Optimized programming algorithms for multilevel RRAM in hardware neural networks

Valerio Milo[1]*, Francesco Anzalone[1], Cristian Zambelli[2], Eduardo Pérez[3], Mamathamba K. Mahadevaiah[3], Óscar G. Ossorio[4], Piero Olivo[2], Christian Wenger[3,5], and Daniele Ielmini[1]

[1]Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano and IU.NET, Piazza Leonardo da Vinci 32, 20133 Milan, Italy, *e-mail: valerio.milo@polimi.it
[2]Dipartimento di Ingegneria, Università degli Studi di Ferrara, Via Giuseppe Saragat 1, 44122 Ferrara, Italy
[3]IHP-Leibniz-Institut für innovative Mikroelektronik, Im Technologiepark 25, 15236 Frankfurt (Oder), Germany
[4]Departamento de Electricidad y Electrónica, Universidad de Valladolid, Paseo de Belén 15, 47011 Valladolid, Spain
[5]BTU Cottbus-Senftenberg, 01968 Cottbus, Germany

2021 IEEE International Reliability Physics Symposium (IRPS) | 978-1-7281-6893-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/IRPS46558.2021.9405119

***Abstract*—A key requirement for RRAM in neural network accelerators with a large number of synaptic parameters is the multilevel programming. This is hindered by resistance imprecision due to cycle-to-cycle and device-to-device variations. Here, we compare two multilevel programming algorithms to minimize resistance variations in a 4-kbit array of HfO_2 RRAM. We show that gate-based algorithms have the highest reliability. The optimized scheme is used to implement a neural network with 9-level weights, achieving 91.5% (vs. software 93.27%) in MNIST recognition.**

***Index Terms* – Resistive-switching random access memory (RRAM); multilevel programming; resistance variability; weight quantization; hardware neural networks; in-memory computing.**

I. Introduction

Artificial intelligence (AI) has recently driven excellent achievements in various emerging application tasks such as image recognition, natural language processing, and desease recognition by massive training of deep neural networks (DNN) in the cloud [1]. However, the extensive use of central processing units (CPU) and graphics processing units (GPU) with von Neumann architecture for cloud computing requires a significant amount of energy and time due to the intensive data transfer between memory and computing units [2]. These energy and latency issues of AI in conventional systems represent a crucial issue for the transition from the cloud to the edge, namely in battery-powered portable devices such as smartphones, wearable sensors and internet of things (IoT).

A promising approach to improve the energy efficiency of AI training and inference is in-memory computing (IMC) with novel memory devices such as resistive-switching random access memory (RRAM) and phase change memory (PCM) [3], [4]. In fact, RRAM and PCM memory devices can perform physical computation where the data storage takes place by the Ohm's law and Kirchhoff's law, thus leading to an efficient execution of the matrix-vector multiplication (MVM) in DNNs in terms of area consumption, speed, and energy [5]. Also, the rich portfolio of the properties of these devices includes the multilevel cell (MLC) operation [6]–[8], which is a key feature for achieving analog synaptic weights enabling DNN training/inference with software-equivalent performance in hardware [9].

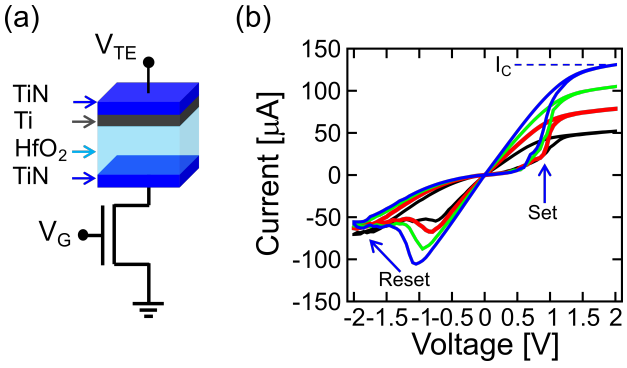

Fig. 1. (a) Sketch of TiN/Ti/HfO_2/TiN RRAM cell with 1T1R structure. The n-channel MOS connected in series to the RRAM is used to select the RRAM device and limit the maximum current flowing during the set transition. (b) Median $I-V$ characteristics of 1T1R RRAM device for increasing V_G-controlled compliance current I_C indicating well-controllable MLC operation in the 1T1R device.

While these major advantages have been explored in neural network arrays in recent years [10]–[13], both device-to-device (D2D) and cycle-to-cycle (C2C) variations of resistance raise strong concerns in achieving high accuracy for weight programming in hardware arrays [14]. Also, the limited number of programmable weight levels hinders the achievement of the software-equivalent training/inference performance [15]. It appears that the co-optimization of weight quantization schemes and MLC programming algorithms used to map the software weights into the hardware arrays is a key requirement to minimize the accuracy loss with respect to the DNNs trained in software.

