Technology Aware Training in Memristive Neuromorphic Systems for Nonideal Synaptic Crossbars

Indranil Chakraborty, Deboleena Roy, and Kaushik Roy, *Fellow, IEEE*

***Abstract*—The advances in the field of machine learning using neuromorphic systems have paved the pathway for extensive research on possibilities of hardware implementations of neural networks. Various memristive technologies such as oxide-based devices, spintronics, and phase change materials have been explored to implement the core functional units of neuromorphic systems, namely the synaptic network, and the neuronal functionality, in a fast and energy efficient manner. However, various nonidealities in the crossbar implementations of the synaptic arrays can significantly degrade performance of neural networks, and hence, impose restrictions on feasible crossbar sizes. In this paper, we build mathematical models of various nonidealities that occur in crossbar implementations such as source resistance, neuron resistance, and chip-to-chip device variations and analyze their impact on the classification accuracy of a fully connected network (FCN) and convolutional neural network (CNN) trained with Backpropagation algorithm. We show that a network trained under ideal conditions can suffer accuracy degradation as large as 59.84% for FCNs and 62.4% for CNNs when implemented on nonideal crossbars for relevant nonideality ranges. This severely constrains the sizes for crossbars. As a solution, we propose a technology aware training algorithm, which incorporates the mathematical models of the nonidealities in the backpropagation algorithm. We demonstrate that our proposed methodology achieves significant recovery of testing accuracy within 1.9% of the ideal accuracy for FCNs and 1.5% for CNNs. We further show that our proposed training algorithm can potentially allow the use of significantly larger crossbar arrays of sizes 784 $\times$ 500 for FCNs and 4096 $\times$ 512 for CNNs with a minor or no tradeoff in accuracy.**

***Index Terms*—Neural networks, memristive crossbar, backpropagation, neuromorphic, image recognition.**

I. Introduction

RECENT developments in computational neuroscience have resulted in a paradigm shift away from Boolean computing in sequential von-Neumann architectures as the research community strives to emulate the functionality of the human brain on neurocomputers. Although extensive research has been done to accelerate computational functions such as matrix operations on general-purpose computers, the parallelism of the human brain has remained elusive to von-Neumann architecture, thus engendering high hardware cost and energy consumption [1]. This has resulted in the exploration of non-von Neumann architectures with 'massively parallel operations in-memory', thus avoiding the overhead cost of exchanging data between memory and processor. Especially with the recent advances in machine learning in various cognitive tasks such as image recognition, natural language processing etc, the search for such energy-efficient 'in-memory computing' platforms has become quintessential. Although standardized hardware implementations of neuromorphic systems like $CAVIAR$ [2], $IBM\ TrueNorth$ [3], $SpiNNaker$ [4] have primarily been dominated by CMOS technology, the memristor-based non-volatile memory (NVM) technology [5]–[9] has naturally evolved into an exciting prospect. To that end, various technologies such as spintronics [10], oxide-based memristors [11], [12], phase change materials (PCM) [13], etc., have shown promising progress in mimicking the functionality of the core computational units of a neural network, i.e., neurons and synapses.

The core functionality of a neuromorphic system is a parallelized dot product between the inputs and the synaptic weights [14]. This has been demonstrated to be efficiently realized by a dense resistive crossbar array [15], [16]. The ability to naturally compute matrix multiplications makes crossbar arrays the most convenient way of implementing neuromorphic systems. However, real crossbars could suffer from various non-idealities including device variations [17], [18], parasitic resistances, non-ideal sources, and neuron resistances. Although neural networks are generally robust against small variations in the crossbar, the aforementioned technological constraints can severely impact accuracy of recognition tasks as well as restrict the crossbar size. Several techniques such as redundancy schemes [19], technology optimization [20] and modified training algorithms [21]–[23] have been explored for both on-chip and ex-situ learning to mitigate specific non-ideal effects such as IR drops, synaptic device variations. However, mathematical modeling of non-idealities and its incorporation in standard training algorithm needs further exploration.

Manuscript received November 24, 2017; revised February 11, 2018; accepted March 4, 2018. Date of publication September 20, 2018; date of current version September 21, 2018. This work was supported in part by the Center for Brain-inspired Computing Enabling Autonomous Intelligence (C-BRIC), one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, in part by the National Science Foundation, in part by the Intel Corporation, in part by the ONR MURI program, and in part by the Vannevar Bush Faculty Fellowship. *(Corresponding author: Indranil Chakraborty.)*

The authors are with the Department of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47906 USA (e-mail: ichakra@purdue.edu; roy77@purdue.edu; kaushik@purdue.edu).

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/TETCI.2018.2829919

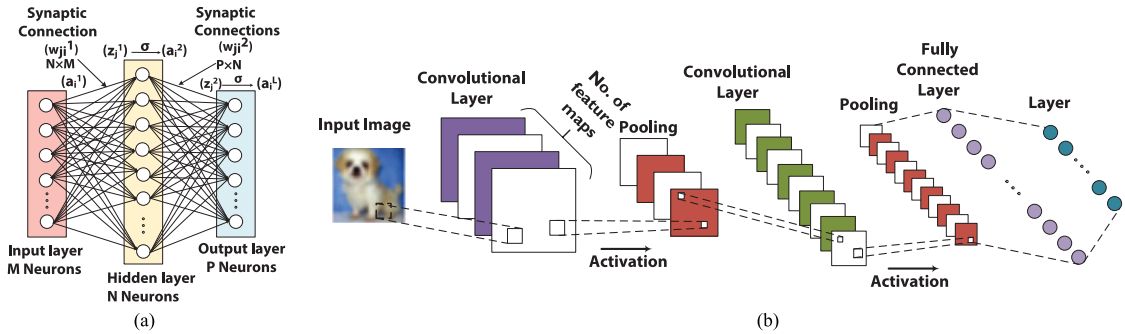

Fig. 1. (a) Fully connected 3-layered neural network showing the input layer, hidden layer, and an output layer. Each neuron in a particular layer is fed by weighted sum of all inputs of the previous layer and it performs a sigmoid operation on the sum to provide the inputs for the next layer. (b) CNN Architecture with different convolutional and pooling layers terminated by a fully connected layer.

In this work, we analyze the impact of non-idealities such as source resistance, neuron resistances, and synaptic weight variations in hardware implementations of neuromorphic crossbars. We show how such non-idealities can significantly degrade the accuracy when traditional training methodologies are employed. The presence of these parasitic elements also severely limits the crossbar sizes. As a solution, we propose an ex-situ technology aware training algorithm that mathematically models the aforementioned non-idealities and accounts for the same in the traditional backpropagation algorithm. Such a technique not only preserves the accuracy of an ideal network appreciably but also allows us to use larger crossbar sizes without significant accuracy degradation. The key highlights of our work are as follows:

1) We mathematically model the effect of source resistance, neuron resistance, and variations in synaptic conductance on the output currents of a neuromorphic crossbar. We establish the validity of our model by comparing against SPICE-like simulations of resistive networks.
2) We analyze the impact of these non-idealities on the accuracy of two types of image recognition tasks with varying amounts of non-ideality within relevant technological limits.
3) We propose a training algorithm which incorporates the mathematical models of the crossbar non-idealities and modifies the standard training algorithm in an effort to restore the ideal accuracy.

II. Crossbar Implementation of Neural Networks

A. Types of Network Topologies

1) Fully Connected Networks: Traditionally, deep neural networks such as deep belief nets (DBNs) comprise of multiple layers of interconnected units. Fully connected networks (FCN) involve a series of neuron layers between the input and the output layers. The output of each neuron in a layer is connected to the inputs of all the neurons in the subsequent layer. Fig. 1(a) shows a 3-layered fully connected network consisting of a single hidden layer between the input and output layers.

