Memristor-Based Circuit Design for Multilayer Neural Networks

Yang Zhang, Xiaoping Wang, *Member, IEEE*, and Eby G. Friedman, *Fellow, IEEE*

***Abstract*— Memristors are promising components for applications in nonvolatile memory, logic circuits, and neuromorphic computing. In this paper, a novel circuit for memristor-based multilayer neural networks is presented, which can use a single memristor array to realize both the plus and minus weight of the neural synapses. In addition, memristor-based switches are utilized during the learning process to update the weight of the memristor-based synapses. Moreover, an adaptive back propagation algorithm suitable for the proposed memristor-based multilayer neural network is applied to train the neural networks and perform the XOR function and character recognition. Another highlight of this paper is that the robustness of the proposed memristor-based multilayer neural network exhibits higher recognition rates and fewer cycles as compared with other multilayer neural networks.**

***Index Terms*— Memristor, synaptic weight, crossbar array, multilayer neural networks, XOR function, character recognition.**

I. Introduction

ARTIFICIAL neural networks have been exploited to solve many problems in the area of pattern recognition, exhibiting the potential to provide high speed computation. One possible device to achieve high speed computation is memristors, the discovery of which greatly broadened the area of hybrid CMOS architectures to nonconventional logic [1] such as threshold logic [2] and neuromorphic computing [3]. Memristors were theoretically postulated by Chua in 1971 [4] and later Williams's team presented a resistance variable device as a memristor at HP Labs in 2008 [5]. As a novel nanoscale device, memristors provide several advantageous features such as non-volatility, high density, low power, and good scalability [6]. Memristors are particularly appealing for realizing synaptic weights in artificial neural networks [7], [8] as the innate property of a reconfigurable resistance with memory makes memristor highly suitable for synapse weight refinement.

Neuron circuits were originally developed in CMOS [9], [10]. Later, hybrid CMOS-memristor synaptic circuits were developed [11]–[13]. The area and power consumption of transistors are however much greater than memristors. A memristor bridge synapse-based neural network and learning are proposed in [14]–[16], which implement multilayer neural networks (MNN) trained by a back propagation (BP) algorithm, and the synapse weight updates are performed by a host computer. The major computational bottleneck is however the learning process itself which could not be completely implemented in hardware with massive memristor-based crossbar arrays.

Many previous memristor-based learning rules have focused on Spike-Timing Dependent Plasticity (STDP) [17]. For example, neuromorphic character recognition system with two PCMO memristors (2M) as a synapse was presented in [18], and a learning rule proposed in [19] for visual pattern recognition with a CMOS neuron. The filamentary switching binary 2M synapse was used for speech recognition [20]. The convergence of STDP-based learning is however not guaranteed for general inputs [21].

New methods have since been proposed for memristor-based neuromorphic architectures. For example, brain-state-in-a-box (BSB) neural networks are presented in [22], which also use 2M crossbar arrays to represent, respectively, plus-polarity and minus-polarity connection matrices. Memristor-based multilayer neural networks with online gradient descent training are proposed in [23] and [24], which use a single memristor and two CMOS transistors (2T1M) per synapse. A training method of a 2M hardware neuromorphic network is proposed in [25]. To reduce the circuit size, fewer memristors and transistors are desired. A memristor-based crossbar array architecture is therefore presented in [26], where both plus-polarity and minus-polarity connection matrices are realized by a single crossbar array and a simple constant-term circuit, thereby reducing the physical size and power dissipation. The memristor-based neural network in [26] is however limited to a single layer neural network (SNN). On-chip learning methods remain a challenge in most memristor-based neural networks.

Neuromorphic processors with memristor-based synapses are investigated in [27]–[29] to achieve the digital pattern recognition. Training algorithms of 2M crossbar neuromorphic processors is proposed in [30] and [31], which could be

Manuscript received February 5, 2017; revised April 25, 2017 and June 28, 2017; accepted July 15, 2017. Date of publication August 11, 2017; date of current version January 25, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61374150, Grant 61272050, and Grant 61672357, in part by the Science Foundation of Guangdong Province under Grant 2014A030313556, and in part by the China Scholarship Council. This paper was recommended by Associate Editor T. Serrano-Gotarredona. *(Corresponding author: Xiaoping Wang.)*

Y. Zhang is with the School of Automation, Huazhong University of Science and Technology, Wuhan 430074, China, with the Shenzhen Key Laboratory of Spatial Smart Sensing and Services, School of Computer Science and Software Engineering, Computer Vision Institute, Shenzhen University, Shenzhen 518060, China, and also with the Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY 14627 USA.

X. Wang is with the School of Automation, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: wangxiaoping@mail.hust.edu.cn).

E. G. Friedman is with the Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY 14627 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2017.2729787

used in MNN; however, two memristors per synapse are required. An on-chip supervised learning rule for an ultra-high density neural crossbar using a memristor for the synapse and neuron is described in [32] to perform XOR and AND logic operations. Realizing the BP algorithm on a 1M crossbar array remains an issue.

The primary contributions of this paper are:

1) A memristor-based AND (MRL) gate [33] is utilized as a memristor-based switch (MS) [2] in updating the synaptic crossbar circuits. A memristive model for synaptic circuits based on experimental data is utilized in the simulations. Formulae for determining the relevant time for the weight updating process are also available. Moreover, an amplifier is added to generate the errors, creating an opportunity for updating the synaptic weights on-chip.
2) The memristor-based SNN in [26] is expanded to MNN and provides enhanced robustness despite the memristance variations. The proposed memristor-based synaptic crossbar circuit uses fewer memristors and no transistors as compared with the synaptic circuits discribed in [11], [12], [14]-[16], [18]-[20], [22], [23], [25], [30], and [31].
3) An adaptive BP algorithm suitable for the proposed memristor-based MNN is developed to train neural networks and perform the XOR function and character recognition. Moreover, the weight adjustment process and the proposed MNNs exhibit higher recognition rates and require fewer cycles.

The remainder of this paper is organized as follows. The memristor and the MS are discussed in section II. The proposed memristor-based SNN and the expanded MNN are presented in section III. The operation of the MNN based on the proposed adaptive BP algorithm is described in Section IV. The robustness of the crossbar architecture is discussed in section V. In section VI, simulation results are provided to demonstrate the superior performance of the proposed memristor-based neural networks. The paper is concluded in section VII.

II. Memristors and MS

Basic background about memristors and the memristor-based switch (MS) are provided in this section. In section II.A, a memristor is characterized analytically, and different memristor models are discussed. In section II.B, the MS is proposed for application to memristor-based neural networks.

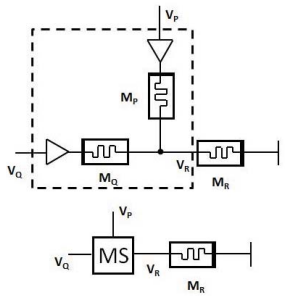
A. Memristor

A charge controlled memristance [4] can be described as

$$M(q) = d\phi/dq, \tag{1}$$

where $M(q)$ is the memristance (in Ω), ϕ is the magnetic flux, and q is the electric charge. The current controlled HP memristor is [5], [34]

$$\left.\begin{aligned} v(t) &= R(t)i(t) \\ R(t) &= R_{\mathrm{ON}}\frac{w(t)}{D} + R_{\mathrm{OFF}}\left(1 - \frac{w(t)}{D}\right) \end{aligned}\right\}, \tag{2}$$



V_P	V_Q	V_R
0	0	0
1	0	0
0	1	0
1	1	1

(b)

Fig. 1. Memristor-based logic switch. (a) A memristor-based logic switch, and (b) truth table for AND operation.

where $w(t)$ is the width of the doped region (initial width $w(0) = w_0 \in R$), $R(t)$ is the memristance ($R(t) = M(q)$), D is the thickness of the TiO_2, R_{ON} denotes the internal low memristance when the memristor is completely doped ($w(t) = D$), R_{OFF} denotes the internal high memristance when the memristor is completely undoped ($w(t) = 0$), and $i(t)$ and $v(t)$ are, respectively, the current and voltage of the memristor [5].

