The Impact of Device Uniformity on Functionality of Analog Passively-Integrated Memristive Circuits

Z. Fahimi, M. R. Mahmoodi, M. Klachko, H. Nili, and D. B. Strukov, *Senior Member, IEEE*

***Abstract*— Passively-integrated memristors are the most prospective candidates for designing high-speed, energy-efficient, and compact neuromorphic circuits. Despite all the promising properties, experimental demonstrations of passive memristive crossbars have been limited to circuits with few thousands of devices until now, which stems from the strict uniformity requirements on the *IV* characteristics of memristors. This paper expands upon this vital challenge and investigates how uniformity impacts the computing accuracy of analog memristive circuits, focusing on neuromorphic applications. Specifically, the paper explores the tradeoffs between computing accuracy, crossbar size, switching threshold variations, and target precision. All-embracing simulations of matrix multipliers and deep neural networks on CIFAR-10 and ImageNet datasets have been carried out to evaluate the role of uniformity on the accuracy of computing systems. Further, we study three post-fabrication methods that increase the accuracy of nonuniform 0T1R neuromorphic circuits: hardware-aware training, improved tuning algorithm, and switching threshold modification. The application of these techniques allows us to implement advanced deep neural networks with almost no accuracy drop, using current state-of-the-art analog 0T1R technology.**

***Index Terms*— Memristor, neuromorphic computing, VMM, memristive circuits, ReRAM.**

I. Introduction

THE cognitive capabilities of the human brain have served as an inspiration for the development of artificial neural networks (ANNs). Despite the fact that ANNs have surpassed humans in terms of prediction accuracy in few applications, e.g., image classification, they are still far inferior in terms of energy efficiency. While offering far more cognition capabilities, the visual cortex consumes several orders of magnitude less energy than state-of-the-art ANN systems. Hence, further progress in the field of neural computation hinges on the use of more efficient hardware as the need for energy and area-efficient neural networks is as great as ever [1]–[3].

Manuscript received April 21, 2021; revised June 11, 2021; accepted June 28, 2021. Date of publication July 26, 2021; date of current version September 30, 2021. This work was supported in part by Semiconductor Research Corporation (SRC) through the Joint University Microelectronics Program (JUMP) Center for Research on Intelligent Storage and Processing-in-memory (CRISP) and the National Science Foundation (NSF)/SRC Energy-Efficient Computing: from Devices to Architectures (E2CDA) under Grant 1740352. This article was recommended by Associate Editor A. James. *(Z. Fahimi and M. R. Mahmoodi contributed equally to this work.) (Corresponding authors: M. R. Mahmoodi; Z. Fahimi.)*

The authors are with the Department of Electrical and Computer Engineering (ECE), University of California at Santa Barbara, Santa Barbara, CA 93106 USA (e-mail: zfahimi@ucsb.edu; mrmahmoodi@ucsb.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2021.3097282.

Digital Object Identifier 10.1109/TCSI.2021.3097282

A key challenge towards the development of efficient large-scale neuromorphic hardware is the lack of a suitable electronic device mimicking synapse functionality. Such functionality is essential to compute a dot-product, the most common operation in inference or training of ANNs. In this context, a very promising approach for neuromorphic inference applications is to employ circuits with memristors. A memristor is a nanodevice with adjustable conductance G - essentially an analog nonvolatile memory cell - that enables efficient implementation of the dot-product operation in the analog domain. Indeed, the most appealing aspect of memristor technology is its scalability prospects. The conductance modulation in filamentary metal-oxide memristors is attributed to the reversible modulation of the concentration of oxygen vacancies. The atomic scale of the vacancy position modulation implies the feasibility of downscaling memristors to sub-deca nanometers [4], [5]. The density of a device could be as small as $4F^2$, limited by the half-pitch metal size F. Emulating these adjustable devices with purely CMOS circuits requires orders of magnitude larger footprint (e.g., the next compact option, redesigned analog-grade floating-gate memories [6], consume $\sim$100 F^2/cell). However, the progress for implementing larger-scale memristive circuits, especially their most dense passively-integrated (0T1R) variety, that are required for practically useful neuromorphic hardware, faces several challenges.

One critical challenge is the presence of large device-to-device variations [7]. The stochastic nature of oxide rupture in such small scales complicates the reproducibility of device parameters, e.g., the voltage required for electroforming and switching. Such variabilities are the very reason for the limited demonstrations of memristive neuromorphic networks so far. One solution to alleviate this issue is the usage of selector transistors (1T1R memories); however, such an approach is in conflict with the main driving force of this technology (i.e., scalability and three-dimensional integration compatibility [5], [8]).

The recent work [9] showed progress in addressing this challenge and demonstrated the successful integration of a 64×64 passive metal-oxide memristor crossbar circuit. This technology features analog-grade memories with $\sim$99% device yield based on a foundry-compatible fabrication process with etch-down patterning and a low-temperature budget, conducive to vertical integration. The crossbar also features excellent analog properties such as long retention and high endurance characteristics. The cell size is $10^4\times$ denser at a similar yield, and the average conductance is $10\times$ less than state-of-the-art

1T1R technology [3], [10]. Besides, the reported uniformity is sufficient for <5% average tuning precision that is slightly worse than ∼3% reported in analog 1T1R memories [10].

Despite the vital importance of uniformity in 0T1R memristor crossbars, it has not been thoroughly investigated in the context of neuromorphic computing to the best of our knowledge. The key open questions include: How does the crossbar uniformity impact the computing accuracy of memristive crossbars? From this perspective, what are the critical factors that affect computing accuracy? How can we improve the performance? How much crossbar uniformity is needed to achieve software-equivalent accuracy and build a large-scale deep neural network? This paper aims to expand upon these important questions and the critical role of switching threshold variations in the computing precision of neuromorphic networks. First, we discuss the preliminaries, motivation, and previously fabricated analog-grade memristor crossbars. Then, a dynamic model and a simulation framework are developed based on experimental data from the fabricated crossbar. Further, extensive simulations of vector-by-matrix multipliers (VMMs) and representative neuromorphic networks are performed to assess the tradeoffs and trends. Finally, three post-fabrication solutions are explored for improving the performance of neuromorphic circuits. The paper is concluded with a thorough discussion of the results and prospects of harnessing 0T1R and 1T1R circuits in neuromorphic circuits.

II. Analog Passive Memristive Crossbar Circuits

A. Basic Structure and Operation

Fig. 1a shows the scanning electron microscope image of our latest fabricated 64 × 64 memristive crossbar [9]. The inset shows the zoomed-in view to a portion of the crossbar, showing top electrodes passing on top of the bottom electrodes. A memristive device is formed at the intersection of each top and bottom electrodes. Such an array of conductance-adjustable devices could be used to implement vector-by-matrix operation in the analog domain by utilizing Ohm and Kirchhoff laws [11].

The devices are typically operated in three phases: forming, programming, and read. Upon fabrication, devices are initially in the pristine state and require a one-time forming process before becoming adjustable memristors. The electroforming process includes applying a current-limited ramp voltage to a device and continuously monitoring its low-voltage conductance. When the device reaches a certain threshold, a conductive filament forms inside it [12], and its low-voltage conductance jumps significantly, enabling subsequent analog-state tuning and storage.

In the second phase, the tuning or programming stage, the conductance of the device is adjusted to a desirable value (G) through the modulation of the impurity profile. We may increase (set) or decrease (reset) the conductivity of the device by applying a moderately large voltage to the device that is about (or slightly larger than) its switching threshold—a device-unique voltage that alters its conductance by, e.g., 20%. Harnessing the write-verify algorithm [7], we keep programming and monitoring the state of the device ($\hat{G}$) until reaching a certain relative tuning error $|\hat{G} - G|/G < \in$. (Note that a relative tuning accuracy is defined as 1 - $\in$.) Ultimately, to implement multiplication, summation, or useful computational tasks, devices are operated in the non-disturbing read (i.e., inference) phase: A relatively low voltage (V) is applied to the device, and the generated current, $I = \hat{G} \cdot V$ is sensed in a CMOS circuity.

When a high-precision readout circuit is available and memristive devices have excellent retention characteristics, $\in$ is almost entirely bounded by the devices' dynamic switching characteristics and their variations. To clarify this, consider the practical V/2 approach [7], [9] of tuning memristive crossbars (Fig. 1b). The voltage applied on the selected device (by peripheral decoders and switch matrix) is V_{set}. Unselected electrodes are pinned to $V_{\text{set}}/2$ to minimize the disturbance on other devices. The applied voltage on the unselected devices is zero; however, $V_{\text{set}}/2$ is dropped on the devices which share an electrode with the selected device (i.e., half selected devices). If the switching threshold of these devices is $\sim V_{\text{set}}/2$ or less, their state shifts undesirably, resulting in an imprecise tuning. A similar idea also holds for the reset operation. Fig. 1c shows the measured *I-V* characteristics of a device (R_0) in the 64 × 64 crossbar. Two hypothetical switching threshold distributions and *I-V* characteristics corresponding to half-selected devices R_1 and R_2 are also shown to clarify our point. When we set R_0, $V_{\text{set,R0}}/2$ drops on both R_1 and R_2. The state of R_1 is expected to alter negligibly since the set threshold of R_1 is much larger than $V_{\text{set,R0}}/2$, unlike R_2 that switches considerably. Hence, when tuning the entire crossbar, the total disturbance is correlated to the variations in the distribution of switching thresholds, and the smaller variations (or higher uniformity) result in a higher tuning precision.