2) Convolutional Networks: Complex image recognition datasets comprise of objectively different classes where global weight mapping like FCNs prove to be less efficient. As an alternative, convolutional neural networks (CNN) have been recognized as a more powerful tool for complex image recognition problems using locally shared weights to learn common spatially local features. As shown in Fig. 1(b), CNNs consist of several layers performing operations like convolution, activation, and pooling, finally terminating with a fully connected layer. The convolution function can be mathematically represented by a 4 dimension tensor. Intuitively, a convolution layer is composed of a number of filter banks. The number of filter banks is equal to the number of output maps. Each output map represents a feature. A filter bank is made up of multiple kernels, one for each input map. Hence each filter bank operates on all the input maps to extract one output feature map. A kernel is mathematically represented as a $n \times n$ weight matrix. During convolution, the kernels of a filter bank are convolved with their respective input maps. The outputs of these convolutions are then summed together to form the corresponding output map of that filter bank. Thus a convolution operation captures the spatially local features of an input image. Convolution of a $m \times m$ input map with a kernel of size $n \times n$ yields an output map of size $((m-n+2p)/s+1) \times ((m-n+2p)/s+1)$, where s is the stride of the filter and p is the padding. In practice, s and p are chosen such that the original input size is preserved. The activation layer which can be RELU [24], sigmoid [25], or other non-linear functions, introduces a non-linearity in the network [26]. The pooling layer reduces the dimensionality of the output map. Most commonly used pooling techniques are average and max-pooling [27]. Finally, the fully connected layer uses the learned features to classify the images. In essence, a fully connected layer could also be represented by a convolutional layer where the kernel size is equal to the input size.

B. Hardware Representations of Neural Networks

In hardware realizations of neural networks based on the non-Von Neumann architecture framework, the synaptic connections between the neurons of two adjacent layers are represented using a resistive crossbar. The weights are represented in terms of conductance and the inputs are encoded as voltages. Convolutional layers have locally concentrated connections, hence each filter bank is represented by a crossbar of equivalent size. The

input to the crossbar is a subset of the image being sampled by the kernel. Each element of the output map is calculated through time multiplexing of the outputs from a particular crossbar for different subsets of the image. This is repeated for each filter bank to obtain different output maps. In contrast, fully connected layers have all possible connections between input and the output and the entire connection matrix can be represented by a crossbar. The basic computational function of any layer is a dot product and can be seamlessly performed by representing the weights as the resistances in a crossbar fashion. The output current of jth neuron of each crossbar is computed as

$$I_j = \sum V_i^{+} G_{ji}^{+} + V_i^{-} G_{ji}^{-} \tag{1}$$

where V_i is the input voltage corresponding to ith input neuron and G_{ji} represents the conductance corresponding to the synaptic weights between the neurons. Two resistive arrays are deployed to account for bipolar weights. The input to the positive array is $+V_i$ whereas the input to the negative array is $-V_i$. The weight matrix $[w_{ji}]$ is mapped to a corresponding conductance range $[G_{\text{low}}, G_{\text{high}}] \subset [G_{\text{on}}, G_{\text{off}}]$. To represent bipolar weights, the conductance of the synapse connecting the jth neuron in the next layer to the ith input is denoted by a positive (G_{ji}^{+}) component and a negative (G_{ji}^{-}) component. For positive (negative) weights, the programming is done such that G_{ji}^{+} (G_{ji}^{-}) $= |w_{ji}| G_{\text{high}}$ and G_{ji}^{-} (G_{ji}^{+}) $= 0$ (no connection). Fig. 2(a) shows a crossbar implementation of a fully connected neural network.

As mentioned earlier, crossbar arrays could suffer from nonideal effects and incur limitations on their sizes. As a result, larger crossbars are divided into smaller crossbars and the output of each crossbar is time-multiplexed to obtain the desired functionality of the entire crossbar. Fig. 2(b) shows how multiple small crossbars can be efficiently mapped to realize the functionality of a large crossbar in a particular layer. The small size of the crossbar reduces fan-out and fan-in, thus minimizing the impact of non-idealities. FCNs, being densely connected, are severely affected by hardware imperfections, especially when implemented on large crossbars. Convolutional layers in CNNs are usually implemented on very small crossbars and are thus insensitive to non-ideal effects. However, the final fully connected layers which acts as a classifier can be significantly affected by these non-idealities due to their large sizes. In this work, we are thus considering the impact of non-idealities on FCNs, and fully connected layers of CNNs.

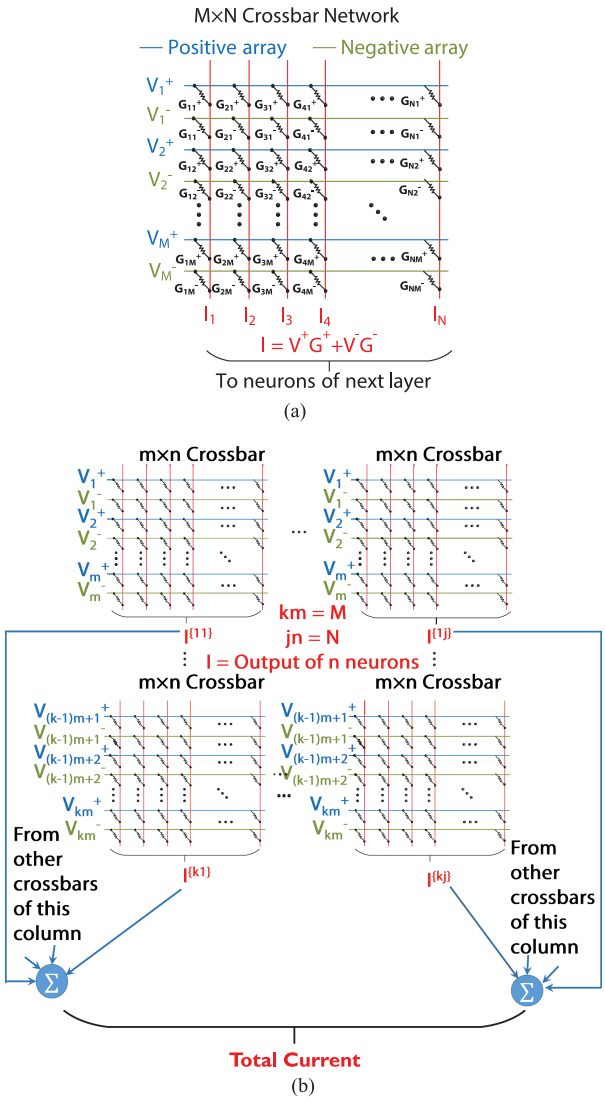

Fig. 2. (a) Hardware implementation of a single fully connected network layer represented by two resistive crossbar arrays. The output of the crossbar will be fed to another crossbar representing the next layer. (b) An arrangement of multiple sub-crossbars to realize the functionality of a large crossbar.

C. Training

The training of Artificial Neural Networks (ANN) are traditionally done off-chip through the standard backpropagation algorithm which updates weight matrices using gradient descent technique [28]. It is important to note down the vital aspects of the algorithm here in relevance to the later sections. The basic algorithm updates weights based on the gradients of a cost function. The cost function depends on the error computed from the feed-forward network which assumes a form : $C = \frac{1}{2} \sum (y_j - a_j)^2$, where y_j is the expected output and a_j is actual output from the jth neuron in the output layer. The sensitivity of the errors for each layer are calculated from the derivatives of the cost function with respect to the outputs and weights and after each iteration, the weights are updated based on those of the corresponding layer. The detailed description of the algorithm is well documented [28]. In this work, we focus on the aspects of the algorithm pertinent to fully connected layers and we build mathematical models to account for the non-idealities experienced by the hardware implementation of neuromorphic crossbars.