To consider the characteristics of the memristors, different memristor models are compared in [35]–[37]. Moreover, different memristor-based SPICE models and circuits are presented in [38]–[40]. A new model is therefore proposed which matches the synaptic behavior of recent memristive devices [41]. The derivative of the state variable in the proposed memristive model is

$$\frac{dw(t)}{dt} = \begin{cases} \mu_v \dfrac{R_{\mathrm{ON}}}{D}\dfrac{i_{\mathrm{off}}}{i(t) - i_0} f(w(t)), & v(t) > V_{\mathrm{T}+} > 0 \\ 0, & V_{\mathrm{T}-} \leqslant v(t) \leqslant V_{\mathrm{T}+} \\ \mu_v \dfrac{R_{\mathrm{ON}}}{D}\dfrac{i(t)}{i_{\mathrm{on}}} f(w(t)), & v(t) < V_{\mathrm{T}-} < 0, \end{cases} \tag{3}$$

where i_0, i_{off}, and i_{on} are constants, μ_v denotes the average ion mobility, and $V_{\mathrm{T}+}$ and $V_{\mathrm{T}-}$ are, respectively, positive and negative threshold voltages. A relationship exists between the change of the conductance and the pulse numbers (time) [42]. It is also possible to add a nonlinear ion drift phenomenon, such as a decrease in the ion drift speed close to the bounds, with a window,

$$f(w(t)) = 1 - \left(\frac{2w(t)}{D} - 1\right)^{2p}, \tag{4}$$

where p is a positive integer.

B. A Memristor-Based Logic Switch

In a memristor-based neural network, the input voltages are V_{H} or 0 V. The proposed memristor-based AND logic switch (MS) is a simplified form of [2], which consists of two memristors P and Q connected by two positive terminals, as shown in Fig. 1(a). Four memristors are not needed in an MS since the input voltages are all positive in a neural network. V_{P} and V_{Q} are the two input voltages, and

V_{R} is the output. To ensure the correctness of the AND logic operation, $R_{\mathrm{R}} \gg R_{\mathrm{P}}, R_{\mathrm{Q}}$, as described in [2]. The truth table for the memristor-based AND operation is shown in Fig. 1(b). To simplify the calculation, the window function is ignored. The nonlinear ion drift phenomenon is considered in the simulation with a window function. The time required for a memristor to change from R_{ON} to R_{OFF} or from R_{OFF} to R_{ON} is assumed to be the same. The initial weight of P and Q memristors is arbitrary and the switching time T_1 for MS is [2]

$$T_1 \approx \frac{2i_{\mathrm{on}}\Delta R R_{\mathrm{OFF}}}{kV_{\mathrm{H}}} = \frac{2\beta_1 i_{\mathrm{on}} D^2}{\mu_v V_{\mathrm{H}}}, \tag{5}$$

where $k = \mu_v \Delta R R_{\mathrm{ON}}/D^2$, ΔR is the difference between R_{OFF} and R_{ON} ($\Delta R = R_{\mathrm{OFF}} - R_{\mathrm{ON}}$), and $\beta_1 = R_{\mathrm{OFF}}/R_{\mathrm{ON}}$. R_{OFF} and R_{ON} are, respectively, the high and low memristances of memristors P and Q, as shown in Fig. 1(a). No threshold in the MS is assumed. The output error V_{e} is [2]

$$V_{\mathrm{e}} \approx \frac{R_{\mathrm{Q}}}{R_{\mathrm{P}} + R_{\mathrm{Q}}} V_{\mathrm{H}} = \frac{R_{\mathrm{ON}}}{R_{\mathrm{OFF}} + R_{\mathrm{ON}}} V_{\mathrm{H}}. \tag{6}$$

To change the memristance of M_{R} from an arbitrary initial memristance $R_{\mathrm{R}i}$ to a final memristance $R_{\mathrm{R}f}$, the relationship between R_{R} and time T is [2]

$$T = \begin{cases} \dfrac{V_{\mathrm{R}}(\ln R_{\mathrm{R}i} - \ln R_{\mathrm{R}f}) - i_0(R_{\mathrm{R}i} - R_{\mathrm{R}f})}{k' i_{\mathrm{off}}}, & V_{\mathrm{R}} > 0 \\ \dfrac{R_{\mathrm{R}i}^2 - R_{\mathrm{R}f}^2}{2k' V_{\mathrm{R}}} i_{\mathrm{on}}, & V_{\mathrm{R}} < 0, \end{cases} \tag{7}$$

where $k' = \mu_v' \Delta R R_{\mathrm{ON}}'/D^2$, and R_{ON}' is the low memristance of M_{R}.

When the memristance of the memristor M_{R} changes from R_{ON}' to R_{OFF}', from (7), the time T_2 is

$$T_2 = \frac{R_{\mathrm{OFF}}'^2 - R_{\mathrm{ON}}'^2}{2k' V_{\mathrm{R}}} i_{\mathrm{on}} = \frac{(\beta_2 + 1) D^2}{2\mu_v' V_{\mathrm{R}}} i_{\mathrm{on}}, \tag{8}$$

where $\beta_2 = R_{\mathrm{OFF}}'/R_{\mathrm{ON}}'$, and R_{OFF}' is the high memristance of M_{R}. T_1 is desired to be as small as possible with respect to T_2, so $\beta_1 < \beta_2$ and $\mu_v > \mu_{v0}$.

III. Memristive Neural Network Circuits

In section III.A, a memristor-based single layer neural network (SNN) is expanded into a multilayer neural network (MNN). Moreover, the modified back propagation (BP) learning method is proposed for on-chip MNN in section III.B.

A. Memristor-Based SNN

In this section, the MS is utilized in the memristor-based synaptic crossbar circuit in [26], where both plus-polarity and minus-polarity connection matrices are realized by a single crossbar array and a simple constant-term circuit [26]. The second amplifier A_2 is changed (as shown in Fig. 2) in the output part [26] to correctly operate the neural network (the connection of the positive and negative terminals of the amplifier are switched). Moreover, amplifier A_3 is added (as shown in Fig. 2) to generate the errors, updating the synaptic weights on chip. The requirement of T_1 to be smaller than T_2 can be achieved by using different memristors in the MS and synapses. The memristance of the synaptic memristor is much higher than in the MS.

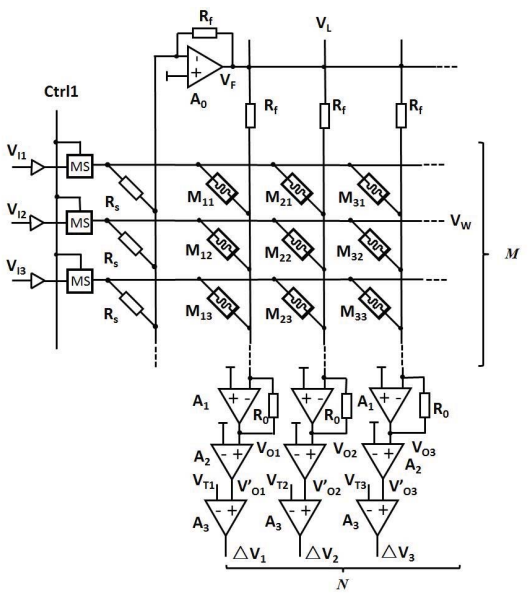


Fig. 2. Proposed memristor-based SNN.

Assume a learning system that operates on K discrete iterations of inputs, indexed by $k = 1, 2, \ldots, K$. During each iteration k, the system receives a pair of two vectors of size M and N: inputs $\mathbf{V}_{\mathrm{I}}{}^{(k)} \in \mathbb{R}^M$ and outputs $\mathbf{V}_{\mathrm{O}}{}^{(k)} \in \mathbb{R}^N$.

For example, assume $\mathbf{W}$ is an adjustable $N \times M$ matrix, and consider the estimator [23],

$$\mathbf{V}_{\mathrm{O}}{}^{(k)} = \mathbf{W}^{(k)} \mathbf{V}_{\mathrm{I}}{}^{(k)}, \tag{9}$$

or

$$V_{\mathrm{O}j}^{(k)} = \sum_{i=1}^{M} W_{ji}^{(k)} V_{\mathrm{I}i}^{(k)}, \tag{10}$$

where $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$.