In this work, we study two MLC approaches, namely (i) incremental step pulse with verify algorithm (ISPVA) [16] and (ii) incremental gate voltage with verify algorithm (IGVVA) based on 100 mV (IGVVA-100) and 10 mV (IGVVA-10) voltage steps. Based on data for 5-level programming of a 4-

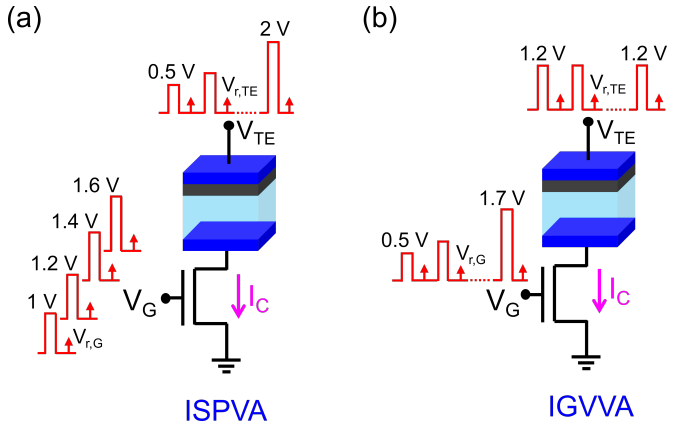

Fig. 2. Schematic of MLC programming by ISPVA and IGVVA. (a) In ISPVA, the LRS levels were programmed by application of set pulses at increasing V_{TE} from 0.5 V to 2 V with 100 mV voltage step, keeping V_G fixed at 1 V, 1.2 V, 1.4 V, and 1.6 V, respectively. (b) In IGVVA, the LRS levels were programmed by tuning V_G from 0.5 V to 1.7 V, keeping V_{TE} fixed at 1.2 V. IGVVA was experimentally investigated using two V_G steps equal to 100 mV and 10 mV, respectively. The read operation was performed for both algorithms by applying a gate voltage $V_{r,G}$ = 1.7 V and a TE voltage $V_{r,TE}$ = 0.2 V.

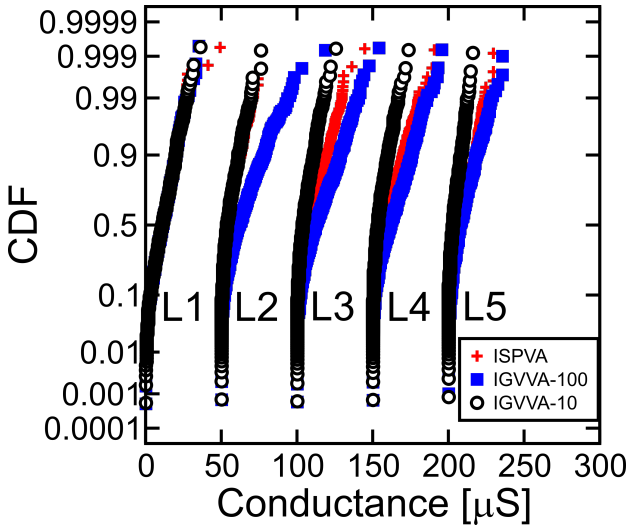

Fig. 3. CDFs of 5 conductance levels programmed by ISPVA, IGVVA-100, and IGVVA-10. IGVVA-10 CDFs show higher accuracy than ISPVA CDFs for all the levels, which in turn display lower variability than IGVVA-100 CDFs.

kbit HfO_2 RRAM array, IGVVA-10 shows the best accuracy, which is explained by the programming characteristics of the one-transistor/one-resistor (1T1R) RRAM cell. The optimized 5-level algorithm is used to program 4-kbit synaptic weights of a 2-layer neural network. We achieve a high inference accuracy namely 91.5% compared to 93.27% software accuracy in MNIST recognition as a result of the co-optimization of quantized-weight training scheme and MLC programming algorithm.