D. Technologies

Various technologies have been explored for crossbar implementations of neural networks. Memristive crossbars based on different material systems (like TaO_x [29], TiO_2 [30], Ag/Si [31] etc) have been proposed to realize neuromorphic functionality in an energy efficient manner. Phase change

TABLE I
RESISTANCE RANGES FOR VARIOUS TECHNOLOGIES

Technology	$[R_{on}, R_{off}]$	Considered Range $[R_{low}, R_{high}]$	$R_s/R_{high}(\%)$	$R_{neu}/R_{high}(\%)$
TiO_2 [35]	15k, 2M	40k,600k	0.033 - 0.13	0 - 0.033
Ag/Si [18]	25k,10M	100k,1.5M	0.013 - 0.053	0 - 0.013
TaOx [36]	1k,1M	20k,300k	0.067 - 0.27	0 - 0.067
Spintronics*	Function of MTJ oxide thickness [37]	40k, 400k	0.05 - 0.2	0-0.05
PCM [13]	10k, 3M	60k, 900k	0.022 - 0.08	0 - 0.022

*The spintronic analysis is done based on predictive measures of R_{off}/R_{on} [32]
R_s range - 200 to 800 Ω, R_{neu} range - 0 to 200 Ω

materials (PCM) [13] have also been investigated as potential candidates for neuromorphic computing due to their high scalability. More recently, neurons and synapses implemented with spintronic devices [10], [16] have shown great promise in performing ultra-low power neuromorphic computing. However, each technology suffers from specific drawbacks. An important metric in regard of resistive crossbars for neuromorphic systems is the ratio of the high resistance state (R_{off}) and the low resistance state (R_{on}) of the synaptic device. Usually, a high $R_{\text{off}}/R_{\text{on}}$ ratio is desired for a near-ideal implementation of the weights in a neuromorphic crossbar. Moreover, in the light of non-ideal systems, higher values of R_{on}, R_{off} may be less significantly impacted by parasitic resistances. In this work, we have chosen a maximum to minimum conductance ($G_{\text{low}} = \alpha/R_{\text{off}}$, $G_{\text{high}} = 15G_{\text{low}}$, α is a parameter of choice) ratio of 15 which is a potentially realizable predictive measure for all memory technologies [13], [32], [33].

Memristor based neuromorphic crossbar designs leverages its inherent capability of matrix multiplication to provide high accuracy at a relatively modest computational cost [34].

However, the memristor technology is still in its nascent stage. Thus, the hardware implementation of such crossbars may suffer various kinds of non-ideal effects arising from memristor device variations, parasitic resistances as well as non-idealities in sources, and sensing neurons.

III. MODELING THE NON-IDEALITIES

In this work, we have considered three kinds of non-idealities that arise in crossbar implementations, namely,

1) Neuron Resistance (R_{neu})
2) Source Resistance (R_s)
3) Memristive resistance variations

To perform an analysis of the impact non-idealities might have on accuracy of recognition task, it is important to note the ratio of non-ideal resistances to the synaptic resistances for a particular technology. Table I shows the range of considered resistance ratios to synaptic resistances R_s/R_{high} and $R_{\text{neu}}/R_{\text{high}}$ for various technologies, considering relevant values of source (R_s) and neuron (R_{neu}) resistances.

A. Neuron Resistance

The resistance offered by the neuron in a neuromorphic crossbar varies from technology to technology. In many cases, such as, PCM technology, the resistance of the neuron is not a hardware issue as the crossbar outputs are sensed through a sense amplifier, where virtual ground at the input eliminates the voltage drop across the neuron. However, in spintronic crossbars [10], crossbar outputs are fed to the neuron as a current stimulus and thus, the resistance of the neuronal device becomes relevant. Fig. 3 shows the effect of neuron resistance on the crossbar output. This can be mathematically modeled to modify (1) as:

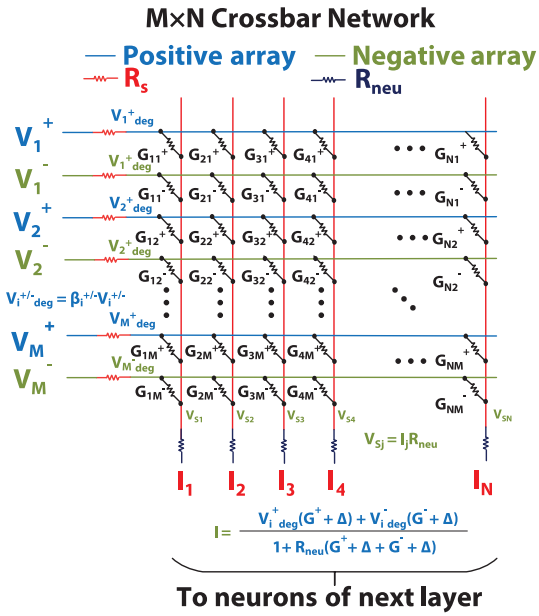

Fig. 3. Crossbar Architecture showing non-ideal elements like source and neuron resistances. The final output current equation is modified by the impact of these non-ideal elements.

$$I_j = \frac{\sum V_i^+ G_{ji}^+ + V_i^- G_{ji}^-}{1 + R_{\text{neu}} \sum G_{ji}^+ + G_{ji}^-} \tag{2}$$

Here, I_j, $V_i^{+/-}$, $G_{ji}^{+/-}$ and R_{neu} carry the same meaning as described in earlier sections. (1) can be derived by applying Kirchoff's law at the output nodes of the crossbar and considering the voltage drop across the neuron to be $V_{j,\text{neu}} = I_j \times R_{\text{neu}}$. It is evident that the denominator is close to 1 for smaller arrays as G_{ji} s are much smaller than neuron conductances (resistances of the order of a few hundred ohms [10]). However, larger arrays could lead to $G_{\text{neu}} = 1/R_{\text{neu}}$ being comparable to sum of the conductances in a particular column. More specifically, a higher number of rows in the crossbar lead to enhanced impact of neuron resistance.

B. Source Resistance

The source resistance (R_s) in a neuromorphic crossbar could arise due to non-ideal voltage sources and input access selectors lumped together. The input voltages to crossbar gets degraded due to R_s and the degradation can be mathematically modeled as:

$$V^{+}_{i,\mathrm{deg}} = V^{+}_{i} \frac{1/R_s}{1/R_s + \sum \frac{1}{R^{+}_{ij}+R_{\mathrm{neu}}}} \tag{3}$$

$$V^{-}_{i,\mathrm{deg}} = V^{-}_{i} \frac{1/R_s}{1/R_s + \sum \frac{1}{R^{-}_{ij}+R_{\mathrm{neu}}}} \tag{4}$$

Here $R^{+/-}_{ij}$ is the resistance of the synaptic element between the ith row and jth column in the positive or negative array. The model ignores the effect of sneak paths. In neuromorphic crossbars, all the inputs are simultaneously active. As the IR drops in the metal lines are negligible, all the nodes in a particular row are supplied by the degraded source voltage of that row. As all the rows are supplied by voltages of same polarity, even the shortest possible current sneak path will experience a low potential difference. Thus, the current through the series connection of the synaptic memristor and neuron would be primarily dependent on the degraded supply voltage and effective series resistance. We have verified the validity of the model by comparing against SPICE-like simulations, which is described in more detail in Section IV-B.

