning mechanisms.

From a robustness and hardware-aware adaptation perspective, reference [215] addresses variability and noise through adaptive synaptic reprogramming, while reference [216] introduces an accelerator with in situ learning for edge deployment. The role of stochasticity is explored in reference [217] via an Extended Synaptic Sampling Machine (ESSM), and heterogeneous dynamics are leveraged in reference [218] to enhance state richness. Input preprocessing is examined in reference [219], showing that parity-based encoding stabilizes readout optimization, while reference [220] demonstrates that tunable conductance timescales in iontronic memristors directly influence training effectiveness.

Finally, reference [221] demonstrates high-accuracy digit recognition using multi-epoch readout training, and reference [222] emphasizes the importance of network topology, showing that scale-free and heterogeneous structures generate more diverse reservoir states, thereby simplifying the learning task for the readout layer.

Overall, reservoir computing represents a particularly suitable approach for memristive systems because the nonlinear dynamics and temporal properties of memristive devices can be exploited without training the reservoir itself. As summarized in Table 2, most approaches restrict training to the readout layer, using methods ranging from closed-form linear and ridge regression to gradient-based LMS, SGD, and Adam optimization. This substantially reduces the training complexity of the memristive part of the system compared with fully trainable neural networks.

Table 2. Alternative learning and reservoir-computing approaches in memristive systems and their key properties.

Ref.	Algorithm	Training Scope	Gradient Use	Memristive Implementation	Architecture	Impl.	Dataset/Task
Gradient-free and evolutionary approaches							
[177]	Binary search (Kalman gain) and random search (PD conductances)	Controller parameters	None	1R	Hybrid analog–digital Kalman filter and PD controller	HW	Mobile inverted-pendulum control data
[178]	Genetic algorithm	Global	None	1R	Memristive-network genetic algorithm for image-edge detection	Sim.	Gray-scale images (Lena, Pepper, etc.)
[179]	Evolutionary learning/genetic algorithm	Network weights	None	1TnM	Binary neural network on a 784 × 100 virtual memristor crossbar	Hybrid	MNIST
[180]	Evolutionary programming	Global	None	1T1R	Brain-inspired evolvable architecture with memristive arrays	Sim.	NARMA20
[182]	Genetic algorithm; Manhattan comparator	Network weights	None (GA); gradient-derived sign (Manhattan)	1R differential pair	Single-layer perceptron with 185 weights (370 memristors)	Sim.	6 × 46 letter images, 5 classes
Mixed evolutionary/gradient-based hardware-update approaches							
[181]	GA outer loop with quantization-aware BP	Network weights	Full BP gradient in fitness loop	Unspecified	ProtoNet, MobileNetV2, and MicroNet	Sim.	MNIST, Fashion-MNIST, CIFAR-10, CIFAR-100

Table 2. *Cont.*

Ref.	Algorithm	Training Scope	Gradient Use	Memristive Implementation	Architecture	Impl.	Dataset/Task
[176]	Stochastic-adaptive pulse update from SGD	Network weights	Full BP gradient direction	1T1R differential pair and 1T1R–1R	LeNet-5 CNN (2 convolutional and 3 fully connected layers)	Hybrid	MNIST
Local/backpropagation-avoiding approaches							
[72]	Perceptron learning rule	Output weights	Local error-driven	1R differential pair	Single-layer perceptron (9 inputs plus bias, 1 output)	Hybrid	34 × 3 binary images (X and T patterns)
[183]	Forward–Forward	Local layer weights	Local gradient	1R differential pair	FF network (37, 74, and 100 neurons; 5 clusters ×9)	Hybrid	ECG heart-beat/arrhythmia classification
[184]	Forward-only learning (SFF, CF)	Local layer weights	Local gradient	1T1R differential pairs; separate 2T1R MAC	Two-layer MLP with 48 hidden units	Hybrid	Four-class ImageNet-1k subset with pretrained ResNet-18 features
Reservoir-computing approaches							
[210]	Genetic algorithm for readout weights	Readout only	None	1R random network	SPICE-simulated graph reservoir (6–40 nodes) with linear readout	Sim.	Triangular- and square-wave discrimination; memory tasks
[197]	Normal-equation regression; stochastic LMS	Readout only	Closed-form; readout gradient	Differential 2R weights	Doubly twisted toroidal ESN with memristor crossbar readout	Sim.	EEG seizure and EMG finger-motion data
[185]	Logistic and linear regression	Readout only	Readout gradient	1R volatile reservoir	WO_x dynamic-memristor array with software readout	Hybrid	Digit recognition and second-order nonlinear dynamics
[211]	Online LMS	Readout only	Readout gradient	Memristive synaptic circuit	Newman–Watts small-world MESN with memristive LMS readout	Sim.	UCI Car Evaluation; EUNITE power-load forecasting
[186]	RMSProp training with one-shot blind updates	Readout only	Readout gradient (RMSProp)	1R reservoir; 1T1R differential readout	Diffusive-Ag reservoir with 110 × 20 drift-memristor readout	Hybrid	MNIST
[198]	Linear readout training	Readout only	Not specified	1R volatile WO_x reservoir	Time-multiplexed WO_x reservoir with virtual nodes and linear readout	Hybrid	TI-46 spoken digits; Mackey–Glass forecasting
[213]	STDP-based readout training	Readout only	Local rule (STDP)	Memristor crossbar	Circuit-level memristive reservoir and readout architecture	Sim.	Wisconsin Breast Cancer
[187]	Reservoir-state separation only	None	None	1R self-rectifying	Single Pt/HfO_2/TaO_x/TiN volatile device with 4-bit pulse encoding	N/A	All 16 four-bit pulse sequences
[205]	Minibatch Adam softmax regression	Readout only	Readout gradient (Adam)	1R optoelectronic reservoir	SnS in-sensor reservoir with software softmax readout	Hybrid	Five spoken and written sentence patterns
[199]	Moore–Penrose linear regression	Readout only	Closed-form	1R volatile reservoir	Serially emulated parallel dynamic-memristor reservoirs with software readout	Hybrid	TI-46 spoken digits and Hénon map
[200]	Linear regression	Readout only	Closed-form	Memristive tunnel-junction network	Quasi-3D nanowire-network reservoir	Sim.	Memory-capacity and nonlinear-transformation benchmarks
[206]	Readout fitting (method not specified)	Readout only	Not specified	Four volatile memristor models	Two modeled memristive reservoir architectures	Sim.	Logistic- and Hénon-map prediction

Table 2. *Cont.*

Ref.	Algorithm	Training Scope	Gradient Use	Memristive Implementation	Architecture	Impl.	Dataset/Task
[189]	Delta-rule readout	Readout only	Readout gradient	1R differential pair	Volatile virtual reservoir and nonvolatile readout	Hybrid	Four neural firing patterns
[201]	Linear regression	Readout only	Not specified	α-In_2Se_3 FeSFET and planar ferroelectric device	Cascaded physical reservoir with virtual nodes	Hybrid	Brazilian energy use and sine and square waveforms
[209]	Offline Adam readout; linear regression for forecasting	Readout only	Readout gradient (Adam); regression solver not specified	Ag nanowire reservoir; TaO_x ReRAM readout	Physical in-materia nanowire reservoir with memristive readout	Hybrid	Spatio-temporal pattern classification; MNIST; Mackey–Glass
[190]	Linear regression	Readout only	Not specified	ZnO memristor model	Modeled ZnO reservoir with added 35-neuron classifier	Sim.	MNIST
[191]	Linear-regression readout	Readout only	Not specified	1R volatile reservoir	Seven perovskite devices with virtual nodes	Hybrid	Fashion-MNIST
[188]	Multinomial logistic readout	Readout only	Not specified	1R volatile molecular reservoir	100-device reservoir with 100 × 3 readout; 14-state device experiment separate	Hybrid	Three-class emoticons; MNIST
[207]	Offline softmax or logistic readout training	Readout only	Readout gradient	Photo-synapse reservoir; differential 1R crossbar readout	DUV in-sensor reservoir with memristor-array readout	Hybrid	FVC2002 latent fingerprints
[69]	Noise-aware linear regression/pseudo-inverse	Readout only	Closed-form	1R reservoir; NVM crossbar readout	24 parallel Ti/TiO_x/Pd nodes with analog readout	Hybrid	MIT-BIH arrhythmia and DVS128 gestures
[223]	Supervised STDP readout	Readout only	Local rule (STDP)	Five volatile PANI devices; PPX synapses	Physical PANI reservoir with software-modeled spiking PPX readout	Hybrid	3 × 3 patterns and 5 × 3 digit patterns
[224]	Adam-trained software readout	Readout only	Readout gradient (Adam)	1R volatile PANI device	Experiment-based PANI temporal-reservoir model	Hybrid	MNIST
[204]	Logistic and linear regression	Readout only	Readout gradient; closed-form	Volatile and nonvolatile ferroelectric diodes	All-ferroelectric reservoir and FC readout	Hybrid	Curvature, digits, waveforms, and Hénon map
[222]	Moore–Penrose linear regression	Readout only	Closed-form	Memristive tunnel-gap network	Scale-free and heterogeneous nanoparticle reservoirs	Sim.	NARMA2 and NARMA10, memory, nonlinear transformation, waveforms
[217]	Adam optimization with stochastic synaptic sampling	Readout only	Readout gradient (Adam)	Memristive stochastic logic	ESSM echo-state reservoir with dense softmax readout	Sim.	ECG, MNIST, Fashion-MNIST, CIFAR-10
[215]	Gradient descent and STDP	Readout plus optional reservoir STDP	Readout gradient; local STDP	HfO_2 1T1R and differential crossbar	Spiking recurrent reservoir with memristive dot-product readout	Sim.	Mackey–Glass, Lorenz, and IRIS classification
[202]	Ridge regression	Readout only	Not specified	Nanoparticle random-network memristor	In-material random network with multiple electrode outputs	Hybrid	Waveform transformation, NARMA2, and memory capacity
[218]	Linear regression; two-layer trained readout	Readout only	Closed-form; readout gradient	Ion-channel 1R devices	Small heterogeneous reservoir of 3–5 dynamic memristors	Hybrid	Second-order dynamics; generated neural-activity classes

Table 2. *Cont.*

Ref.	Algorithm	Training Scope	Gradient Use	Memristive Implementation	Architecture	Impl.	Dataset/Task
[193]	Adam-optimized readout	Readout only	Readout gradient (Adam)	$Pt/Ta_2O_5/HfO_2$/TiN second-order memristor	Heterogeneous device-response reservoir with trained readout	Hybrid	MNIST
[214]	Online in situ readout learning	Readout only	Not specified	Memristive synapses	SPICE-level memristor liquid-state machine	Sim.	MNIST, TI-46, drone audio, and MIDI
[194]	Multi-epoch MLP training	Readout only	Not specified	1R MFIS device	196 device -derived state channels feeding a 100–10 MLP	Hybrid	MNIST
[192]	Linear regression	Readout only	Not specified	Ag@TiO_2 nanowire network	Nanowire-network reservoir with linear readout	Hybrid	Sine and square waveform classification
[203]	Ridge regression	Readout only	Closed-form	Single modeled dynamic memristor	Nonmasking virtual-node reservoir using HOG features	Sim.	MNIST, Fashion-MNIST, CIFAR-10, Cropped SVHN
[216]	Online LMS with L_2 regularization	Readout only	Readout gradient	Memristor crossbars	ESN accelerator with in situ readout learning	Sim.	PJM Energy, Mackey–Glass, Daily Temperature, NARMA10
[225]	Single-epoch SGD; nonlinear model fitting	Readout and model	Readout gradient; nonlinear least-squares model fit	Ion-channel 1R devices	Model-trained reservoir with physical inference	Hybrid	20 × 20-cropped MNIST
[195]	Linear software readout	Readout only	Not specified	Bi_2Se_3 channel network	Physical network reservoir for analog robot control	Hybrid	Rover tracking and drone-lever balancing
[219]	SGD-trained logistic regression	Readout only	Readout gradient	Volatile 1R reservoir	RC with 1D, 2D, and parity preprocessing	Sim.	MNIST
[196]	Ridge regression	Readout only	Closed-form	Discrete-memristor model	Two-dimensional chaotic-map reservoir	Sim.	Nonlinear regression and logistic-map prediction
[226]	FC classifier trained for 10 epochs	Readout only	Not specified	1R volatile reservoir	Single TiN/NbO_x/Pt node with virtual states and 196–256–128–64–10 classifier	Hybrid	MNIST and Hénon map
[220]	Ridge regression	Readout only	Closed-form	Aqueous 1R models	Leaky echo-state and band-pass reservoirs	Sim.	Chaotic time-series forecasting
[208]	Adam optimization	Readout only	Readout gradient (Adam)	1R optoelectronic device	Measured Au/TiO_x/ITO device parameterizing a modeled 500 × 2 reservoir	Hybrid	ECG; digit recognition with 4-bit encoding
[212]	Recursive least squares (RLS)	Readout only	Non-gradient RLS	Dynamic 1R model	Single-node delay reservoir with 2D masking	Sim.	Hénon and Lorenz series and multivariate signals
[221]	200-epoch hardware-aware readout training	Readout only	Not specified	WO_x reservoir; TiO_x crossbar	Dual-memory reservoir and 40 × 64 VMM readout	Hybrid	Isolated spoken digits and Mackey–Glass
[227]	Linear readout training	Readout only	Not specified	1R optoelectronic reservoir	Organic in-sensor analog reservoir with virtual nodes	Hybrid	CASIA-FingerprintV5

Implementation: HW = task-level computation demonstrated using physical memristive hardware; Hybrid = physical memristive devices are involved but software-based training, readout, emulation, or simulation remains an important part of the demonstrated task path; Sim. = modeled or simulated memristive system; N/A = physical device or reservoir-property experiment without a trained task-level system. "Not specified" is used when the cited paper does not establish the optimizer or gradient form precisely enough for a stronger classification.

4. Bio-Inspired Algorithms

Bio-inspired algorithms form a distinct branch of memristive neuromorphic computing because they do not merely transfer conventional digital learning rules onto memristive hardware. Instead, they exploit the intrinsic analog, nonlinear, and history-dependent dynamics of memristive devices as part of the computational mechanism itself [70,228–230]. In this sense, memristive systems are especially attractive for implementing biologically grounded functions such as synaptic plasticity, neuronal excitability, slow regulatory modulation, and adaptive interaction with living neural tissue (Figure 12) [70,231,232].

From a biological viewpoint, the literature can be organized hierarchically. At the lowest level, memristive devices emulate synapses through activity-dependent conductance updates. At the next level, they contribute to neuron-like thresholding, integration, and spike generation. More recent studies extend this framework toward astrocyte-inspired regulation, while larger assemblies give rise to network-level learning and temporal dynamics. The most ambitious direction is represented by hybrid systems, in which memristive devices are coupled to living neurons in a closed loop [70,83,231,233–235].

A critical observation is that the maturity of these directions is uneven. Synaptic emulation is already well established, neuron-like implementations remain less consolidated, astrocyte-inspired systems are still emerging, and hybrid systems represent the most visionary yet technologically demanding branch [230,232,236]. This asymmetry should be kept in mind when assessing the significance of bio-inspired memristive approaches.

The topic is covered through five subsections, addressing memristive synapses, memristive neurons, memristive neuronal networks, neuron–astrocyte networks, and memristive hybrid systems. The classification is mainly based on the level of biological organization that is simulated.

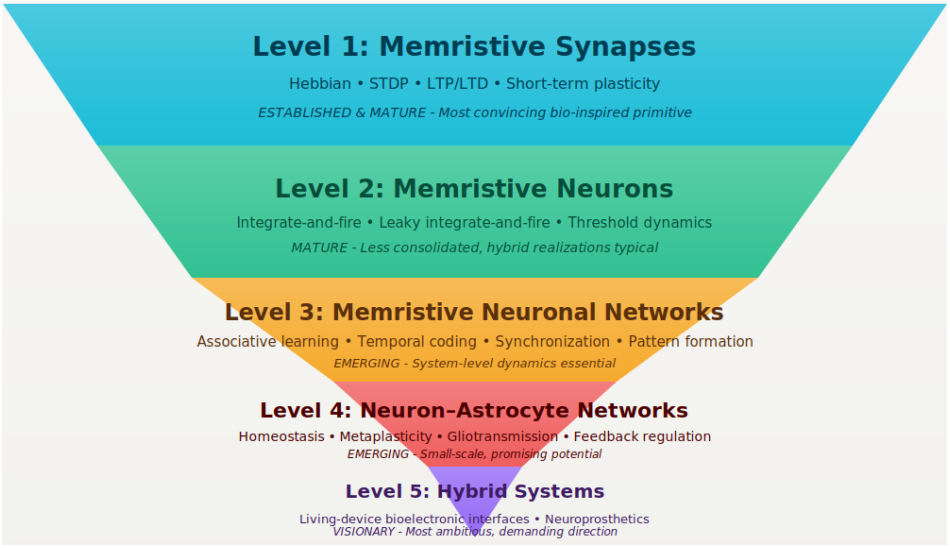


Figure 12. Colour gradient —from cyan (established) through green, amber, and red (emerging) to purple (visionary), text descriptions on each level: indicating the maturity status.

4.1. Memristive Synapses

Synaptic emulation is the most mature and most convincing bio-inspired application of memristive devices. This follows from the close functional similarity between a synapse and a memristor: both are adaptive connection elements whose present state depends on previous activity. Accordingly, a large body of work has shown that memristive devices can reproduce analog weight modulation, potentiation and depression, short-term plasticity, and STDP [81,224,230,237].

A combination of RC with a spiking system was reported in reference [223]. The reservoir was fabricated with volatile polyaniline-based memristive devices, while the readout layer was done from non-volatile parylene memristive devices. This system was applied to the digit classification.

Within this field, the work of Victor Erokhin and collaborators is especially important because it established the organic memristive route as a serious neuromorphic paradigm. In that line, organic memristive devices were treated not as simple memory cells, but as adaptive bio-inspired elements with synapse-like learning behavior and relevance for soft neuroelectronic interfaces [224,228,238]. Experimental studies further showed that conductance evolution in such devices depends strongly on the signal waveform and frequency, indicating that part of the learning rule can be embodied directly in the material dynamics [81].

A particularly important milestone was the demonstration of coupling cortical neurons through an electronic memristive synapse [233]. In that work, an organic memristive device mediated interaction between living cortical neurons, showing that memristors can function not only as artificial synapse models, but also as prosthetic synaptic elements in hybrid biological-electronic systems. This result strongly connects synaptic memristors to adaptive neurointerfaces and future neuroprosthetic applications [70,234].

Overall, memristive synapses remain the strongest bio-inspired primitive in the field. Their main advantage is the close match between biological function and device physics, while their main limitations remain variability, nonlinearity, endurance, and reproducibility under large-scale training conditions [230,232].

Table 3 shows that the field is still dominated by Hebbian, STDP-like, and LTP/LTD-inspired schemes, reflecting the close functional match between synaptic plasticity and history-dependent conductance modulation. More recent directions extend this paradigm toward short-term dynamics, astrocyte-assisted stabilization, and hybrid prosthetic synapses, indicating a shift from phenomenological plasticity emulation toward adaptive bioelectronic interfacing [81,230,233,234,236].

Table 3. Comparison of major bio-inspired synaptic algorithms and memristive synapse paradigms.

Paradigm	Biological Basis	Memristive Mechanism	Main Trade-Off
Hebbian	Co-activity	Correlated conductance increase	Simple and local, but unstable without regulation
STDP	Spike timing	Timing-dependent pulse overlap	Biologically expressive, but sensitive to asymmetry and variability
LTP/LTD-like	Persistent strengthening/weakening	Multi-pulse potentiation/depression	Easy benchmarking, but less temporally rich
STP	Short-timescale facilitation/depression	Volatile conductance dynamics	Useful for temporal filtering, but limited retention control
Astrocytic regulation	Homeostasis and metaplasticity	Refresh/recovery-assisted tuning	Improves stability, but still immature experimentally
Hybrid prosthetic synapse	Adaptive biological coupling	Closed-loop memristive interface	Highest bio-relevance, but most complex technologically

4.2. Memristive Neurons

Compared with synapses, memristive neuron implementations are conceptually important but technologically less mature. A biological neuron performs temporal integration, threshold activation, refractoriness, and excitability, which are more difficult to reproduce in compact hardware than adaptive conductance alone. As a result, many reported memristive neurons remain hybrid realizations in which the memristive device contributes thresholding or nonlinear dynamics, while other circuit blocks provide part of the temporal behavior [239].