A new synaptic array composed of a single crossbar array of $M^-(G_{ji})$ and the constant-term circuit of G_s is shown in Fig. 2. Here, G_s ($G_s = 1/R_s$) is the conductance of R_s, and G_{ji} ($G_{ji} = 1/R_{ji}$) is the memristor conductance at the crossing point between the i^{th} row and j^{th} column. $V_{\mathrm{I}i}$ is the input voltage applied to the i^{th} row. According to Kirchhoff's law, V_{F} is [26]

$$V_{\mathrm{F}} = -\sum_{i=1}^{M} \frac{R_f}{R_s} V_{\mathrm{I}i}. \tag{11}$$

The output voltage of the j^{th} column $V_{\mathrm{O}j}$ is

$$V_{\mathrm{O}j} = -\left[\sum_{i=1}^{M} (R_0 \times G_{ji} \times V_{\mathrm{I}i}) + \frac{R_0}{R_f} V_{\mathrm{F}}\right]. \tag{12}$$

Combining (12) with (11), $V_{\mathrm{O}j}$ is

$$V_{\mathrm{O}j} = \sum_{i=1}^{M} R_0 \times (G_s - G_{ji}) \times V_{\mathrm{I}i}. \tag{13}$$

The synaptic weight is

$$W_{ji} = R_0 \times (G_s - G_{ji}). \tag{14}$$

The comparator enables $V'_{\mathrm{O}j}$ as

$$V'_{\mathrm{O}j} = f(V_{\mathrm{O}j}) = \begin{cases} V_{\mathrm{H}} & \text{if } V_{\mathrm{O}j} > 0 \\ V_{\mathrm{L}} & \text{if } V_{\mathrm{O}j} \le 0, \end{cases} \tag{15}$$

where V_{H} and V_{L} ($V_{\mathrm{L}} = 0$) are, respectively, the high and low voltages of the comparator. The *output* of the estimator $\mathbf{V}_{\mathrm{O}} = \mathbf{W}\mathbf{V}_{\mathrm{I}}$ predicts the *target* output $\mathbf{V}_{\mathrm{T}}$ for new unseen *inputs* $\mathbf{V}_{\mathrm{I}}$. To solve this problem, the synapse *weights* $\mathbf{W}$ are updated to minimize the error between the outputs and target outputs over a K_0 long subset of the training set ($k = 1, 2, \ldots, K_0$). The error vector is

$$\Delta\mathbf{V}^{(k)} = \mathbf{V}_{\mathrm{T}}{}^{(k)} - \mathbf{V}'_{\mathrm{O}}{}^{(k)}. \tag{16}$$

A common measure is the mean square error (MSE) [23] which is

$$\mathrm{MSE} = \sum_{k=1}^{K_0} \|\Delta\mathbf{V}^{(k)}\|^2. \tag{17}$$

The performance of the outputs is tested over another subset, called the test set ($k = K_0 + 1, K_0 + 2, \ldots, K$).

A feasible iterative algorithm for minimizing the objective (17) is

$$\mathbf{W}^{(k+1)} = \mathbf{W}^{(k)} - \frac{1}{2}\eta\nabla_{\mathbf{W}^{(k)}}\|\Delta\mathbf{V}^{(k)}\|^2, \tag{18}$$

where η is the *learning rate*. Using the chain rule (9) and (16), $\nabla_{\mathbf{W}^{(k)}}\|\Delta\mathbf{V}^{(k)}\|^2 = -2(\mathbf{V}_{\mathrm{T}}{}^{(k)} - \mathbf{V}'_{\mathrm{O}}{}^{(k)})(\mathbf{V}_{\mathrm{I}}{}^{(k)})^{\top}$. Therefore, defining $\Delta\mathbf{W}^{(k)} \equiv \mathbf{W}^{(k+1)} - \mathbf{W}^{(k)}$, the *outer product* is

$$\Delta\mathbf{W}^{(k)} = \eta\Delta\mathbf{V}^{(k)}(\mathbf{V}_{\mathrm{I}}{}^{(k)})^{\top}, \tag{19}$$

or

$$\Delta W_{ji}^{(k)} = W_{ji}^{(k+1)} - W_{ji}^{(k)} = \eta\Delta V_j^{(k)}(V_{\mathrm{I}i}^{(k)}). \tag{20}$$

Specifically, the MNNs are commonly trained using the BP algorithm, which is an efficient form of online gradient descent [43]. Importantly, note that the update rule in (20) is local, i.e., the change in the synaptic weight $W_{ji}^{(k)}$ depends only on the input $V_{\mathrm{I}i}^{(k)}$ and error $\Delta V_j^{(k)}$.

To implement BP training for the neural network, the relevant time is determined for each step. From (14),

$$\Delta W_{ji}^{(k)} = -R_0\Delta G_{ji}^{(k)}. \tag{21}$$

From (20),

$$\begin{aligned}\Delta W_{ji}^{(k)} &= \eta(V_{\mathrm{T}j}^{(k)} - V_{\mathrm{O}j}^{\prime(k)})V_{\mathrm{I}i}^{(k)} \\ &= \begin{cases} 0 & \text{if } V_{\mathrm{T}j}^{(k)} = V_{\mathrm{O}j}^{\prime(k)} \text{ or } V_{\mathrm{I}i}^{(k)} = 0 \\ \eta V_{\mathrm{H}}^2 & \text{if } V_{\mathrm{I}i}^{(k)} = V_{\mathrm{H}}, V_{\mathrm{T}j}^{(k)} = V_{\mathrm{H}}, V_{\mathrm{O}j}^{\prime(k)} = 0 \\ -\eta V_{\mathrm{H}}^2 & \text{if } V_{\mathrm{I}i}^{(k)} = V_{\mathrm{H}}, V_{\mathrm{T}j}^{(k)} = 0, V_{\mathrm{O}j}^{\prime(k)} = V_{\mathrm{H}}. \end{cases}\end{aligned} \tag{22}$$

If $\Delta W_{ji}^{(k)} \neq 0$,

$$|\ \Delta G_{ji}^{(k)}\ | = |\ \frac{1}{R_{ji}^{(k+1)}} - \frac{1}{R_{ji}^{(k)}}\ | = \frac{\eta V_{\mathrm{H}}^2}{R_0}. \tag{23}$$

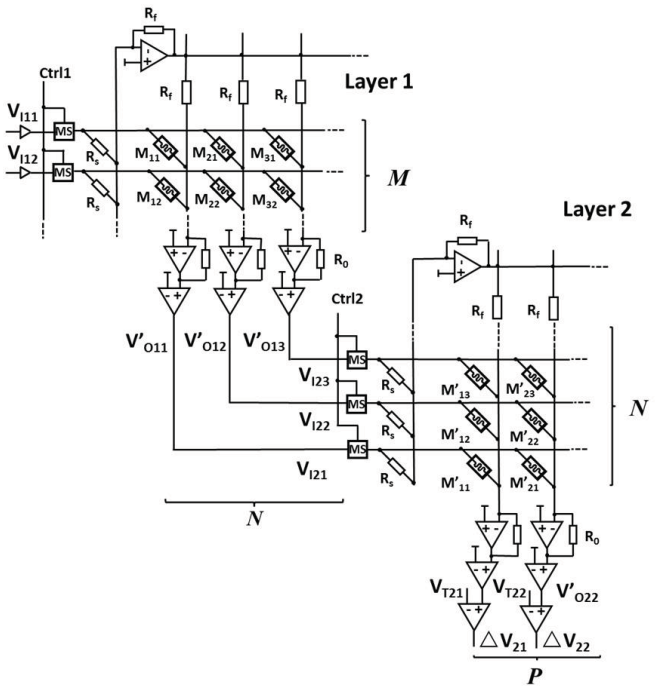


Fig. 3. Proposed memristor-based MNN.