B. Experimental Demonstrations and Their Challenges

Retention, yield, and *I-V* variations and on-off conductance dynamic range are critical factors for analog-grade passive crossbar circuits that are used in neuromorphic inference computing. A 32 × 32 WO_x memristor crossbar is reported in [13] with $G_{\text{on}}/G_{\text{off}} = 3\ \mu\text{S}\ /1\ \mu\text{S}$ and >35% tuning error, though it is not clear if this reported precision is obtained after programming the entire crossbar or otherwise. A 108 × 54 crossbar made of 126 6 × 8 subarrays with ∼600 $\mu m^2 WO_x$ devices are integrated on CMOS in [14] with dot-product operation experimentally demonstrated despite minutes-to-hours-scale room-temperature retention. Ref. [15] demonstrates low normalized variations (∼3.75%), excellent retention, and high switching endurance on a 16 × 1 crossbar of 100 μm^2 SiGe devices. Ref. [16] also demonstrates passive crossbars using two-dimensional materials with 98% yield and 12.3% (5.7%) normalized variations in the set (reset) switching distributions.

TiO_2 memristors have been used in designing 10 × 2 [1], 12 × 12 [4], and 20 × 20 [2], and 64 × 64 [9] crossbar circuits, with excellent retention (>20 hrs in 100°C), endurance (> 10^6 analog switching cycles), and close to 100% yield. The normalized variations in these works are 10%, 11%, 18%,

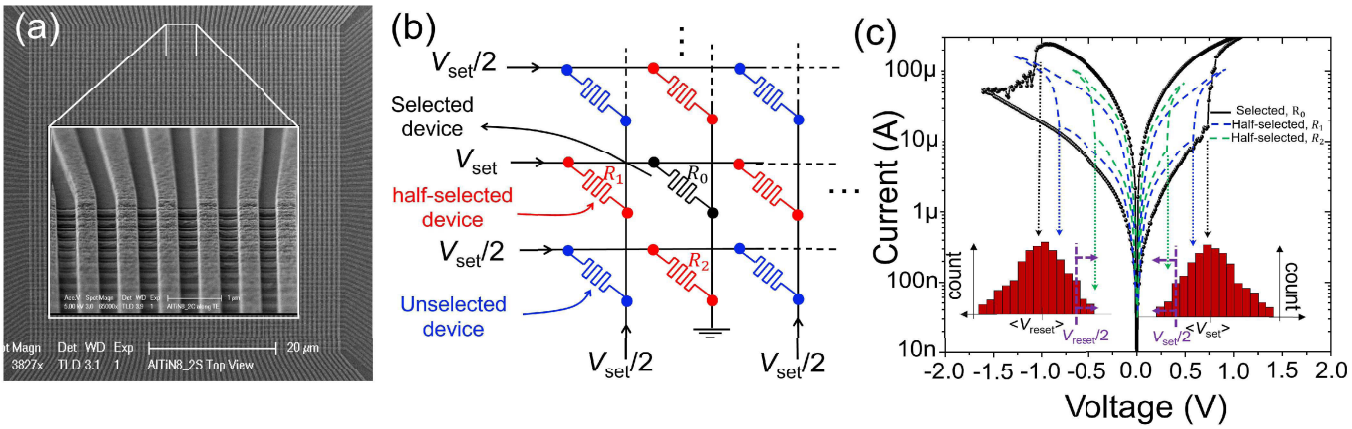

Fig. 1. (a) The SEM image of the fabricated 64 × 64 crossbar [9]. All experiments in this paper are performed with this chip. The inset shows the zoom-in view of the portion of the crossbar. (b) The schematic of the 3 × 3 portion of the crossbar and the V/2 tuning scheme with highlighted selected, unselected, and half-selected devices. Panel (c) shows a typical *I-V* characteristic of a device and reveals why the tight distribution of switching voltage is critical.

and 26%, respectively. The same stack is also used in the only analog-grade 3D integrated demonstration [5] using two layers of 10×10 memristor array and reporting ~13.6% normalized variations.

The natural trend that is observed in the TiO_2 crossbar circuits is that normalized device variations grow with increasing the crossbar size. This is in part due to the larger forming current required for electroforming increases because of the increase in leakage currents. A larger compliance voltage/current is required as more and more devices are electroformed, which increases electrical stress and ultimately leads to a higher device variability. When forming our 64×64 crossbars, the maximum electroforming current is set to ~50 μA at the beginning, but it is raised to $\sim 1 \div 5$ mA at the end. Additionally, the more devices are in the crossbar; the more disturbance is created during tuning. These two factors make high precision tuning in larger crossbars significantly more challenging. The good news is that the crossbar circuits do have to be too large. For example, our preliminary architectural studies show that for many computing applications, e.g., deep learning accelerators, the optimum crossbar dimension is in the range of 64×64 as choosing enormous crossbar modules underutilizes the hardware resources and reduces the overall performance [17].

In this paper, we are interested in investigating how the parameters of a 0T1R memristor technology, i.e., the variations in the switching thresholds, impact the tuning error ($\in$) and, in turn, the computational accuracy of memristive neuromorphic networks. The relationship between the variations in the switching threshold voltages and the crossbar size with circuit fidelity was not studied earlier. To clarify it, we first use plentiful experimental data to develop a reliable dynamic model for the memristor that relates the conductance change to the switching thresholds and the applied voltage. Then, we use this model to emulate the tuning process of ex-situ weight transfer and find the relationship between the accuracy, block size, and normalized variations in general VMM blocks and representative neuromorphic circuits.

III. Model Development and Simulation Framework

In order to study the role of uniformity in memristive crossbar circuits, we have developed a dynamic model that describes the changes of the conductance (memory state) as a function of its initial state after the application of a pulse with an amplitude V_p and a fixed duration t_p.

Note that we could not use previous models (including our recent work [18]) because they do not accurately predict the half-select drift. Though we could consider a very general model, we prefer to use a model that is more representative of our fabricated stack that meets the essential requirements for analog computing (analog tunability, high retention, endurance, etc.). Furthermore, since our candidate device and other metal-oxide memristors share some similar switching characteristics, we expect to see resembling trends of the results for other devices as well. For simplicity of the model development, we consider square-shaped write pulses (which also helped with reducing cycle-to-cycle variations as was shown in Ref. [19]).

First, we measure a massive number of experimental data points (~35×10^4) from our 64×64 crossbar and develop a model that describes an average behavior of relative change in the conductance among all devices. Then, we use a device-unique multiplicative factor α that models the variations in the switching thresholds by effectively scaling the applied voltage. Instead of relying on physics-based models, we use an empirical fitting function for modeling purposes. The relative change in the conductance of a device with conductance G, subjected to a set/reset square-wave pulse stimulation with the amplitude of V, is modeled with [16]

$$\frac{\Delta G}{G} \approx \exp\left[\frac{\beta_1}{1+\beta_2 (\alpha V)^2}\right] \sinh\left[\beta_3 \frac{\alpha V}{1+\beta_2 (\alpha V)^2}\right] \times \left(\gamma_1 + \gamma_2 \sqrt{G} + \gamma_3 G\right), \quad (1)$$

where $\beta_1, \beta_2, \beta_3, \gamma_1, \gamma_2$, and γ_3 are fitting parameters. The form of the function includes exp and sinh functions, which were used in previous works [4], [16], [34], [35] to describe