It is to note that it was possible to make a memristive-based circuit, emulating not only the learning of a simple animal (pond snail), but even the architecture of the part of the nervous system, responsible for this learning [240].

Even so, progress has been substantial. Recent studies show that memristive devices can be used to realize integrate-and-fire, leaky integrate-and-fire, and oscillator-like neuron models, moving memristive hardware beyond static weight storage toward time-domain computation [82,232,239]. Classic conditioning (Pavlov's dog learning) has been realized based on memristive devices and using the STDP algorithm [241]. However, the literature remains heterogeneous, and many implementations reproduce only selected neuron-like features. Thus, memristive neurons should still be regarded as an emerging rather than mature technological class.

4.3. Memristive Neuronal Networks

When synaptic, neuronal, and regulatory motifs are combined, bio-inspired memristive computation reaches the network level. The focus then shifts from isolated biological analogies to emergent functions such as associative learning, temporal coding, synchronization, sequence generation, and adaptive sensorimotor dynamics [83,232]. This shift is important because the value of bio-inspired algorithms lies less in single-device resemblance than in the system-level dynamics that arise when many adaptive elements interact.

A representative example is the memristive circuit-based model of a central pattern generator proposed by Masaev et al. (Figure 13), in which a memristive schematic capable of self-learning according to bio-plausible STDP was used to organize electrical activity underlying a walking pattern [83]. Figure 13 shows the schematic of a memristive spinal central pattern generator (CPG) implementation, leaky neurons (LN1, LN2), memristive neurons with self-adaptation (memN3, memN4), and a motor neuron (MN), with memristive neurons at multiple levels and non-memristive neurons feeding into an motor-neuron output stage that drives the walking pattern (represented by the rodent limb).

Such studies move the discussion beyond static benchmarks and toward embodied timing and neuroprosthetic relevance. At the same time, network-level studies remain heterogeneous, and a useful distinction must be made between genuinely bio-inspired systems based on local physical adaptation and architectures that use memristors mainly as compact programmable weights.

Single- and double-layer perceptrons based on organic memristive devices were fabricated and studied. In the case of a single-layer perceptron [242] classification of linearly separable objects was performed according NAND and NOR logic operations, while in the case of a double-layer perceptrons [243] it was done for linearly not separable objects according to XOR logic operation.

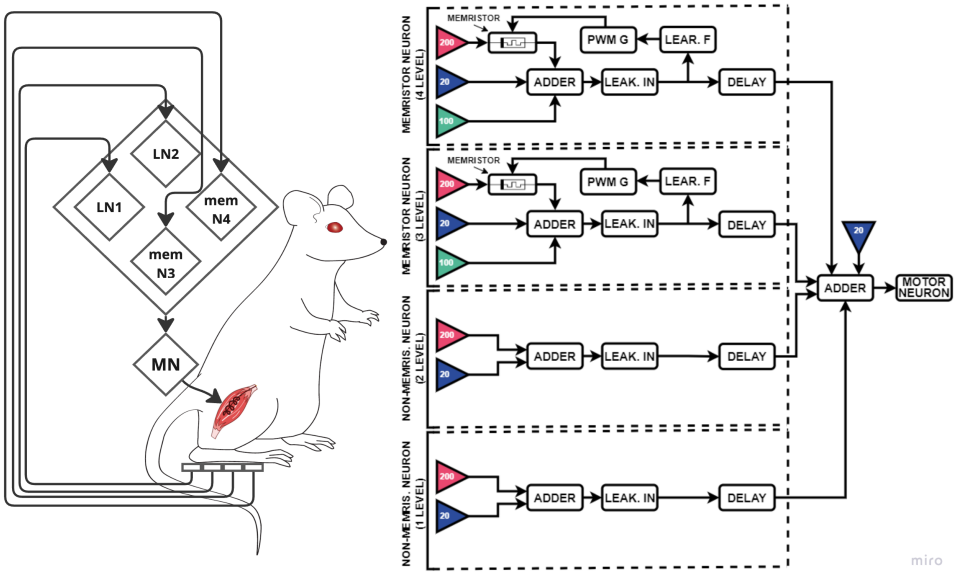


Figure 13. The schematic representation of the memristive spinal CPG implementation reworked from reference [83].

4.4. Neuron–Astrocyte Networks, Towards Memristive Implementation

According to recent findings, astrocyte signaling plays a critical role in synaptic plasticity, neuronal network oscillations, information processing, and memory [244]. Although incapable of generating action potentials, astrocytes respond to neuronal activity through elevations in intracellular calcium. This enables them to detect neurotransmitters and release gliotransmitters, such as glutamate and ATP, which in turn provide feedback to neurons, modulating synaptic transmission, excitability, and plasticity [245,246]. Notably, the timescale of both astrocytic calcium dynamics and their modulation of synaptic transmission is much slower in duration relative to neuronal spiking [247]. This temporal property is crucially important for retaining a record of prior activity and facilitating plastic changes during synaptic plasticity. Consequently, the multiscale spatiotemporal integration of astrocytes with neuronal networks may enable higher-order/intelligent information processing [248].

These predictions have been confirmed by computational models, which show that astrocytic integration enhances network performance [249,250]. To construct a biologically precise model and clarify how astrocytes regulate neural activity, an astrocyte is integrated into the neuron model. Intracellular calcium dynamics in astrocytes are traditionally described within the Hodgkin–Huxley formalism [251] and are represented by biophysical models of intracellular calcium concentration dynamics based on systems of second- or third-order nonlinear differential equations (the Li–Rinzel and Ullah models; [252,253]). As dynamical system, the astrocyte model represents a nonlinear oscillator. The oscillations typically emerges from supercritical Andronov-Hopf bifurcation. In its excitable mode the astrocyte can respond on external stimulation imitating, for example, a glutamate puff by single pulses. At variance with neuronal all-or-none response calcium pulses in astrocytes have gradually changing amplitudes depending on glutamate concentration. Detailed bifurcation analysis of single astrocyte dynamics and oscillations in coupled cells demonstrated possibility of complex oscillatory modes including a chaotic dynamics [254,255].

Astrocytic modulation of synaptic transmission is also described using biophysical models, either by modifying the Tsodyks–Markram model of short-term synaptic plasticity [256,257] to incorporate astrocytic influence [258], or through more simplified approaches in which astrocytic modulation is represented by varying synaptic weight or in the excitability of the pre-

or postsynaptic neuron as a function of astrocytic activity [259,260]. Based on such biophysical models of neurons, astrocytes, and their bidirectional interaction, large-scale multilayer networks have been constructed to describe signal processing in neuron–astrocyte networks of the brain. These models have demonstrated that astrocytes increase firing synchronicity [261] and synaptic coordination [262]. It has also been shown that astrocytes induce dynamic coordination in neuronal ensembles by stimulating spatiotemporal patterns of neuronal activity on the astrocytic timescale, thereby facilitating the generation of integrated information within the neuronal network [263,264]. Furthermore, astrocytic modulation of synaptic transmission and short-term synaptic plasticity has been shown to result in local spatial synchronization in neuronal ensembles [265], a mechanism underlying multi-item short-term memory in neuron–astrocyte networks [266–269]. While interest in the biological and computational modeling of neuron–astrocyte signaling is increasing, the neuromorphic circuitry underlying this crosstalk remains less explored [270]. For hardware implementation, a phenomenological functional approach describing the neuron–astrocyte information exchange loop is preferred over detailed biophysical models. From engineering perspective, astrocytes provide slow feedback to both pre- and postsynaptic neurons, thereby regulating their signaling.

Early hardware implementations were reported almost simultaneously. Soleimani et al. [271] proposed a modified astrocyte model enabling convenient digital implementation and reproducing relevant biological astrocyte behaviors for feedback control of neuronal activity. Their model was implemented on FPGA using two coupled limit-cycle Hopf oscillators with weak astrocytic coupling. Similarly, Nazari et al. [272] proposed a digital neuromorphic circuit for neuron–astrocyte interaction utilizing the Izhikevich neuron model [273] and a functional astrocyte calcium model [274]. Employing single constant multiply (SCM) and linear approximations for low-cost FPGA implementation, they demonstrated that the digital astrocyte alters neuronal firing patterns via bidirectional communication, thereby enhancing information processing for reconfigurable neuromorphic devices. Hayati et al. [275] advanced to a simple two-neuron-one-astrocyte network using FitzHugh–Nagumo (FHN) neurons. FPGA implementation confirmed that this model mimics biological regulation of postsynaptic activity and synaptic transmission, including LTP. More recently, Hong et al. Hong et al. [276] proposed a memristive self-repairing network inspired by astrocytic endocannabinoid signaling and tripartite neuron–astrocytic synapse biology. When synaptic faults occur, the circuit auto-detects and repairs damaged neurons by enhancing healthy neurons, analogous to biological astrocyte repair. Robot experiments confirmed improved performance and reliability. Stasenko et al. [277] developed a SNN model integrating memristor-based plasticity alongside astrocytic regulation. This allowed the network to switch between "burst" and "asynchronous" modes, with astrocytes restoring normal burst dynamics in glutamatergic synapses, demonstrating their crucial role in controlling network-level behavior. Recently, a novel neuron–astrocyte network comprising one astrocyte and two Hindmarsh–Rose (HR) neurons coupled via memristive synapses [278] was introduced, exploring the effects of astrocyte heterogeneity and memristor parameters on firing activities. This work employs a liquid memristor as a biomimetic device emulating the neuronal microenvironment, including chemical and electrical synapses. The first dedicated "artificial astrocyte memristor" was demonstrated by Cheng et al. [236] in one material structure represented by a traditional memristive device encapsulated with the oxygen vacancy-rich layer modulating the memristor switching dynamics. This device uniquely features recoverable linearity, meaning the nonlinear distortions that appear over time can be reversed via a refresh operation—directly mimicking the astrocyte's role in maintaining normal neuronal activity. This recovery of linearity boosted

digit classification accuracy from 62.98% to 94.75%, showcasing its practical value for neuromorphic computing.

All these recent studies represent initial steps toward integrating astrocytes into learning algorithms for memristive neuromorphic devices. However, they are limited to small networks (a single astrocyte with one or a few neurons), primarily investigating astrocytic effects on neuronal synchronization. Further development requires the implementation of neuron–astrocyte networks directly on neuromorphic chips at scales sufficient to solve applied tasks such as robotic navigation control or classification of noisy images and video.

4.5. Memristive Hybrid Systems

The most ambitious realization of bio-inspired memristive algorithms is found in hybrid systems, where artificial adaptive devices interact with living neural systems in real time. Here the objective is not simply to emulate biology, but to communicate with it through adaptive electronic interfaces [70,231,234,235]. This direction is especially important for neuroprosthetics, biosensing, and closed-loop neuromodulation.

A foundational reference in this direction is the neurohybrid memristive CMOS-integrated architecture proposed by Mikhaylov et al., which combines living neuronal cultures, microelectrode arrays, memristive arrays, mixed-signal CMOS circuitry, and spiking modules in a bidirectional neurointerface [70]. Partial implementation of the proposed concept is reported in reference [231] using the example of a memristor-based adaptive system for stimulating a spatially ordered culture of hippocampal cells. More recent work on memristor-based neuromodulation devices and stochastic memristive interfaces further supports the idea that memristors can act as adaptive signal-processing and coupling elements in biologically relevant closed-loop systems [234,235].

Taken together, the literature suggests that the long-term value of bio-inspired memristive algorithms may lie less in isolated neural metaphors and more in the use of adaptive device physics to build robust, low-power, closed-loop systems approaching the organizational principles of living neural computation [70,232].

5. Reinforcement Learning Algorithms with Memristors

Reinforcement Learning (RL) is a machine learning technique in which an "agent" learns to make decisions by performing actions in an environment to maximize a cumulative reward [279]. Unlike supervised learning, which uses labelled datasets, RL relies on a trial-and-error approach similar to how humans or animals learn.

The reinforcement learning process is typically modeled as a Markov Decision Process (MDP), which consists of five core elements:

- Agent is the learner or decision-maker in the system.
- Environment represents everything the agent interacts with.
- State is the current situation or configuration of the environment.
- Action is the choice or movement the agent makes to interact with the environment and.
- Reward represents a numerical feedback signal (positive or negative) that tells the agent how successful an action was.

In the following subsections, we provide a comprehensive overview of reinforcement learning algorithms implemented with memristive technologies. We begin with temporal difference algorithms in Section 5.1, exploring their principles and hardware adaptations. Next, in Section 5.2 we examine bioinspired reinforcement learning methods that leverage neural plasticity concepts. Following this, policy search techniques are discussed, highlighting their applications and benefits in Section 5.3, while the integration of a Monte Carlo method with memristive systems is presented in Section 5.4. Finally, the

section concludes with a summary that combines key insights and comparisons across these algorithmic approaches.

5.1. Temporal Difference RL

Temporal Difference (TD) RL [279] is a fundamental approach in machine learning where an agent learns to predict and optimize future rewards by updating estimates based on the difference between successive predictions. Key algorithms include TD(0), SARSA, and Q-learning [279].

Ref. [280] presented the first memristive deep Q-learning implementation, addressing RRAM training limitations like backpropagation and energy-intensive weight updates. They proposed an architecture for direct RRAM training using a variability-free tuning scheme and gradually-write circuits to reduce read/write frequency and manage RRAM asymmetry. By reordering training flow, they eliminated weight copying between networks. This approach achieved 126× better energy efficiency and 70× faster performance than GPU implementations, proving memristive crossbars as efficient platforms for DNN.

Authors in reference [281] proposed a hybrid 1T-1R memristor-based platform with an optimized deep Q-learning algorithm. As shown in Figure 14, vector-matrix multiplications for the Deep Q Network (DQN) forward pass are offloaded to memristive crossbars, leveraging Ohm's and Kirchhoff's laws for high efficiency, while both backpropagation and weight updates are also accelerated by the crossbars. Using memristive crossbars for processing and non-volatile memory offers superior throughput and energy efficiency, demonstrated in cart-pole and mountain car simulations.

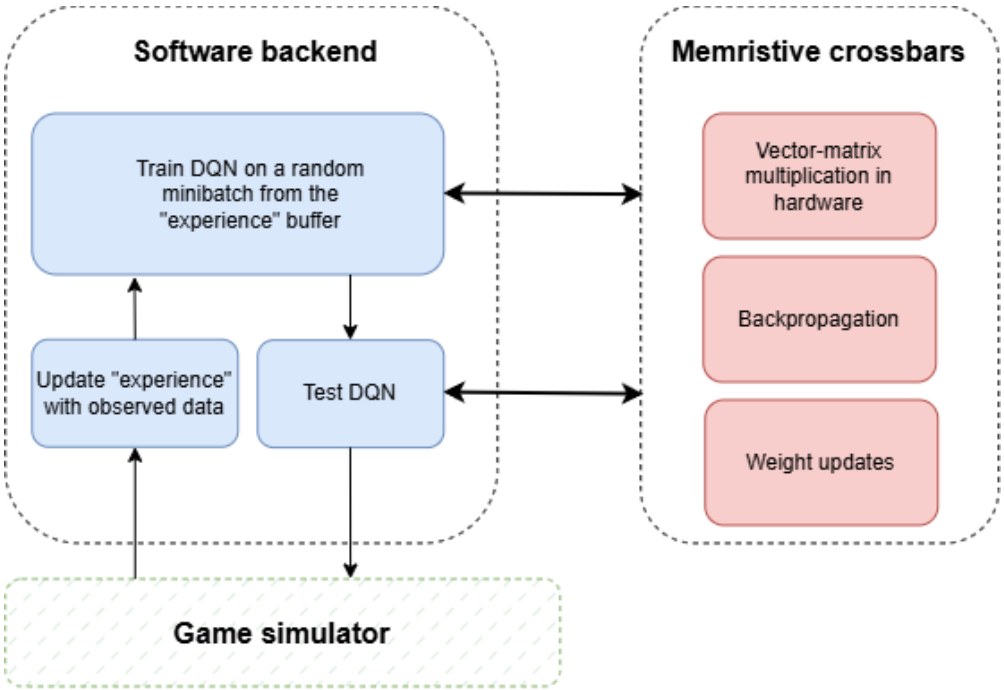


Figure 14. Flowchart of the deep-Q reinforcement learning algorithm [281].

A similar approach was proposed in reference [282], where the authors implemented a deep Q-learning neural network, termed MEM-DRL, using a memristive crossbar with two hidden layers. Consistent with the methodology in reference [281], this memristor-based network functions as a Q-function approximator for current and future states, training the agent via experience replay. The system updates memristor crossbar conductances in-loop using error gradients derived from backpropagation. To minimize energy consumption, the design incorporates an analog ReLU circuit, and the authors demonstrated that

low-precision (4-bit) representations are sufficient for effective on-chip analog training. Furthermore, the system's robustness was validated by injecting noise to simulate memristor device variability. A key contribution of this work is the significant reduction in energy consumption compared to optimized digital computing systems [282].

Memristive crossbars have been effectively utilized in prior research [283], where the authors introduced a path-finding algorithm termed MRNS Q-learning. This approach enhances training efficiency and convergence by integrating reward-shaping Q-learning with a memristive Radial Basis Function (RBF) neural network. The reward-shaping mechanism leverages a priori environmental knowledge to provide supplementary feedback, accelerating the learning process. Structurally, the RBF network comprises three functional layers: a transparent input layer for data distribution, a hidden layer for high-dimensional nonlinear mapping, and a linear output layer for final action prediction. Within the framework presented in reference [283], RBF synaptic weights are updated based on the TD error, while the reward signal is augmented by the shaping function.

A framework for TD learning based on memristor technology, wherein memristors fulfill a tripartite function, is presented in reference [284]. In this framework, memristors serve not only as synaptic weights within an actor-critic network trained online and in-memory but also play a crucial role in determining the network's subsequent actions. Additionally, memristors are employed to compute the weight updates associated with the TD error, a critical element of the training algorithm. The applicability of this framework is validated through two well-established neuroscience navigation tasks: the T-maze and the Morris water maze. By implementing two distinct neural networks one responsible for action selection and the other for estimating the value function necessary for TD error computation the authors demonstrate that their bio-inspired approach enables fully in-memory training while maintaining the requisite accuracy.

In situ RL with memristive hardware offers efficiency but faces challenges regarding device endurance and operational safety. To address this, a two-fold method, first proposed in reference [285] and subsequently expanded in reference [286], combines ex situ simulation with in situ deployment, reducing sample complexity and preserving hardware longevity. Depending on data availability, the approach uses either direct retraining or off-policy pre-training within an actor-critic framework. This system utilizes one-step TD learning and is specifically optimized to mitigate memristive non-idealities and device variations.

SARSA (State-Action-Reward-State-Action) [279] is an on-policy TD RL algorithm. It learns the action-value function Q(s,a) by updating estimates based on the current state, action, reward received, next state, and next action taken by the agent. The update rule uses the TD error calculated from these elements to improve the policy gradually. SARSA(λ) extends SARSA by incorporating eligibility traces, which allow the algorithm to update not only the most recent state-action pair but also previous ones, weighted by how recently and frequently they were visited. The parameter λ ($0 \leq \lambda \leq 1$) controls the decay rate of these traces, balancing between one-step TD learning ($\lambda = 0$) and Monte Carlo methods ($\lambda = 1$). This leads to faster and more efficient learning, especially in environments with delayed rewards.

SARSA (State-Action-Reward-State-Action) is an on-policy RL algorithm updating Q-values via state-action sequences, while SARSA(λ) adds eligibility traces, controlled by λ, to update past pairs for faster, more efficient learning in delayed-reward environments [279].