Assuming $W_{ji} \in [-1, 1]$ while $R_{ji} \in [R'_{\mathrm{ON}}, R'_{\mathrm{OFF}}]$, R_0, R_s, R_f are determined from (14). From (7), the time $T^{(k)}$ in each step is

$$T^{(k)} = \begin{cases} \dfrac{V_{\mathrm{W}}(\ln R_{ji}^{(k)} - \ln R_{ji}^{(k+1)}) - i_0(R_{ji}^{(k)} - R_{ji}^{(k+1)})}{k' i_{\mathrm{off}}}, & V_{\mathrm{W}} > 0 \\ \dfrac{(R_{ji}^{(k)})^2 - (R_{ji}^{(k+1)})^2}{2k' V_{\mathrm{W}}} i_{\mathrm{on}}, & V_{\mathrm{W}} < 0. \end{cases} \tag{24}$$

B. Memristor-Based MNN

A simple two-layer neural network can be expanded to a multilayer NN, as shown in Fig. 3. A two-layer crossbar is considered. In layer 1, an $N \times M$ matrix $\mathbf{W}_{(1)}$ (W_{ji}) corresponds to N neurons and M inputs. In layer 2, a $P \times N$ matrix $\mathbf{W}_{(2)}$ (W_{pj}) corresponds to P neurons and N inputs.

An adaptive BP algorithm is [30]

1) Initialize the memristors with a high memristance R'_{OFF}.
2) Randomly apply the weight updating voltage V_{W} to all of the memristors to record all of the random values of the initial weights ($W_{ji}^{(1)}$ and $W_{pj}^{(1)}$) and memristances ($R_{ji}^{(1)}$ and $R_{pj}^{(1)}$).
3) Apply the input patterns $\mathbf{V}_{\mathrm{I}}$ to the crossbar circuit and evaluate any hidden and output neuron values.
4) For output layer neurons, determine the error $\Delta\mathbf{V}$ between the neuron output $\mathbf{V}'_{\mathrm{O}}$ and the target output $\mathbf{V}_{\mathrm{T}}$.

$$\Delta V_{1j}^{(k)} = \sum_{p=1}^{P} \Delta V_{2p}^{(k)} W_{pj}^{(k)}. \tag{25}$$

5) Back propagate the error.

$$\Delta W_{pj}^{(k)} = \eta(V_{T2p}^{(k)} - V_{O2p}^{\prime(k)})V_{I2j}^{(k)} = \begin{cases} 0 & \text{if } V_{T2p}^{(k)} = V_{O2p}^{\prime(k)} \text{ or } V_{I2j}^{(k)} = 0 \\ \eta V_H^2 & \text{if } V_{I2j}^{(k)} = V_H, V_{T2p}^{(k)} = V_H, V_{O2p}^{\prime(k)} = 0 \\ -\eta V_H^2 & \text{if } V_{I2j}^{(k)} = V_H, V_{T2p}^{(k)} = 0, V_{O2p}^{\prime(k)} = V_H, \end{cases} \quad (26)$$

where neuron p is a connected with the previous layer neuron j.

6) Apply write pulses to the crossbar with the pulse width proportional to ΔW_{pj} to update the memristor conductance. If $\Delta W_{pj}^{(k)} \neq 0$,

$$| \Delta G_{pj}^{(k)} | = | \frac{1}{R_{pj}^{(k+1)}} - \frac{1}{R_{pj}^{(k)}} | = \frac{\eta V_H^2}{R_0}. \quad (27)$$

7) Determine $\Delta W_{ji}^{(k)}$ to ensure that each memristor conductance is changed.

$$\Delta W_{ji}^{(k)} = \eta \Delta V_{1j}^{(k)} V_{I1i}^{(k)} = \begin{cases} 0 & \text{if } V_{I1i}^{(k)} = 0 \\ \eta V_H \sum_{p=1}^{P} \Delta V_{2p}^{(k)} W_{pj}^{(k)} & \text{if } V_{I1i}^{(k)} \neq 0. \end{cases} \quad (28)$$

8) Apply write pulses to the crossbar with the pulse width proportional to $\Delta W_{ji}^{(k)}$ to update the memristor conductance. If $\Delta W_{ji}^{(k)} \neq 0$,

$$| \Delta G_{ji}^{(k)} | = | \frac{1}{R_{ji}^{(k+1)}} - \frac{1}{R_{ji}^{(k)}} | = \left| \frac{\eta V_H \sum_{p=1}^{P} \Delta V_{2p}^{(k)} W_{pj}^{(k)}}{R_0} \right|. \quad (29)$$

9) If the error does not converge to a sufficiently small value, return to step 2).

Remark 1: Using a similar method, (24)-(29) can be written as a general expression describing N layers.

IV. System Operation

The methods of operating the proposed BP training and synaptic weight adjustment process on-chip are introduced in this section, which exhibit the advantages of no sneak paths. In section IV.A, a four step BP on-chip training method is presented. Sneak path methods that change the synaptic weight in the neural network array are described in section IV.B.

A. BP Training Circuit

The circuit implementation of BP training for the neural network circuits is shown in Fig. 3. The training is composed of four steps [30]:

1 Apply input voltages to layer 1 and record layer 2 neuron output errors.
2 Back propagate layer 2 errors through the second layer weights and record layer 1 errors.

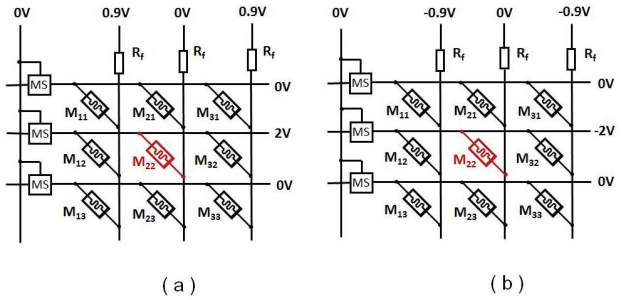


Fig. 4. Memristor-based synaptic weight adjustment method. (a) M_{22} is adjusted from 1 to -1, and (b) M_{22} is adjusted from -1 to 1.

3 Update layer 2 synaptic weights based on layer 2 errors.
4 Update layer 1 synaptic weights based on layer 1 errors.

The four-step operation of the proposed BP training is described as follows:

Step 1: The input signals, Ctrl1 and Ctrl2, are set to V_H, turning on the MS. A set of input voltages is applied to the layer 1 neurons, and the layer 2 neuron outputs $V_{O21}, V_{O22}, \ldots, V_{O2p}, \ldots, V_{O2P}$ are compared with the expected outputs $V_{T21}, V_{T22}, \ldots, V_{T2p}, \ldots, V_{T2P}$. This process is shown in Fig. 3. The error terms $\Delta V_{21}, \Delta V_{22}, \ldots, \Delta V_{2p}, \ldots, \Delta V_{2P}$ are based on the difference between the observed outputs and the expected outputs. These values are generated using comparators that provide a discretized error value of V_H, $-V_H$, or 0. These errors are amplified (to 2 V, -2 V, or 0) and applied to the synaptic array to change the value of the memristances, as shown in Fig. 4. The value of $|\Delta W_{ji}|$ can be obtained in testing process and the corresponding $|\Delta G_{ji}|$ (or $|\Delta R_{ji}|$) can be calculated with FPGA or LUT (lookup table). The initial states of memristors are recorded and the calculation of adjustment time can be processed in FPGA. Each synaptic adjustment is controlled with peripheral circuits.

Step 2: To back propagate the errors of layer 2, Ctrl1 and Ctrl2 are set to V_H. The layer 2 errors $(\Delta V_{21}, \Delta V_{22}, \ldots, \Delta V_{2p}, \ldots, \Delta V_{2P})$ are applied to the layer 2 weights to generate the layer 1 errors $(\Delta V_{11}, \Delta V_{12}, \ldots, \Delta V_{1j}, \ldots, \Delta V_{1N})$.

Step 3: Ctrl1 and Ctrl2 are set to 0 (MS is turned off) to isolate the second layer crossbar from the first layer. A training unit amplifies the layer 2 errors along with the layer 2 intermediate outputs to generate a set of training pulses. These pulses are applied to the layer 2 memristor crossbar to update the layer 2 synaptic weights.

Step 4: To isolate the first layer crossbar from the input voltages, Ctrl1 and Ctrl2 are set to 0. A process similar to *step 3* is applied to update the synaptic weights in the memristor crossbar array of layer 1.