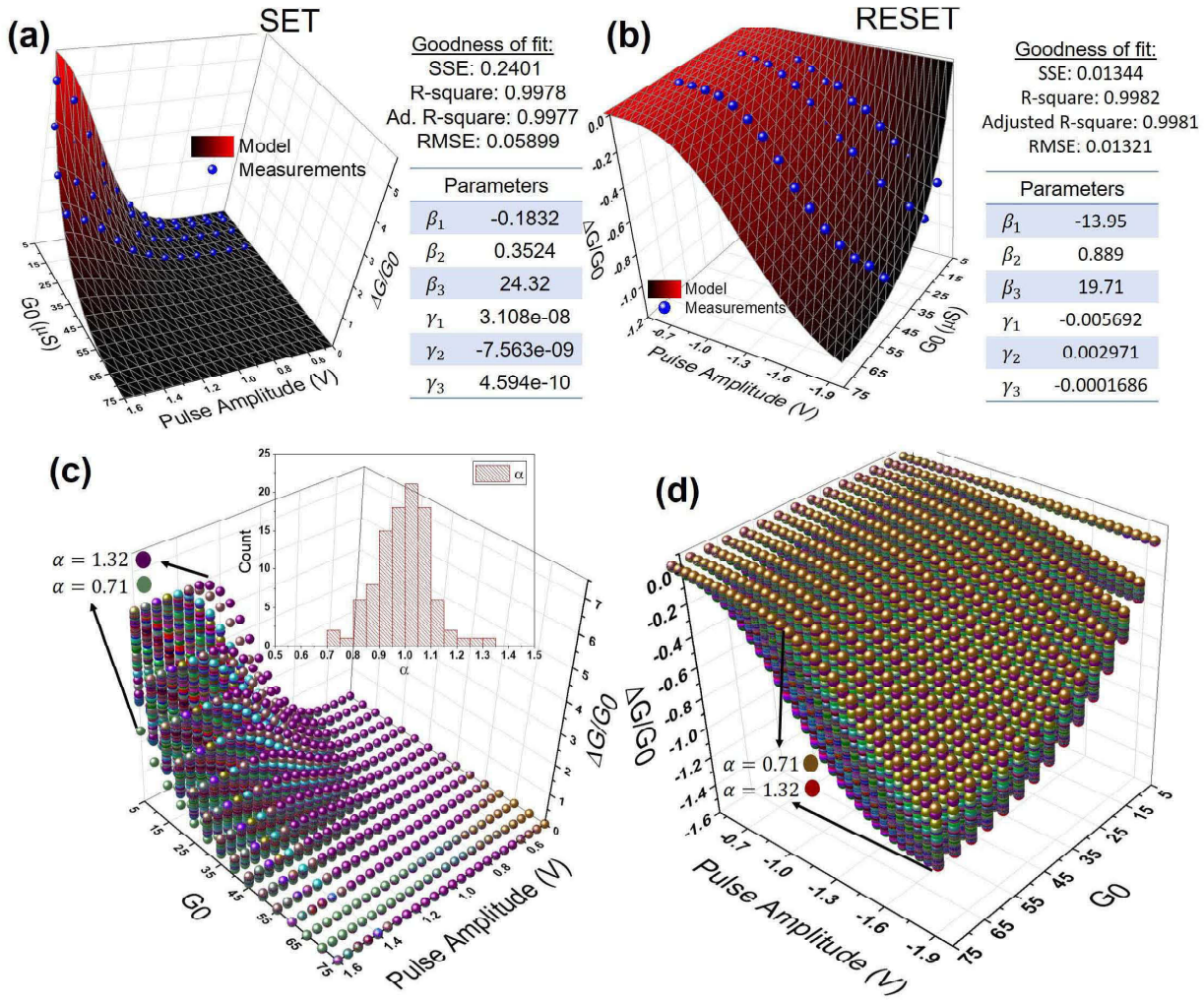

Fig. 2. The modeling results for the average set (a) and reset (b) operations. Panels show the relative change in the conductance for different initial conductances when subjected to square write pulse with a specified amplitude and constant, 2-ms duration. The inset tables show the corresponding goodness of the fit and model parameters. Panels (c-d) show the set/reset characteristics for 100 devices with 10% normalized variations. The inset shows the corresponding distribution of α.

the nonlinear switching dynamics of memristive devices. This model captures the exponential switching kinetics with applied voltage in nonvolatile ionic memories [20], [21]. Also, we include an extra term $(\gamma_1 + \gamma_2\sqrt{G} + \gamma_3 G)$, which models the nonlinearity with respect to an initial conductance G. This update function is multiplied by a simple exponential window function to avoid out-of-range updates.

Specifically, the average switching characteristics of 500 devices are measured in the crossbar using 2 ms pulse width and several initial conductance points. The trust-region algorithm [33] is used for nonlinear least-squares to optimize the model parameters. Fig. 2 shows the experimental data and modeling results for set and reset operations within the useful range of device conductance ($\sim$5 μs to $\sim$75 μs). The model parameters closely reproduce the measurement results. The fitting parameters and goodness of fit for both set and reset operations are also shown in Fig. 2. Note that, by definition, $\alpha = 1$ is used in the average model. Given a certain switching threshold (V_{set}), α is obtained using $\alpha = \overline{V_{set}}/V_{set}$, where $\overline{V_{set}}$ is the average set threshold. A similar definition is also used for the reset operation. When studying the normalized variations, we use fixed average thresholds ($\overline{V_{set}} = 1$ V and $\overline{V_{reset}} = -1.2$ V, obtained from the measurements under the condition $\Delta G/G = 0.2$), employ log-normal distributions with lower bound clipped to 0.5 V, and change their standard deviation parameter. Fig. 2c-d shows the simulated set/reset characteristics for 100 devices with 10% normalized variations. In the following, we discuss how we emulate the tuning and perform the ex-situ training using this dynamic model.

In ex-situ training of a neuromorphic network, synaptic weights are calculated on a precursor software-based network and then imported sequentially into the crossbar circuits. Networks are typically composed of many crossbar blocks which are programmed in parallel or sequentially. However, within a crossbar, the devices are tuned into their corresponding predetermined desired states individually (one-by-one). Due to the stochastic nature of the switching mechanism in memristors, particularly analog-grade devices often require

multiple pulses to reach the desired accuracy. This is executed using the well-known write-verify algorithm [7].

In every simulation case, the devices' conductances are initially randomized using a Gaussian distribution with an average of 36.25 μS (midrange conductance) and a standard deviation of 9 μS. Then, conductances are adjusted one by one using the write-verify algorithm and the dynamic model. We reconstruct the exact procedure that we employ in the experiments when tuning the devices [7], [9]. The devices within any crossbar block are tuned in raster order. More importantly, to increase the tuning speed, we progressively increase the pulse amplitude (set/reset) starting from 0.5 V with 10 mV steps to the switching voltage of the device. The tuning direction (setting or resetting) is alternated whenever we pass the target conductance. To avoid overstressing the memristors, creating too much disturbance, and reducing the tuning time, we limit the tuning process for every device to 5 rounds. The algorithm is aborted (and restarted with the next device) whenever it reaches the desired tuning accuracy or the maximum permitted pulse per device. The half-select disturbance is simulated for every applied pulse and every device by updating the state of devices sharing either top/bottom electrode with the V/2 rule.

The procedure of tuning all crossbar devices is repeated 10 times (rounds) to improve the results by re-tuning the devices disturbed by half-select effect.

IV. Computing Precision in Nonuniform Crossbar

VMM is the most critical operation in inference accelerators and most neuromorphic tasks. The fidelity of most neural network models closely follows the computing precision in their VMMs. Here, we consider $N \times N$ two-quadrant VMM circuits, which are implemented in the analog domain by two separate $N \times N$ memristive crossbars. VMM size, variations in switching thresholds, and target precision are variables of this research. For every case study, 20 crossbars with random log-normally distributed switching thresholds and 20 different normally distributed weight matrices with zero mean are generated. The mapping function $G_{ij}^{\pm} = G_{\min} + (1 \pm W_{ij})(G_{\max} - G_{\min})/2$ in which $W_{ij} = [-1, +1]$ is the normalized weight and $G_{\max}$ and $G_{\min}$ are upper and lower conductance bounds are used to convert dimensionless weights into device conductances [22]. For each VMM, we randomly generate 1k input voltage vectors, with elements uniformly distributed in the range 0 to 0.1 V. VMM computing errors are then calculated over the output current (I) and defined by $|I_{\text{actual}} - I_{\text{ideal}}|/I_{\max}$. Ideal currents ($I_{\text{ideal}}$) are obtained directly from the mathematical vector-by-matrix multiplication of the input voltage vector and conductance matrix, actual currents (I_{actual}) are obtained from the circuit simulation after all devices are tuned, and $I_{\max}$ is the maximum absolute pre-activation current over all input combinations.

First, the half-disturbance issue is investigated for 64×64 VMMs and 5% and 25% variations in switching threshold voltages. Fig. 3a shows the tuning error for 50 devices (in the crossbar that implements G^{+}) during 10 rounds of the programming phase in the case with 5% variations. Specifically, each curve shows how the tuning error for each device evolves, starting from the first tuning round to the last one. One curve is highlighted for better clarity. The steep drops in each curve denote the moments the device is tuned. For the highlighted curve, the device is initially tuned with $<1\%$ error, but the disturbance moves its state leading to $\sim$5% error by the end of the 1st round. The device is retuned in the 2nd round, and the disturbance alters it to $\sim$3% of the target. Less disturbance generated in the 2nd round stems from the fact that some devices are within the target accuracy by the end of the 1st round. So, the total number of pulses (and hence overall disturbance) decreases in each round. The state of most devices stabilizes by the end of the 4th round. The conductance error distributions and related statistics, shown in Fig. 3b, confirm these findings as well.

The assumption of 5% variations in a 64×64 crossbar is too optimistic with the current technology. Figs. 3b and 3c show the result from the simulations of crossbars with 25% variations in the switching thresholds. Though the results slightly improve in the first 4 rounds, many devices remain in imprecisely tuned states after that. The periodic state evolution of many devices (e.g., the highlighted curve) in Fig. 3c is because of the large disturbance and strong dependencies, making the tuning effectively unstable for many devices. Fig. 3e compares the ultimate distribution of conductance error for both cases. The 99 percentiles of the tuning error are $\sim$14.4 % and $\sim$1.0 % for 25% and 5% variations, respectively. The huge gap between the realistic and ideal case signifies the importance of variations in passive crossbars. Imprecise tuning results in a large error in the output signal, as expected. Fig. 3f shows the VMM error distribution for both cases. The 99 percentiles of the distributions are $\sim$7.0 % and $\sim$3.7 % for $\sigma_{\text{n}} = 25\%$ and $\sigma_{\text{n}} = 5\%$ variations, respectively.

Fig. 4 summarizes our VMM-level simulation results in which the role of VMM size, switching threshold variations, and target tuning error are studied. Every data point is obtained by considering 400 VMM instances (20 different sets of weights and 20 crossbars) to characterize the worst-case error statistics (99 percentiles of the output error among 10^3 patterns), a useful parameter to evaluate the computational accuracy. Note that the VMM size and normalized variations are increased exponentially and linearly, respectively. As a convenient baseline, the dashed red line shows the expected intrinsic error resulting from the imprecise tuning of individual devices, i.e., without the half-select problem that would be representative of 1T1R circuits. Such intrinsic error is roughly linearly proportional to the target tuning error.