C. Memristive Conductance Variations

The weights obtained from the training algorithm are usually discretized in order to be represented as memristive synapses. In this work, we have used a 4-bit discretization technique where we have used a $R_{\mathrm{high}}/R_{\mathrm{low}}$ ratio of 15, relevant to the technologies considered. We have mapped the weights such that the maximum weight always maintains the $R_{\mathrm{high}}/R_{\mathrm{low}}$ ratio to the minimum weight. We have chosen the maximum and minimum weight limits so as to minimize the accuracy degradation due to discretization. To analyze the impact of chip-to-chip variation of weights, we have introduced weight variations in terms of standard deviation (σ) errors, ranging from -2σ to $+2\sigma$ after discretization. This implies that all the memristive devices on a neuromorphic chip suffer the same variation at a particular process corner. The weight variations are incorporated in the mathematical model as a Δ variation to the conductances.

D. Proposed Training Algorithm

The mathematical representations of the non-idealities are finally collated and incorporated in the feed-forward path and the backpropagation algorithm for training the ANN. Weights w_{ji} and inputs a_i replaces the conductances G_{ji} and voltages V_i respectively in (1) and (2). The symbol z_j is used to represent the current output of the crossbars I_j corresponding to jth neuron of the next layer. We assume that the neuronal function receives a current input and provides a voltage output. For the sake of simplicity, we assume ideal mathematical representations of activation functions like RELU [24] and sigmoid [25]. As described in Section II-A, the ideal crossbar output of the jth column in any layer is given by $z_j = \sum_i a_i \times w_{ji}$. The modified crossbar output can be computed as follows:

$$z^{l}_{j} = \frac{\sum a^{+}_{i,\mathrm{deg}} w^{+}_{ji,\mathrm{vary}} + a^{-}_{i,\mathrm{deg}} w^{-}_{ji,\mathrm{vary}}}{\gamma_j}$$
$$\gamma_j = 1 + R_{\mathrm{neu}} \sum_i w^{+}_{ji,\mathrm{vary}} + w^{-}_{ji,\mathrm{vary}} \tag{5}$$

where,

$$a^{+}_{i,\mathrm{deg}} = a_i \frac{1/R_s}{\beta^{+}_{i}}$$
$$a^{-}_{i,\mathrm{deg}} = -a_i \frac{1/R_s}{\beta^{-}_{i}}$$
$$w^{+/-}_{ij,\mathrm{vary}} = w^{+/-}_{ij} + \Delta$$
$$\beta^{+/-}_{i} = 1/R_s + \sum \frac{1}{R^{+/-}_{ij} + R_{\mathrm{neu}}}$$
$$R^{+/-}_{ij} = 1/w^{+/-}_{ij,\mathrm{vary}}$$

As described earlier, two weight matrices are deployed to account for bipolar weights in the original weight matrix $W = [w_{ji}]$. Positive (Negative) inputs are fed to the positive (negative) weight array. The weight matrices are created such that $w^{+}_{ji}\ (w^{-}_{ji}) = 0$ for all i,j for which $W_{ji} < 0 (> 0)$ and $w^{+}_{ji}\ (w^{-}_{ji}) = W_{ji}$ for all i,j for which $W_{ji} > 0 (< 0)$. Note that mapping the weights to a particular conductance range is equivalent to multiplication by a scaling factor as we have already discretized the weights based on a maximum to minimum weight ratio equal to $G_{\mathrm{high}}/G_{\mathrm{low}} = 15$. Thus an equivalent representation in terms of conductance would be $G^{+/-}_{ji} = W_{ji} G_{\mathrm{high}}$.

The output of each crossbar is passed as inputs to the next crossbar through a sigmoid function such that $a^{L+1}_{i} = \sigma(z^{L}_{i})$ (where L is the layer index). The backpropagation algorithm is modified to account for the modified crossbar functionality. As described earlier, learning in neural networks relies on computation of gradients of a cost function. Here, it is calculated from the error between the expected and the actual output of the output layer neurons in the form of $C = \frac{1}{2}\sum (y_j - a^{L}_{j})^2$. The delta-rule in the backpropagation algorithm [38] involves calculation of δ for each layer accounting for the change in the cost function for unit change in inputs to that particular layer. Thus, δ for layer l can be written as:

For output layer,

$$\delta^{L}_{j} = \frac{\partial C}{\partial z^{L}_{j}} = \sum \frac{\partial C}{\partial a^{L}_{j}} \frac{\partial a^{L}_{j}}{\partial z^{L}_{j}} = (a^{L}_{j} - y_j)\sigma'(a^{L}_{j}) \tag{6}$$

For other layers,

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_k \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_k \delta_k^{l+1} \frac{\partial z_k^{l+1}}{\partial z_j^l}$$

$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = \frac{\partial z_k^{l+1}}{\partial a_j^l} \frac{\partial a_j^l}{\partial z_j^l} = \frac{\partial z_k^{l+1}}{\partial a_j^l} \sigma'(a_j^l) \tag{7}$$

$$\frac{\partial z_k^{l+1}}{\partial a_j^l} = \frac{\frac{a_{j,\deg}^{+,l}}{a_j^l} w_{jk,\text{vary}}^{+} - \frac{a_{j,\deg}^{-,l}}{a_j^l} w_{jk,\text{vary}}^{-}}{\gamma_j} \tag{8}$$

Finally, the δ s of each layer are used to compute the weight updates as:

$$dw_{jk}^l = \frac{\partial C}{\partial w_{jk}^l} = \frac{\partial C}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_{jk}^l} = \delta_j^l \frac{\partial z_j^l}{\partial w_{jk}^l} \tag{9}$$

$$\frac{\partial z_j^l}{\partial w_{jk}^l} = \frac{\gamma_j (a_{k,\deg}^{+} (1 - \frac{w_{kj}^{+}}{\beta_k^{+}}) + a_{k,\deg}^{-} (1 - \frac{w_{kj}^{-}}{\beta_k^{-}})) - R_{\text{neu}} z_j^l \gamma_j}{\gamma_j^2} \tag{10}$$

To simulate the impact of non-idealities on varying crossbar size, we divide the large crossbars of size $M \times N$ into several smaller crossbars of size $m \times n$. Fig. 1(c) shows the network architecture of combining smaller crossbars to realize the neuromorphic functionality of larger crossbars. The source degradation factor β_i is more prominent for larger number of columns as it depends on the term $\sum_j 1/(R_{ij} + R_{\text{neu}})$ summed over the columns. The neuron resistance degradation factor γ_j, on the other hand, increases with the number of rows due to its dependence on the term $\sum_i w_{ji}$, summed over the rows. Thus, the combined effect of these two non-idealities is expected to have a higher impact on the network for larger crossbars.

IV. Simulation Framework

A. Model Simulations

The model described in the previous section was implemented on FCNs using the MATLAB Deep Learning Toolbox [39] and CNNs using MatConvNet [40].

1) FCN: A 3-layered neural network was employed to recognize digits from the MNIST Dataset. The training set consists of 60000 images, while the testing set consists of 10000 images. The input layer consists of 784 neurons designated to carry the information of each pixel of each 28 × 28 image. The hidden layer consists of 500 neurons and the output layer has 10 neurons to recognize 10 digits. The neuron transfer function was chosen to be the sigmoid function which can be written as $\sigma(x) = \frac{1}{1+e^{-x}}$.