Approach proposed in reference [287] utilizes the conductance drift in Ti/TiO_x/Pt memristors, where the intrinsic decay property of the devices was leveraged to accelerate the SARSA (λ) RL algorithm. By implementing this mechanism within a memristor array, the authors demonstrated faster convergence of SARSA (λ) relative to classical RL algorithms, as validated through a path-planning task [287]. Within the memristor-based RL system,

an agent interacts with the environment to identify the shortest optimal path (Figure 15a). The Sarsa(λ) algorithm updates the Q-values Q(S,A) for all state-action pairs stored in the Q-table by incorporating a decay factor λ. The implementation of Sarsa(λ) involved encoding path information onto a TiO_x-based memristor after 32 training cycles. The decay mechanism of λ is realized via conductance drift, attributed to the diffusion of oxygen vacancies within the TiO_x memristor, as depicted in the upper panel of Figure 15a. Distinct actions correspond to different conductance states, which diminish over time. Steps closer to the target path endpoint carry increased importance and higher corresponding values. Figure 15b presents the schematic structure of the TiO_x memristor (upper panel) alongside its scanning electron microscopy (SEM) image (lower panel). Subsequently, conductance values representing the training rounds are input into the memristor array in parallel, enabling computation of Q(S,A) through the intrinsic MVM capabilities of the memristor array, as illustrated in Figure 15c.

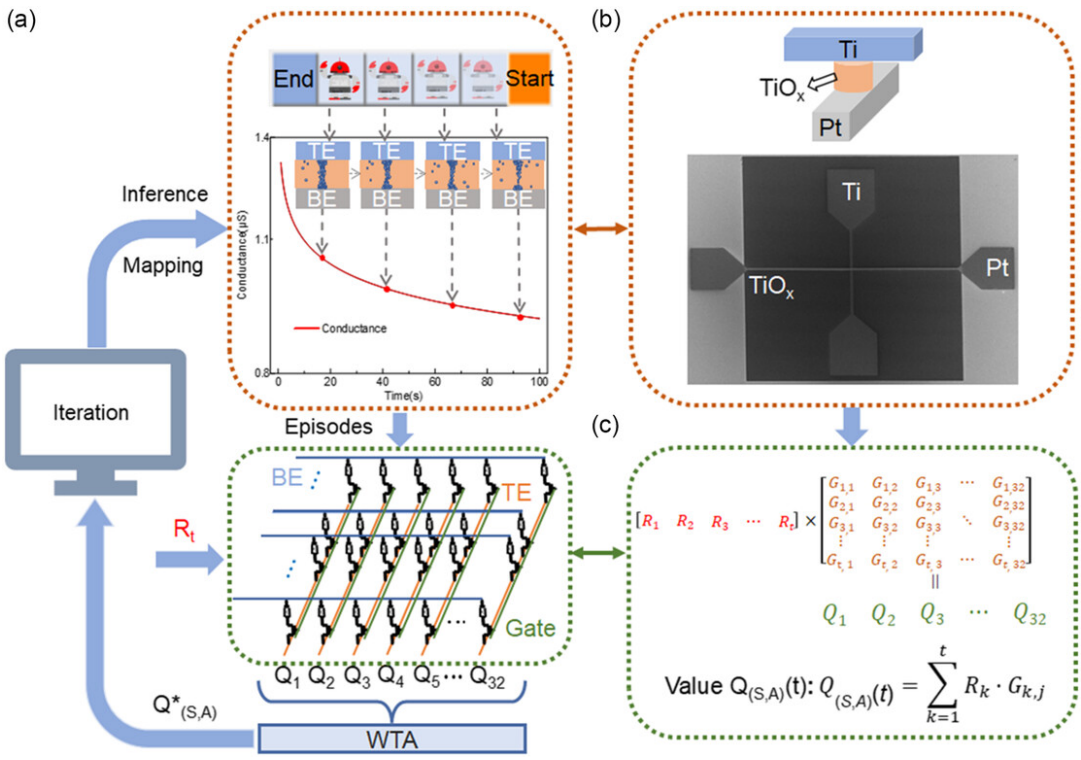


Figure 15. (**a**) Path planning RL system diagram (**b**) memristor structure diagram and SEM image (**c**) Matrix vector multiplication performed by memristor array, adapted with permission under the Creative Commons Attribution (CC BY 4.0) license from reference [287].

5.2. Biologically Inspired RL

Even though memristor-based Neural Network (NN)s allow in-memory computation, which is fast and efficient, they are not suitable for traditional learning methods, such as error back-propagation. SNNs are highly promising from this aspect, since they allow weight updates based on "local" knowledge, instead of requiring information collected from the entire network, as in the case of error back-propagation. A memristive synapse circuit with one memristor and four transistors, hence called the 1M4T synapse circuit, was presented in reference [71]. The proposed 1M4T circuit used as a synapse in SNN is suitable for so-called Reward-modulated Spike-Timing Dependent Plasticity (R-STDP) learning, which is demonstrated on a simple pattern classification task. R-STDP learning assumes that the memristor (within the 1M4T synapse circuit) conductivity can be updated, increased or decreased according to the current reward/punishment in supervised learning,

respectively. When the current classification is correct, synapse conductance is increased and, on the opposite, decreased when the classification is not correct. Standard STDP assumes that synapse conductivities (representing NN weights) are updated based on a temporal difference of a pre-synaptic and post-synaptic spike occurrence: synapse is strengthened (weight increased) when a post-synaptic spike arrives after a pre-synaptic one and, on the opposite, weakened (weight decreased) when it arrives before. In the 1M4T synapse circuit and its R-STDP learning proposed in reference [71] the same effect of increasing/decreasing memristor conductance can be achieved with the post-synaptic spike always arriving later than the pre-synaptic one, which simplifies the training process significantly. Although only supervised training on a simple pattern recognition task was demonstrated in reference [71], the proposed 1M4T synapse circuit can be used to realize on-chip low-cost and energy-efficient neuromorphic RL.

In ref. [288], the authors introduce MSNN-RL, a reinforcement learning framework utilizing Memristive Spiking Neural Networks (MSNN). This architecture integrates the SARSA temporal difference algorithm with a Remote Supervised Method (ReSuMe) to modulate synapses via STDP rules. Within this framework, the MSNN serves as a function approximator for action values, effectively addressing challenges associated with high-dimensional state spaces. The implementation utilizes the ThrEshold Adapted Memristor (TEAM) model in a network architecture restricted to input and output layers, omitting hidden layers. To bridge the gap between continuous-domain data and the spiking domain, the authors employ encoding and decoding mechanisms for bidirectional signal processing. The validity of this approach was demonstrated through a simulation of the acrobot system.

In ref. [289], the authors pioneered the integration of reinforcement learning in SNNs using memristive synaptic weights. Their approach utilizes a time-coding scheme that employs Gaussian receptive fields for input encoding. Learning is driven by a binarized reinforcement signal with adaptive amplitude, where the reward gradient calculated between consecutive time steps dictates the polarity of the feedback. Specifically, a positive reward difference triggers an external rectangular current pulse to the output neuron, while a negative difference results in a punishing pulse. The magnitude of these pulses is dynamically attenuated as the training episode progresses toward completion. In a subsequent study [290], the authors developed an online RL algorithm for SNNs using memristive STDP synapses. Benchmarked on CartPole, the system uses Leaky Integrate-and-Fire (LIF) neurons and incremental learning based on spike timing, eliminating state-action histories. By prioritizing early spikes and adjusting weights via inter-spike intervals, the approach proved robust across various memristive devices despite non-ideal hardware characteristics.

While addressing the challenge of implementing reinforcement learning in SNN with memristive STDP synapses and focusing on real-time training, the authors in reference [291] introduced the concept of trace variables for continuous reinforcement learning. This approach, as described by the authors, enables the development of energy-efficient neuromorphic processors capable of learning and fine-tuning parameters throughout their entire operational lifetime. The proposed trace variables function by accumulating synaptic weight changes modulated by reward signals, thereby endowing the agent with a form of short-term memory that facilitates the integration of successive actions into coherent behavioral patterns. The efficacy of this learning methodology was demonstrated on the Cartpole control task, employing the same input state encoding scheme as in reference [290]. In particular, due to the incorporation of a pretraining phase, the SNN in reference [291] required significantly fewer episodes to achieve successful task performance.

A Generative Adversarial Network (GAN) is a type of deep learning architecture consisting of two neural networks, a Generator and a Discriminator, that compete against each other in a "zero-sum game" to create realistic new data. The generator's goal is to create

synthetic data that is indistinguishable from real data, while the discriminator is a binary classifier that evaluates data to determine its authenticity. While GANs produce high-fidelity data, their three-stage training process demands significant computational complexity and memory overhead. To mitigate these challenges, the authors in reference [28] propose a GAN architecture entirely implemented via memristive SNNs. By utilizing SNNs for both the generator and discriminator, the system enhances computational efficiency, while a specialized 1-memristor 4-transistor (1M4T) synaptic circuit enables computation-in-memory, drastically reducing memory access costs. Training is achieved through reward-modulated STDP alongside a novel gradient-free backpropagation mechanism. The high-speed, energy-efficient performance of this memristive SNN GAN was validated using the MNIST [107] and Fashion-MNIST datasets [292].

5.3. Policy Search Methods

Policy search is a broad category of RL methods that aim to find an optimal policy directly, without necessarily relying on value functions. Policy gradient methods are a specific subset of policy search methods that directly optimize an agent's strategy, called a policy, in order to maximize the rewards it receives. In contrast to value-based reinforcement learning (e.g., Q-learning), which derives an optimal strategy indirectly by estimating state-action value functions, policy gradient methods perform direct optimization within the policy space to maximize expected returns. The policy is typically represented by a deep neural network that maps a given state to a stochastic distribution over the action space. The primary objective is to iteratively refine the network parameters through gradient ascent, thus increasing the likelihood of selecting actions that maximize the expected cumulative return. Proximal Policy Optimization (PPO) [293] is preferred over traditional policy gradient methods due to its superior training stability and resistance to policy collapse. While traditional policy gradient methods are sensitive to learning rate fluctuations, PPO utilizes a clipped surrogate objective function. This mechanism constrains the probability ratio between the new and old policies, effectively preventing large-scale parameter updates that could lead to catastrophic divergence in the agent's behavior.

In reference [294], the authors introduce Decay-based Phase-change Proximal Policy Optimization (DP-PPO), a deep reinforcement learning framework that integrates phase-change memristive technology with the standard PPO algorithm. By leveraging the conductance drift inherent in memristive PCM to derive a dynamic decay function for the clipping hyperparameter ϵ, the framework mitigates the environment under-exploration typical of constant ϵ while reducing computational overhead [294]. The proposed RL framework utilizes a shared CNN architecture for both the actor and critic networks, which differ only in their final fully-connected layers. The actor network outputs a selection from eight possible actions, while the critic network produces a single scalar representing the state value.

The approach from reference [295] is presenting the hardware-efficient In-Memory Reinforcement Learning (mRL) system using Ferroelectric Tunnel Junction (FTJ) memristors. The core innovation lies in leveraging the intrinsic stochastic conductance switching of FTJs to facilitate environment exploration and optimal policy search by mimicking a random search process that can outperform deterministic heuristics. The proposed training algorithm uses a WTA selection method and a biphasic pulsing to refine policies according to received rewards. It integrates leaky capacitors to create a memory trace, allowing the system to retroactively reinforce the specific sequence of actions that led to a successful outcome. Experimental and simulation results, including a CartPole balancing task, show that moderate levels of stochasticity significantly improve learning speed and sample efficiency. Conversely, the study found that either zero or excessive stochasticity degrades

performance, highlighting a "sweet spot" for device variability in autonomous learning. This approach demonstrates a path toward compact, low-power RL systems for mobile deployment by moving away from traditional von-Neumann architectures.

Unlike deterministic deep learning models that utilize point estimates for weights and biases, Bayesian Deep Neural Network (BDNN)s treat parameters as probability distributions. This probabilistic framework enables the quantification of predictive uncertainty, a feature essential for safety-critical applications such as medical diagnostics and autonomous systems. Recent research [296] demonstrates that the intrinsic physical mechanisms of memristive devices induce fluctuations in read currents. By leveraging the ability to tune this stochasticity through conductance programming, memristors can effectively represent the weight probability distributions required for hardware-based BDNN implementations. To validate this approach, the authors utilized the Risk-Sensitive RL "Storm Coast" task. The experimental framework employed the Proximal Policy Optimization (PPO) algorithm to optimize the policy, aiming to maximize cumulative rewards while accounting for the inherent stochasticity of the environment. The implementation utilized a four-layer BDNN to decouple aleatoric (due to inherent variations within the environment) and epistemic (due to lack of knowledge) uncertainties, leveraging the intrinsic stochasticity of memristive devices. When compared to conventional digital architectures, the proposed memristor-based hardware demonstrated a tenfold increase in execution speed and a 150-fold improvement in energy efficiency during uncertainty decomposition tasks

To optimize and accelerate multi-level memristor conductance programming relative to the conventional write-and-verify technique, the authors in reference [297] propose an effective control framework termed Probe-then-Program (PtP). During the initial probe phase, the statistical distribution of parameters is inferred from a physics-based memristor model using sequential Bayesian inference. Once the model is accurately fitted, it functions as a surrogate model of the memristor, enabling the training of a memristor tuning agent. This agent predicts the optimal programming pulse amplitude during the subsequent program phase. The underlying neural network agent is trained via the PPO algorithm, while experimental results demonstrate that this approach achieves a tuning speed approximately twice as fast as traditional write-and-verify methods.

The initial application of Deep Reinforcement Learning (DRL) for the synchronization of Memristive Chaotic Systems (MCS), as an alternative to conventional linear and nonlinear control methods that rely on precise mathematical models, was introduced in reference [298]. Utilizing a well-characterized memristive chaotic circuit comprising a memristor, two capacitors, two resistors, and an inductor, the authors developed a DRL framework based on the PPO algorithm. This framework aims to minimize the synchronization error, defined as the difference between the outputs of the master and slave oscillators, thereby achieving synchronization between the two systems. Furthermore, the proposed PPO algorithm incorporates a modified reward function, which facilitates rapid convergence and demonstrates robustness to Gaussian noise interference. The effectiveness of the DRL-based synchronization approach for MCS is validated through its application to image encryption.

5.4. Monte Carlo RL Method

Monte Carlo (MC) reinforcement learning methods are a class of algorithms that learn value functions and policies based on averaging complete returns from episodes. Unlike TD methods, MC methods wait until the end of an episode to calculate the total accumulated reward (return) and then update value estimates accordingly. This approach does not require knowledge of the environment's dynamics and does not bootstrap from other estimates, making it straightforward but typically limited to episodic tasks. MC

methods are useful for problems where the environment naturally breaks into episodes and provide unbiased estimates of value functions by relying on actual observed returns.

In ref. [299], the authors leveraged memristor cycle-to-cycle variability for in situ Bayesian learning using a 16,384-device 1T1R array. By applying Markov Chain Monte Carlo (MCMC) sampling across device rows, they achieved accelerated learning and high accuracy in supervised and reinforcement tasks. The approach significantly reduces power consumption and remains robust against device aging and variability, making it ideal for energy-efficient edge hardware applications.

Table 4 summarizes key reinforcement learning algorithms implemented with memristive networks, highlighting their synaptic circuits, architectures, hardware presence, network types, and benchmarks. In this table, a "Yes" in the HW column indicates the presence of actual physical components, such as memristive crossbar arrays and memristive devices, employed to implement either part or the whole neural network. Conversely, a "No" in this column signifies that only hardware simulations have been performed and documented.

Table 4. Reinforcement learning algorithms with memristive networks.

Reference	Algorithm	Synaptic Circuit	Architecture	HW	MNN/SNN	Benchmark
[280]	Deep Q-learning	1M	CNN	No	MNN	The Arcade Learning Environment [300]
[281]	Deep Q-learning	1T1M	MLP	Yes	MNN	Cart pole and Mountain car
[282]	Deep Q-learning	1M	MLP	No	MNN	Cart pole
[283]	Q-learning, TD	1M	MLP	No	MNN	Path finding on 40 × 40 grid
[286]	TD, gradient descent	1M	MLP	proposed HW implementation	MNN	Inverted pendulum
[285]	TD, gradient descent	1M	MLP	No	MNN	Inverted pendulum
[284]	TD	1M	SLP	Yes	MNN	T-maze and Morris water maze
[287]	SARSA (λ) with modulation of memristor's conductivity	1T1M	memristor array	Yes	MNN	Path planning task
[28]	R-STDP	1M4T	SLP	No	SNN	MNIST, Fashion-MNIST
[71]	R-STDP	1M4T	SLP	No	SNN	Pattern classification (X/O)
[288]	SARSA learning, STDP	1M	SLP	Proposed platform for weights update	SNN	Acrobot system
[289]	TD, STDP	1M	SLP	No	SNN	Cart pole
[290]	TD, STDP	1M	SLP	memristive devices	SNN	Cart pole
[291]	STDP	1M	SLP	No	SNN	Cart pole
[296]	PPO	1M1T	MLP	Yes	MNN	Storm coast task
[297]	PPO	1M	crossbar array	No	MNN	/
[295]	Policy search	1M	memristive crossbar	Ferroelectric Tunnel Junction (FTJ) memristors	MNN	Cart pole
[299]	Metropolis-Hastings Markov Chain Monte Carlo	1T1M	memristor array	Yes	MNN	Cart pole

6. Future Directions, Challenges and Conclusions

Neural networks incorporating memristive devices have a broad range of potential applications, including pattern recognition and classification [107,179,223,301], time-series forecasting [185,221], biomedical applications [69,195,302], sensing [303,304] Edge AI [176], and optoelectronic neural networks [227,305], as well as various neuromorphic computing applications [306].

The full realization of the application potential of MNNs requires addressing the current challenges associated with both training and inference. In the case of inference-only implementations, the performance of the overall system can be degraded due to the mismatch between the programmed weight values and the actual physical conductance states of memristive devices. In addition, such approaches may lose part of the inherent advantage of memristive systems, since they do not fully exploit the reduced data movement, lower energy consumption, and improved latency enabled by in situ training. Recently, Liu et al. [37] summarized the main challenges of analog memristive training as precision limitations related to non-idealities in VMM computation, together with efficiency limitations caused by temporal variations, including nonlinearity and cycle-to-cycle variation, as well as spatial device-to-device variation. As a possible direction for overcoming these limitations, they proposed the error-aware probabilistic update method (EaPU), which adapts the algorithmic weight-update process to the stochastic nature of memristive conductance changes, leading to more stable training and more reliable weight updates. However, the full hardware realization of large-scale neural networks relying on EaPU remains an open challenge, since demonstrated hardware realizations are shown only for small-scale image-processing networks, whereas larger architectures such as ResNet and Vision Transformers are verified through simulation. Mixed-precision training approaches, such as those proposed by Nandakumar et al. [64] and Wu et al. [25] represent another direction for mitigating device non-idealities in the weight-update process. By combining memristive VMM acceleration with high-precision digital gradient accumulation and update control, these methods relax the requirements on analog conductance updates; however, they still require high-precision digital weight storage and remain limited in terms of fully integrated large-scale experimental verification.

Besides mitigating device non-idealities through programming and training methods, such as write-and-verify, mixed-precision training [64], and EaPU [37], another important direction is to improve memristor characteristics through materials and interface engineering. Relevant requirements include sufficient conductance states, controllable bidirectional updates, and acceptable device-to-device and cycle-to-cycle variability, as discussed for gradient-based training by [31,116]. Promising recent advances include CMOS-integrated memristors with 2048 conductance levels [307] and interface-engineered devices with improved cycling uniformity, multilevel switching, retention, and endurance [308]. Furthermore, Dalgaty et al. [299] exploited intrinsic variability for gradient-free probabilistic learning, while Ming et al. [309] engineered oxygen-gradient-driven temporal dynamics to support learning rate adaptation in reinforcement learning. In generally, technological advances should be pursued through hardware–algorithm co-design, taking into account the specific requirements of different algorithmic approaches.

One of the challenges that arise in memristive systems is the influence of delays, uncertainties, and perturbations on the dynamical stability of neural networks and, where synchronization is required, their speed of convergence to synchronization. Fan et al. [310] addressed this issue by deriving sufficient conditions for finite-time Mittag–Leffler synchronization under feedback control for fractional-order multi-delayed memristive neural networks (FMMNNs). Related recent Lyapunov-based synchroniza-

tion analysis using a non-decomposition approach for fractional-order octonion-valued fuzzy bidirectional associative memory neural networks (FOOVFBAMNN) [311], as well as stability analysis of linear time-delay systems using two new lemmas and genetic-algorithm-based optimization of stability-criterion parameters [312], could, with appropriate model adaptation, provide a basis for further research into memristive network dynamics. This perspective could complement the discussion of training methods in this review by considering the dynamical stability of memristive networks under device non-idealities, in addition to the convergence of learning algorithms.