The common observation from Fig. 4 data is that the median worst-case error increases exponentially with variations, more evidently for $N > 30$ (here, the median refers to the statistics over 400 VMM instances). It also increases roughly exponentially with respect to VMM size for low variations and super-exponentially in large variations. This is due to the unstable tuning process in larger circuits, in which the disturbance of half-select devices overwhelms the tuning of individual devices (Fig. 3c). The spread of the worst-case VMM error distribution among different instances also extends with increasing crossbar size and/or device variations for high precision tuning cases since the chances of hitting worse

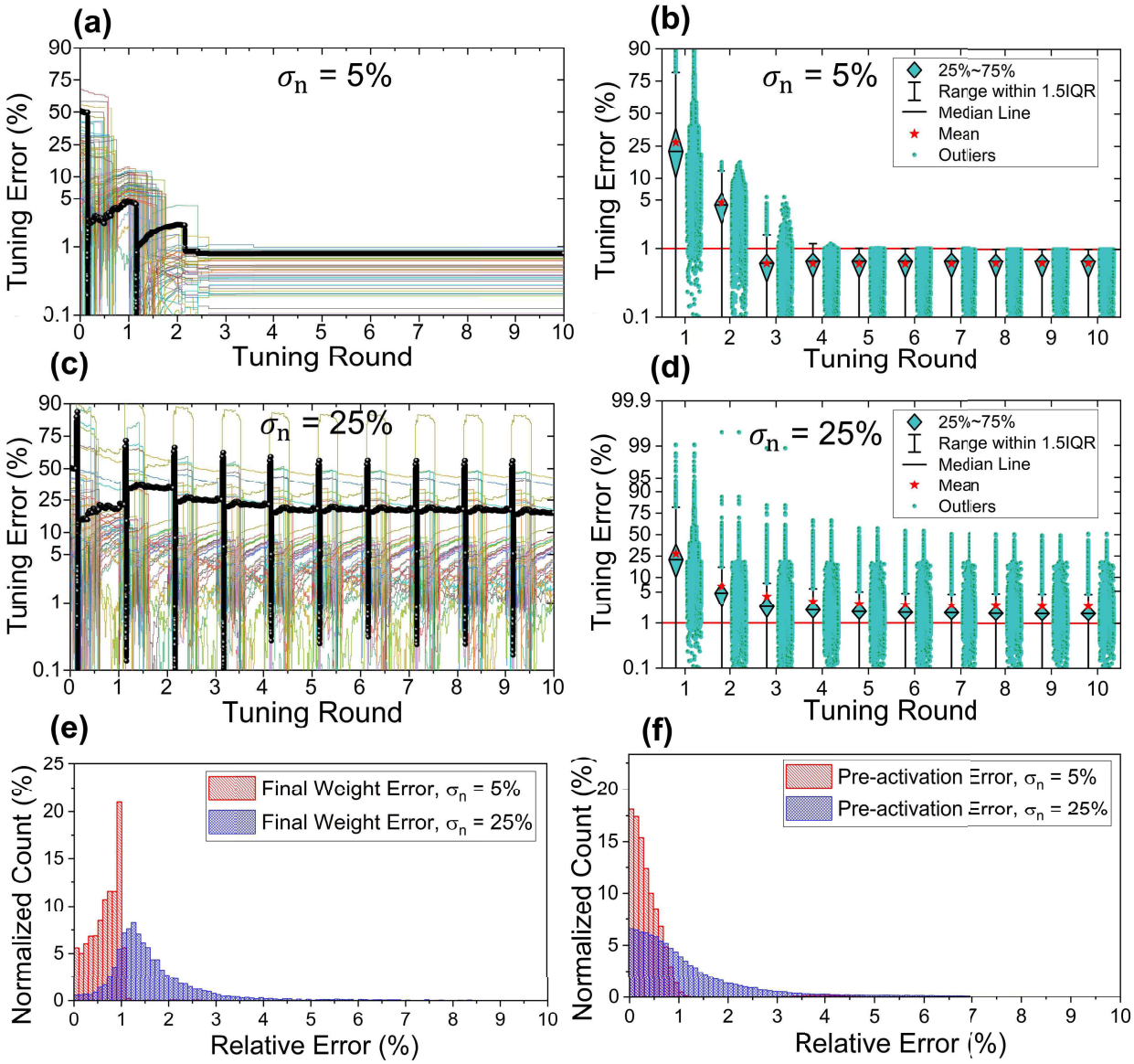

Fig. 3. Tuning analysis in a 64 × 64 VMM with random weights for two cases of 5% and 25% of normalized variations in switching thresholds. Panels (a,b) and (c,d) correspond to weight tuning error for 5% and 25% variations, respectively. Panels (a and c) show the evolution of tuning error for 50 devices during 10 rounds of tuning the crossbar. Each line corresponds to the tuning error for each device. Panels (b and d) denote the error distribution for all devices at the end of each round. Panels (e) and (f) show the distribution of weight and VMM errors (at the end of the 10^{th} round) for the two cases, respectively.

corner cases increase. This issue becomes particularly important in high-precision computing tasks with tight error margins.

For small VMMs (e.g., $N < 16$), the error follows the intrinsic trend even in the presence of large variations because the total disturbance is low enough to be fully recovered after running several tuning rounds. In moderate VMM sizes (e.g., $N = 32$), the error tends to increase for high precision tuning cases (e.g., $< 4\%$), particularly when the variations are high. This error escalation originates from an increase in the number of applied pulses for achieving a better tuning precision, which in turn leads to a larger disturbance. For large VMMs, variations become more prominent such that the computational accuracy is adversely impacted. For $N = 64$, the drastic change for $\sigma > 0.25$ also stems from the exponential growth of the severe half-select disturbance cases. To clarify this, let us look at a fraction of the devices in the crossbar circuit that is disturbed during write operation with half write voltages exceeding their switching threshold (Fig. 4). When the standard deviation of the set threshold distribution increases from 0.25 to 0.3, this number soars by a factor of $\sim$10, indicating a surge in the cases of severe disturbances.

Another subtle point is related to the reduced computational accuracy in cases with even no variations. For instance, comparing the case of no half-select (no HS) with $\sigma_n = 0$, we observe a $\sim$4.3% increase of the average error in the case of $N = 64$ and 1% target. Even with no variations, the voltage drop on other devices could have a slightly disturbing effect (non-zero changes in the conductive state at half of the write voltages), which could become potentially noticeable when the

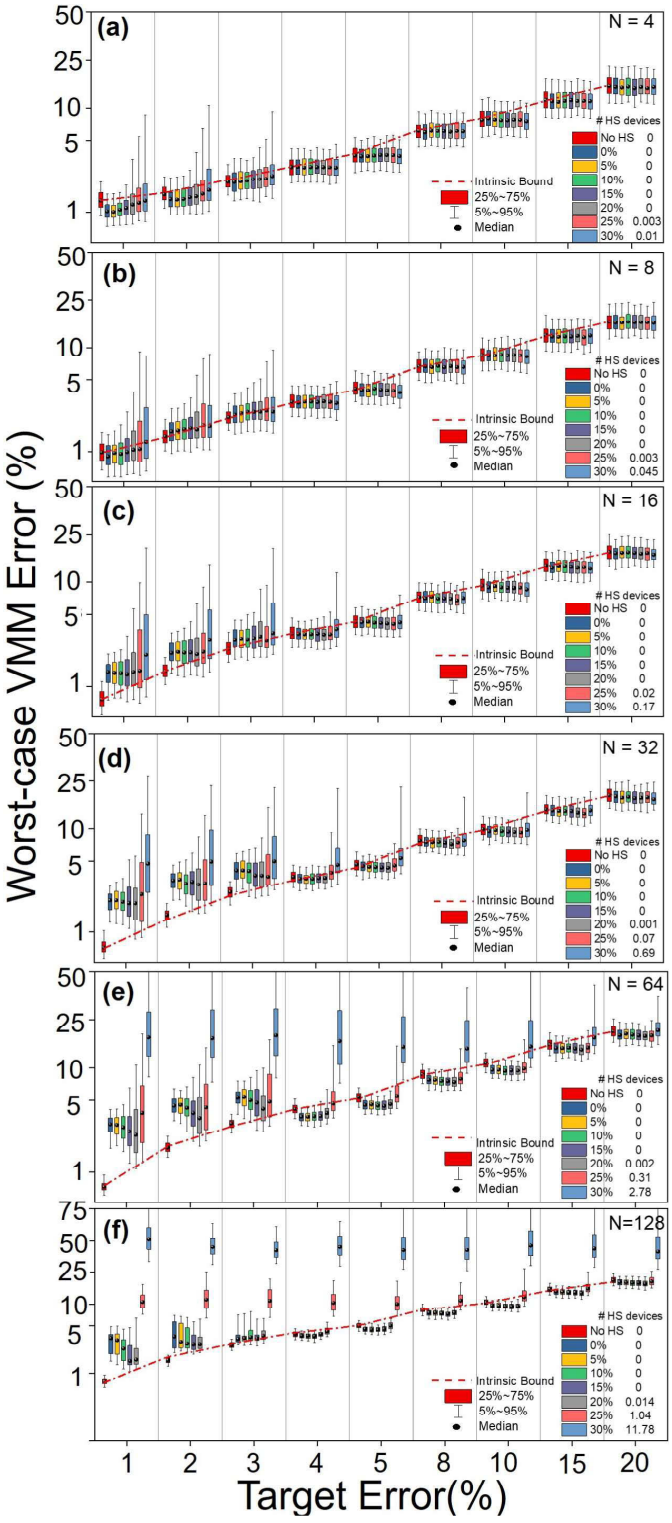

Fig. 4. The distribution of the worst-case VMM error (99 percentile of the absolute error distribution for 10^3 patterns) among 400 instances of a) 4 × 4, b) 8 × 8, c) 16 × 16, d) 32 × 32, e) 64 × 64 and f) 128 × 128 VMMs. No HS: no half-select. #HS devices: the average number of devices affected with V/2 disturbance normalized by N^2, which is a metric that shows the overall disturbance level in a certain crossbar. The median for the 'no half-select' case is the intrinsic bound that shows how accurately we can compute given a semi-exponential tuning error distribution (Fig. 3e) in the crossbar. Note the log-scale of the y-axis in all panels.

total number of pulses grows very large. This issue would get worse for device technologies with inferior memory retention. On the other hand, the slight improvements beyond the intrinsic error (e.g., see the case of $N = 4$ and target error = 5%) originate from the regularization impact of half-select disturbance, which slightly improves the accuracy.