2) CNN: For the classification of more complex dataset CIFAR-10, we have used a network with RELU-activated convolutional layers and a sigmoid-activated fully connected layer. The architecture is represented as 32 × 32 × 3-64c5-2s-128c5-2s-256c3-2s-512o-10o.The details of the layers are provided in Table II. Essentially, the different layers represent subsequent operations such as convolution, max-pooling or average-pooling and activation. The operations are described in detail in Secton II-A. Each convolutional layer is followed by a batch-normalization layer for better performance. We concentrate our analysis on the fully connected layers of the network as the initial convolutional layers possess local connections implemented on small crossbars equal to the kernel sizes.

TABLE II
CNN Architecture

Input 32 × 32 RGB image
5 × 5 conv. 64 RELU 2 × 2 max-pooling stride 2
5 × 5 conv. 128 RELU 2 × 2 max-pooling stride2
3 × 3 conv. 256 RELU 2 × 2 avg-pooling stride 2
4 × 4 conv. 512 Sigmoid (fully connected) 0.5 Dropout
1 × 1 conv. 10 (fully connected)
10-way softmax

B. SPICE-Like Simulations for Validation

Each fully connected layer for both FCNs and CNNs can be implemented in a crossbar architecture comprising of all possible connections. A SPICE-like framework was implemented in MATLAB by creating a netlist of all connections, voltage source, source and neuron resistances in such resistive crossbars and evaluating the voltages at each node by solving the conductance matrix: $[V] = [G]^{-1}[I]$. The framework was benchmarked with HSPICE. This framework was used to calculate the output of non-ideal crossbars on application of the inputs from the MNIST dataset as voltages. The resistances of the crossbar elements R_{ji} were determined such that $R_{ji} = 1/w_{ji}$, where w_{ji} are the weights determined by the ideal training scheme described in the previous section. The output obtained by showing 100 images of the testing set was averaged and the distribution was compared with the mathematical model simulations. Fig. 4(a) shows the comparison in the distribution of output currents of a crossbar where the approximate model shows good agreement with the exact SPICE-like simulations. Fig. 4(b) shows that the normalized root mean square deviation (NRMSD) between the two techniques for various $(R_s + R_{\text{neu}})/R_{\text{high}}$ combinations remains very close to zero for relevant values. As SPICE simulations automatically takes account of possible sneak paths, the agreement of our model to SPICE simulations means that the effect of sneak paths, even if not absolutely zero, is insignificant. Thus, the dominant issues in the crossbar to be considered are source and neuron resistances. It is to be noted that the validation of our approximate model was important in the context of reducing the time required for simulating the training and inferencing of each network for the entire dataset as the matrix operations could be more efficiently performed using the mathematical model. This eliminated simulating the network for each input image in HSPICE and the subsequent iterative steps involving MATLAB-SPICE interfacing.

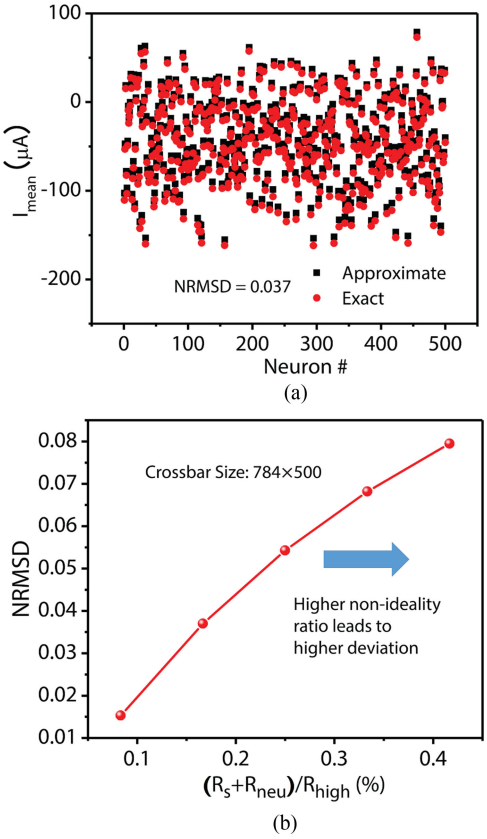

Fig. 4. (a) Distribution of output currents (I_{mean}), averaged over 100 images, across 500 neurons in the hidden layer comparing the approximate model to SPICE-like simulation framework. (b) Variation of Normalized Root Mean Square Deviation (NRMSD) with non-ideality ratio. NRMSD is close to zero for the relevant range of non-idealities.

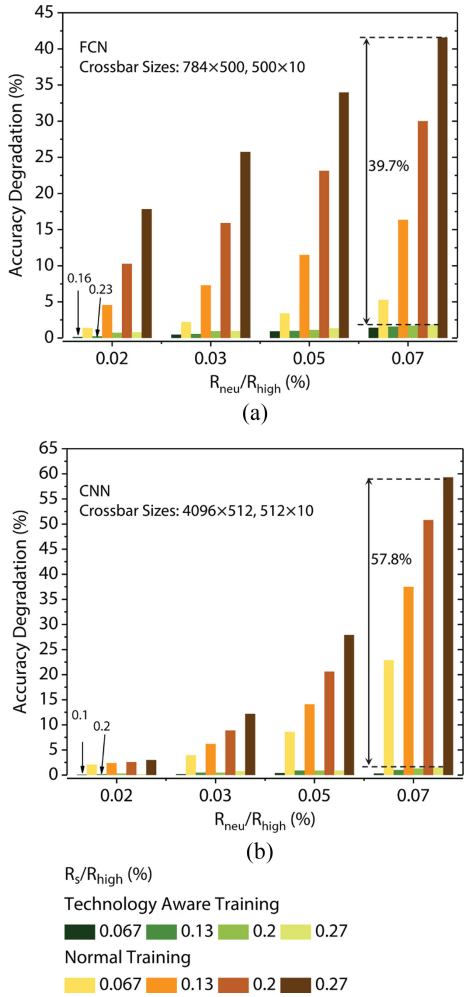

Fig. 5. Accuracy degradation v/s varying $R_{\text{neu}}/R_{\text{high}}$ ratio for different R_s/R_{high} combinations comparing technology aware training scheme with normal training for (a) FCN and (b) CNN.

V. Results and Discussion

We analyzed the impact of technological constraints in crossbar implementations on both FCNs and CNNs. As fully connected layers form the crux of classification in both network topologies, it is expected that such non-ideal conditions will have similar detrimental effects on both. We present the detailed impact of each non-ideality on FCNs and CNNs for better understanding.

We consider a 3-layered FCN and a CNN architecture described in Table II to analyze the impact of the non-idealities on the accuracy of recognition task on MNIST and CIFAR-10 datasets respectively. The other convolutional layers in the CNN are usually implemented using small crossbars and hence do not suffer significant effects of non-ideal resistances.

First, the neural networks were trained under ideal conditions using the training set. Then, the non-ideal model was included in the feed-forward path and the ideally trained network was tested using the testing set to determine the performance degradation due to the non-idealities. Next, the technology aware training algorithm was implemented by incorporating the mathematical formulation of the non-idealities in the standard training iterations of feed-forward and backpropagation as described in the Section III-D. For each iteration, the weights were discretized as described in Section III-C. The testing accuracy of an ideally trained FCN with a sigmoid neuronal function was 98.12% on MNIST and that of an ideally trained CNN was 85.6% on CIFAR-10 datasets. The accuracy degradations discussed in this section has been calculated with respect to these ideal testing accuracies such that Accuracy Degradation (%) = Ideal Accuracy (%) − Accuracy Obtained (%).