Another important limitation of memristive NNs is the overhead introduced by peripheral ADC and DAC units, which can significantly affect energy consumption, chip area and latency [10]. Possible solutions discussed in reference [10] include the use of lower-precision converters, the sharing of ADC/DAC units between multiple columns/rows, and the integration of these blocks on-chip together with the memristive crossbar. Other promising approaches that avoid the need for ADC and DAC blocks include the implementation of a dedicated circuit for storing the product of two analog voltage signals directly in a memristive device [135], as well as the implementation of the contrastive-divergence algorithm for memristive DBNs, as proposed in reference [159], although scalability still remains an open issue. In a recent study, Hong et al. [313] introduced a memristive ADC with adaptive quantization, which can be integrated into CIM systems using content-addressable memory (CAM) cells, achieving more than 50% reduction in energy overhead and more than 25% reduction in area overhead for neural network applications, while DAC overhead remains unaddressed.

Regarding the future potential of in situ training of memristive neural networks, mixed precision learning by Nandakumar el al. offer a promising solution by combining high-precision digital accumulation of weight updates with imprecise conductance updates within computational memory unit [64]. Other notable approaches include Chopped-TTv2 (c-TTv2), which mitigates residual reference offsets in TTv2 through chopper-based sign changes, and Analog Gradient Accumulation with Dynamic reference (AGAD), which dynamically estimates reference values and supports devices with symmetric or asymmetric update characteristics [314]. More recently, error-aware probabilistic update (EaPU) has emerged as a promising hardware-aware approach that accounts for memristor stochasticity while substantially reducing physical weight updates [37]. In contrast, ex situ noise-aware software training, exemplified by Joshi et al. [315], offers considerable potential for inference acceleration rather than on-device learning, while combining hardware-aware training and mapping with multicore architectures provides a practical route toward scalable hardware, as illustrated by the 48-core NeuRRAM inference chip demonstrated by Wan et al. [26].

From the perspective of future developments, recent findings by Wang et al. [316] expose the potential of Memristive Spiking Neural Networks for high-speed processing combined with high energy efficiency. The proposed architecture incorporates analog spike response model (SRM) neurons with time-scaling ability and 1T1M RRAM synapses. Although high accuracy of 93.06% on the DVS128 Gesture dataset [317] and 101.05 TSOPS/W energy efficiency were achieved, a complete hardware implementation was not realized, since only the final layer of the neural network was experimentally implemented on the 128 × 24 RRAM crossbar, while the first two layers were emulated using a model with included device variations. In a recent review, Song et al. [12] emphasized the need to increase the capacity of memristor chips up to 1 Gb in order to support dominant AI models, as well as the requirement for extendability to enable the implementation of more complex AI architectures. Motivated by these challenges, Wang et al. [318] propose an inference-only architecture for deploying LLMs on memristor crossbars, combining a dense memristor crossbar for parameter storage with a computation crossbar. The approach targets key

LLM challenges: large model size, non-weight-stationary attention multiplications, and nonlinear operations; however, the reported results represent hardware-level simulations using parameters derived from experimental data. Successfully addressing open issues, including the potential for large-scale production, will require close collaboration among researchers in materials science, device physics, hardware architecture, and algorithm development, as well as hardware–software co-design.

Unique properties of memristive devices open additional advantages for realizing real neuromorphic bio-inspired systems: combination of sensing, memory and processing properties in a unified system. This concept is rather new and only a few publications with the utilization of memristive devices are dedicated to this argument. Probably, the first use of memristive devices for in-memory sensing was applied for registering cancer markers [319]. This concept was further developed, demonstrating the practical application of this approach for cancer medicine [320], prostatitis detection [321], and pulmonary hypertension monitoring [322]. A similar approach introducing the "all-in-one" concept (sensing, memory and processing) was used for optical information analysis [323] and IR detection [324].

Successful implementation of in-memory processing and in-memory sensing is expected to provide a breakthrough in such areas as AI, autonomous robots, implantable sensors and neuroprostheses. For their reliable working, it will be necessary to use training approaches, considered in this review.

To conclude, future neuromorphic systems must have important features, similar to those of the brain. Namely, the same elements must be used for memorizing and processing of the information, the system must allow parallel information processing, and learning of the system must follow bio-plausible algorithms (STDP). The last requirement assumes working in the spike mode. Current efforts in this direction are mainly based on the use of software networks, which require huge consumption of energy. As the energy consumption is among key problems of the modern world, this approach does not seem to be a winning one. Thus, it is necessary to think about new computing hardware paradigms, taking the brain as a biological benchmark. Obviously, it is not the easy task to change completely the architecture and logistics of the hardware. Nevertheless, some preliminary steps have already been done in this direction. Let us try to summarize important features of these future computational hardware systems. They will perform in-memory processing and in-memory sensing functions, which will significantly decrease the energy consumption. Analysis of stimuli of different origin and simulation of cognitive processes will require studying and effectively using such phenomena, as cross-talk of elements and effect of noise, that play an important role in living beings but usually are considered as negative features in modern computational systems. Now we are just beginning to make the first steps in this direction, but, in the case of success, we will have completely new data treatment systems and, probably, will understand better how our brain works.

Author Contributions: Conceptualization, N.M.S., V.V., P.T., S.S., S.G., A.M., M.T., V.K., R.S. and V.E.; methodology, N.M.S., V.V., P.T., S.S., S.G., A.M., M.T., V.K., R.S. and V.E.; formal analysis, V.V., J.M. and S.S.; investigation, N.M.S., V.V., P.T., S.G. and M.T.; resources, N.M.S., V.V., P.T., S.G., A.M. and M.T.; data curation, N.M.S., V.V., P.T. and S.S.; writing—original draft preparation, N.M.S., V.V., P.T., J.M., S.S., S.G., A.M., M.T., V.K., R.S. and V.E.; writing—review and editing, N.M.S., V.V., P.T., A.M., M.T., R.S. and V.E.; visualization, N.M.S., V.V., P.T., J.M. and M.T.; supervision, N.M.S., A.M., M.T., V.K., R.S. and V.E.; project administration, N.M.S., V.V. and P.T.; funding acquisition, N.M.S. and M.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been supported by the Ministry of Science, Technological Development and Innovation (Contract No. 451-03-34/2026-03/200156) and the Faculty of Technical Sciences, University of Novi Sad, through the project "Scientific and Artistic Research Work of Researchers in Teaching and Associate Positions at the Faculty of Technical Sciences, University of Novi Sad 2026" (No. 01-3609/1). This study was also conducted within the framework of the scientific program of the National Center for Physics and Mathematics, section No. 9, "Artificial intelligence and big data in technical, industrial, natural and social systems'', at the facilities of the Laboratory of memristor nanoelectronics (state assignment for the creation of new laboratories for the electronics industry No. FSWR-2025-0006).

Data Availability Statement: No new data were created or analyzed during this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

1T1R	One Transistor, One Memristor
AAM	Amplitude-Adaptive Matching
ABC	Artificial Bee Colony Optimization
AGAD	Analog Gradient Accumulation with Dynamic reference
ANN	Artificial Neural Network
ASIC	Application Specific Integrated Circuit
BDNN	Bayesian Deep Neural Network
BNN	Binary Neural Network
BP	Backpropagation
BP-GA	Backpropagation with Gradient Accumulation
BPTT	Backpropagation Through Time
C2C	Cycle-to-Cycle
CD	Contrastive Divergence
CIL	Chip-in-the-Loop
CIM	Computing-in-Memory
CKFO	Convolution Kernel First Operated
CNN	Convolutional Neural Networks
c-TTV2	Chopped Tiki-Taka Algorithm Version 2
D2D	Device-to-Device
DBN	Deep Belief Networks
DFA	Direct Feedback Alignment
DFATT	Direct Feedback Alignment Through Time
DM-BPNN	Discrete Memristors Backpropagation Neural Network
DNN	Deep Neural Networks
DP-PPO	Decay-based Phase-change Proximal Policy Optimization
DQN	Deep Q Network
DRL	Deep Reinforcement Learning
EaPU	Error-Aware Probabilistic Update
EDP	Energy-Delay Product
ELM	Extreme Learning Machines

ESN	Echo State Network
FC	Fully Connected
FCN	Fully Convolutional Network
FET	Field-Effect Transistor
FGSM	Fast Gradient Sign Method
FMMNN	Fractional-order Multi-delayed Memristive Neural Networks
FOOVFBAMNN	Fractional-Order Octonion-Valued Fuzzy Bidirectional Associative Memory Neural Networks
FPGA	Field-Programmable Gate Array
FSCA	Full-Size Convolutional Algorithm
FSM	Finite State Machine
GA	Genetic Algorithm
GAN	Generative Adversarial Network
HOG	Histogram of Oriented Gradients
HRS	High Resistive State
HTM	Hierarchical Temporal Memory
HW	Hardware
HWA	Hardware-Aware
IFA	Indirect Feedback Alignment
IMC	In-Memory Computing
ITO	Indium Tin Oxide
LGP	Local Gradient Potential
LIF	Leaky Integrate-and-Fire
LFW	Labelled Faces in the Wild dataset
LMS	Least Mean Square
LRS	Low Resistive State
LSTM	Long Short-Term Memory
LTD	Long-Term Potentiation
LTP	Long-Term Potentiation
MAC	Multiply-Accumulate
MbLSTM	Memristor-based Long Short-Term Memory
MCA	Memristor Crossbar Array
MCMC	Markov Chain Monte Carlo
mCNN	Memristor-based Convolutional Neural Network
MCS	Memristive Chaotic Systems
MFCN	Memristive Fully Convolutional Network
MIF	Memristive Integrate-and-Fire
MLNN	Multi-Layer Neural Network
MLP	Multi-Layer Perceptron
MLTNN	Multiple Neural Networks (MLTNN)
MNGA	Memristive Network-based Genetic Algorithm
MNIST	Modified National Institute of Standards and Technology Database

MNN	Memristive Neural Networks
MSE	Mean Square Error
MSNN	Memristive Spiking Neural Networks
MVM	Matrix-vector Multiplication
NN	Neural Network
NVM	Nonvolatile Memory
PCM	Phase Change Memory
PE	Processing Elements
PPO	Proximal Policy Optimization
RBM	Restricted Boltzmann Machines
RC	Reservoir Computing
ReLU	Rectifying Linear Unit
RL	Reinforcement Learning
RMSProp	Root Mean Square Propagation
RNN	Recurrent Neural Network
RPU	Resistive Processing Unit
RRAM/ReRAM	Resistive Random Access Memory
R-W	Read–Write
SBP	Sign Backpropagation Algorithm
SDFA	Spiking Direct Feedback Alignment
SGDM	Stochastic Gradient Descent with Momentum
SGD	Stochastic Gradient Descent
SLNN	Single-Layer Neural Network
SLP	Single-Layer Perceptron
SNN	Spiking Neural Networks
SNU	Single Neuron Unit
SO	Set Only
SDFA	Spiking Direct Feedback Alignment (SDFA)
SR	Set Reset
SSparse DFA	Single-connection Sparse Direct Feedback Alignment
STDP	Spike-Timing Dependent Plasticity
TD	Temporal Difference
TNSA	Transposable Neurosynaptic Array
TT	Tiki-Taka Algorithm
TT v2	Tiki-Taka Algorithm Version 2
VMM	Vector–Matrix Multiplication
WH	Widrow–Hoff Algorithm
WOx	Tungsten Oxide
WTA	Winner-Take-All
ZnOx	Zinc Oxide

References

1. Morra, L.; Mohanty, S.P.; Lamberti, F. Artificial intelligence in consumer electronics. *IEEE Consum. Electron. Mag.* **2020**, *9*, 46–47. [CrossRef]
2. Yaseen, M.; Durai, P.; Gokul, P.; Justin, S.; Anand, A.J. Artificial Intelligence Based Automated Appliances in Smart Home. In *Proceedings of the 2023 Seventh International Conference on Image Information Processing (ICIIP), Solan, India, 22–24 November 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 442–445. [CrossRef]
3. Bajwa, J.; Munir, U.; Nori, A.; Williams, B. Artificial intelligence in healthcare: Transforming the practice of medicine. *Future Healthc. J.* **2021**, *8*, e188–e194. [CrossRef] [PubMed]
4. Bickley, S.J.; Chan, H.F.; Torgler, B. Artificial intelligence in the field of economics. *Scientometrics* **2022**, *127*, 2055–2084. [CrossRef]
5. Park, Y.J.; Jones-Jang, S.M. Surveillance, security, and AI as technological acceptance. *AI Soc.* **2023**, *38*, 2667–2678. [CrossRef]
6. Chatterjee, A. Art in an age of artificial intelligence. *Front. Psychol.* **2022**, *13*, 1024449. [CrossRef] [PubMed]
7. Wang, S.; Wang, F.; Zhu, Z.; Wang, J.; Tran, T.; Du, Z. Artificial intelligence in education: A systematic literature review. *Expert Syst. Appl.* **2024**, *252*, 124167. [CrossRef]
8. Mehonic, A.; Ielmini, D.; Roy, K.; Mutlu, O.; Kvatinsky, S.; Serrano-Gotarredona, T.; Linares-Barranco, B.; Spiga, S.; Savel'ev, S.; Balanov, A.G.; et al. Roadmap to neuromorphic computing with emerging technologies. *APL Mater.* **2024**, *12*, 109201. [CrossRef]
9. Mehonic, A.; Kenyon, A.J. Brain-inspired computing needs a master plan. *Nature* **2022**, *604*, 255–260. [CrossRef] [PubMed]
10. Aguirre, F.; Sebastian, A.; Le Gallo, M.; Song, W.; Wang, T.; Yang, J.J.; Lu, W.; Chang, M.F.; Ielmini, D.; Yang, Y.; et al. Hardware implementation of memristor-based artificial neural networks. *Nat. Commun.* **2024**, *15*, 1974. [CrossRef] [PubMed]
11. Shooshtari, M.; Serrano-Gotarredona, T.; Linares-Barranco, B. Review of Memristors for In-Memory Computing and Spiking Neural Networks. *Adv. Intell. Syst.* **2026**, *8*, e202500806. [CrossRef]
12. Song, M.K.; Kang, J.H.; Zhang, X.; Ji, W.; Ascoli, A.; Messaris, I.; Demirkol, A.S.; Dong, B.; Aggarwal, S.; Wan, W.; et al. Recent advances and future prospects for memristive materials, devices, and systems. *ACS Nano* **2023**, *17*, 11994–12039. [CrossRef] [PubMed]
13. CHUA, L.O. Memristor-The Missing Circuit Element. *IEEE Trans. Circuit Theory* **1971**, *18*, 507–519. [CrossRef]
14. Strukov, D.B.; Snider, G.S.; Stewart, D.R.; Williams, R.S. The missing memristor found. *Nature* **2008**, *453*, 80–83. [CrossRef] [PubMed]
15. Adhikari, S.P.; Sah, M.P.; Kim, H.; Chua, L.O. Three fingerprints of memristor. In *Handbook of Memristor Networks*; Springer: Cham, Swizterland, 2019; pp. 165–196. [CrossRef]
16. Chua, L.O. Memristors on 'edge of chaos'. *Nat. Rev. Electr. Eng.* **2024**, *1*, 614–627. [CrossRef]
17. Chua, L.O. MEMRISTORS on Edge of Chaos: Quo Vadis? *Int. J. Bifurc. Chaos* **2025**, *35*, 2540001. [CrossRef]
18. Dautovic, S.; Samardzic, N.; Juhas, A.; Ascoli, A.; Tetzlaff, R. Analytical solutions for charge and flux in HP ideal generic memristor model with Joglekar and Prodromakis window functions. *IEEE Access* **2024**, *12*, 94870–94884. [CrossRef]
19. Dautovic, S.; Samardzic, N.; Juhas, A. Takacs Model of Hysteresis in Mathematical Modeling of Memristors. *Radioengineering* **2020**, *29*, 147–158. [CrossRef]
20. Teodorovic, P.; Dautovic, S.; Malbasa, V. Recursive Boolean formula minimization algorithms for implication logic. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2013**, *32*, 1829–1833. [CrossRef]
21. Ascoli, A.; Slesazeck, S.; Mähne, H.; Tetzlaff, R.; Mikolajick, T. Nonlinear dynamics of a locally-active memristor. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2015**, *62*, 1165–1174. [CrossRef]
22. Ascoli, A.; Corinto, F.; Senger, V.; Tetzlaff, R. Memristor model comparison. *IEEE Circuits Syst. Mag.* **2013**, *13*, 89–105. [CrossRef]
23. Biolek, D.; Kolka, Z.; Biolková, V.; Biolek, Z.; Kvatinsky, S. (V) TEAM for SPICE simulation of memristive devices with improved numerical performance. *IEEE Access* **2021**, *9*, 30242–30255. [CrossRef]
24. Biolek, Z.; Biolek, D.; Biolkova, V. SPICE model of memristor with nonlinear dopant drift. *Radioengineering* **2009**, *18*, 210–214.
25. Wu, Y.; Wang, Q.; Wang, Z.; Wang, X.; Ayyagari, B.; Krishnan, S.; Chudzik, M.; Lu, W.D. Bulk-switching memristor-based compute-in-memory module for deep neural network training. *Adv. Mater.* **2023**, *35*, 2305465. [CrossRef] [PubMed]
26. Wan, W.; Kubendran, R.; Schaefer, C.; Eryilmaz, S.B.; Zhang, W.; Wu, D.; Deiss, S.; Raina, P.; Qian, H.; Gao, B.; et al. A compute-in-memory chip based on resistive random-access memory. *Nature* **2022**, *608*, 504–512. [CrossRef] [PubMed]
27. Khaddam-Aljameh, R.; Stanisavljevic, M.; Mas, J.F.; Karunaratne, G.; Brändli, M.; Liu, F.; Singh, A.; Müller, S.M.; Egger, U.; Petropoulos, A.; et al. HERMES-core—A 1.59-TOPS/mm 2 PCM on 14-nm CMOS in-memory compute core using 300-ps/LSB linearized CCO-based ADCs. *IEEE J. Solid-State Circuits* **2022**, *57*, 1027–1038. [CrossRef]
28. Tian, M.; Lu, J.; Gao, H.; Wang, H.; Yu, J.; Shi, C. A lightweight spiking GAN model for memristor-centric silicon circuit with on-chip reinforcement adversarial learning. In *Proceedings of the 2022 IEEE International Symposium on Circuits and Systems (ISCAS), Austin, TX, USA, 27 May–1 June 2022*; IEEE: Piscataway, NJ, USA, 2022; pp. 3388–3392. [CrossRef]