The computational accuracy of state-of-the-art 1T1R and 0T1R crossbars can be compared in Fig.4. For practical VMM sizes (e.g., $N = 64$) and considering state-of-the-art reported 1T1R (i.e., with no half-select) tuning error of ∼3% in [10] using highly conductive devices, the computational accuracy of 0T1R VMMs ($\sigma < 30$) is the same as 1T1R when tuned with 1% target error. The computational accuracy is worse by ∼1% when 0T1R circuits are tuned with a similar 3% error. We believe that such results are very encouraging and give hope for using 0T1R design in large-scale neuromorphic computing networks.

The final takeaway is that the computational accuracy in passive crossbars is a function of the total number of applied pulses or, equivalently, the total disturbance imposed by the tuning algorithm. The general trends are that the larger the VMM is, the higher the number of tuning pulses is required. The larger the variations are, the more pulses are needed to tune the devices in multiple rounds. And the smaller the tuning target error is, the higher the number of pulses is required. Consequently, assuming system-level and architecture considerations determine an optimum kernel size (N) to optimize the functional performance, there are two natural options for mitigating the half-select disturbance and improving the computational accuracy in neuromorphic systems based on passively-integrated memristors, namely fabricating more uniform crossbars that lead to tighter variations and developing more optimum tuning algorithms that directly reduce the total disturbance and the number of applied pulses. Furthermore, the most efficient and accurate circuit is not necessarily obtained when the device is pushed to its high precision limit. Hence, extensive simulations are required to find the optimum tuning margin for a given technology, kernel size, and the computing model.

V. Modeling Neuromorphic Inference Applications

We consider two representative neural network models: a moderate-size convolutional neural network (ConvNet) and ResNet-18. The former is a modified Lenet-5 architecture that includes 2 convolutional, 2 pooling, and 2 fully-connected layers (see [23] for more details on the structure). The model is trained with 50k images and tested on the remaining 10k images of the CIFAR-10 dataset. Standard data augmentation techniques are employed to improve the accuracy. Each image is zero-padded with two pixels before we crop a random 32 × 32 region and perform random horizontal flipping of images. We use ADAM optimizer, cross-entropy cost function, a batch size of 64, a learning rate of 0.001, and 220 epochs to achieve 87.25% inference accuracy.

The ResNet-18 implementation is based on the pre-trained model available at the official model zoo of PyTorch. It includes 21 + 2 layers: a convolutional layer with 7 × 7 kernels and stride of 2, a max-pooling layer with 3 × 3 kernels and stride of 2, 4 convolutional blocks with

residual connections, each including 4 convolutional layers based on 3 × 3 kernels and strides of 2 and 1, a 7 × 7 average-pooling layer with the stride of 7, and finally a 512 × 1000 fully-connected layer that provides the output prediction corresponding to 1000 classes. The network is trained on ∼1.3M images of the ImageNet dataset for 150 epochs with a batch size of 256, the learning rate of 0.1 that is divided by 0.1 every 30 epochs (step scheduling), cross-entropy cost function, weight decay of 0.0001, and stochastic gradient descent optimization with a momentum of 0.9. The model achieves an average classification accuracy of ∼70.2% tested on 50k images of the dataset. The networks are trained with 32-bit floating-point precision on Nvidia Titan X GPUs, and the learned parameters achieving the highest test accuracy are used as the baseline model.

Note that the baseline classification accuracy is somewhat worse compared to the state-of-the-art numbers that are achieved with more complex neural network models. However, the chosen models in this paper allow performing simulations reasonably, focusing on the impact of the uniformity.

In every model, the VMM operations are partitioned to nonoverlapping $N \times N$ kernels - see the example of such partitioning in general-purpose mixed-signal deep neural networks [17]. In other words, in order to conduct this study using GPUs, we trustfully modeled the entire circuit, e.g., mapping each weight to a unique pair of two adjustable devices. On the other hand, since this paper focuses on device uniformity rather than any other nonideality, we assumed ideal peripheral transfer functions and pooling layers.

Similar to the VMM study, the obtained weights are mapped into target device conductances. The conductance tuning process for the constructed VMM kernels is then emulated using the device model and previously discussed tuning algorithm. The imprecise tuned weights are then imported backed to the simulation setup. Subsequently, the inference tasks are performed on the generated models, and the classification drop is recorded for each data point. For every case study, 12 model instances are generated by using 12 sets of randomly generated switching threshold distributions.

Fig. 5 shows the accuracy drop of running the inference test on both benchmarks versus the crossbar uniformity for various VMM sizes. The box plot is obtained by simulating 12 random hardware instances - note that tuning simulations are extremely slow even when performed on a powerful server. The destructive impact of crossbar half-select disturbance is evident in both benchmarks, especially in ResNet-18 that performs the more complex ImageNet classification. The trends are consistent with VMM simulations in that the accuracy drops roughly exponentially when the VMM size or normalized variations are increased. Notably, with 25% normalized variations and 64 × 64 crossbars, we achieve ∼9% accuracy drop in the ConvNet and 18.5% on ResNet-18. In the next section, we introduce several methods, which restore this accuracy drop.

VI. Improving the Accuracy

The most straightforward solution to cope with the destructive impact of variations in the switching thresholds is to improve the fabrication process and device properties. The switching threshold variations in metal-oxide memristors depend on multiple factors. Forming voltage and current overshoot during the forming significantly contribute to device variability and can be tuned by a combination of oxide layer thickness and stoichiometry adjustment and optimized annealing conditions [4], [25]. Here, we focus on some post-fabrication techniques for mitigating the disturbance.

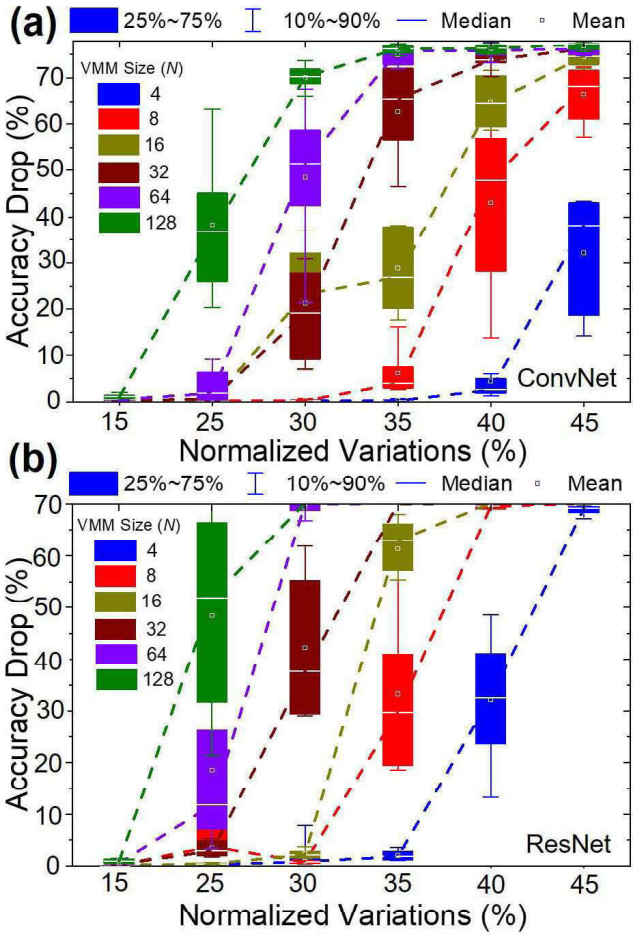

Fig. 5. The accuracy drop in deep neuromorphic networks versus crossbar uniformity: (a) ConvNet, (b) ResNet-18. Each box plot is obtained from the simulation of 12 different switching threshold distributions (e.g., corresponding to 12 emulated chips). The dashed lines connect the median of the boxes.

A. Hardware-Aware Training

In our recent works [23], [26], imperfections of synaptic devices such as noise, temperature dependency, stuck-at fault, retention, and tuning error are compensated by the method of hardware-aware training: The training is performed fully ex-situ (no extra hardware cost), with the only subtle difference of including the device models and imperfections in the training phase for the purpose of generating more robust models.