We use the parameters R_s/R_{high} and $R_{\text{neu}}/R_{\text{high}}$ to denote the ratios of the non-ideal resistances and the maximum synaptic resistance.

1) Source and Neuron Resistance: Fig. 5(a) and (b) shows the accuracy degradation for different $R_{\text{neu}}/R_{\text{high}}$ and R_s/R_{high} combinations in FCN and CNN, respectively. The effect of the non-ideal resistances on the performance of the network predictably worsens monotonically with higher R_s/R_{high} and $R_{\text{neu}}/R_{\text{high}}$ ratios. It can be observed that with normal training methods, the non-ideal resistances result in accuracy degradation for FCN: up to 41.58% for $R_{\text{neu}}/R_{\text{high}} = 0.07\%$ and $R_s/R_{\text{high}} = 0.27\%$. Our proposed training scheme incorporates the impact of non-idealities and achieves significant restoration

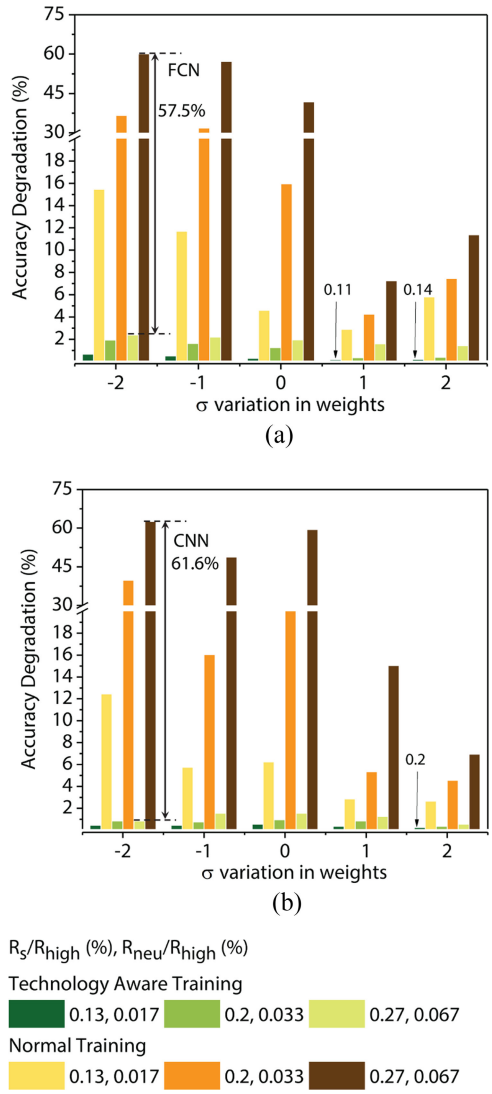

Fig. 6. Accuracy degradation v/s σ variations in weights for various R_s/R_{high} and $R_{\text{neu}}/R_{\text{high}}$ combinations comparing the technology aware training scheme with normal training for (a) FCN and (b) CNN.

of accuracy, within 1.9% of the ideal accuracy, for the worst case combination of resistances considered, shown in Fig. 5(a).

In case of CNNs, we show that due to the large crossbar sizes of the fully connected layers in the CNN, it can suffer up to 59.3% degradation in accuracy for the worst case non-ideal resistances considered. Our proposed algorithm, on the other hand, achieves an accuracy within 1.5% of the ideal accuracy [Fig. 5(b)], considering the largest crossbar sizes for the architecture.

2) Weight Variations: On-chip crossbar implementations suffer from chip-to-chip device variations. To account for such variations, we form a defect weight matrix, and include it in the feed-forward network, as described in detail in Section III-C. We have considered up to $\pm 2\sigma$ variation in the synaptic weights. Fig. 6 shows the impact of such device variations on the accuracy of FCN and CNN for different combinations of R_s/R_{high} and $R_{\text{neu}}/R_{\text{high}}$. Predictably, changes in the positive direction reduces the accuracy degradation from the nominal (no variation) case as it enhances the significance of the neurons. However, changes in the negative direction slightly degrades the accuracy from the nominal case. It is observed that a -2σ variation can result in an accuracy degradation of up to 59.9% for $R_{\text{neu}}/R_{\text{high}} = 0.067\%$ and $R_s/R_{\text{high}} = 0.27\%$ in FCN. By accounting for these variations in the backpropagation algorithm, our proposed training methodology successfully restores the accuracy within 2.34% of the ideal accuracy for worst case of non-idealities considered, as shown in Fig. 6(a).

Weight variations in the negative direction also adversely affect CNNs where -2σ variation can result in an accuracy degradation of 62.4% considering the non-ideal resistances mentioned above. Our proposed algorithm achieves an accuracy within 0.8% of the ideal testing accuracy as shown in Fig. 6(b).

3) Crossbar Size: Non-idealities in crossbars usually establish restrictions on the allowable crossbar sizes due to the dependence of their performance on fan-in and fan-out. For example, the impact of R_s on the crossbar depends on the parallel combination of column resistances and a higher number of columns (and hence, higher fan-out) result in severe performance degradation. Also, the impact of R_{neu} intensifies with increasing number of rows in the crossbar as it leads to more fan-in. As observed in Fig. 7(a), the combined effect of these resistances and variations can result in significant accuracy degradation (41.58%) when the network is implemented on crossbars of sizes 784×500 and 500×10 for the respective layers in the FCN. Under the same non-ideal conditions, accuracy degradation drops to 1.2% when smaller crossbars of sizes 112×100 and 100×10 are used to represent the functionality of the network. In contrast, considering the same R_s and R_{neu}, our proposed training algorithm achieves an accuracy degradation within $\sim$1.89% for sizes $784 \times 500, 500 \times 10$ and $\sim$0.3% for sizes $112 \times 100, 100 \times 10$. Thus, the proposed algorithm ensures that a network implemented on larger crossbars can parallel the performance of ideally trained networks implemented on smaller crossbars with minimal degradation.

The convolutional layers in CNNs are implemented on smaller crossbars. For the fully connected layers in the CNN architecture, we have considered significantly larger crossbars of sizes 4096×512 and 512×10. Due to large sizes of the last 2 layers of the considered architecture, we show in Fig. 7(b), that the network, when trained under ideal conditions, can suffer as large as 59.3% degradation in accuracy for the worst case resistance constraints considered. On the other hand, using smaller crossbars of sizes $512 \times 64, 64 \times 10$ reduces the accuracy degradation to 2.4% for the same conditions. In comparison, a network trained with the proposed technology aware training algorithm restores the accuracy to within $\sim$1.5% of the ideal accuracy even for the highest crossbar sizes ($4096 \times 512, 512 \times 10$). Thus, the proposed algorithm ensures that a CNN with fully connected layers implemented on crossbars of size in the order of 4096×512 can achieve better performance than for crossbars of size 512×64 with standard training algorithms. Such a provision of using large crossbars for implementing neuromorphic systems could potentially reduce overheads of repeating inputs, time multiplexing outputs, thus ensuring faster operations.