29. Leroux, N.; Manea, P.P.; Sudarshan, C.; Finkbeiner, J.; Siegel, S.; Strachan, J.P.; Neftci, E. Analog in-memory computing attention mechanism for fast and energy-efficient large language models. *Nat. Comput. Sci.* **2025**, *5*, 813–824. [CrossRef] [PubMed]
30. Kataeva, I.; Merrikh-Bayat, F.; Zamanidoost, E.; Strukov, D. Efficient training algorithms for neural networks based on memristive crossbar circuits. In *Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 12–17 July 2015*; IEEE: Piscataway, NJ, USA, 2015; pp. 1–8. [CrossRef]
31. Gokmen, T.; Vlasov, Y. Acceleration of deep neural network training with resistive cross-point devices: Design considerations. *Front. Neurosci.* **2016**, *10*, 333. [CrossRef] [PubMed]
32. Prezioso, M.; Merrikh-Bayat, F.; 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] [PubMed]
33. Yao, P.; Wu, H.; Gao, B.; Eryilmaz, S.B.; Huang, X.; Zhang, W.; Zhang, Q.; Deng, N.; Shi, L.; Wong, H.S.P.; et al. Face classification using electronic synapses. *Nat. Commun.* **2017**, *8*, 15199. [CrossRef] [PubMed]
34. Ambrogio, S.; Narayanan, P.; Tsai, H.; Shelby, R.M.; Boybat, I.; Di Nolfo, C.; Sidler, S.; Giordano, M.; Bodini, M.; Farinha, N.C.; et al. Equivalent-accuracy accelerated neural-network training using analogue memory. *Nature* **2018**, *558*, 60–67. [CrossRef] [PubMed]
35. Zhou, E.; Fang, L.; Yang, B. Memristive spiking neural networks trained with unsupervised STDP. *Electronics* **2018**, *7*, 396. [CrossRef]
36. Dong, S.; Chen, Y.; Fan, Z.; Chen, K.; Qin, M.; Zeng, M.; Lu, X.; Zhou, G.; Gao, X.; Liu, J.M. A backpropagation with gradient accumulation algorithm capable of tolerating memristor non-idealities for training memristive neural networks. *Neurocomputing* **2022**, *494*, 89–103. [CrossRef]
37. Liu, J.; Lu, J.; Tang, S.; Zhou, R.; Ma, H.; Lyu, B.; Tian, Y.; Shi, T.; Liu, Q. Error-aware probabilistic training for memristive neural networks. *Nat. Commun.* **2025**, *16*, 11494. [CrossRef] [PubMed]
38. Samardzic, N.M.; Dautovic, S. Implementation of Direct Feedback Alignment Through Time Algorithm in Training of Shallow Fully Memristive Spiking Neural Networks. *Int. J. Circuit Theory Appl.* **2026**, *54*, 2853–2863. [CrossRef]
39. Indiveri, G.; Linares-Barranco, B.; Legenstein, R.; Deligeorgis, G.; Prodromakis, T. Integration of nanoscale memristor synapses in neuromorphic computing architectures. *Nanotechnology* **2013**, *24*, 384010. [CrossRef] [PubMed]
40. Pantazi, A.; Woźniak, S.; Tuma, T.; Eleftheriou, E. All-memristive neuromorphic computing with level-tuned neurons. *Nanotechnology* **2016**, *27*, 355205. [CrossRef] [PubMed]
41. Kudithipudi, D.; Schuman, C.; Vineyard, C.M.; Pandit, T.; Merkel, C.; Kubendran, R.; Aimone, J.B.; Orchard, G.; Mayr, C.; Benosman, R.; et al. Neuromorphic computing at scale. *Nature* **2025**, *637*, 801–812. [CrossRef] [PubMed]
42. Kim, S.Y.; Zhang, H.; Rivera-Sierra, G.; Fenollosa, R.; Rubio-Magnieto, J.; Bisquert, J. Introduction to neuromorphic functions of memristors: The inductive nature of synapse potentiation. *J. Appl. Phys.* **2025**, *137*, 111101. [CrossRef]
43. Wang, X.; Zhu, Y.; Zhou, Z.; Chen, X.; Jia, X. Memristor-based spiking neuromorphic systems toward brain-inspired perception and computing. *Nanomaterials* **2025**, *15*, 1130. [CrossRef] [PubMed]
44. Kang, S.M.; Choi, D.; Eshraghian, J.K.; Zhou, P.; Kim, J.; Kong, B.S.; Zhu, X.; Demirkol, A.S.; Ascoli, A.; Tetzlaff, R.; et al. How to build a memristive integrate-and-fire model for spiking neuronal signal generation. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2021**, *68*, 4837–4850. [CrossRef]
45. Huang, J.; Serb, A.; Stathopoulos, S.; Prodromakis, T. Text classification in memristor-based spiking neural networks. *Neuromorphic Comput. Eng.* **2023**, *3*, 014003. [CrossRef]
46. Dalgaty, T.; Moro, F.; Demirağ, Y.; De Pra, A.; Indiveri, G.; Vianello, E.; Payvand, M. Mosaic: In-memory computing and routing for small-world spike-based neuromorphic systems. *Nat. Commun.* **2024**, *15*, 142. [CrossRef] [PubMed]
47. Serb, A.; Corna, A.; George, R.; Khiat, A.; Rocchi, F.; Reato, M.; Maschietto, M.; Mayr, C.; Indiveri, G.; Vassanelli, S.; et al. Memristive synapses connect brain and silicon spiking neurons. *Sci. Rep.* **2020**, *10*, 2590. [CrossRef] [PubMed]
48. Makarov, V.A.; Lobov, S.A.; Shchanikov, S.; Mikhaylov, A.; Kazantsev, V.B. Toward reflective spiking neural networks exploiting memristive devices. *Front. Comput. Neurosci.* **2022**, *16*, 859874. [CrossRef] [PubMed]
49. Chen, P.; Zhang, B.; He, E.; Xiao, Y.; Liu, F.; Lin, P.; Wang, Z.; Pan, G. Towards scalable memristive hardware for spiking neural networks. *Mater. Horiz.* **2025**, *12*, 2820–2839. [CrossRef] [PubMed]
50. Saïghi, S.; Mayr, C.G.; Serrano-Gotarredona, T.; Schmidt, H.; Lecerf, G.; Tomas, J.; Grollier, J.; Boyn, S.; Vincent, A.F.; Querlioz, D.; et al. Plasticity in memristive devices for spiking neural networks. *Front. Neurosci.* **2015**, *9*, 51. [CrossRef] [PubMed]
51. Mehonic, A.; Vrajitoarea, A.; Cueff, S.; Hudziak, S.; Howe, H.; Labbe, C.; Rizk, R.; Pepper, M.; Kenyon, A. Quantum conductance in silicon oxide resistive memory devices. *Sci. Rep.* **2013**, *3*, 2708. [CrossRef] [PubMed]
52. Samardzić, N.; Mionić, M.; Dakić, B.; Hofmann, H.; Dautović, S.; Stojanović, G. Analysis of quantized electrical characteristics of microscale TiO 2 ink-jet printed memristor. *IEEE Trans. Electron Devices* **2015**, *62*, 1898–1904. [CrossRef]
53. Samardzic, N.; Bajac, B.; Srdic, V.; Stojanovic, G. Conduction mechanisms in multiferroic multilayer BaTiO3/NiFe2O4/BaTiO3 memristors. *J. Electron. Mater.* **2017**, *46*, 5492–5496. [CrossRef]

54. Milano, G.; Aono, M.; Boarino, L.; Celano, U.; Hasegawa, T.; Kozicki, M.; Majumdar, S.; Menghini, M.; Miranda, E.; Ricciardi, C.; et al. Quantum conductance in memristive devices: Fundamentals, developments, and applications. *Adv. Mater.* **2022**, *34*, 2201248. [CrossRef] [PubMed]
55. Tzouvadaki, I.; De Micheli, G.; Carrara, S. Memristive biosensors for ultrasensitive diagnostics and therapeutics. In *Applications of Emerging Memory Technology: Beyond Storage*; Springer: Cham, Switerland, 2020; pp. 133–157. [CrossRef]
56. Bouzouita, M.; Pathak, S.; Zayer, F.; Belgacem, H.; Tzouvadaki, I. Advanced memristive architectures based on nanomaterials for biomedical applications: A mini review. *Front. Nanotechnol.* **2025**, *7*, 1558743. [CrossRef]
57. Bouzouita, M.; Zayer, F.; Tzouvadaki, I.; Carrara, S.; Belgacem, H. Advancing Biosensing with Memristive Technologies and Machine Learning: A Review. *IEEE Sens. J.* **2026**, *26*, 3558–3576. [CrossRef]
58. Shchanikov, S.; Bordanov, I.; Kucherik, A.; Gryaznov, E.; Mikhaylov, A. Neuromorphic Analog Machine Vision Enabled by Nanoelectronic Memristive Devices. *Appl. Sci.* **2023**, *13*, 13309. [CrossRef]
59. Wang, Z.; Joshi, S.; Savel'ev, S.E.; Jiang, H.; Midya, R.; Lin, P.; Hu, M.; Ge, N.; Strachan, J.P.; Li, Z.; et al. Memristors with diffusive dynamics as synaptic emulators for neuromorphic computing. *Nat. Mater.* **2017**, *16*, 101–108. [CrossRef] [PubMed]
60. Samardžić, N.M.; Bajac, B.; Bajić, J.; Đurđić, E.; Miljević, B.; Srdić, V.V.; Stojanović, G.M. Photoresistive switching of multiferroic thin film memristors. *Microelectron. Eng.* **2018**, *187–188*, 139–143. [CrossRef]
61. Shooshtari, M.; Serrano Gotarredona, T.; Linares-Barranco, B. Enhancing the performance of HfO2-based memristors with a thin Al2O3 layer: A comparative study. *J. Phys. D Appl. Phys.* **2025**, *58*, 455101. [CrossRef]
62. Chanthbouala, A.; Garcia, V.; Cherifi, R.O.; Bouzehouane, K.; Fusil, S.; Moya, X.; Xavier, S.; Yamada, H.; Deranlot, C.; Mathur, N.D.; et al. A ferroelectric memristor. *Nat. Mater.* **2012**, *11*, 860–864. [CrossRef] [PubMed]
63. Kuzum, D.; Jeyasingh, R.G.; Lee, B.; Wong, H.S.P. Nanoelectronic programmable synapses based on phase change materials for brain-inspired computing. *Nano Lett.* **2012**, *12*, 2179–2186. [CrossRef] [PubMed]
64. Nandakumar, S.; Le Gallo, M.; Piveteau, C.; Joshi, V.; Mariani, G.; Boybat, I.; Karunaratne, G.; Khaddam-Aljameh, R.; Egger, U.; Petropoulos, A.; et al. Mixed-precision deep learning based on computational memory. *Front. Neurosci.* **2020**, *14*, 406. [CrossRef] [PubMed]
65. Mikhaylov, A.N.; Gryaznov, E.G.; Koryazhkina, M.N.; Bordanov, I.A.; Shchanikov, S.A.; Telminov, O.A.; Kazantsev, V.B. Neuromorphic Computing Based on CMOS-Integrated Memristive Arrays: Current State and Perspectives. *Supercomput. Front. Innov.* **2023**, *10*, 77–103. [CrossRef]
66. Yao, P.; Wu, H.; Gao, B.; Tang, J.; Zhang, Q.; Zhang, W.; Yang, J.J.; Qian, H. Fully hardware-implemented memristor convolutional neural network. *Nature* **2020**, *577*, 641–646. [CrossRef] [PubMed]
67. Hasan, R.; Yakopcic, C.; Taha, T.M. Ex-situ training of dense memristor crossbar for neuromorphic applications. In *Proceedings of the 2015 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH 15), Boston, MA, USA, 8–10 July 2015*; IEEE: Piscataway, NJ, USA, 2015; pp. 75–81. [CrossRef]
68. Zhang, Y.; Wang, X.; Friedman, E.G. Memristor-based circuit design for multilayer neural networks. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2018**, *65*, 677–686. [CrossRef]
69. Zhong, Y.; Tang, J.; Li, X.; Liang, X.; Liu, Z.; Li, Y.; Xi, Y.; Yao, P.; Hao, Z.; Gao, B.; et al. A memristor-based analogue reservoir computing system for real-time and power-efficient signal processing. *Nat. Electron.* **2022**, *5*, 672–681. [CrossRef]
70. Mikhaylov, A.; Pimashkin, A.; Pigareva, Y.; Gerasimova, S.; Gryaznov, E.; Shchanikov, S.; Zuev, A.; Talanov, M.; Lavrov, I.; Demin, V.; et al. Neurohybrid Memristive CMOS-Integrated Systems for Biosensors and Neuroprosthetics. *Front. Neurosci.* **2020**, *14*, 358. [CrossRef] [PubMed]
71. Shi, C.; Lu, J.; Wang, Y.; Li, P.; Tian, M. Exploiting memristors for neuromorphic reinforcement learning. In *Proceedings of the 2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS), Washington DC, USA, 6–9 June 2021*; IEEE: Piscataway, NJ, USA, 2021; pp. 1–4. [CrossRef]
72. Alibart, F.; Zamanidoost, E.; Strukov, D.B. Pattern classification by memristive crossbar circuits using ex situ and in situ training. *Nat. Commun.* **2013**, *4*, 2072. [CrossRef] [PubMed]
73. Burr, G.W.; Shelby, R.M.; Sidler, S.; Di Nolfo, C.; Jang, J.; Boybat, I.; Shenoy, R.S.; Narayanan, P.; Virwani, K.; Giacometti, E.U.; et al. Experimental demonstration and tolerancing of a large-scale neural network (165,000 synapses) using phase-change memory as the synaptic weight element. *IEEE Trans. Electron Devices* **2015**, *62*, 3498–3507. [CrossRef]
74. Xu, W.; Wang, J.; Yan, X. Advances in memristor-based neural networks. *Front. Nanotechnol.* **2021**, *3*, 645995. [CrossRef]
75. Wang, Z.; Li, Z.; Xia, Z.; Sun, X.; Meng, J.; Wang, T. Emerging CMOS Compatible Memristor for Storage Technology and Neuromorphic Computing Applications. *Chip* **2026**, *5*, 100183. [CrossRef]
76. 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] [PubMed]

77. Nikiruy, K.; Emelyanov, A.; Demin, V.; Sitnikov, A.; Minnekhanov, A.; Rylkov, V.; Kashkarov, P.; Kovalchuk, M. Dopamine-like STDP modulation in nanocomposite memristors. *AIP Adv.* **2019**, *9*, 065116. [CrossRef]
78. Chua, L. Memristor, Hodgkin-Huxley, and edge of chaos. In *Handbook of Memristor Networks*; Springer: Cham, Switzerland, 2019; pp. 287–313. [CrossRef]
79. Rozenberg, M.; Schneegans, O.; Stoliar, P. An ultra-compact leaky-integrate-and-fire model for building spiking neural networks. *Sci. Rep.* **2019**, *9*, 11123. [CrossRef] [PubMed]
80. Tan, Y.; Wang, C. A simple locally active memristor and its application in HR neurons. *Chaos Interdiscip. J. Nonlinear Sci.* **2020**, *30*, 053118. [CrossRef] [PubMed]
81. Gerasimov, Y.; Zykov, E.; Prudnikov, N.; Talanov, M.; Toschev, A.; Erokhin, V. On the Organic Memristive Device Resistive Switching Efficacy. *Chaos Solitons Fractals* **2021**, *143*, 110549. [CrossRef]
82. Kipelkin, I.M.; Gerasimova, S.A.; Guseinov, D.V.; Belov, A.I.; Tyukin, I.Y.; Mikhaylov, A.N.; Kazantsev, V.B. Periodic and quasi-periodic dynamics of neuronal oscillators based on an inorganic memristive device. *Chaos Solitons Fractals* **2026**, *208*, 118304. [CrossRef]
83. Masaev, D.N.; Suleimanova, A.A.; Prudnikov, N.V.; Serenko, M.V.; Emelyanov, A.V.; Demin, V.A.; Lavrov, I.A.; Talanov, M.O.; Erokhin, V.V. Memristive Circuit-Based Model of Central Pattern Generator to Reproduce Spinal Neuronal Activity in Walking Pattern. *Front. Neurosci.* **2023**, *17*, 1124950. [CrossRef] [PubMed]
84. Krestinskaya, O.; James, A.P.; Chua, L.O. Neuromemristive circuits for edge computing: A review. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 4–23. [CrossRef] [PubMed]
85. Sung, C.; Hwang, H.; Yoo, I.K. Perspective: A review on memristive hardware for neuromorphic computation. *J. Appl. Phys.* **2018**, *124*, 151903. [CrossRef]
86. Huang, H.M.; Wang, Z.; Wang, T.; Xiao, Y.; Guo, X. Artificial neural networks based on memristive devices: From device to system. *Adv. Intell. Syst.* **2020**, *2*, 2000149. [CrossRef]
87. Yang, F.; Ma, J.; Wu, F. Review on memristor application in neural circuit and network. *Chaos Solitons Fractals* **2024**, *187*, 115361. [CrossRef]
88. Bisquert, J.; Shim, W.; Kim, S.Y.; Linares-Barranco, B. Synaptic function in memristor devices for neuromorphic circuit applications. *Adv. Electron. Mater.* **2025**, *11*, 2400903. [CrossRef]
89. Peng, H.; Gan, L.; Guo, X. Memristor-based spiking neural networks: Cooperative development of neural network architecture/algorithms and memristors. *Chip* **2024**, *3*, 100093. [CrossRef]
90. Schuman, C.D.; Potok, T.E.; Patton, R.M.; Birdwell, J.D.; Dean, M.E.; Rose, G.S.; Plank, J.S. A survey of neuromorphic computing and neural networks in hardware. *arXiv* **2017**, arXiv:1705.06963. [CrossRef]
91. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
92. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [CrossRef]
93. Haykin, S.O. *Neural Networks and Learning Machines*, 3rd ed.; Pearson Education: Upper Saddle River, NJ, USA; McMaster University: Hamilton, ON, Canada, 2009.
94. Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychol. Rev.* **1958**, *65*, 386–408. [CrossRef] [PubMed]
95. Widrow, B.; Hoff, M.E. Adaptive switching circuits. In *Neurocomputing: Foundations of Research*; MIT Press: Cambridge, MA, USA, 1988; pp. 123–134. [CrossRef]
96. Lin, Y.P.; Bennett, C.H.; Cabaret, T.; Vodenicarevic, D.; Chabi, D.; Querlioz, D.; Jousselme, 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]
97. Chabi, D.; Zhao, W.; Querlioz, D.; Klein, J.O. Robust neural logic block (NLB) based on memristor crossbar array. In *Proceedings of the 2011 IEEE/ACM International Symposium on Nanoscale Architectures, San Diego, CA, USA, 8–9 June 2011*; IEEE: Piscataway, NJ, USA, 2011; pp. 137–143. [CrossRef]
98. Zhang, X.; Wang, X.; Ge, Z.; Li, Z.; Wu, M.; Borah, S. A novel memristive neural network circuit and its application in character recognition. *Micromachines* **2022**, *13*, 2074. [CrossRef] [PubMed]
99. Cai, F.; Correll, J.M.; Lee, S.H.; Lim, Y.; Bothra, V.; Zhang, Z.; Flynn, M.P.; Lu, W.D. A fully integrated reprogrammable memristor–CMOS system for efficient multiply–Accumulate operations. *Nat. Electron.* **2019**, *2*, 290–299. [CrossRef]
100. Schiffmann, W.; Joost, M.; Werner, R. *Optimization of the Backpropagation Algorithm for Training Multilayer Perceptrons*; Technical Report; University of Koblenz, Institute of Physics: Koblenz, Germany, 1994.
101. Belhumeur, P.N.; Hespanha, J.P.; Kriegman, D.J. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1997**, *19*, 711–720. [CrossRef]