The simulation results of the previous section indicate that variations in switching thresholds lead to random tuning errors in the devices. Note that the tuning errors remain fixed during the inference, assuming devices have adequate

retention. Nevertheless, tuning errors are chip-dependent, model-dependent, and unpredictable because of the intrinsic chip-specific distribution of switching thresholds. Though accounting for individual device tuning errors in the training phase is not feasible, the error distribution is predictable due to the uniform shape of weight distribution in a neural network model, especially when using the same crossbar sizes and tuning algorithm (see modular accelerator architectures, e.g., aCortex [17]).

We model the tuning error during the training to increase the robustness of the trained model against half-select disturbance during the inference of the neural hardware. Specifically, prior to computing the activation values in each update, the weights are converted to memristor conductances. Built-in uniform random number generator with the parameter ζ is then used to perturb conductances (both G^+ and G^- in the differential implementation). After computing imprecise preactivations, the ideal weights are then restored before proceeding with the rest of the training operations. Note that ζ is optimized for a given network model and overall disturbance, which is a function of VMM size, switching threshold variations, and target accuracy.

Fig. 6a shows the performance improvement achieved by this technique on the ConvNet benchmark implemented with 64×64 VMM blocks. The figure shows the accuracy drop versus the normalized variations for various values of ζ. The robustness of the deployed model is obviously increased with this method. For 15%, 25%, and 30% normalized variations, the optimum performance is achieved when ζ is set to 5%, 20%, and 30%, respectively. Notably, in the case of 25% normalized variations and 64×64 crossbars, the ~9% average accuracy drop is now reduced to ~1.87% using $\zeta = 20\%$. The same trends of improvements are also observed in the case of ResNet-18 implemented with 64×64 crossbars (Fig.7b). For example, using $\zeta = 3\%$ (20%) diminishes the average accuracy drop from 18.5% to 3.5% (6.1%) for $\sigma = 25\%$.

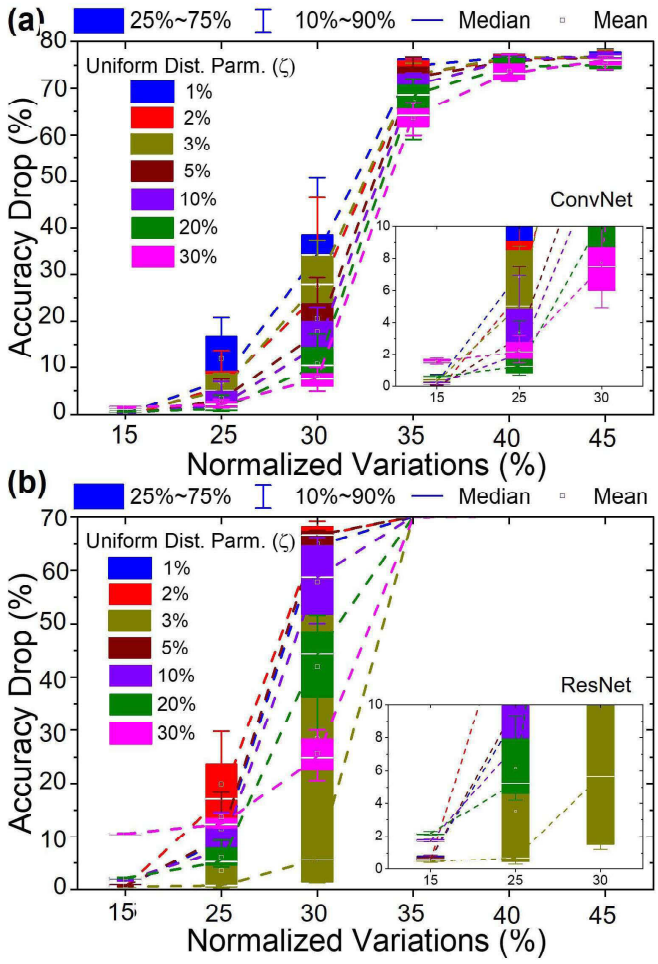

Fig. 6. Reducing the accuracy drop in (a) ConvNet and (b) ResNet-18 (both with 64×64 VMMs) using the hardware-aware training technique. By emulating the distribution of the tuning error during the training, the network becomes more resilient toward the half-select disturbance. The inset shows the zoomed-in to the lower portion of the figure.

B. Improved Tuning Algorithm

In Ref. [9], we propose a novel crossbar tuning procedure consisting of two methodologies for reducing the tail of tuning error distribution. First, the write voltage amplitudes are limited to a specific voltage, which is decreased gradually within each tuning round. The consequences of restricting the maximum applied write voltage within each round are gradual reduction of net disturbance in each round and large (final) tuning error in high threshold devices. The former stems from the fact that low-to-moderate threshold devices become disturbed less and less as the tuning algorithm advances. The latter is due to not sufficient write voltages to switch the high threshold devices.

In the second method, we initially identify devices with large set (reset) switching thresholds and switch them to the highest (lowest) conductive state prior to executing the first tuning round. Then, we take advantage of the possibility to encode the same weight with different target conductances in the differential pair implementation. In every round, when tuning a disturbed device with a threshold higher than the maximum voltage limit imposed by the first methodology, the state of the paired device is adjusted rather than the high voltage device. The application of these two novel techniques significantly reduces the tail of disturbed devices.

Fig. 7a demonstrates the effectiveness of using these novel tuning algorithms with and without applying the hardware-aware training technique. When no hardware-aware training is applied, the novel tuning algorithm reduces the accuracy drop, especially when variations are higher than 20%. When the two techniques are both applied, the results are even better. A sub-percent accuracy drop is now feasible even with 30% normalized variations. For the notable case of 25%, the average drop now becomes insignificant when $\zeta = 2\%$ is used in the hardware-aware training.

The simulation results of the ResNet-18 benchmark are also promising (Fig.7b). For example, in the case of $N = 64$ and $\sigma = 25\%$, the improved tuning algorithm solely reduces the accuracy drop to 1.88%. Combined with the hardware-aware training ($\zeta = 3\%$), we can decrease the average accuracy drop to just ~0.4%. In the initial simulation, we observe that the model generates almost random outputs (~70% accuracy

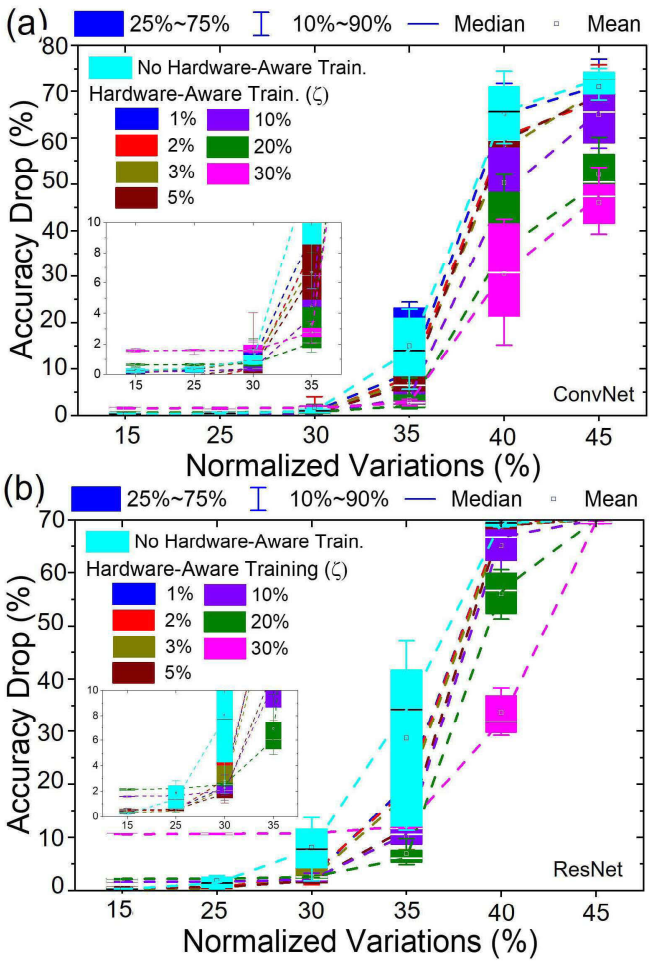

Fig. 7. Reducing the accuracy drop in (a) ConvNet and (b) ResNet-18 (64 × 64 VMMs) using the novel tuning algorithm with and without the hardware-aware training. The inset shows the zoomed-in to the lower portion of the figure.