B. Synaptic Weight Updating

In the testing process, the applied voltage is the input voltage 0 or V_H. The final output voltages are error signal voltages 0, V_H or $-V_H$. The applied weight update voltages are amplified to 0, 2 V or -2 V, which are controlled with

TABLE I
WEIGHT UPDATE DIRECTION

ΔV_j	ΔW_{ji}	W_{ji}	G_{ji}	R_{ji}	V_{W}
$+$	$+$	Increase	Decrease	Increase	$-$
$-$	$-$	Decrease	Increase	Decrease	$+$

peripheral circuits. The memristor weight updating process utilized by the training unit is presented in this section.

All the weights within a crossbar can be updated using the following method. Different grid sizes of the crossbar arrays have been evaluated, and a 3×3 crossbar array is considered as an example. When a voltage greater than the memristor threshold is applied across a memristor, the memristance increases or decreases depending upon the terminal of the device. The two possible update situations of ΔV_j are listed in Table I along with the direction that the weight is changed, or $\Delta V_j = 0$ and the weight remains unchanged.

1) Decreasing Synaptic Weight: If $\Delta V_j < 0$, $V_{\mathrm{W}} > 0$. Assume only memristor M_{22} is changed and the other memristors remain unchanged. One possible solution is to apply different voltages on different rows and columns. For $V_{\mathrm{W}} = 2$ V on row 2 and the other rows are connected to GND, column 2 is connected to GND and 0.9 V is applied to the other columns, as shown in Fig. 4(a). The threshold voltage (V_{T}) of the synapse memristors is 1.5 V, and R_f is much smaller than the memristance of M_{ji}. Only the voltage of memristor M_{22} is 2 V, which is larger than V_{T}. The memristance of M_{22} therefore decreases. The voltage of memristors M_{11}, M_{13}, M_{31}, and M_{33} is -0.9 V, while the voltage of memristors M_{12} and M_{32} is 1.1 V and the voltage of memristors M_{21} and M_{23} is 0 V. The absolute values are all smaller than V_{T}, the memristances therefore remain the same.

2) Increasing Synaptic Weight: If $\Delta V_j > 0$, $V_{\mathrm{W}} < 0$. A similar method is utilized, as shown in Fig. 4(b).

To make changes in the memristor conductances, a voltage of appropriate magnitude and polarity for a suitable duration across the memristor is applied [44]–[46]. In each step, the required durations $T_2^{(1)}, T_2^{(2)}, \ldots, T_2^{(k)}, \ldots, T_2^{(K)}$ can be determined from (27) and (24), and $T_1^{(1)}, T_1^{(2)}, \ldots, T_1^{(k)}, \ldots, T_1^{(K)}$ can be determined from (29) and (24).

V. ROBUSTNESS ANALYSIS

Process variations and noise can significantly affect circuit performance. In this section, these noise sources, the impact of physical design challenges, and sneak path currents are discussed.

A. Sources of Noise

Electrical noise from the power supplies and neighboring wires can significantly degrade the quality of the analog signals. Different from process variations that remain unchanged once the circuit is fabricated, signal fluctuations vary during circuit operation. When character recognition is operated on-chip, the input voltages can be affected by random noise. The inputs are changed from 0 to V_{H} to evaluate the robustness of the proposed MS and synaptic circuit.

TABLE II
PARAMETERS OF THE MEMRISTOR-BASED MNN

Parameters of the MNN	***Values***
R_{ON} (Ω)	100
R_{OFF} (Ω)	9000
R'_{ON} (MΩ)	1
R'_{OFF} (MΩ)	200
R_0 (MΩ)	2.01
R_{s} (MΩ)	1.99
R_{f} (kΩ)	2
V_{T} (V)	1.5
V_{H} (V)	0.9
V_{L} (V)	0
V_{W} (V)	±2
D (nm)	1
i_0 (A)	9e-9
i_{on} (A)	1
i_{off} (A)	8.8e-16
μ_v ($\mathrm{m^2 s^{-1} \Omega^{-1}}$)	1e-5
μ'_v ($\mathrm{m^2 s^{-1} \Omega^{-1}}$)	1e-7

B. Physical Challenges

There are four major physical constraints: 1) The memristors in the MS and synaptic array are different; 2) The boundary voltage V_{H} is smaller than the V_{T} of the memristors; 3) The sum of V_{H} and the maximum amplitude of the noise sources is smaller than V_{T}. 4) The precision of the synaptic adjustment process depends upon the precision of the conductance of the memristors and is adjusted by applying a different number of voltage pulses.

C. Sneak Path Currents

The training process is not affected by sneak path currents [47]. Only one write line is raised to reach the weight changing voltage V_{W}. Other lines are protected to ensure the absolute value of the voltages remain smaller than V_{T}, as shown in Fig. 4. V_{W} is set higher than V_{T} since only a small change in the memristance is needed during each training step. Hence, the voltage drop on the other memristors is smaller than V_{T}, and does not result in unexpected changes in the memristance. However, in larger arrays, after a significant number of pulses (such as a 16 kb array after 5×10^6 pulses), the write operation is disturbed and exhibits 164 false bits [48].

VI. SIMULATION RESULTS

PSPICE is used to evaluate the proposed memristor-based MNN circuits. The circuits and learning process are also evaluated in Matlab. The basic memristor model [38] and the synaptic model with threshold voltages [41] are both used in the simulations. The parameters of the memristor-based MNN are listed in Table II. The drive voltages of amplifiers A_0 and A_1 are ±5 V, 0.9 V and 0 V for A_2, and ±0.9 V for A_3. Many different voltage levels are required, ±5 V voltages is utilized to produce ±0.9 V and ±2 V voltages through step down circuits. The range of weights supported by a memristor-based synapse is $[-1, +1]$.

A. Synaptic Weight Updating

Different grid sizes of crossbars have been considered. A 3×3 crossbar array, for example, considers that memristor

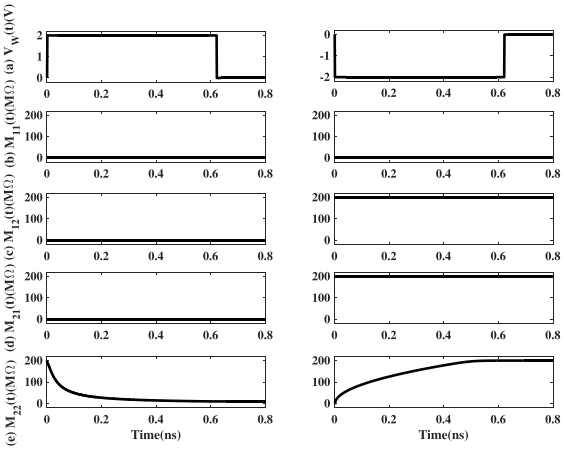


Fig. 5. Simulation results of memristor-based synaptic weight adjustment process. (a) Input voltage V_W for weight adjustment operation, (b) memristance of memristor M_{11} (M_{13}, M_{31}, M_{33} are the same), (c) memristance of memristor M_{12} (M_{32} is the same), (d) memristance of memristor M_{21} (M_{23} is the same), and (e) memristance of memristor M_{22}. X represents time and Y represents voltage.

M_{22} in row two, column two is accessed. Using the synaptic weight updating approach, the simulation results demonstrate that only M_{22} can be updated from R'_{OFF} to R'_{ON} or from R'_{ON} to R'_{OFF} (shown in Fig. 5) while the other memristors remain the same during the updating process, verifying that the sneak path currents have been eliminated during the weight updating process.

The switching time (gate delay) is approximately equal to 0.02 ns from (5), and the error is approximately equal to 0.01 V from (6). A pulse of amplitude 2 V and width 0.61 ns from (8) change the memristance from R'_{OFF} to R'_{ON} (the weight of the synapse from 1 to -1). Similarly, a pulse of amplitude -2 V and width 0.61 ns change the memristance from R'_{ON} to R'_{OFF} (the weight of the synapse from -1 to 1), as shown in Fig. 5.