102. Baughman, D.; Liu, Y. Fundamental and practical aspects of neural computing. In *Neural Networks in Bioprocessing and Chemical Engineering*; Academic Press: Cambridge, MA, USA, 1995; pp. 21–109. [CrossRef]
103. Adhikari, S.P.; Yang, C.; Kim, H.; Chua, L.O. Memristor bridge synapse-based neural network and its learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2012**, *23*, 1426–1435. [CrossRef] [PubMed]
104. Hasan, R.; Taha, T.M. Enabling back propagation training of memristor crossbar neuromorphic processors. In *Proceedings of the 2014 International Joint Conference on Neural Networks (IJCNN), Beijing, China, 6–11 July 2014*; IEEE: Piscataway, NJ, USA, 2014; pp. 21–28. [CrossRef]
105. Wolberg, W.; Mangasarian, O.; Street, N.; Street, W. Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository, 1993. Available online: https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic (accessed on 12 October 2026). [CrossRef]
106. Zhang, Y.; Shen, Y.; Wang, X.; Cao, L. A novel design for memristor-based logic switch and crossbar circuits. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2015**, *62*, 1402–1411. [CrossRef]
107. Deng, L. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Process. Mag.* **2012**, *29*, 141–142. [CrossRef]
108. Soudry, D.; Di Castro, D.; Gal, A.; Kolodny, A.; Kvatinsky, S. Memristor-based multilayer neural networks with online gradient descent training. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 2408–2421. [CrossRef] [PubMed]
109. Rosenthal, E.; Greshnikov, S.; Soudry, D.; Kvatinsky, S. A fully analog memristor-based neural network with online gradient training. In *Proceedings of the 2016 IEEE International Symposium on Circuits and Systems (ISCAS), Montreal, QC, Canada, 22–25 May 2016*; IEEE: Piscataway, NJ, USA, 2016; pp. 1394–1397. [CrossRef]
110. Krestinskaya, O.; Salama, K.N.; James, A.P. Analog backpropagation learning circuits for memristive crossbar neural networks. In *Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 27–30 May 2018*; IEEE: Piscataway, NJ, USA, 2018; pp. 1–5. [CrossRef]
111. Krestinskaya, O.; Salama, K.N.; James, A.P. Learning in memristive neural network architectures using analog backpropagation circuits. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2019**, *66*, 719–732. [CrossRef]
112. Mehonic, A.; Joksas, D.; Ng, W.H.; Buckwell, M.; Kenyon, A.J. Simulation of inference accuracy using realistic RRAM devices. *Front. Neurosci.* **2019**, *13*, 593. [CrossRef] [PubMed]
113. Milo, V.; Zambelli, C.; Olivo, P.; Pérez, E.; K Mahadevaiah, M.; G Ossorio, O.; Wenger, C.; Ielmini, D. Multilevel HfO2-based RRAM devices for low-power neuromorphic networks. *APL Mater.* **2019**, *7*, 081120. [CrossRef]
114. Ansari, M.; Fayyazi, A.; Banagozar, A.; Maleki, M.A.; Kamal, M.; Afzali-Kusha, A.; Pedram, M. PHAX: Physical characteristics aware ex-situ training framework for inverter-based memristive neuromorphic circuits. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2018**, *37*, 1602–1613. [CrossRef]
115. Vahdat, S.; Kamal, M.; Afzali-Kusha, A.; Pedram, M. Offline training improvement of inverter-based memristive neural networks using inverter voltage characteristic smoothing. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**, *67*, 3442–3446. [CrossRef]
116. Gokmen, T.; Haensch, W. Algorithm for training neural networks on resistive device arrays. *Front. Neurosci.* **2020**, *14*, 103. [CrossRef] [PubMed]
117. Gokmen, T. Enabling training of neural networks on noisy hardware. *Front. Artif. Intell.* **2021**, *4*, 699148. [CrossRef] [PubMed]
118. Kim, Y.; Gokmen, T.; Miyazoe, H.; Solomon, P.; Kim, S.; Ray, A.; Doevenspeck, J.; Khan, R.S.; Narayanan, V.; Ando, T. Neural network learning using non-ideal resistive memory devices. *Front. Nanotechnol.* **2022**, *4*, 1008266. [CrossRef]
119. Li, C.; Belkin, D.; Li, Y.; Yan, P.; Hu, M.; Ge, N.; Jiang, H.; Montgomery, E.; Lin, P.; Wang, Z.; et al. Efficient and self-adaptive in-situ learning in multilayer memristor neural networks. *Nat. Commun.* **2018**, *9*, 2385. [CrossRef] [PubMed]
120. Prajapat, S.; Mondal, M.N.; Sur-Kolay, S. Efficient and Fault-Tolerant Memristive Neural Networks with In-Situ Training. *arXiv* **2025**, arXiv:2507.20193. [CrossRef]
121. Kiani, F.; Yin, J.; Wang, Z.; Yang, J.J.; Xia, Q. A fully hardware-based memristive multilayer neural network. *Sci. Adv.* **2021**, *7*, eabj4801. [CrossRef] [PubMed]
122. Liu, X.; Zeng, Z. Memristor crossbar architectures for implementing deep neural networks. *Complex Intell. Syst.* **2022**, *8*, 787–802. [CrossRef]
123. Feng, R.; Li, J.; Xie, S.; Mao, X. Efficient training method for memristor-based array using 1T1M synapse. *IEEE Trans. Circuits Syst. II Express Briefs* **2023**, *70*, 2410–2414. [CrossRef]
124. Kvatinsky, S.; Ramadan, M.; Friedman, E.G.; Kolodny, A. VTEAM: A general model for voltage-controlled memristors. *IEEE IEEE Trans. Circuits Syst. II Express Briefs* **2015**, *62*, 786–790. [CrossRef]
125. Chua, L.O.; Kang, S.M. Memristive devices and systems. *Proc. IEEE* **1976**, *64*, 209–223. [CrossRef]
126. Yakopcic, C.; Taha, T.M.; Subramanyam, G.; Pino, R.E. Generalized memristive device SPICE model and its application in circuit design. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2013**, *32*, 1201–1214. [CrossRef]

127. Oblea, A.S.; Timilsina, A.; Moore, D.; Campbell, K.A. Silver chalcogenide based memristor devices. In *Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, Spain, 18–23 July 2010*; IEEE: Piscataway, NJ, USA, 2010; pp. 1–3. [CrossRef]
128. Miller, K.; Nalwa, K.S.; Bergerud, A.; Neihart, N.M.; Chaudhary, S. Memristive behavior in thin anodic titania. *IEEE Electron Device Lett.* **2010**, *31*, 737–739. [CrossRef]
129. Mehta, S. NASA: Asteroids Classification. Kaggle Dataset, 2018. Data Collected From the NASA NeoWs API. Availabe online: https://www.kaggle.com/datasets/shrutimehta/nasa-asteroids-classification (accessed on 12 October 2026).
130. Fisher, R.A. Iris. UCI Machine Learning Repository, 1936. Availabe online: https://archive.ics.uci.edu/dataset/53/iris (accessed on 12 October 2026). [CrossRef]
131. Prajaprati, S.; Mondal, M.N.; Sur-Kolay, S. Memristive neural network with efficient in-situ supervised training. In *Proceedings of the 2022 IEEE 35th International System-on-Chip Conference (SOCC), Belfast, UK, 5–8 September 2022*; IEEE: Piscataway, NJ, USA, 2022; pp. 1–6. [CrossRef]
132. Mikhaylov, A.N.; Morozov, O.A.; Ovchinnikov, P.E.; Antonov, I.N.; Belov, A.I.; Korolev, D.S.; Sharapov, A.N.; Gryaznov, E.G.; Gorshkov, O.N.; Pigareva, Y.I.; et al. One-Board Design and Simulation of Double-Layer Perceptron Based on Metal-Oxide Memristive Nanostructures. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 371–379. [CrossRef]
133. Joksas, D.; Muñoz-González, L.; Lupu, E.; Mehonic, A. Nonideality-aware training makes memristive networks more robust to adversarial attacks. *APL Mach. Learn.* **2025**, *3*, 016111. [CrossRef]
134. Günakın, S.; Çam Taşkıran, Z.G. A memristive synaptic circuit and optimization algorithm for synaptic control. *Cogn. Neurodynamics* **2025**, *19*, 73. [CrossRef] [PubMed]
135. Hasan, R.; Alam, M.S.; Taha, T.M. Energy-Efficient Training of Memristor Crossbar-Based Multi-Layer Neural Networks. *Chips* **2025**, *4*, 38. [CrossRef]
136. Zhang, Q.; Wu, H.; Yao, P.; Zhang, W.; Gao, B.; Deng, N.; Qian, H. Sign backpropagation: An on-chip learning algorithm for analog RRAM neuromorphic computing systems. *Neural Netw.* **2018**, *108*, 217–223. [CrossRef] [PubMed]
137. He, S.; Sun, K.; Peng, Y.; Wang, L. Modeling of discrete fracmemristor and its application. *Aip Adv.* **2020**, *10*, 015332. [CrossRef]
138. He, S.; Liu, J.; Wang, H.; Sun, K. A discrete memristive neural network and its application for character recognition. *Neurocomputing* **2023**, *523*, 1–8. [CrossRef]
139. Werbos, P.J. Backpropagation through time: What it does and how to do it. *Proc. IEEE* **2002**, *78*, 1550–1560. [CrossRef]
140. Mozer, M.C. A Focused Backpropagation Algorithm for Temporal Pattern Recognition. *Complex Syst.* **1989**, *3*, 349–381.
141. Guo, W.; Fouda, M.E.; Eltawil, A.M.; Salama, K.N. Efficient training of spiking neural networks with temporally-truncated local backpropagation through time. *Front. Neurosci.* **2023**, *17*, 1047008. [CrossRef] [PubMed]
142. Zhou, P.; Choi, D.U.; Kang, S.M.; Eshraghian, J.K. Backpropagating errors through memristive spiking neural networks. In *Proceedings of the 2023 IEEE International Symposium on Circuits and Systems (ISCAS), Monterey, CA, USA, 21–25 May 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 1–5. [CrossRef]
143. Zhou, P.; Choi, D.U.; Lu, W.D.; Kang, S.M.; Eshraghian, J.K. Gradient-based neuromorphic learning on dynamical RRAM arrays. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2022**, *12*, 888–897. [CrossRef]
144. Neftci, E.O.; Mostafa, H.; Zenke, F. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Process. Mag.* **2019**, *36*, 51–63. [CrossRef]
145. Wang, Z.; Joshi, S.; Savel'ev, S.; Song, W.; Midya, R.; Li, Y.; Rao, M.; Yan, P.; Asapu, S.; Zhuo, Y.; et al. Fully memristive neural networks for pattern classification with unsupervised learning. *Nat. Electron.* **2018**, *1*, 137–145. [CrossRef]
146. Samardzic, N.M.; Bajic, J.S.; Sekulic, D.L.; Dautovic, S. Volatile memristor in leaky integrate-and-fire neurons: Circuit simulation and experimental study. *Electronics* **2022**, *11*, 894. [CrossRef]
147. Smagulova, K.; James, A.P. A survey on LSTM memristive neural network architectures and applications. *Eur. Phys. J. Spec. Top.* **2019**, *228*, 2313–2324. [CrossRef]
148. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
149. Gers, F.A.; Schmidhuber, J.; Cummins, F. Learning to forget: Continual prediction with LSTM. *Neural Comput.* **2000**, *12*, 2451–2471. [CrossRef] [PubMed]
150. Liu, X.; Zeng, Z.; Wunsch II, D.C. Memristor-based LSTM network with in situ training and its applications. *Neural Netw.* **2020**, *131*, 300–311. [CrossRef] [PubMed]
151. Gokmen, T.; Rasch, M.J.; Haensch, W. Training LSTM networks with resistive cross-point devices. *Front. Neurosci.* **2018**, *12*, 745. [CrossRef] [PubMed]
152. Smagulova, K.; Adam, K.; Krestinskaya, O.; James, A.P. Design of cmos-memristor circuits for lstm architecture. In *Proceedings of the 2018 IEEE International Conference on Electron Devices and Solid State Circuits (EDSSC), Shenzhen, China, 6–8 June 2018*; IEEE: Piscataway, NJ, USA, 2018; pp. 1–2. [CrossRef]

153. Adam, K.; Smagulova, K.; James, A.P. Memristive LSTM network hardware architecture for time-series predictive modeling problems. In *Proceedings of the 2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), Chengdu, China, 26–30 October 2018*; IEEE: Piscataway, NJ, USA, 2018; pp. 459–462. [CrossRef]
154. Smagulova, K.; Krestinskaya, O.; James, A.P. A memristor-based long short term memory circuit. *Analog. Integr. Circuits Signal Process.* **2018**, *95*, 467–472. [CrossRef]
155. Adam, K.; Smagulova, K.; Krestinskaya, O.; James, A.P. Wafer quality inspection using memristive LSTM, ANN, DNN and HTM. In *Proceedings of the 2018 IEEE Electrical Design of Advanced Packaging and Systems Symposium (EDAPS), Chandigarh, India, 16–18 December 2018*; IEEE: Piscataway, NJ, USA, 2018; pp. 1–3. [CrossRef]
156. Li, C.; Wang, Z.; Rao, M.; Belkin, D.; Song, W.; Jiang, H.; Yan, P.; Li, Y.; Lin, P.; Hu, M.; et al. Long short-term memory networks in memristor crossbar arrays. *Nat. Mach. Intell.* **2019**, *1*, 49–57. [CrossRef]
157. Hinton, G.E. Training products of experts by minimizing contrastive divergence. *Neural Comput.* **2002**, *14*, 1771–1800. [CrossRef] [PubMed]
158. Sheri, A.M.; Rafique, A.; Pedrycz, W.; Jeon, M. Contrastive divergence for memristor-based restricted Boltzmann machine. *Eng. Appl. Artif. Intell.* **2015**, *37*, 336–342. [CrossRef]
159. Wang, W.; Hoffer, B.; Greenberg-Toledo, T.; Li, Y.; Zou, M.; Herbelin, E.; Ronen, R.; Xu, X.; Zhao, Y.; Yang, J.; et al. Efficient training of the memristive deep belief net immune to non-idealities of the synaptic devices. *Adv. Intell. Syst.* **2022**, *4*, 2100249. [CrossRef]
160. Nøkland, A. Direct feedback alignment provides learning in deep neural networks. In *Advances in Neural Information Processing Systems, NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016*; Curran Associates Inc.: Red Hook, NY, USA, 2016; Volume 29.
161. Crafton, B.; Parihar, A.; Gebhardt, E.; Raychowdhury, A. Direct Feedback Alignment with Sparse Connections for Local Learning. *Front. Neurosci.* **2019**, *13*, 525. [CrossRef] [PubMed]
162. Crafton, B.; West, M.; Basnet, P.; Vogel, E.; Raychowdhury, A. Local Learning in RRAM Neural Networks with Sparse Direct Feedback Alignment. In *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED), Lausanne, Switzerland, 29–31 July 2019*; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6. [CrossRef]
163. Bhattacharjee, A.; Shi, J.; Chen, W.C.; Wang, X.; Panda, P. Low-Power Spike-Based Wearable Analytics on RRAM Crossbars. In *Proceedings of the 2025 IEEE International Symposium on Circuits and Systems (ISCAS), London, UK, 25–28 May 2025; IEEE: Piscataway, NJ, USA, 2025*; pp. 1–5. [CrossRef]
164. Ren, H.; He, Y.; Wong, K.; Bao, R.; Lin, N.; Wang, Z.; Shang, D. When Pipelined In-Memory Accelerators Meet Spiking Direct Feedback Alignment: A Co-Design for Neuromorphic Edge Computing. In *Proceedings of the 2025 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Munich, Germany, 26–30 October 2025*; IEEE: Piscataway, NJ, USA, 2025; pp. 1–9. [CrossRef]
165. Yakopcic, C.; Alom, M.Z.; Taha, T.M. Memristor crossbar deep network implementation based on a convolutional neural network. In *Proceedings of the 2016 International Joint Conference on Neural Networks (IJCNN), Vancouver, BC, Canada, 24–29 July 2016*; IEEE: Piscataway, NJ, USA, 2016; pp. 963–970. [CrossRef]
166. Liu, J.; Li, Z.; Tang, Y.; Hu, W.; Wu, J. 3D Convolutional Neural Network based on memristor for video recognition. *Pattern Recognit. Lett.* **2020**, *130*, 116–124. [CrossRef]
167. Yi, W.; Perner, F.; Qureshi, M.S.; Abdalla, H.; Pickett, M.D.; Yang, J.J.; Zhang, M.X.M.; Medeiros-Ribeiro, G.; Williams, R.S. Feedback write scheme for memristive switching devices. *Appl. Phys. A* **2011**, *102*, 973–982. [CrossRef]
168. Yakopcic, C.; Alom, M.Z.; Taha, T.M. Extremely parallel memristor crossbar architecture for convolutional neural network implementation. In *Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017*; IEEE: Piscataway, NJ, USA, 2017; pp. 1696–1703. [CrossRef]
169. Wen, S.; Wei, H.; Zeng, Z.; Huang, T. Memristive fully convolutional network: An accurate hardware image-segmentor in deep learning. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 324–334. [CrossRef]
170. Wen, S.; Chen, J.; Wu, Y.; Yan, Z.; Cao, Y.; Yang, Y.; Huang, T. CKFO: Convolution kernel first operated algorithm with applications in memristor-based convolutional neural network. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2021**, *40*, 1640–1647. [CrossRef]
171. Chellapilla, K.; Puri, S.; Simard, P. High performance convolutional neural networks for document processing. In *Proceedings of the Tenth International Workshop on Frontiers in Handwriting Recognition, La Baule, France, 23–26 October 2006*; Suvisoft: Pirkanmaa, Finland, 2006.
172. Tan, J.; Shen, S.; Duan, S.; Wang, L. An efficient full-size convolutional computing method based on memristor crossbar. *Artif. Intell. Rev.* **2024**, *57*, 158. [CrossRef]
173. Lin, P.; Li, C.; Wang, Z.; Li, Y.; Jiang, H.; Song, W.; Rao, M.; Zhuo, Y.; Upadhyay, N.K.; Barnell, M.; et al. Three-dimensional memristor circuits as complex neural networks. *Nat. Electron.* **2020**, *3*, 225–232. [CrossRef]