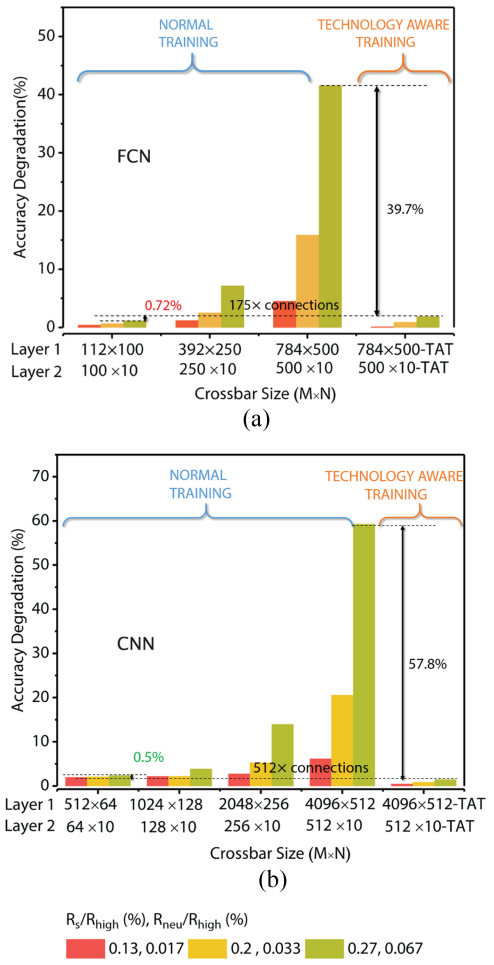

Fig. 7. Accuracy degradation v/s crossbar size for various R_s/R_{high} and $R_{\text{neu}}/R_{\text{high}}$ combinations comparing the technology aware training scheme with normal training for (a) FCN and (b) CNN. Larger crossbars show higher accuracy degradation.

VI. Conclusion

Hardware implementations of neuromorphic systems in crossbar architecture could suffer from various non-idealities resulting in severe performance degradation when employed in machine learning applications such as recognition tasks, natural language processing, etc. In this work, we analyzed, by means of mathematical modeling, the impact of non-idealities such as source resistance, neuron resistance and chip-to-chip device variations on performance of a 3-layered FCN on MNIST and a state-of-the-art CNN architecture on CIFAR-10. Severe degradation in recognition accuracy, up to 59.84%, was observed in FCNs. Although convolution layers in CNN can be implemented on smaller crossbars, the large fully connected layers at the end made them prone to performance degradation (up to 62.4% for our example). As a solution, we proposed a technology aware training algorithm which incorporates the mathematical models of the non-idealities in the training algorithm. Considering relevant ranges of non-idealities, our proposed methodology recovered the performance of the network implemented on non-ideal crossbars to within 2.34% of the ideal accuracy for FCNs and 1.5% for CNNs. We further show that the proposed technology aware training algorithm enables the use of larger crossbars of sizes in the order of 4096×512 for CNNs and 784×500 for FCNs without significant performance degradation. Thus, we believe that the proposed work potentially paves the way for implementation of neuromorphic systems on large crossbars which otherwise is rendered unfeasible using standard training algorithms.

References

[1] W. A. Wulf and S. A. McKee, "Hitting the memory wall," *ACM SIGARCH Comput. Arch. News*, vol. 23, no. 1, pp. 20–24, Mar. 1995. [Online]. Available: https://doi.org/10.1145%2F216585.216588

[2] R. Serrano-Gotarredona *et al.*, "CAVIAR: A 45 k neuron, 5 m synapse, 12 g connects/s AER hardware sensory-processing-learning-actuating system for high-speed visual object recognition and tracking," *IEEE Trans. Neural Netw.*, vol. 20, no. 9, pp. 1417–1438, Sep. 2009. [Online]. Available: https://doi.org/10.1109%2Ftnn.2009.2023653

[3] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, and D. S. Modha, "A digital neurosynaptic core using embedded crossbar memory with 45 pj per spike in 45 nm," in *Proc. 2011 IEEE Custom Integr. Circuits Conf.*, Sep. 2011. [Online]. Available: https://doi.org/10.1109%2Fcicc.2011.6055294

[4] X. Jin, M. Lujan, L. A. Plana, S. Davies, S. Temple, and S. B. Furber, "Modeling spiking neural networks on SpiNNaker," *Comput. Sci. Eng.*, vol. 12, no. 5, pp. 91–97, Sep. 2010. [Online]. Available: https://doi.org/10.1109%2Fmcse.2010.112

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

[6] H. Shiga *et al.*, "A 1.6 gb/s ddr2 128 mb chain feram with scalable octal bitline and sensing schemes," *IEEE J. Solid-State Circuits*, vol. 45, no. 1, pp. 142–152, Jan. 2010.

[7] K. Osada *et al.*, "Phase change ram operated with 1.5-v CMOS as low cost embedded memory," in *Proc. IEEE 2005 Custom Int. Circuits Conf.*, 2005, pp. 431–434.

[8] S.-S. Sheu *et al.*, "A 4 mb embedded SLC resistive-ram macro with 7.2 ns read-write random-access time and 160 ns mlc-access capability," in *Proc. 2011 IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2011, pp. 200–202.

[9] S. Chung *et al.*, "Fully integrated 54 nm STT-RAM with the smallest bit cell dimension for high density memory application," in *Proc. 2010 IEEE Int. Electron Devices Meeting*, 2010, pp. 12.7.1–12.7.4.

[10] A. Sengupta, Y. Shim, and K. Roy, "Proposal for an all-spin artificial neural network: Emulating neural and synaptic functionalities through domain wall motion in ferromagnets," *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 6, pp. 1152–1160, Dec. 2016. [Online]. Available: https://doi.org/10.1109%2Ftbcas.2016.2525823

[11] 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, no. 7550, pp. 61–64, May 2015. [Online]. Available: https://doi.org/10.1038%2Fnature14441

[12] C. Liu *et al.*, "A memristor crossbar based computing engine optimized for high speed and accuracy," in *Proc. 2016 IEEE Comput. Soc. Annu. Symp. VLSI*, Jul. 2016. [Online]. Available: https://doi.org/10.1109%2Fisvlsi.2016.46

[13] S. B. Eryilmaz *et al.*, "Brain-like associative learning using a nanoscale non-volatile phase change synaptic device array," *Front. Neurosci.*, vol. 8, Jul. 2014. [Online]. Available: https://doi.org/10.3389%2Ffnins.2014.00205

[14] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015. [Online]. Available: https://doi.org/10.1016%2Fj.neunet.2014.09.003

[15] P. Gu *et al.*, "Technological exploration of RRAM crossbar array for matrix-vector multiplication," in *Proc. 20th Asia South Pac. Des. Autom. Conf*, Jan. 2015. [Online]. Available: https://doi.org/10.1109%2Faspdac.2015.7058989

[16] A. Sengupta and K. Roy, "A vision for all-spin neural networks: A device to system perspective," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 63, no. 12, pp. 2267–2277, Dec. 2016. [Online]. Available: https://doi.org/10.1109%2Ftcsi.2016.2615312