B. XOR Function

A simple two layer neural network is simulated to verify the correctness of the learning method. The XOR function is learned on a network of 2 inputs $\times$ 3 hiddens $\times$ 1 output, as shown in Fig. 7. In layer 1, a 3×2 memristor matrix corresponds to three neurons and two inputs. In layer 2, a 1×3 memristor matrix corresponds to three inputs and one output. One cycle contains four iterations for four different input patterns and the learning rate $\eta = 0.1$. The output voltages during each iteration are shown in Figs. 6. The error of the hardware training process for each neuron during each cycle is shown in Fig. 8 (1,000 events are tested to produce different samples). The error E_e is

$$E_e = \sqrt{\frac{1}{K_0}\mathrm{MSE}}. \tag{30}$$

C. Character Recognition

A more complex function is simulated on two different networks for character recognition. One network is a single layer composed of 30 inputs $\times$ 10 outputs. The network is trained on black and white images of size 5×6 pixels, as shown in Fig. 9. Consider, for example, image 5. The input and output voltages are shown in Table III ("1" represents V_{H}). One cycle contains ten iterations for ten different input patterns and the learning rate $\eta = 0.1$. The error ($K_0 = 10$) of hardware

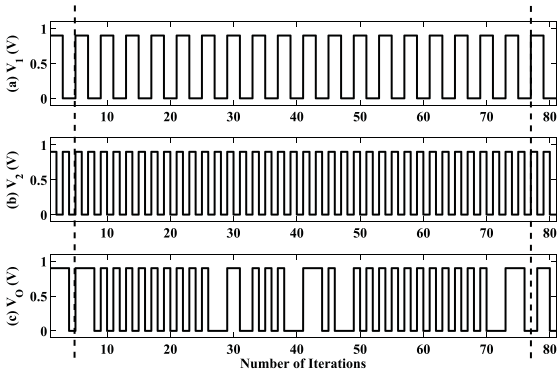
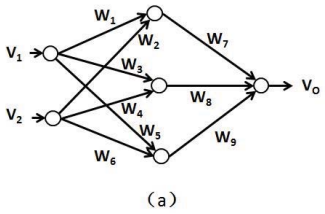


Fig. 6. Changes in input and output voltages during each iteration of the XOR operation. (a) Input voltage V_{I11}, (b) input voltage V_{I12}, and (c) output voltage V'_{O21}.



V_1	V_2	V_O
0	0	0
1	0	1
0	1	1
1	1	0

Fig. 7. XOR operation, (a) MNN, and (b) truth table.

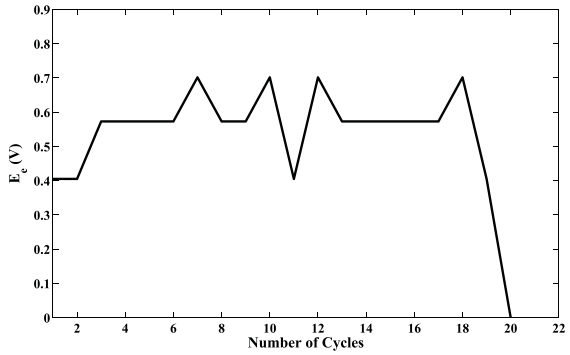


Fig. 8. Training error in each cycle for the XOR operation in the MNN.

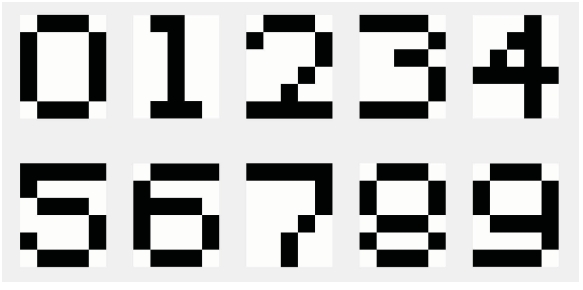


Fig. 9. Ten digit images used as an input to the network for character recognition.

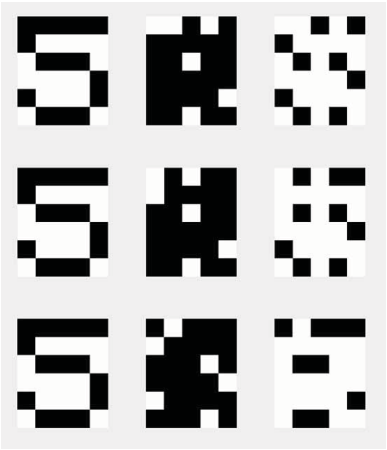


Fig. 10. Input image 5 in the left column is shown. Images with 20% random pixels as white noise are shown in the middle column to produce a noisy image mask. The resulting noisy images used for testing are shown in the right column.

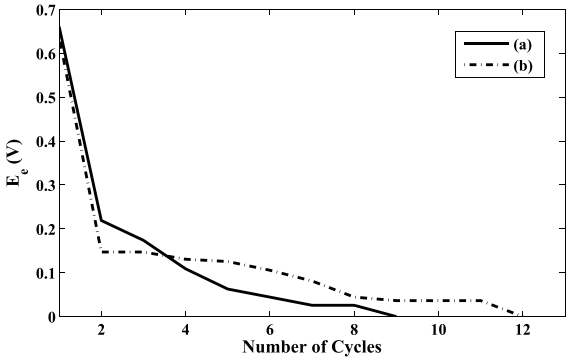


Fig. 11. Training error in each cycle for character recognition in the SNN. (a) Input images are without noise, and (b) input images are with 20% noise.

TABLE III
INPUT AND OUTPUT VOLTAGE ARRAYS OF IMAGE 5

Input Voltages	$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$	
Output Voltages	SNN	$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$
	MNN	$\begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$

training for each neuron in each cycle is shown in Fig. 11(a) (1,000 events are tested to produce different samples).

The robustness of the memristive neural network circuits is tested using following method. Noise is added to input images

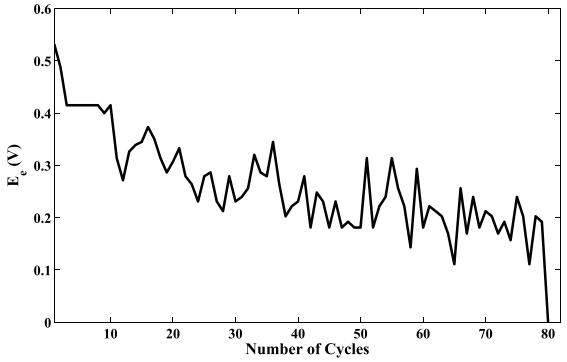


Fig. 12. Training error in each cycle for character recognition in the MNN.

TABLE IV
COMPARISON OF RECOGNITION RATE FOR DIFFERENT NOISE LEVELS

Noise Level	*Correct Recognition in the SNN [18]*	*Correct Recognition in the Proposed SNN*
5%	99.8%	100.0%
10%	99.6%	99.8%
15%	99.1%	99.4%
20%	98.3%	99.0%
30%	72.5%	97.4%

(e.g. number 5) to generate noisy images, as shown in Fig. 10. The original images for number 5 are shown in the left column, and uniform random noise mask images are shown in the middle column. Noisy images are acquired by flipping all of the pixels in the character images wherever there is a white pixel in the noise mask image (assuming 20% noise). The average number of cycles for the correct recognition is shown in Fig. 11(b). The recognition rate of the network is 99.0% when 20% noise is added to the images, as listed in Table IV.

Another example network is two layers of 30 inputs $\times$ 6 hiddens $\times$ 4 outputs. Consider again, for example, image 5. The input and output voltages are listed in Table III. One cycle contains ten iterations for ten different input patterns and the learning rate $\eta = 0.04$. The error ($K_0 = 10$) of hardware training for each neuron in each cycle is shown in Fig. 12 (1,000 events are tested to produce different samples). From simulation, the recognition rate of the network is 95.4% when 20% noise is added to the images.

The recognition rate of the proposed memristor-based neural networks when noise is added to the images is compared with [18] in Table IV. The proposed memristor-based BP algorithm is compared with the winner-take-all algorithm [18] in Table V. Note that the proposed system exhibits a higher recognition rate and requires fewer cycles.

D. Effects of Memristance Variations

In a practical fabricated memristor array, the the updating process is achieved by applying different numbers of positive or negative voltage pulses. In each step, the memristance or conductance of memristors are not adjusted precisely, the effects of memristance variations are therefore discussed in this section.