174. Jiang, N.; Jiang, M.; Huang, H.; Yang, L.; Huang, Y.; Zeng, Z. A parallel read-write circuit with fast amplitude-adaptive matching scheme to memristor crossbar array. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2025**, *72*, 5898–5911. [CrossRef]
175. Wen, S.; Wei, H.; Yang, Y.; Guo, Z.; Zeng, Z.; Huang, T.; Chen, Y. Memristive LSTM network for sentiment analysis. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *51*, 1794–1804. [CrossRef]
176. Zhang, W.; Pan, L.; Yan, X.; Zhao, G.; Chen, H.; Wang, X.; Tay, B.K.; Zhong, G.; Li, J.; Huang, M. Hardware-friendly stochastic and adaptive learning in memristor convolutional neural networks. *Adv. Intell. Syst.* **2021**, *3*, 2100041. [CrossRef]
177. Chen, B.; Yang, H.; Song, B.; Meng, D.; Yan, X.; Li, Y.; Wang, Y.; Hu, P.; Ou, T.H.; Barnell, M.; et al. A memristor-based hybrid analog-digital computing platform for mobile robotics. *Sci. Robot.* **2020**, *5*, eabb6938. [CrossRef] [PubMed]
178. Yu, Y.; Yang, C.; Deng, Q.; Nyima, T.; Liang, S.; Zhou, C. Memristive network-based genetic algorithm and its application to image edge detection. *J. Syst. Eng. Electron.* **2021**, *32*, 1062–1070. [CrossRef]
179. Kim, D.H.; Kim, Y.S.; Cheong, W.H.; Song, H.; Rhee, H.; Kay, S.N.; Han, J.W.; Kim, K.M. Evolutionary learning of binary neural network using a TaOx memristor via stochastic stateful logic. *Adv. Intell. Syst.* **2022**, *4*, 2200058. [CrossRef]
180. Wang, Z.; Shi, X.; Yao, X. A brain-inspired hardware architecture for evolutionary algorithms based on memristive arrays. *ACM Trans. Des. Autom. Electron. Syst.* **2023**, *28*, 1–32. [CrossRef]
181. Zhang, W.; Zhang, Q.; Wu, H. Genetic Algorithm-based Training Method for Memristor Neural Networks. *Int. J. Comput. Appl.* **2024**, *186*, 7–13. [CrossRef]
182. Edwards, K.N.; Shen, X. Comparison of update and genetic training algorithms in a memristor crossbar perceptron. *AIP Adv.* **2022**, *12*, 025327. [CrossRef]
183. Imbert, B.; Renaudineau, A.; Diallo, M.H.; Aguirre-Morales, J.D.; Iftakher, M.A.; Harabi, K.E.; Turck, C.; Drouhin, M.; Hirtzlin, T.; Vianello, E.; et al. Forward-forward learning exploiting low-voltage reset of RRAM. In *Proceedings of the 2024 IEEE International Electron Devices Meeting (IEDM), San Francisco, CA, USA, 7–11 December 2024*; IEEE: Piscataway, NJ, USA, 2024; pp. 1–4. [CrossRef]
184. Renaudineau, A.; Diallo, M.H.; Dupuis, T.; Imbert, B.; Iftakher, M.A.; Harabi, K.E.; Turck, C.; Hirtzlin, T.; Bonnet, D.; Melul, F.; et al. Forward-only learning in memristor arrays with month-scale stability. *arXiv* **2026**, arXiv:2601.09903. [CrossRef]
185. Du, C.; Cai, F.; Zidan, M.A.; Ma, W.; Lee, S.H.; Lu, W.D. Reservoir computing using dynamic memristors for temporal information processing. *Nat. Commun.* **2017**, *8*, 2204. [CrossRef] [PubMed]
186. Midya, R.; Wang, Z.; Asapu, S.; Zhang, X.; Rao, M.; Song, W.; Zhuo, Y.; Upadhyay, N.; Xia, Q.; Yang, J.J. Reservoir computing using diffusive memristors. *Adv. Intell. Syst.* **2019**, *1*, 1900084. [CrossRef]
187. Ryu, H.; Kim, S. Implementation of a reservoir computing system using the short-term effects of Pt/HfO2/TaOx/TiN memristors with self-rectification. *Chaos Solitons Fractals* **2021**, *150*, 111223. [CrossRef]
188. Zhang, G.; Xiong, Z.Y.; Gong, Y.; Zhu, Z.; Lv, Z.; Wang, Y.; Yang, J.Q.; Xing, X.; Wang, Z.P.; Qin, J.; et al. Polyoxometalate accelerated cationic migration for reservoir computing. *Adv. Funct. Mater.* **2022**, *32*, 2204721. [CrossRef]
189. John, R.A.; Demirağ, Y.; Shynkarenko, Y.; Berezovska, Y.; Ohannessian, N.; Payvand, M.; Zeng, P.; Bodnarchuk, M.I.; Krumeich, F.; Kara, G.; et al. Reconfigurable halide perovskite nanocrystal memristors for neuromorphic computing. *Nat. Commun.* **2022**, *13*, 2074. [CrossRef] [PubMed]
190. Wang, L.; Zhang, Y.; Guo, Z.; Wu, Z.; Chen, X.; Du, S. Reservoir computing-based design of zno memristor-type digital identification circuits. *Micromachines* **2022**, *13*, 1700. [CrossRef] [PubMed]
191. Yang, J.; Zhang, F.; Xiao, H.M.; Wang, Z.P.; Xie, P.; Feng, Z.; Wang, J.; Mao, J.; Zhou, Y.; Han, S.T. A perovskite memristor with large dynamic space for analog-encoded image recognition. *ACS Nano* **2022**, *16*, 21324–21333. [CrossRef] [PubMed]
192. Weng, Z.; Ji, T.; Yu, Y.; Fang, Y.; Lei, W.; Shafie, S.; Jindapetch, N.; Zhao, Z. Memristive devices based on necklace-like structure Ag@ TiO2 nanowire networks for neuromorphic learning and reservoir computing. *ACS Appl. Nano Mater.* **2024**, *7*, 21018–21025. [CrossRef]
193. Ghenzi, N.; Park, T.W.; Kim, S.S.; Kim, H.J.; Jang, Y.H.; Woo, K.S.; Hwang, C.S. Heterogeneous reservoir computing in second-order Ta 2 O 5/HfO 2 memristors. *Nanoscale Horiz.* **2024**, *9*, 427–437. [CrossRef] [PubMed]
194. Ju, D.; Noh, M.; Kim, G.; Park, Y.; Lee, S.; Kim, S. Reservoir computing system with diverse input patterns in HfAlO-based ferroelectric memristor. *ACS Appl. Mater. Interfaces* **2024**, *16*, 66250–66261. [CrossRef] [PubMed]
195. Chen, M.; An, X.; Ki, S.J.; Liu, X.; Sekhon, N.; Boyarov, A.; Acharya, A.; Tawil, J.; Bederman, M.; Liang, X. Nanoelectronics-enabled reservoir computing hardware for real-time robotic controls. *Sci. Adv.* **2025**, *11*, eadu2663. [CrossRef] [PubMed]
196. Deng, Y.; Zhang, S.; Yuan, F.; Li, Y.; Wang, G. Reservoir computing system using discrete memristor for chaotic temporal signal processing. *Chaos Solitons Fractals* **2025**, *194*, 116230. [CrossRef]
197. Kudithipudi, D.; Saleh, Q.; Merkel, C.; Thesing, J.; Wysocki, B. Design and analysis of a neuromemristive reservoir computing architecture for biosignal processing. *Front. Neurosci.* **2016**, *9*, 502. [CrossRef] [PubMed]
198. Moon, J.; Ma, W.; Shin, J.H.; Cai, F.; Du, C.; Lee, S.H.; Lu, W.D. Temporal data classification and forecasting using a memristor-based reservoir computing system. *Nat. Electron.* **2019**, *2*, 480–487. [CrossRef]

199. Zhong, Y.; Tang, J.; Li, X.; Gao, B.; Qian, H.; Wu, H. Dynamic memristor-based reservoir computing for high-efficiency temporal signal processing. *Nat. Commun.* **2021**, *12*, 408. [CrossRef] [PubMed]
200. Daniels, R.; Mallinson, J.B.; Heywood, Z.; Bones, P.; Arnold, M.D.; Brown, S.A. Reservoir computing with 3D nanowire networks. *Neural Netw.* **2022**, *154*, 122–130. [CrossRef] [PubMed]
201. Liu, K.; Dang, B.; Zhang, T.; Yang, Z.; Bao, L.; Xu, L.; Cheng, C.; Huang, R.; Yang, Y. Multilayer reservoir computing based on ferroelectric α-In2Se3 for hierarchical information processing. *Adv. Mater.* **2022**, *34*, 2108826. [CrossRef] [PubMed]
202. Srikimkaew, O.; Banerjee, D.; Azhari, S.; Usami, Y.; Tanaka, H. High performance of an in-material reservoir computing device achieved by complex dynamics in a nanoparticle random network memristor. *ACS Appl. Electron. Mater.* **2024**, *6*, 688–695. [CrossRef]
203. Wu, X.; Lin, Z.; Deng, J.; Li, J.; Feng, Y. Nonmasking-based reservoir computing with a single dynamic memristor for image recognition. *Nonlinear Dyn.* **2024**, *112*, 6663–6678. [CrossRef]
204. Chen, Z.; Li, W.; Fan, Z.; Dong, S.; Chen, Y.; Qin, M.; Zeng, M.; Lu, X.; Zhou, G.; Gao, X.; et al. All-ferroelectric implementation of reservoir computing. *Nat. Commun.* **2023**, *14*, 3585. [CrossRef] [PubMed]
205. Sun, L.; Wang, Z.; Jiang, J.; Kim, Y.; Joo, B.; Zheng, S.; Lee, S.; Yu, W.J.; Kong, B.S.; Yang, H. In-sensor reservoir computing for language learning via two-dimensional memristors. *Sci. Adv.* **2021**, *7*, eabg1455. [CrossRef] [PubMed]
206. Hossain, M.R.; Paul, P.S.; Sadia, M.; Dhungel, A.; Najem, J.S.; Hasan, M.S. Memristor based reservoir network for chaotic time series prediction. In *Proceedings of the 2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), Maldives, 16–18 November 2022*; IEEE: Piscataway, NJ, USA, 2022; pp. 1–6. [CrossRef]
207. Zhang, Z.; Zhao, X.; Zhang, X.; Hou, X.; Ma, X.; Tang, S.; Zhang, Y.; Xu, G.; Liu, Q.; Long, S. In-sensor reservoir computing system for latent fingerprint recognition with deep ultraviolet photo-synapses and memristor array. *Nat. Commun.* **2022**, *13*, 6590. [CrossRef] [PubMed]
208. Li, Z.; Gu, D.; Xie, X.; Li, P.; Sun, B.; Liao, C.; Hu, X.; Yan, J.; Wang, L.; Duan, S.; et al. Photoelectric reservoir computing based on TiO_x memristor for analog signal processing. *ACS Appl. Nano Mater.* **2025**, *8*, 6591–6603. [CrossRef]
209. Milano, G.; Pedretti, G.; Montano, K.; Ricci, S.; Hashemkhani, S.; Boarino, L.; Ielmini, D.; Ricciardi, C. In materia reservoir computing with a fully memristive architecture based on self-organizing nanowire networks. *Nat. Mater.* **2022**, *21*, 195–202. [CrossRef] [PubMed]
210. Kulkarni, M.S.; Teuscher, C. Memristor-based reservoir computing. In Proceedings of the 2012 IEEE/ACM International Symposium on Nanoscale Architectures, Amsterdam, The Netherlands, 4–6 July 2012; pp. 226–232. [CrossRef]
211. Wen, S.; Hu, R.; Yang, Y.; Huang, T.; Zeng, Z.; Song, Y.D. Memristor-based echo state network with online least mean square. *IEEE Trans. Syst. Man Cybern. Syst.* **2019**, *49*, 1787–1796. [CrossRef]
212. Lin, Z.; Deng, J.; Chen, Z.; Zhang, J.; Liang, S.; Feng, Y. An efficient reservoir computing system based on 2D mask processing and dynamic memristor. *Nonlinear Dyn.* **2025**, *113*, 20191–20209. [CrossRef]
213. Sayyaparaju, S.; Shawkat, M.S.A.; Adnan, M.M.; Rose, G.S. Circuit techniques for efficient implementation of memristor based reservoir computing. In *Proceedings of the 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 12–14 October 2020*; IEEE: Piscataway, NJ, USA, 2020; pp. 1–5. [CrossRef]
214. Henderson, A.; Yakopcic, C.; Merkel, C.; Hazan, H.; Harbour, S.; Taha, T.M. Memristor based liquid state machine with method for in-situ training. *IEEE Trans. Nanotechnol.* **2024**, *23*, 376–385. [CrossRef]
215. Rathore, M.; Febbo, R.; Foshie, A.; Tushar, S.N.B.; Das, H.; Rose, G.S. Reliability analysis of memristive reservoir computing architecture. In Proceedings of the Great Lakes Symposium on VLSI 2023, Knoxville, TN, USA, 5–7 June 2023; pp. 131–136. [CrossRef]
216. Zyarah, A.; Kudithipudi, D. Time-series forecasting and sequence learning using memristor-based reservoir system. *ACM Trans. Embed. Comput. Syst.* **2025**, *24*, 1–17. [CrossRef]
217. Nair, V.V.; Reghuvaran, C.; John, D.; Choubey, B.; James, A. Essm: Extended synaptic sampling machine with stochastic echo state neuro-memristive circuits. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2023**, *13*, 965–974. [CrossRef]
218. Armendarez, N.X.; Mohamed, A.S.; Dhungel, A.; Hossain, M.R.; Hasan, M.S.; Najem, J.S. Brain-inspired reservoir computing using memristors with tunable dynamics and short-term plasticity. *ACS Appl. Mater. Interfaces* **2024**, *16*, 6176–6188. [CrossRef] [PubMed]
219. Daniels, R.; Wattad, D.; Ronen, R.; Saad, D.; Kvatinsky, S. Preprocessing Methods for Memristive Reservoir Computing for Image Recognition. In *Proceedings of the 2025 IEEE International Conference on Metrology for eXtended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE), Ancona, Italy, 22–24 October 2025*; IEEE: Piscataway, NJ, USA, 2025; pp. 1164–1169. [CrossRef]
220. Kamsma, T.M.; Teijema, J.J.; van Roij, R.; Spitoni, C. Echo state and band-pass networks with aqueous memristors: Leaky reservoir computing with a leaky substrate. *Chaos Interdiscip. J. Nonlinear Sci.* **2025**, *35*, 093133. [CrossRef] [PubMed]
221. Singh, A.; Choi, S.; Wang, G.; Daimari, M.; Lee, B.G. Analysis and fully memristor-based reservoir computing for temporal data classification. *Neural Netw.* **2025**, *182*, 106925. [CrossRef] [PubMed]

222. Mallinson, J.; Heywood, Z.; Daniels, R.; Arnold, M.; Bones, P.; Brown, S. Reservoir computing using networks of memristors: Effects of topology and heterogeneity. *Nanoscale* **2023**, *15*, 9663–9674. [CrossRef] [PubMed]
223. Matsukatova, A.N.; Prudnikov, N.V.; Kulagin, V.A.; Battistoni, S.; Minnekhanov, A.A.; Trofimov, A.D.; Nesmelov, A.A.; Zavyalov, S.A.; Malakhova, Y.N.; Parmeggiani, M.; et al. Combination of organic-based reservoir computing and spiking neuromorphic systems for a robust and efficient pattern classification. *Adv. Intell. Syst.* **2023**, *5*, 2200407. [CrossRef]
224. Prudnikov, N.V.; Kulagin, V.A.; Battistoni, S.; Demin, V.A.; Erokhin, V.V.; Emelyanov, A.V. Polyaniline-Based Memristive Devices as Key Elements of Robust Reservoir Computing for Image Classification. *Phys. Status Solidi (a)* **2023**, *220*, 2200700. [CrossRef]
225. Armendarez, N.; Hasan, M.S.; Najem, J. Nonlinear memristor model with exact solution allows for ex situ reservoir computing training and in situ inference. *Nanoscale* **2025**, *17*, 2068–2077. [CrossRef] [PubMed]
226. Hua, S.; Zhang, L.; Wang, L.; Zheng, R.; Gan, P.; Guo, X. Electrochemical Preparation of Reliable and High Yield Memristors for Efficient Reservoir Computing Systems. *ACS Appl. Mater. Interfaces* **2025**, *17*, 53691–53703. [CrossRef] [PubMed]
227. Zhou, J.; Li, W.; Chen, Y.; Qian, H.; Lin, Y.H.; Li, R.; Wang, Z.; Wang, J.; Shi, W.; Tao, X.; et al. Optoelectronic polymer memristors with dynamic control for power-efficient in-sensor edge computing. *Light. Sci. Appl.* **2025**, *14*, 309. [CrossRef] [PubMed]
228. Erokhin, V. *Fundamentals of Organic Neuromorphic Systems*; Springer: Cham, Swizterland, 2022. [CrossRef]
229. Erokhin, V. Memristive devices for neuromorphic applications: Comparative analysis. *BioNanoScience* **2020**, *10*, 834–847. [CrossRef]
230. Kim, S.G.; Han, J.S.; Kim, H.; Kim, S.Y.; Jang, H.W. Recent Advances in Memristive Materials for Artificial Synapses. *Adv. Mater. Technol.* **2018**, *3*, 1800457. [CrossRef]
231. Shchanikov, S.; Zuev, A.; Bordanov, I.; Danilin, S.; Lukoyanov, V.; Korolev, D.; Belov, A.; Pigareva, Y.; Gladkov, A.; Pimashkin, A.; et al. Designing a bidirectional, adaptive neural interface incorporating machine learning capabilities and memristor-enhanced hardware. *Chaos Solitons Fractals* **2021**, *142*, 110504. [CrossRef]
232. Jin, B.; Wang, Z.; Wang, T.; Meng, J. Memristor-Based Artificial Neural Networks for Hardware Neuromorphic Computing. *Research* **2025**, *8*, 0758. [CrossRef] [PubMed]
233. Juzekaeva, E.; Nasretdinov, A.; Battistoni, S.; Berzina, T.; Iannotta, S.; Khazipov, R.; Erokhin, V.; Mukhtarov, M. Coupling Cortical Neurons through Electronic Memristive Synapse. *Adv. Mater. Technol.* **2019**, *4*, 1800350. [CrossRef]
234. Dias, C.; Castro, D.; Aroso, M.; Ventura, J.; Aguiar, P. Memristor-Based Neuromodulation Device for Real-Time Monitoring and Adaptive Control of Neuronal Populations. *ACS Appl. Electron. Mater.* **2022**, *4*, 2380–2387. [CrossRef] [PubMed]
235. Gerasimova, S.A.; Belov, A.I.; Korolev, D.S.; Guseinov, D.V.; Lebedeva, A.V.; Koryazhkina, M.N.; Mikhaylov, A.N.; Kazantsev, V.B.; Pisarchik, A.N. Stochastic Memristive Interface for Neural Signal Processing. *Sensors* **2021**, *21*, 5587. [CrossRef] [PubMed]
236. Cheng, C.; Wang, Y.; Xu, L.; Liu, K.; Dang, B.; Lu, Y.; Yan, X.; Huang, R.; Yang, Y. Artificial Astrocyte Memristor with Recoverable Linearity for Neuromorphic Computing. *Adv. Electron. Mater.* **2022**, *8*, 2100669. [CrossRef]
237. Demin, V.; Nekhaev, D.; Surazhevsky, I.; Nikiruy, K.; Emelyanov, A.; Nikolaev, S.; Rylkov, V.; Kovalchuk, M. Necessary conditions for STDP-based pattern recognition learning in a memristive spiking neural network. *Neural Netw.* **2021**, *134*, 64–75. [CrossRef] [PubMed]
238. Erokhin, V. Organic Memristive Device as Key Element for Neuromorphic Networks. *AIP Conf. Proc.* **2015**, *1648*, 280006. [CrossRef]
239. Bian, J.; Liu, Z.; Tao, Y.; Wang, Z.; Zhao, X.; Lin, Y.; Xu, H.; Liu, Y. Advances in Memristor Based Artificial Neuron Fabrication-Materials, Models, and Applications. *Int. J. Extrem. Manuf.* **2024**, *6*, 012002. [CrossRef]
240. Erokhin, V.; Berzina, T.; Camorani, P.; Smerieri, A.; Vavoulis, D.; Feng, J.; Fontana, M.P. Material memristive device circuits with synaptic plasticity: Learning and memory. *BioNanoScience* **2011**, *1*, 24–30. [CrossRef]
241. Minnekhanov, A.A.; Emelyanov, A.V.; Lapkin, D.A.; Nikiruy, K.E.; Shvetsov, B.S.; Nesmelov, A.A.; Rylkov, V.V.; Demin, V.A.; Erokhin, V.V. Parylene based memristive devices with multilevel resistive switching for neuromorphic applications. *Sci. Rep.* **2019**, *9*, 10800. [CrossRef] [PubMed]
242. Demin, V.; Erokhin, V.; Emelyanov, A.; Battistoni, S.; Baldi, G.; Iannotta, S.; Kashkarov, P.; Kovalchuk, M. Hardware elementary perceptron based on polyaniline memristive devices. *Org. Electron.* **2015**, *25*, 16–20. [CrossRef]
243. Emelyanov, A.; Lapkin, D.; Demin, V.; Erokhin, V.; Battistoni, S.; Baldi, G.; Dimonte, A.; Korovin, A.; Iannotta, S.; Kashkarov, P.; et al. First steps towards the realization of a double layer perceptron based on organic memristive devices. *AIP Adv.* **2016**, *6*, 111301. [CrossRef]
244. Santello, M.; Toni, N.; Volterra, A. Astrocyte function from information processing to cognition and cognitive impairment. *Nat. Neurosci.* **2019**, *22*, 154–166. [CrossRef] [PubMed]
245. Araque, A.; Parpura, V.; Sanzgiri, R.P.; Haydon, P.G. Tripartite synapses: Glia, the unacknowledged partner. *Trends Neurosci.* **1999**, *22*, 208–215. [CrossRef] [PubMed]
246. Araque, A.; Carmignoto, G.; Haydon, P.G.; Oliet, S.H.; Robitaille, R.; Volterra, A. Gliotransmitters travel in time and space. *Neuron* **2014**, *81*, 728–739. [CrossRef] [PubMed]