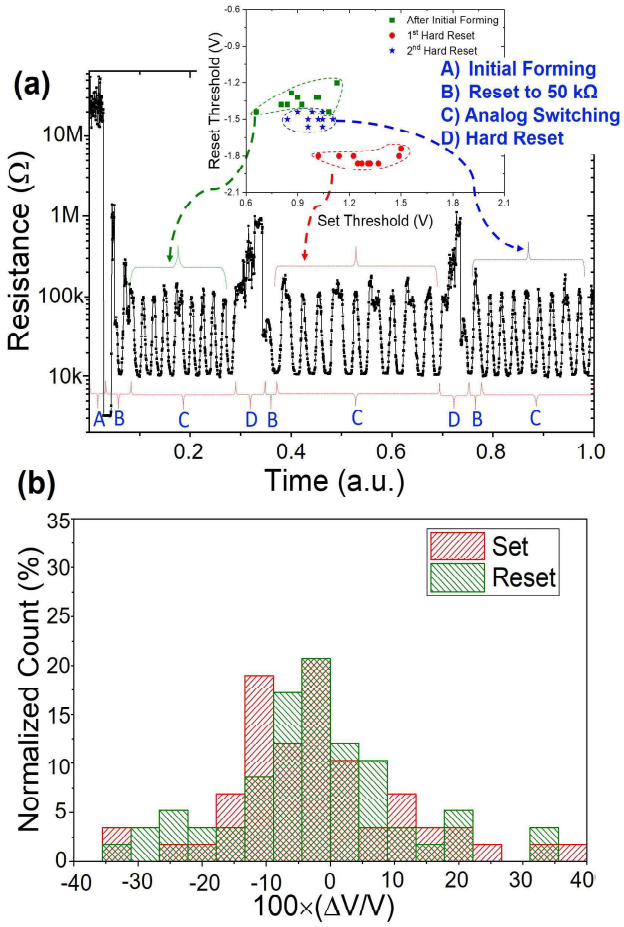

Fig. 8. Modulation of switching thresholds: (a) The experimental results of the modification process applied on a virgin device in the crossbar. The process includes initial electroforming, tuning the device to 50 kΩ, analog switching for ten times between 10 kΩ and 100 kΩ, which are close to typical the lowest and highest resistance states, and hard reset to a state of more than 1 MΩ. The inset shows the switching thresholds measured in each round. (b) The histogram of the stochastic relative change ($100 \times \Delta V / V$) in the average set/reset switching thresholds (V) for 60 devices after performing the hard reset and revival processes.

drop) when the variations are $\sigma = 35\%$ and larger. While the two proposed techniques enable 6.9% and 17.2% average accuracy drops, utilizing $\zeta = 20\%$ and $\zeta = 3\%$, respectively.

C. Modifying Switching Thresholds

Modifying the switching thresholds of outlier devices is another method for reducing the impact of variations in the switching thresholds. This correction process includes an unconventional continuous hard reset operation, which pushes the outlier device close to its virgin state, followed by a voltage-controlled reforming procedure, which revives the device with slightly shifted switching characteristics. Our experiments show that the correction process results in a stochastic shift in the switching threshold of devices, which means the refreshed device could have improved switching properties. Applying this technique to outlier devices (that feature low voltage thresholds) reduces the spread of variations, which in turn improves the accuracy of the implemented model.

Fig. 8 shows the results of the experiments developed to confirm this idea. First, a virgin device in the crossbar is formed and tuned to 50 kΩ. Then, its switching thresholds are measured by tuning it repeatedly to 100 kΩ and 10 kΩ. After 10 rounds, the device is hard reset to >1MΩ and then revived and tuned to 50 kΩ. Switching thresholds are measured again in a similar fashion. The process is performed one more time just to make the results more illustrative. In another experiment, a one-time switching threshold modification method is applied on 60 devices with various initial switching thresholds (Fig. 8b). The results of Fig. 8b clearly show the average shift of threshold voltages. For example, one device has an average (over ten switching events) +0.8 V set and -0.8 reset switching thresholds. After applying the threshold modification method, an average set and reset voltages became 0.95 V and -1.2 V, respectively, for this device, which is significantly better and closer to the typical average switching thresholds of the crossbar. (Note that due to the limitations of our experimental setup, we can not validate the impact of this method with direct system-level experimental results. The study of the impact of this technique on large-scale neuromorphic architectures is an important future work.)

VII. Discussion

The major focus of this paper is on device uniformity. We propose several novel methods for coping with large device variations and investigate the impact of the device uniformity on the accuracy of neuromorphic inference circuits implementing representative neural models. The findings in this paper confirm the encouraging prospects for using 0T1R crossbars in neuromorphic computing. The results of our paper are complementary to prior works [27]–[50] that have focused on other nonidealities, e.g., IR drop, static nonlinearities, retention loss, with a specific focus on devices with selectors, i.e., 1T1R circuits with inferior density with respect to passively integrated crossbars.

Specifically, in the presented general VMM study (section IV), the major contributions include:

- The relationship between the computational error and the crossbar size, uniformity, and target tuning error is thoroughly investigated.

- We present the periodic and instability of tuning error (Fig. 3c) in large nonuniform crossbars in addition to the linear and exponential dependency of computing accuracy to uniformity at small, moderate, and large VMM sizes.

- It is shown that in large VMMs, very precise tuning of devices requires a large number of pulses, which in turn may lead to more disturbance and reduction of the ultimate computing accuracy.

- Slight increase in the computational error is inevitable in very large 0T1R crossbars even with zero variations since even a small half-select voltage drop could become potentially noticeable when the total number of pulses grows very large.

- We compare the computational accuracy of state-of-the-art 1T1R ($\sim$3% target error reported in [10] based on extremely conductive devices) and 0T1R crossbars (1% target tuning and $\sigma \sim 25\%$ reported in [9]) and report similar computing accuracy when the 0T1R crossbars are tuned with 1% target precision or worse by only $\sim$1% when using the same (as 1T1R) tuning precision of 3%.

Furthermore, three techniques are explored for mitigating the impact of nonuniform *I-V* characteristics of 0T1R memristors in neuromorphic circuits. The simulation results indicate that these techniques enable software-equivalent accuracy on both ResNet-18 and ConvNet benchmarks, in the case of $N = 64$ and 25% normalized variations, which corresponds to the features of our recent fabricated crossbar. In addition, the presented data in Fig. 7 suggest that a sub-percent accuracy drop is achievable in advanced neuromorphic circuits with even $\sim$30% normalized variations using 64×64 crossbars, which leads to a balanced resource utilization at system levels, as predicted by theoretical architectural studies [17].

Let us also mention several limitations of these mitigation techniques. First, hardware-aware training is not a viable option in some neuromorphic tasks, e.g., neurooptimization [24], in which the weights are fixed and predetermined by some constraints of the applications. In such cases, the practical solutions are improved tuning algorithm, fabrication process, outlier correction, and, if needed, reducing the crossbar dimensions. Second, the switching threshold modification method should only be used for outlier devices once or a few times to prevent damaging the devices or reducing their endurance life.

VIII. Conclusion

The excellent scalability prospects of memristors are promising for designing energy-efficient and compact neuromorphic circuits. However, the strict uniformity requirements on the *I-V* characteristics of memristors make the scaling dimensions of 0T1R memristor crossbars challenging. In this paper, we have conducted an in-depth analysis of this problem and studied the tradeoffs between computing accuracy, crossbar size, switching threshold variations, and target precision. The tradeoffs are first studied for vector-matrix multiplication circuits. The impact of crossbar uniformity is then investigated for two representative deep neural networks. Most importantly, we proposed and evaluated three solutions - hardware-aware training, improved tuning algorithm, and switching threshold modification - for improving the performance. It is shown that the current state-of-the-art analog-grade 0T1R technology can offer software-equivalent accuracy of advanced deep neural networks. Although the paper has mainly focused on uniformity, the primary challenge of upscaling 0T1R crossbars, other nonidealities, including limited retention time, endurance, device noise, and temperature dependency, are also important require in-depth analysis.

References

[1] 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, no. 1, Oct. 2013, Art. no. 2072.

[2] F. M. Bayat, M. Prezioso, B. Chakrabarti, H. Nili, I. Kataeva, and D. Strukov, "Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits," *Nature Commun.*, vol. 9, no. 1, Dec. 2018, Art. no. 2331.

[3] M. Hu *et al.*, "Dot-product engine for neuromorphic computing: Programming 1T1M crossbar to accelerate matrix-vector multiplication," in *Proc. 53rd Annu. Design Autom. Conf. (DAC)*, Austin, TX, USA, Jun. 2016, pp. 1–6.

[4] 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.

[5] G. C. Adam, B. D. Hoskins, M. Prezioso, F. Merrikh-Bayat, B. Chakrabarti, and D. B. Strukov, "3-D memristor crossbars for analog and neuromorphic computing applications," *IEEE Trans. Electron Devices*, vol. 64, no. 1, pp. 312–318, Jan. 2017.

[6] X. Guo *et al.*, "Fast, energy-efficient, robust, and reproducible mixed-signal neuromorphic classifier based on embedded NOR flash memory technology," in *IEDM Tech. Dig.*, San Francisco, CA, USA, Dec. 2017, pp. 6.5.1–6.5.4.

[7] F. Alibart, L. Gao, B. D. Hoskins, and D. B. Strukov, "High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm," *Nanotechnology*, vol. 23, no. 7, 2012, Art. no. 075201.

[8] H. Nili *et al.*, "Hardware-intrinsic security primitives enabled by analogue state and nonlinear conductance variations in integrated memristors," *Nature Electron.*, vol. 1, no. 3, Mar. 2018, Art. no. 197.

[9] H. Kim, H. Nili, M. Mahmoodi, and D. Strukov, "4K-memristor analog-grade passive crossbar circuit," 2019, *arXiv:1906.12045*. [Online]. Available: http://arxiv.org/abs/1906.12045

[10] Z. Wang *et al.*, "Reinforcement learning with analogue memristor arrays," *Nature Electron.*, vol. 2, no. 3, Mar. 2019, Art. no. 115.

[11] M. Bavandpour *et al.*, "Mixed-signal neuromorphic inference accelerators: Recent results and future prospects," in *IEDM Tech. Dig.*, San Francisco, CA, USA, Dec. 2018, pp. 20.4.1–20.4.4.

[12] D. Ielmini and R. Waser, *Resistive Switching: From Fundamentals of Nanoionic Redox Processes to Memristive Device Applications*. Hoboken, NJ, USA: Wiley, 2015.