To obtain a desired change in the memristance, a voltage of appropriate magnitude and polarity over a suitable duration

TABLE V
COMPARISON OF THE PROPOSED MEMRISTOR-BASED BP ALGORITHM WITH THE WINNER-TAKE-ALL ALGORITHM IN [18]

	Neural Networks in [18]	***Proposed Neural Networks***	
Synaptic Structure	2M	1M	
Algorithm	Winner-Take-All	BP	
Layer(s)	Single	Single	Double
Memristors	600	300	204
Cycles (Epochs)	116	9 average	80 average
Error	0	0	0
Output Bits	10	10	4

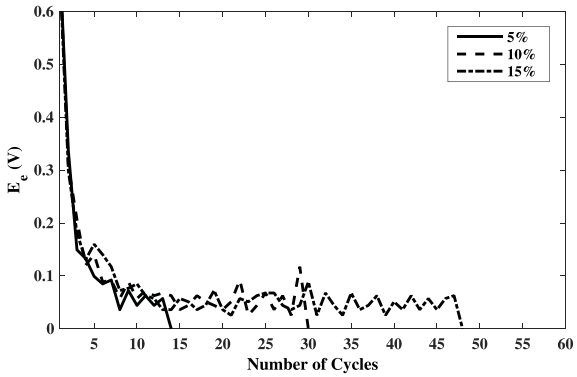


Fig. 13. Training error during each cycle with different levels of memristance variations for character recognition in the SNN.

(pulse number) is applied across the memristor. However, due to cycle-to-cycle variations, the change during each step cannot be accurately controlled. Different random noise ranging from 5% to 15% is therefore added to the memristance during each step, as shown in Fig. 13. The corresponding cycle numbers are, respectively, 14, 30, and 48. The proposed memristor-based MNN is shown to be inherently tolerant to memristance variations.

VII. CONCLUSIONS

A single memristor-based synaptic architecture for multilayer neural networks with on-chip learning is proposed. Moreover, an adaptive BP algorithm suitable for the proposed memristor-based multilayer neural network is applied to train neural networks and perform the XOR function and character recognition. A simple, compact, and reliable neural network can be used for applications in pattern recognition by combining the advantages of the memristor-based synaptic architecture with the proposed BP weight change algorithm. The advantages of the proposed architecture are verified through simulations, demonstrating that the proposed adaptive BP algorithm exhibits higher recognition rates and requires fewer cycles.

REFERENCES

[1] Y. Zhang, Y. Shen, X. Wang, and Y. Guo, "A novel design for memristor-based OR gate," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 62, no. 8, pp. 781–785, Aug. 2015.

[2] Y. Zhang, Y. Shen, X. Wang, and L. Cao, "A novel design for memristor-based logic switch and crossbar circuits," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 5, pp. 1402–1411, May 2015.

[3] G. Indiveri, B. Linares-Barranco, R. Legenstein, G. Deligeorgis, and T. Prodromakis, "Integration of nanoscale memristor synapses in neuromorphic computing architectures," *Nanotechnol.*, vol. 24, no. 38, p. 384010, Sep. 2013.

[4] L. Chua, "Memristor-The missing circuit element," *IEEE Trans. Circuit Theory*, vol. 18, no. 5, pp. 507–519, Sep. 1971.

[5] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, pp. 80–83, May 2008.

[6] P. Junsangsri and F. Lombardi, "Design of a hybrid memory cell using memristance and ambipolarity," *IEEE Trans. Nanotechnol.*, vol. 12, no. 1, pp. 71–80, Jan. 2013.

[7] Y. Zhang, Y. Li, X. Wang, and E. G. Friedman, "Synaptic characteristics of Ag/AgInSbTe/Ta-based memristor for pattern recognition applications," *IEEE Trans. Electron Devices*, vol. 64, no. 4, pp. 1806–1811, Apr. 2017.

[8] L. Wang, Y. Shen, Q. Yin, and G. Zhang, "Adaptive synchronization of memristor-based neural networks with time-varying delays," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 9, pp. 2033–2042, Sep. 2015.

[9] M. Walker, P. Hasler, and L. A. Akers, "A CMOS neural network for pattern association," *IEEE Micro*, vol. 9, no. 5, pp. 68–74, Oct. 1989.

[10] B. V. Benjamin *et al.*, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proc. IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[11] K. D. Cantley, A. Subramaniam, H. J. Stiegler, R. A. Chapman, and E. M. Vogel, "Neural learning circuits utilizing nano-crystalline silicon transistors and memristors," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 4, pp. 565–573, Jun. 2012.

[12] H. Manem, J. Rajendran, and G. S. Rose, "Stochastic gradient descent inspired training technique for a CMOS/nano memristive trainable threshold gate array," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 5, pp. 1051–1060, May 2012.

[13] Z. Wang, W. Zhao, W. Kang, Y. Zhang, J.-O. Klein, and C. Chappert, "Ferroelectric tunnel memristor-based neuromorphic network with 1T1R crossbar architecture," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2014, pp. 29–34.

[14] S. P. Adhikari, H. Kim, R. K. Budhathoki, C. Yang, and L. O. Chua, "A circuit-based learning architecture for multilayer neural networks with memristor bridge synapses," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 1, pp. 215–223, Jan. 2015.

[15] S. P. Adhikari, C. Yang, H. Kim, and L. O. Chua, "Memristor bridge synapse-based neural network and its learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 9, pp. 1426–1435, Sep. 2012.

[16] H. Kim, M. P. Sah, C. Yang, T. Roska, and L. O. Chua, "Neural synaptic weighting with a pulse-based memristor circuit," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 1, pp. 148–158, Jan. 2012.

[17] C. Zamarreño-Ramos, L. A. Camuñas-Mesa, J. A. Pérez-Carrasco, T. Masquelier, T. Serrano-Gotarredona, and B. Linares-Barranco, "On spike-timing-dependent-plasticity, memristive devices, and building a self-learning visual cortex," *Frontiers Neurosci.*, vol. 5, p. 26, Mar. 2011.

[18] A. M. Sheri, H. Hwang, M. Jeon, and B.-G. Lee, "Neuromorphic character recognition system with two PCMO memristors as a synapse," *IEEE Trans. Ind. Electron.*, vol. 61, no. 6, pp. 2933–2941, Jun. 2014.

[19] M. Chu *et al.*, "Neuromorphic hardware system for visual pattern recognition with memristor array and CMOS neuron," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2410–2419, Apr. 2015.

[20] S. N. Truong, S.-J. Ham, and K.-S. Min, "Neuromorphic crossbar circuit with nanoscale filamentary-switching binary memristors for speech recognition," *Nanoscale Res. Lett.*, vol. 9, no. 1, pp. 1–9, Nov. 2014.

[21] R. Legenstein, C. Naeger, and W. Maass, "What can a neuron learn with spike-timing-dependent plasticity?" *Neural Comput.*, vol. 17, no. 11, pp. 2337–2382, 2005.

[22] M. Hu, H. Li, Y. Chen, Q. Wu, G. S. Rose, and R. W. Linderman, "Memristor crossbar-based neuromorphic computing system: A case study," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 10, pp. 1864–1878, Oct. 2014.

[23] D. Soudry, D. Di Castro, A. Gal, A. Kolodny, and S. Kvatinsky, "Memristor-based multilayer neural networks with online gradient descent training," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2408–2421, Oct. 2015.

[24] E. Rosenthal, S. Greshnikov, D. Soudry, and S. Kvatinsky, "A fully analog memristor-based neural network with online gradient training," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2016, pp. 1394–1397.

[25] M. Prezioso, F. Merrikh-Bayat, B. D. Hoskins, G. C. Adam, K. K. Likharev, and D. B. Strukov, "Training and operation of an integrated neuromorphic network based on metal-oxide memristors," *Nature*, vol. 521, pp. 61–64, May 2015.

[26] S. N. Truong and K.-S. Min, "New memristor-based crossbar array architecture with 50-% area reduction and 48-% power saving for matrix-vector multiplication of analog neuromorphic computing," *J. Semicond. Tech. Sci.*, vol. 14, no. 3, pp. 356–363, Jun. 2014.