247. Semyanov, A.; Henneberger, C.; Agarwal, A. Making sense of astrocytic calcium signals—From acquisition to interpretation. *Nat. Rev. Neurosci.* **2020**, *21*, 551–564. [CrossRef] [PubMed]
248. Fields, R.D.; Araque, A.; Johansen-Berg, H.; Lim, S.S.; Lynch, G.; Nave, K.A.; Nedergaard, M.; Perez, R.; Sejnowski, T.; Wake, H. Glial biology in learning and cognition. *Neuroscientist* **2014**, *20*, 426–431. [CrossRef] [PubMed]
249. Gordleeva, S.; Tsybina, Y.A.; Krivonosov, M.I.; Tyukin, I.Y.; Kazantsev, V.B.; Zaikin, A.; Gorban, A.N. Situation-based neuromorphic memory in spiking neuron-astrocyte network. *IEEE Trans. Neural Netw. Learn. Syst.* **2025**, *36*, 881–895. [CrossRef] [PubMed]
250. Kozachkov, L.; Kastanenka, K.V.; Krotov, D. Building transformers from neurons and astrocytes. *Proc. Natl. Acad. Sci. USA* **2023**, *120*, e2219150120. [CrossRef] [PubMed]
251. Hodgkin, A.L.; Huxley, A.F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J. Physiol.* **1952**, *117*, 500–544. [CrossRef] [PubMed]
252. Li, Y.X.; Rinzel, J. Equations for InsP3 receptor-mediated [Ca2+] i oscillations derived from a detailed kinetic model: A Hodgkin-Huxley like formalism. *J. Theor. Biol.* **1994**, *166*, 461–473. [CrossRef] [PubMed]
253. Ullah, G.; Jung, P.; Cornell-Bell, A.H. Anti-phase calcium oscillations in astrocytes via inositol (1, 4, 5)-trisphosphate regeneration. *Cell Calcium* **2006**, *39*, 197–208. [CrossRef] [PubMed]
254. Matrosov, V.V.; Kazantsev, V.B. Bifurcation mechanisms of regular and chaotic network signaling in brain astrocytes. *Chaos Interdiscip. J. Nonlinear Sci.* **2011**, *21*, 023103. [CrossRef] [PubMed]
255. Matrosov, V.; Gordleeva, S.; Boldyreva, N.; Ben-Jacob, E.; Kazantsev, V.; De Pittà, M. Emergence of regular and complex calcium oscillations by inositol 1, 4, 5-trisphosphate signaling in astrocytes. In *Computational Glioscience*; Springer: Cham, Switerland, 2019; pp. 151–176. [CrossRef]
256. Tsodyks, M.V.; Markram, H. The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. *Proc. Natl. Acad. Sci. USA* **1997**, *94*, 719–723. [CrossRef] [PubMed]
257. Markram, H.; Tsodyks, M. Redistribution of synaptic efficacy between neocortical pyramidal neurons. *Nature* **1996**, *382*, 807–810. [CrossRef] [PubMed]
258. De Pittà, M.; Brunel, N.; Volterra, A. Astrocytes: Orchestrating synaptic plasticity? *Neuroscience* **2016**, *323*, 43–61. [CrossRef] [PubMed]
259. Gordleeva, S.Y.; Stasenko, S.V.; Semyanov, A.V.; Dityatev, A.E.; Kazantsev, V.B. Bi-directional astrocytic regulation of neuronal activity within a network. *Front. Comput. Neurosci.* **2012**, *6*, 92. [CrossRef] [PubMed]
260. Makovkin, S.Y.; Shkerin, I.; Gordleeva, S.Y.; Ivanchenko, M. Astrocyte-induced intermittent synchronization of neurons in a minimal network. *Chaos Solitons Fractals* **2020**, *138*, 109951. [CrossRef]
261. Makovkin, S.; Kozinov, E.; Ivanchenko, M.; Gordleeva, S. Controlling synchronization of gamma oscillations by astrocytic modulation in a model hippocampal neural network. *Sci. Rep.* **2022**, *12*, 6970. [CrossRef] [PubMed]
262. Pankratova, E.V.; Kalyakulina, A.I.; Stasenko, S.V.; Gordleeva, S.Y.; Lazarevich, I.A.; Kazantsev, V.B. Neuronal synchronization enhanced by neuron–astrocyte interaction. *Nonlinear Dyn.* **2019**, *97*, 647–662. [CrossRef]
263. Kanakov, O.; Gordleeva, S.; Ermolaeva, A.; Jalan, S.; Zaikin, A. Astrocyte-induced positive integrated information in neuron-astrocyte ensembles. *Phys. Rev. E* **2019**, *99*, 012418. [CrossRef] [PubMed]
264. Abrego, L.; Gordleeva, S.; Kanakov, O.; Krivonosov, M.; Zaikin, A. Estimating integrated information in bidirectional neuron-astrocyte communication. *Phys. Rev. E* **2021**, *103*, 022410. [CrossRef] [PubMed]
265. Gordleeva, S.Y.; Ermolaeva, A.V.; Kastalskiy, I.A.; Kazantsev, V.B. Astrocyte as spatiotemporal integrating detector of neuronal activity. *Front. Physiol.* **2019**, *10*, 294. [CrossRef] [PubMed]
266. Gordleeva, S.Y.; Tsybina, Y.A.; Krivonosov, M.I.; Ivanchenko, M.V.; Zaikin, A.A.; Kazantsev, V.B.; Gorban, A.N. Modeling working memory in a spiking neuron network accompanied by astrocytes. *Front. Cell. Neurosci.* **2021**, *15*, 631485. [CrossRef] [PubMed]
267. De Pittà, M.; Brunel, N. Multiple forms of working memory emerge from synapse–astrocyte interactions in a neuron–glia network model. *Proc. Natl. Acad. Sci. USA* **2022**, *119*, e2207912119. [CrossRef] [PubMed]
268. Becker, S.; Nold, A.; Tchumatchenko, T. Modulation of working memory duration by synaptic and astrocytic mechanisms. *PLoS Comput. Biol.* **2022**, *18*, e1010543. [CrossRef] [PubMed]
269. Tsybina, Y.; Kastalskiy, I.; Krivonosov, M.; Zaikin, A.; Kazantsev, V.; Gorban, A.N.; Gordleeva, S. Astrocytes mediate analogous memory in a multi-layer neuron–astrocyte network. *Neural Comput. Appl.* **2022**, *34*, 9147–9160. [CrossRef]
270. Amiri, M.; Montaseri, G.; Bahrami, F. A phase plane analysis of neuron–astrocyte interactions. *Neural Netw.* **2013**, *44*, 157–165. [CrossRef] [PubMed]
271. Soleimani, H.; Bavandpour, M.; Ahmadi, A.; Abbott, D. Digital implementation of a biological astrocyte model and its application. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 127–139. [CrossRef] [PubMed]
272. Nazari, S.; Faez, K.; Amiri, M.; Karami, E. A digital implementation of neuron–astrocyte interaction for neuromorphic applications. *Neural Netw.* **2015**, *66*, 79–90. [CrossRef] [PubMed]
273. Izhikevich, E.M. Simple model of spiking neurons. *IEEE Trans. Neural Netw.* **2003**, *14*, 1569–1572. [CrossRef] [PubMed]

274. Postnov, D.E.; Koreshkov, R.; Brazhe, N.A.; Brazhe, A.R.; Sosnovtseva, O.V. Dynamical patterns of calcium signaling in a functional model of neuron–astrocyte networks. *J. Biol. Phys.* **2009**, *35*, 425–445. [CrossRef] [PubMed]
275. Hayati, M.; Nouri, M.; Haghiri, S.; Abbott, D. A digital realization of astrocyte and neural glial interactions. *IEEE Trans. Biomed. Circuits Syst.* **2016**, *10*, 518–529. [CrossRef] [PubMed]
276. Hong, Q.; Chen, H.; Sun, J.; Wang, C. Memristive circuit implementation of a self-repairing network based on biological astrocytes in robot application. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 2106–2120. [CrossRef] [PubMed]
277. Stasenko, S.V.; Mikhaylov, A.N.; Fedotov, A.A.; Smirnov, V.A.; Kazantsev, V.B. Astrocyte control bursting mode of spiking neuron network with memristor-implemented plasticity. *Chaos Solitons Fractals* **2024**, *181*, 114648. [CrossRef]
278. Chen, K.; Li, Z. Firing activities analysis of neuron–astrocyte network with biomimetic memristor synapse. *Nonlinear Dyn.* **2025**, *113*, 21769–21787. [CrossRef]
279. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 1998.
280. 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 54th Annual Design Automation Conference 2017, Austin, TX, USA, 18–22 June 2017; pp. 1–6. [CrossRef]
281. Wang, Z.; Li, C.; Song, W.; Rao, M.; Belkin, D.; Li, Y.; Yan, P.; Jiang, H.; Lin, P.; Hu, M.; et al. Reinforcement learning with analogue memristor arrays. *Nat. Electron.* **2019**, *2*, 115–124. [CrossRef]
282. Alam, M.; Yakopcic, C.; Taha, T.M. On-Chip Optimization and Deep Reinforcement Learning in Memristor Based Computing. In Proceedings of the 18th ACM International Symposium on Nanoscale Architectures, Dresden, Germany, 18–20 December 2023; pp. 1–7. [CrossRef]
283. Song, W.; Zhou, Y.; Hu, X.; Duan, S.; Lai, H. Memristive neural network based reinforcement learning with reward shaping for path finding. In *Proceedings of the 2018 5th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS), Hangzhou, China, 16–19 August 2018*; IEEE: Piscataway, NJ, USA, 2018; pp. 200–205. [CrossRef]
284. Portner, K.; Zellweger, T.; Martinelli, F.; Bégon-Lours, L.; Bragaglia, V.; Weilenmann, C.; Jubin, D.; Falcone, D.F.; Hermann, F.; Hrynkevych, O.; et al. Actor–critic networks with analogue memristors mimicking reward-based learning. *Nat. Mach. Intell.* **2025**, *7*, 1939–1953. [CrossRef] [PubMed]
285. Wu, N.; Vincent, A.F.; Strukov, D. Preliminary Results Towards Reinforcement Learning with Mixed-Signal Memristive Neuromorphic Circuits. In *Proceedings of the 2019 IEEE International Symposium on Circuits and Systems (ISCAS), Sapporo, Japan, 26–29 May 2019*; IEEE: Piscataway, NJ, USA, 2019; pp. 1–5. [CrossRef]
286. Wu, N.; Vincent, A.; Strukov, D.; Xie, Y. Memristor hardware-friendly reinforcement learning. *arXiv* **2020**, arXiv:2001.06930. [CrossRef]
287. Dai, Y.; Guo, W.; Feng, Z.; Xu, Z.; Zhu, Y.; Yang, F.; Wu, Z. Intrinsic decay property of Ti/TiOx/Pt memristor for reinforcement learning. *Adv. Intell. Syst.* **2023**, *5*, 2200455. [CrossRef]
288. Ji, X.; Zhang, Y.; Li, C.; Wu, T.; Hu, X. Reinforcement learning in memristive spiking neural networks through modulation of resume. *AIP Conf. Proc.* **2019**, *2073*, 020094. [CrossRef]
289. Vlasov, D.; Rybka, R.; Sboev, A.; Serenko, A.; Minnekhanov, A.; Demin, V. Reinforcement learning in a spiking neural network with memristive plasticity. In *Proceedings of the 2022 6th Scientific School Dynamics of Complex Networks And Their Applications (DCNA), Kaliningrad, Russia, 14–16 September 2022*; IEEE: Piscataway, NJ, USA, 2022; pp. 300–302. [CrossRef]
290. Vlasov, D.; Minnekhanov, A.; Rybka, R.; Davydov, Y.; Sboev, A.; Serenko, A.; Ilyasov, A.; Demin, V. Memristor-based spiking neural network with online reinforcement learning. *Neural Netw.* **2023**, *166*, 512–523. [CrossRef] [PubMed]
291. Kulagin, V.; Matsukatova, A.; Ryl'kov, V.; Demin, V. Reinforcement learning of spiking neural networks using trace variables for synaptic weights with memristive plasticity. *Russ. Microelectron.* **2025**, *54*, 230–239. [CrossRef]
292. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv* **2017**, arXiv:1708.07747. [CrossRef]
293. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347. [CrossRef]
294. Go, S.X.; Jiang, Y.; Loke, D.K. A Phase-Change Memristive Reinforcement Learning for Rapidly Outperforming Champion Street-Fighter Players. *Adv. Intell. Syst.* **2023**, *5*, 2300335. [CrossRef]
295. Berdan, R.; Marukame, T.; Kabuyanagi, S.; Ota, K.; Saitoh, M.; Fujii, S.; Deguchi, J.; Nishi, Y. In-memory reinforcement learning with moderately-stochastic conductance switching of ferroelectric tunnel junctions. In *Proceedings of the 2019 Symposium on VLSI Technology, Kyoto, Japan, 9–14 June 2019*; IEEE: Piscataway, NJ, USA, 2019; pp. T22–T23. [CrossRef]
296. Lin, Y.; Zhang, Q.; Gao, B.; Tang, J.; Yao, P.; Li, C.; Huang, S.; Liu, Z.; Zhou, Y.; Liu, Y.; et al. Uncertainty quantification via a memristor Bayesian deep neural network for risk-sensitive reinforcement learning. *Nat. Mach. Intell.* **2023**, *5*, 714–723. [CrossRef]

297. Yang, M.J.; Yu, Z.; Pedretti, G.; Neftci, E.; Strachan, J.P. Improved memristor control using device physics and deep reinforcement learning. In *Proceedings of the 2025 IEEE 7th International Conference on Artificial Intelligence Circuits and Systems (AICAS), Bordeaux, France, 28–30 April 2025*; IEEE: Piscataway, NJ, USA, 2025; pp. 1–5. [CrossRef]
298. Jin, S.; Chen, J.; Wu, J.; Wang, X.; Luan, X.; Fu, J.; Wen, G. A deep reinforcement learning approach for synchronization between two memristor chaotic systems and application for image encryption. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2026**, *73*, 1380–1393. [CrossRef]
299. Dalgaty, T.; Castellani, N.; Turck, C.; Harabi, K.E.; Querlioz, D.; Vianello, E. In situ learning using intrinsic memristor variability via Markov chain Monte Carlo sampling. *Nat. Electron.* **2021**, *4*, 151–161. [CrossRef]
300. Bellemare, M.G.; Naddaf, Y.; Veness, J.; Bowling, M. The arcade learning environment: An evaluation platform for general agents. *J. Artif. Intell. Res.* **2013**, *47*, 253–279. [CrossRef]
301. Zafar, I.; Cui, Y.; Bai, Q.; Yang, Y. Classifying beers with memristor neural network algorithm in a portable electronic nose system. *Front. Phys.* **2022**, *10*, 907644. [CrossRef]
302. Li, C.; Lammie, C.; Dong, X.; Amirsoleimani, A.; Azghadi, M.R.; Genov, R. Seizure detection and prediction by parallel memristive convolutional neural networks. *IEEE Trans. Biomed. Circuits Syst.* **2022**, *16*, 609–625. [CrossRef] [PubMed]
303. Chen, J.; Carrara, S. Neuromorphic Sensing-and-Computing for Multi-Panel Cancer Diagnostics by Memristive Biosensors. In *Proceedings of the 2026 IEEE International Symposium on Circuits and Systems (ISCAS), Shanghai, China, 24–28 May 2026*; IEEE: Piscataway, NJ, USA, 2026; pp. 3441–3445. [CrossRef]
304. Bouzouita, M.; Zayer, F.; Tzouvadaki, I.; Carrara, S.; Belgacem, H. Novel AI-empowered Memristive Biosensing Computational Approach for Biomarker Detection. *IEEE Sens. J.* **2026**, *26*, 17055–17061. [CrossRef]
305. Zhang, Z.; Wang, Q.; Shi, G.; Ma, Y.; Zeng, J.; Liu, G. Neural networks based on in-sensor computing of optoelectronic memristor. *Microelectron. Eng.* **2024**, *291*, 112201. [CrossRef]
306. Khan, A.H.; Cao, X.; Luo, C.; Zhang, S.; Guo, W.; Katsikis, V.N.; Li, S. Spiking neural networks: A comprehensive survey of training methodologies, hardware implementations and applications. *Artif. Intell. Sci. Eng.* **2025**, *1*, 175–207. [CrossRef]
307. Rao, M.; Tang, H.; Wu, J.; Song, W.; Zhang, M.; Yin, W.; Zhuo, Y.; Kiani, F.; Chen, B.; Jiang, X.; et al. Thousands of conductance levels in memristors integrated on CMOS. *Nature* **2023**, *615*, 823–829. [CrossRef] [PubMed]
308. Chen, S.; Yang, Z.; Hartmann, H.; Besmehn, A.; Yang, Y.; Valov, I. Electrochemical ohmic memristors for continual learning. *Nat. Commun.* **2025**, *16*, 2348. [CrossRef] [PubMed]
309. Ming, J.; Wang, R.; Fu, J.; Liu, J.; Wu, S.; Liu, S.; Shao, H.; Li, Y.; Wang, J.; Wang, Y.; et al. Intrinsic gradient oxygen-driven second-order memristors for continual reinforcement learning. *Nat. Commun.* **2026**, *17*, 3367. [CrossRef] [PubMed]
310. Fan, H.; Chen, X.; Shi, K.; Liang, Y.; Wang, Y.; Wen, H. Mittag-Leffler synchronization in finite time for uncertain fractional-order multi-delayed memristive neural networks with time-varying perturbations via information feedback. *Fractal Fract.* **2024**, *8*, 422. [CrossRef]
311. Xiao, J.; Shi, K.; Teng, Y.; Qi, J.; Fan, H. Relaxed Research on Synchronization Problem of Fractional-Order Fuzzy Octonion-Valued BAM Neural Networks by the Non-Decomposition Method on the High-Dimension Oblique Field. *Fractal Fract.* **2026**, *10*, 414. [CrossRef]
312. Nie, Y.; Zhao, C.; Shi, K.; Chen, Y. Novel stability analysis for time-delay systems via two new lemmas and genetic algorithm. *Chaos Solitons Fractals* **2026**, *208*, 118232. [CrossRef]
313. Hong, H.; Du, Z.; Jiang, M.; Mao, R.; Ren, Y.; Li, F.; Mao, W.; Peng, M.; Zhang, W.; Liu, Z.; et al. Memristor-based adaptive analog-to-digital conversion for efficient and accurate compute-in-memory. *Nat. Commun.* **2025**, *16*, 9749. [CrossRef] [PubMed]
314. Rasch, M.J.; Carta, F.; Fagbohungbe, O.; Gokmen, T. Fast and robust analog in-memory deep neural network training. *Nat. Commun.* **2024**, *15*, 7133. [CrossRef] [PubMed]
315. Joshi, V.; Le Gallo, M.; Haefeli, S.; Boybat, I.; Nandakumar, S.R.; Piveteau, C.; Dazzi, M.; Rajendran, B.; Sebastian, A.; Eleftheriou, E. Accurate deep neural network inference using computational phase-change memory. *Nat. Commun.* **2020**, *11*, 2473. [CrossRef] [PubMed]
316. Wang, Z.; Wang, S.; Du, Z.; Mao, R.; Xiao, Y.; So, H.K.H.; Lin, P.; Li, C. Fully Integrated Memristive Spiking Neural Network with Analog Neurons for High-Speed Event-Based Data Processing. *arXiv* **2025**, arXiv:2509.04960. [CrossRef]
317. Amir, A.; Taba, B.; Berg, D.; Melano, T.; McKinstry, J.; Di Nolfo, C.; Nayak, T.; Andreopoulos, A.; Garreau, G.; Mendoza, M.; et al. A low power, fully event-based gesture recognition system. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7243–7252. [CrossRef]
318. Wang, Z.; Luo, T.; Liu, C.; Liu, W.; Goh, R.S.M.; Wong, W.F. Enabling energy-efficient deployment of large language models on memristor crossbar: A synergy of large and small. *IEEE Trans. Pattern Anal. Mach. Intell.* **2025**, *47*, 916–933. [CrossRef] [PubMed]
319. Heim, D.; Barbruni, G.L.; Carrara, S. A novel approach in edge computing: In-memory sensing of cancer markers. In *Proceedings of the 2022 IEEE International Symposium on Circuits and Systems (ISCAS), Austin, TX, USA, 27 May–1 June 2022*; IEEE: Piscataway, NJ, USA, 2022; pp. 306–310. [CrossRef]

320. Carrara, S.; Chen, J.; Bhardwaj, K.; Golparvar, A.; Barbruni, G.L. In-memory sensing and computing for cancer diagnostics: A perspective paper. *IEEE Trans. Biomed. Circuits Syst.* **2024**, *18*, 361–368. [CrossRef] [PubMed]
321. Chen, J.; Carrara, S. Computational study of simultaneous memristive effect and biosensing for prostate-specific antigen detection. *BioNanoScience* **2025**, *15*, 239. [CrossRef]
322. Cao, Z.; Liu, Y.; Sun, B.; Zhou, G.; Gao, K.; Sun, S.; Cui, Y.; Wang, M.; Yan, X.; Zhao, T.; et al. A high-stability pressure-sensitive implantable memristor for pulmonary hypertension monitoring. *Adv. Mater.* **2025**, *37*, 2411659. [CrossRef] [PubMed]
323. Lu, C.; Meng, J.; Song, J.; Xu, K.; Wang, T.; Zhu, H.; Sun, Q.q.; Zhang, D.W.; Chen, L. Reconfigurable selector-free all-optical controlled neuromorphic memristor for in-memory sensing and reservoir computing. *ACS Nano* **2024**, *18*, 29715–29723. [CrossRef] [PubMed]
324. Feng, W.; Qin, T.; Tang, X. Advances in infrared detectors for in-memory sensing and computing. *Photonics* **2024**, *11*, 1138. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.