[13] P. M. Sheridan, F. Cai, C. Du, W. Ma, Z. Zhang, and W. D. Lu, "Sparse coding with memristor networks," *Nature Nanotechnol.*, vol. 12, no. 8, Aug. 2017, Art. no. 784.

[14] F. Cai *et al.*, "A fully integrated reprogrammable memristor–CMOS system for efficient multiply–accumulate operations," *Nature Electron.*, vol. 2, no. 7, pp. 290–299, Jul. 2019.

[15] S. Choi *et al.*, "SiGe epitaxial memory for neuromorphic computing with reproducible high performance based on engineered dislocations," *Nature Mater.*, vol. 17, no. 4, Apr. 2018, Art. no. 335.

[16] S. Chen *et al.*, "Wafer-scale integration of two-dimensional materials in high-density memristive crossbar arrays for artificial neural networks," *Nature Electron.*, vol. 3, no. 10, pp. 638–645, Oct. 2020.

[17] M. Bavandpour, M. R. Mahmoodi, and D. B. Strukov, "ACortex: An energy-efficient multipurpose mixed-signal inference accelerator," *IEEE J. Explor. Solid-State Computat. Devices Circuits*, vol. 6, no. 1, pp. 98–106, Jun. 2020.

[18] H. Nili, A. F. Vincent, M. Prezesio, M. R. Mahmoodi, I. Kataeva, and D. B. Strukov, "Comprehensive compact phenomenological modeling of integrated metal-oxide memristors," *IEEE Trans. Nanotechnol.*, vol. 19, pp. 344–349, 2020.

[19] M. Prezioso *et al.*, "Spike-timing-dependent plasticity learning of coincidence detection with passively integrated memristive circuits," *Nature Commun.*, vol. 9, no. 1, Dec. 2018, Art. no. 5311.

[20] D. B. Strukov and R. S. Williams, "Exponential ionic drift: Fast switching and low volatility of thin-film memristors," *Appl. Phys. A, Solids Surf.*, vol. 94, no. 3, pp. 515–519, Mar. 2009.

[21] R. Waser, R. Dittmann, G. Staikov, and K. Szot, "Redox-based resistive switching memories–nanoionic mechanisms, prospects, and challenges," *Adv. Mater.*, vol. 21, nos. 25–26, pp. 2632–2663, Jul. 2009.

[22] M. R. Mahmoodi, A. F. Vincent, H. Nili, and D. B. Strukov, "Intrinsic bounds for computing precision in memristor-based vector-by-matrix multipliers," *IEEE Trans. Nanotechnol.*, vol. 19, pp. 429–435, 2020.

[23] M. Klachko, M. R. Mahmoodi, and D. Strukov, "Improving noise tolerance of mixed-signal neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Budapest, Hungary, Jul. 2019, pp. 1–8.

[24] M. R. Mahmoodi, M. Prezioso, and D. B. Strukov, "Versatile stochastic dot product circuits based on nonvolatile memories for high performance neurocomputing and neurooptimization," *Nature Commun.*, vol. 10, no. 1, pp. 1–10, Dec. 2019.

[25] K. G. Young-Fisher *et al.*, "Leakage current-forming voltage relation and oxygen gettering in HfO_x RRAM devices," *IEEE Electron Device Lett.*, vol. 34, no. 6, pp. 750–752, Jun. 2013.

[26] Z. Fahimi, M. R. Mahmoodi, M. Klachko, H. Nili, H. Kim, and D. B. Strukov, "Mitigating imperfections in mixed-signal neuromorphic circuits," 2021, *arXiv:2107.04236*. [Online]. Available: https://arxiv.org/abs/2107.04236

[27] O. Krestinskaya, A. Irmanova, and A. P. James, "Memristive non-idealities: Is there any practical implications for designing neural network chips?" in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2019, pp. 1–5.

[28] Q. Wang, X. Wang, S. H. Lee, F.-H. Meng, and W. D. Lu, "A deep neural network accelerator based on tiled RRAM architecture," in *IEDM Tech. Dig.*, Dec. 2019, pp. 14.4.1–14.4.4.

[29] A. Mehonic, D. Joksas, W. H. Ng, M. Buckwell, and A. J. Kenyon, "Simulation of inference accuracy using realistic RRAM devices," *Frontiers Neurosci.*, vol. 13, p. 593, Jun. 2019.

[30] X. Peng, S. Huang, H. Jiang, A. Lu, and S. Yu, "DNN+NeuroSim V2.0: An end-to-end benchmarking framework for compute-in-memory accelerators for on-chip training," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, early access, Dec. 14, 2020, doi: 10.1109/TCAD.2020.3043731.

[31] C. Lammie, W. Xiang, B. Linares-Barranco, and M. R. Azghadi, "MemTorch: An open-source simulation framework for memristive deep learning systems," Apr. 2020, *arXiv:2004.10971*. [Online]. Available: http://arxiv.org/abs/2004.10971

[32] G. Yuan *et al.*, "An ultra-efficient memristor-based DNN framework with structured weight pruning and quantization using ADMM," in *Proc. IEEE/ACM Int. Symp. Low Power Electron. Design (ISLPED)*, Jul. 2019, pp. 1–6.

[33] R. H. Byrd, R. B. Schnabel, and G. A. Shultz, "A trust region algorithm for nonlinearly constrained optimization," *SIAM J. Numer. Anal.*, vol. 24, no. 5, pp. 1152–1170, 1987.

[34] F. M. Bayat, B. Hoskins, and D. B. Strukov, "Phenomenological modeling of memristive devices," *Appl. Phys. A, Solids Surf.*, vol. 118, pp. 779–786, Jan. 2015.

[35] 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)*, Jul. 2015, pp. 1–8.

[36] Z. Chai, "Impact of RTN on pattern recognition accuracy of RRAM-based synaptic neural network," *IEEE Electron Device Lett.*, vol. 39, no. 11, pp. 1652–1655, Nov. 2018.

[37] S. Agarwal, R. L. Schiek, and M. J. Marinella, "Compensating for parasitic voltage drops in resistive memory arrays," in *Proc. IEEE Int. Memory Workshop (IMW)*, May 2017, pp. 1–4.

[38] A. Chen, "A comprehensive crossbar array model with solutions for line resistance and nonlinear device characteristics," *IEEE Trans. Electron Devices*, vol. 60, no. 4, pp. 1318–1326, Apr. 2013.

[39] D. Joksas *et al.*, "Committee machines—A universal method to deal with non-idealities in memristor-based neural networks," *Nature Commun.*, vol. 11, no. 1, pp. 1–10, Dec. 2020.

[40] J. Kang *et al.*, "Time-dependent variability in RRAM-based analog neuromorphic system for pattern recognition," in *IEDM Tech. Dig.*, San Francisco, CA, USA, Dec. 2017, pp. 6.4.1–6.4.4.

[41] L. Xia *et al.*, "Stuck-at fault tolerance in RRAM computing systems," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 8, no. 1, pp. 102–115, Mar. 2018.

[42] V. Joshi *et al.*, "Accurate deep neural network inference using computational phase-change memory," *Nature Commun.*, vol. 11, no. 1, pp. 1–13, Dec. 2020.

[43] A. Mohanty, X. Du, P.-Y. Chen, J.-S. Seo, S. Yu, and Y. Cao, "Random sparse adaptation for accurate inference with inaccurate multi-level RRAM arrays," in *IEDM Tech. Dig.*, Dec. 2017, pp. 6.3.1–6.3.4.

[44] B. Liu, H. Li, Y. Chen, X. Li, Q. Wu, and T. Huang, "Vortex: Variation-aware training for memristor X-bar," in *Proc. 52nd ACM/EDAC/IEEE Annu. Design Autom. Conf.*, San Francisco, CA, USA, Jun. 2015, pp. 1–6.

[45] L. Chen *et al.*, "Accelerator-friendly neural-network training: Learning variations and defects in RRAM crossbar," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Lausanne, Switzerland, Mar. 2017, pp. 19–24.

[46] C. Liu, M. Hu, J. P. Strachan, and H. Li, "Rescuing memristor-based neuromorphic design with high defects," in *Proc. 54th Annu. Design Autom. Conf.*, Austin, TX, USA, Jun. 2017, pp. 1–6.

[47] T. Gokmen, M. J. Rasch, and W. Haensch, "The marriage of training and inference for scaled deep learning analog hardware," in *IEDM Tech. Dig.*, San Francisco, CA, USA, Dec. 2019, pp. 22.3.1–22.3.4.

[48] F. M. Bayat, M. Prezioso, B. Chakrabarti, I. Kataeva, and D. Strukov, "Memristor-based perceptron classifier: Increasing complexity and coping with imperfect hardware," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Irvine, CA, USA, Nov. 2017, pp. 549–554.

[49] P.-Y. Chen and S. Yu, "Reliability perspective of resistive synaptic devices on the neuromorphic system performance," in *Proc. IEEE Int. Rel. Phys. Symp. (IRPS)*, Mar. 2018, pp. 5C.4-1–5C.4-4, doi: 10.1109/irps.2018.8353615.

[50] C. Lammie, M. R. Azghadi, and D. Ielmini, "Empirical metal-oxide RRAM device endurance and retention model for deep learning simulations," *Semicond. Sci. Technol.*, vol. 36, no. 6, Apr. 2021, Art. no. 065003, doi: 10.1088/1361-6641/abf29d.