[17] S. Yu, X. Guan, and H.-S. P. Wong, "On the stochastic nature of resistive switching in metal oxide RRAM: Physical modeling, monte carlo simulation, and experimental characterization," in *Proc. 2011 IEEE Int. Electron Devices Meeting*, Dec. 2011. [Online]. Available: https://doi.org/10.1109%2Fiedm.2011.6131572
[18] K.-H. Kim *et al.*, "A functional hybrid memristor crossbar-array/CMOS system for data storage and neuromorphic applications," *Nano Lett.*, vol. 12, no. 1, pp. 389–395, Jan. 2012. [Online]. Available: https://doi.org/10.1021%2Fnl203687n
[19] S. Kannan, N. Karimi, R. Karri, and O. Sinanoglu, "Detection, diagnosis, and repair of faults in memristor-based memories," in *Proc. 2014 IEEE 32nd VLSI Test Symp.*, Apr. 2014. [Online]. Available: https://doi.org/10.1109%2Fvts.2014.6818762
[20] P.-Y. Chen *et al.*, "Technology-design co-optimization of resistive cross-point array for accelerating learning algorithms on chip," in *Proc. IEEE 2015 Conf. Des., Autom. Test Eur. Conf. Exhib. (DATE)*, 2015. [Online]. Available: https://doi.org/10.7873%2Fdate.2015.0620
[21] B. Liu *et al.*, "Reduction and IR-drop compensations techniques for reliable neuromorphic computing systems," in *Proc. 2014 IEEE/ACM Int. Conf. Comput. Aided Des.*, Nov. 2014. [Online]. Available: https://doi.org/10.1109%2Ficcad.2014.7001330
[22] P.-Y. Chen *et al.*, "Mitigating effects of non-ideal synaptic device characteristics for on-chip learning," in *Proc. 2015 IEEE/ACM Int. Conf. Comput. Aided Des.*, Nov. 2015. [Online]. Available: https://doi.org/10.1109%2Ficcad.2015.7372570
[23] C. Liu, M. Hu, J. P. Strachan, and H. H. Li, "Rescuing memristor-based neuromorphic design with high defects," in *Proc. 54th Annu. Des. Autom. Conf. 2017*, 2017. [Online]. Available: https://doi.org/10.1145%2F3061639.3062310
[24] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 807–814.
[25] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proc. Nat. Acad. Sci.*, vol. 81, no. 10, pp. 3088–3092, 1984.
[26] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
[27] K. Jarrett *et al.*, "What is the best multi-stage architecture for object recognition?" in *Proc. 2009 IEEE 12th Int. Conf. Comput. Vis.*,, 2009, pp. 2146–2153.
[28] D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representations by error propagation," in *Readings in Cognitive Science*. New York, NY, USA: Elsevier, 1988, pp. 399–421. [Online]. Available: https://doi.org/10.1016%2Fb978-1-4832-1446-7.50035-2
[29] I.-T. Wang, Y.-C. Lin, Y.-F. Wang, C.-W. Hsu, and T.-H. Hou, "3d synaptic architecture with ultralow sub-10 fJ energy per spike for neuromorphic computation," in *Proc. 2014 IEEE Int. Electron Devices Meeting*, Dec. 2014. [Online]. Available: https://doi.org/10.1109%2Fiedm.2014.7047127
[30] F. Alibart, E. Zamanidoost, and D. B. Strukov, "Pattern classification by memristive crossbar circuits using ex situ and in situ training," *Nature Commun.*, vol. 4, Jun. 2013. [Online]. Available: https://doi.org/10.1038%2Fncomms3072
[31] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, "Nanoscale memristor device as synapse in neuromorphic systems," *Nano Lett.*, vol. 10, no. 4, pp. 1297–1301, Apr. 2010. [Online]. Available: https://doi.org/10.1021%2Fnl904092h
[32] A. Hirohata *et al.*, "Roadmap for emerging materials for spintronic device applications," *IEEE Trans. Magn.*, vol. 51, no. 10, pp. 1–11, Oct. 2015. [Online]. Available: https://doi.org/10.1109%2Ftmag.2015.2457393
[33] J. Borghetti, G. S. Snider, P. J. Kuekes, J. J. Yang, D. R. Stewart, and R. S. Williams, "'memristive' switches enable 'stateful' logic operations via material implication," *Nature*, vol. 464, no. 7290, pp. 873–876, Apr. 2010. [Online]. Available: https://doi.org/10.1038%2Fnature08940
[34] M. Hu *et al.*, "Dot-product engine for neuromorphic computing: Programming 1t1m crossbar to accelerate matrix-vector multiplication," in *Proc. 2016 53nd ACM/EDAC/IEEE Des. Autom. Conf.*, 2016, pp. 1–6.
[35] R. Berdan, E. Vasilaki, A. Khiat, G. Indiveri, A. Serb, and T. Prodromakis, "Emulating short-term synaptic dynamics with memristive devices," *Sci. Rep.*, vol. 6, no. 1, Jan. 2016. [Online]. Available: https://doi.org/10.1038%2Fsrep18639
[36] K. M. Kim *et al.*, "Voltage divider effect for the improvement of variability and endurance of TaOx memristor," *Sci. Rep.*, vol. 6, no. 1, Feb. 2016. [Online]. Available: https://doi.org/10.1038%2Fsrep20085
[37] X. Fong, S. K. Gupta, N. N. Mojumder, S. H. Choday, C. Augustine, and K. Roy, "KNACK: A hybrid spin-charge mixed-mode simulator for evaluating different genres of spin-transfer torque MRAM bit-cells," in *Proc. 2011 Int. Conf. Simul. Semicond. Processes Devices*, Sep. 2011. [Online]. Available: https://doi.org/10.1109%2Fsispad.2011.6035047
[38] R. Hecht-Nielsen *et al.*, "Theory of the backpropagation neural network," *Neural Netw.*, vol. 1, no. Suppl. 1, pp. 445–448, 1988.
[39] R. B. Palm, "Prediction as a candidate for learning deep hierarchical models of data," Master's thesis, Dept. Inform. Math. Modeling, Tech. Univ. Denmark, Lyngby, Denmark, 2012.
[40] A. Vedaldi and K. Lenc, "MatConvNet," in *Proc. 23rd 2015 ACM Int. Conf. Multimedia*, 2015. [Online]. Available: https://doi.org/10.1145%2F2733373.2807412

Indranil Chakraborty received the B.Engg. degree in electronics and telecommunication engineering from Jadavpur University, Kolkata, India, in 2013, and the M.Tech. degree in electrical engineering from Indian Institute of Technology Bombay, Mumbai, India, in 2016. Since Fall 2016, he has been working toward the Ph.D. degree with the Nanoelectronics Research Laboratory, Purdue University, West Lafayette, IN, USA. His primary research interests include hardware for neuromorphic computing using CMOS and post-CMOS technologies. He was the recipient of best M.Tech thesis award and academic excellence award during his time at IIT Bombay for his academic performance.

Deboleena Roy received the B.Tech and M.Tech Dual degree in electronics and electrical communications engineering from Indian Institute of Technology Kharagpur, Kharagpur, India, in 2014. She is currently working toward the Ph.D. degree in electrical and computer engineering at Purdue University, West Lafayette, IN, USA. Prior to that, she was a Design Engineer with Qualcomm Bangalore Design Center, Bengaluru, India, from 2014 to 2016. Her current research interests include neuro-inspired algorithms for cognitive applications such as perception, reasoning and decision making.

Kaushik Roy received the B.Tech. degree in electronics and electrical communications engineering from Indian Institute of Technology Kharagpur, Kharagpur, India, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Illinois at Urbana− Champaign, Champaign, IL, USA, in 1990. He was with the Semiconductor Process and Design Center, Texas Instruments Incorporated, Dallas, TX, USA, where he was involved in FPGA architecture development and low-power circuit design. In 1993, he was with the Faculty of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, where he is currently Edward G. Tiedemann Jr. Distinguished Professor. He has authored more than 600 papers in refereed journals and conferences, holds 15 patents, graduated 75 Ph.D. students, and has coauthored two books on *Low Power CMOS VLSI Design* (Wiley and McGraw Hill). His current research interests include neuromorphic and cognitive computing, spintronics, device-circuit codesign for nanoscale Silicon and non-Silicon technologies, low-power electronics for portable computing and wireless communications, and new computing models enabled by emerging technologies.