[27] Q. Wang, Y. Kim, and P. Li, "Architectural design exploration for neuromorphic processors with memristive synapses," in *Proc. 14th Int. Conf. Nanotechnol. (IEEE-NANO)*, Aug. 2014, pp. 962–966.

[28] D. Querlioz, O. Bichler, A. F. Vincent, and C. Gamrat, "Bioinspired programming of memory devices for implementing an inference engine," *Proc. IEEE*, vol. 103, no. 8, pp. 1398–1416, Aug. 2015.

[29] D. Zhang *et al.*, "All spin artificial neural networks based on compound spintronic synapse and neuron," *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 4, pp. 828–836, Aug. 2016.

[30] R. Hasan and T. M. Taha, "Enabling back propagation training of memristor crossbar neuromorphic processors," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Beijing, China, Jul. 2014, pp. 21–28.

[31] I. Kataeva, F. Merrikh-Bayat, E. Zamanidoost, and D. Strukov, "Efficient training algorithms for neural networks based on memristive crossbar circuits," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2015, pp. 1–8.

[32] D. Chabi, Z. Wang, W. Zhao, and J.-O. Klein, "On-chip supervised learning rule for ultra high density neural crossbar using memristor for synapse and neuron," in *Proc. Int. Symp. Nanoscale Archit. (NANOARCH)*, 2014, pp. 7–12.

[33] S. Kvatinsky, N. Wald, G. Satat, A. Kolodny, U. C. Weiser, and E. G. Friedman, "MRL–memristor ratioed logic," in *Proc. 13th Int. Workshop Cellular Nanoscale Netw. Appl.*, Aug. 2012, pp. 1–6.

[34] S. P. Adhikari, M. P. Sah, H. Kim, and L. O. Chua, "Three fingerprints of memristor," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 11, pp. 3008–3021, Nov. 2013.

[35] S. Kvatinsky, E. G. Friedman, A. Kolodny, and U. C. Weiser, "TEAM: Threshold adaptive memristor model," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 1, pp. 211–221, Jan. 2013.

[36] S. Kvatinsky, M. Ramadan, E. G. Friedman, and A. Kolodny, "VTEAM: A general model for voltage-controlled memristors," *IEEE Trans. Circuits Syst. II, Express Briefs*, vol. 62, no. 8, pp. 786–790, Aug. 2015.

[37] A. Ascoli, F. Corinto, V. Senger, and R. Tetzlaff, "Memristor model comparison," *IEEE Circuits Syst. Mag.*, vol. 13, no. 2, pp. 89–105, 2nd Quart., 2013.

[38] Z. Biolek, D. Biolek, and V. Biolkova, "SPICE model of memristor with nonlinear dopant drift," *Radioengineering*, vol. 18, no. 2, pp. 210–214, Jun. 2009.

[39] D. Batas and H. Fiedler, "A memristor SPICE implementation and a new approach for magnetic flux-controlled memristor modeling," *IEEE Trans. Nanotechnol.*, vol. 10, no. 2, pp. 250–255, Mar. 2011.

[40] C. Yakopcic, T. M. Taha, G. Subramanyam, and R. E. Pino, "Generalized memristive device SPICE model and its application in circuit design," *IEEE Trans. Comput.-Aided Design Integr.*, vol. 32, no. 8, pp. 1201–1214, Aug. 2013.

[41] Y. Zhang, X. Wang, Y. Li, and E. G. Friedman, "Memristive model for synaptic circuits," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 64, no. 7, pp. 767–771, Jul. 2017.

[42] Y. Li *et al.*, "Activity-dependent synaptic plasticity of a chalcogenide electronic synapse for neuromorphic systems," *Sci. Rep.*, vol. 4, p. 4906, May 2014.

[43] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural Networks: Tricks of the Trade*, 2nd ed. Berlin, Germany: Springer, 2012, pp. 9–48.

[44] L. Gao, F. Alibart, and D. B. Strukov, "Programmable CMOS/memristor threshold logic," *IEEE Trans. Nanotechnol.*, vol. 12, no. 2, pp. 115–119, Mar. 2013.

[45] G. Papandroulidakis, I. Vourkas, N. Vasileiadis, and G. C. Sirakoulis, "Boolean logic operations and computing circuits based on memristors," *IEEE Trans. Circuits Syst. II, Express Briefs*, vol. 61, no. 12, pp. 972–976, Dec. 2014.

[46] I. Vourkas and G. C. Sirakoulis, "On the generalization of composite memristive network structures for computational analog/digital circuits and systems," *Microelectr. J.*, vol. 45, no. 11, pp. 1380–1391, Nov. 2014.

[47] M. A. Zidan, H. A. H. Fahmy, M. M. Hussain, and K. N. Salama, "Memristor-based memory: The sneak paths problem and solutions," *Microelectron. J.*, vol. 44, no. 2, pp. 176–183, 2013.

[48] H. Li *et al.*, "Write disturb analyses on half-selected cells of cross-point RRAM arrays," in *Proc. IEEE Int. Rel. Phys. Symp. (IRPS)*, Jun. 2014, pp. MY.3.1–MY.3.4.

Yang Zhang received the B.S and Ph.D. degrees in control science and engineering from the School of Automation, Huazhong University of Science and Technology, in 2013 and 2017, respectively.

From 2015 to 2016, he was a Visiting Ph.D. Student with the University of Rochester. He is currently a Researcher with the Shenzhen Key Laboratory of Spatial Smart Sensing and Services, School of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include memristor-based memory, deep neural networks, and image recognition.

Xiaoping Wang (M'14) was born in Hubei, China, in 1974. She received the B.S. degree and the M.S. degree in automation from Chongqing University, Chongqing, China, in 1997 and 2000, respectively, and the Ph.D. degree in systems engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2003. Since 2011, she has been a Professor with the School of Automation, Huazhong University of Science and Technology. Her current research interests include memristors and its application to memory storage, modeling, simulation, and optimization.

Eby G. Friedman (S'80–M'81–SM'90–F'00) received the B.S. degree from Lafayette College in 1979, and the M.S. and Ph.D. degrees from the University of California at Irvine, Irvine, CA, USA, in 1981 and 1989, respectively, all in electrical engineering.

From 1979 to 1991, he was with Hughes Aircraft Company, promoted to Manager of the Signal Processing Design and Test Department, responsible for the design and test of high performance digital and analog ICs. He has been with the Department of Electrical and Computer Engineering, University of Rochester, since 1991, where he is a Distinguished Professor and the Director of the High Performance VLSI/IC Design and Analysis Laboratory. He is also a Visiting Professor with the Technion âŁ" Israel Institute of Technology. He has authored over 500 papers and book chapters, 13 patents, and the author or editor of 18 books in the fields of high speed and low power CMOS design techniques, 3-D design methodologies, high speed interconnect, and the theory and application of synchronous clock and power distribution networks. His current research and teaching interests are in high performance synchronous digital and mixed-signal microelectronic design and analysis with application to high speed portable processors and low power wireless communications.

Dr. Friedman is a Senior Fulbright Fellow. He was a recipient of the IEEE Circuits and Systems Charles A. Desoer Technical Achievement Award, the University of Rochester Graduate Teaching Award, the College of Engineering Teaching Excellence Award, and a member of the University of California at Irvine Engineering Hall of Fame. He is an Editor-in-Chief of the *Microelectronics Journal*, a member of the editorial boards of the *Journal of Low Power Electronics* and the *Journal of Low Power Electronics and Applications*, and a member of the technical program committee of numerous conferences. He was an Editor-in-Chief and the Chair of the Steering Committee of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS, the Regional Editor of the *Journal of Circuits, Systems and Computers*, a member of the Editorial Board of the PROCEEDINGS OF THE IEEE, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II: ANALOG AND DIGITAL SIGNAL PROCESSING, the IEEE JOURNAL ON EMERGING AND SELECTED TOPICS IN CIRCUITS AND SYSTEMS, *Analog Integrated Circuits and Signal Processing*, and the *Journal of Signal Processing Systems*, a member of the Circuits and Systems Society, Board of Governors, and Program and Technical chair of several IEEE conferences.