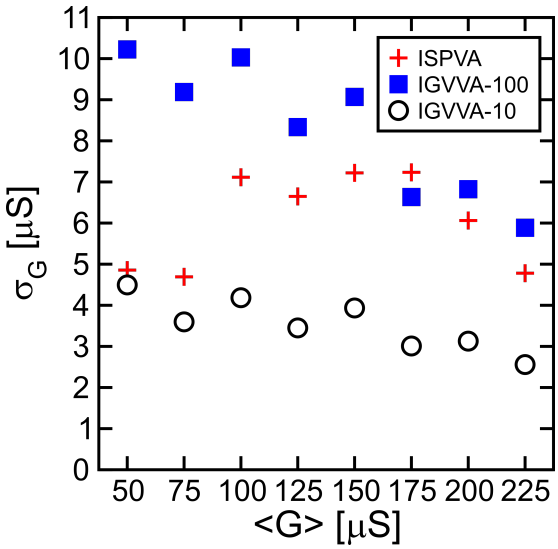

Fig. 4. Standard deviation of CDFs in Fig. 3 for increasing median conductance <G>. The levels programmed by IGVVA-10 exhibit the lowest variation.

II. Device Structure and MLC Operation

Fig. 1(a) shows the 1T1R structure of our RRAM device used in a 64x64 array of 4-kbit. The RRAM is based on a stack including a TiN top electrode (TE), a Ti oxygen scavenger layer, an amorphous HfO_2 switching layer, and a TiN bottom electrode (BE). Also, the RRAM device is connected in series with a n-channel MOS manufactured in 0.25 μm CMOS technology, serving as both select device and current limiter by a compliance current I_C [17]. Fig. 1(b) shows the measured current-voltage ($I-V$) characteristics of RRAM for increasing I_C, exhibiting an abrupt set transition at positive voltages from the high resistance state (HRS) to the low resistance state (LRS), and a more gradual reset transition at negative voltages from the LRS to the HRS. These $I-V$ curves indicate that the 1T1R HfO_2 RRAM device provides a well-controllable MLC operation by tuning I_C via the gate voltage V_G [18].

To map 5 levels in the 1T1R RRAM cell, two programming strategies referred to as ISPVA and IGVVA were designed and experimentally tested at device level. Fig. 2(a) shows the ISPVA for MLC programming. In ISPVA, set pulses at increasing TE voltage V_{TE} were applied to the HRS or L1 from 0.5 V to 2 V with 100 mV step and pulse width t_{pulse} = 1 μs [16]. V_G was kept constant to control I_C depending on the target level of the LRS. In particular, I_C is targeted to 10 μA, 20 μA, 30 μA and 40 μA for L2, L3, L4 and L5, respectively. Fig. 2(b) shows the IGVVA, where V_{TE} was kept constant to 1.2 V (larger than the set voltage V_{set} = 0.9 V) while V_G was increased step-by-step from 0.5 V to 1.7 V until the desired LRS was reached. In particular, two values of the voltage step ΔV_G were used in IGVVA, namely ΔV_G = 100 mV (named IGVVA-100) and 10 mV (IGVVA-10). Note that in both MLC schemes any programming pulse at gate terminal and TE is followed by a read pulse with

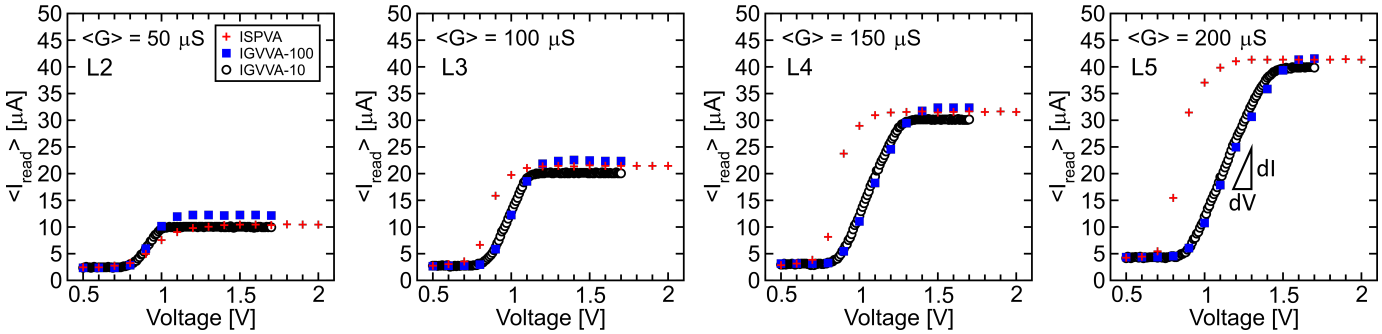

Fig. 5. Median $I-V$ characteristics of 4 LRS levels (L2∼L5) programmed by ISPVA, IGVVA-100, and IGVVA-10. The abrupt set transitions obtained by the application of ISPVA suggest that ISPVA is not suitable to finely tune the level conductance. On the other hand, the smoother conductance increase achieved by IGVVA-100 and IGVVA-10 indicates that the current modulation by V_G is more precise than the modulation by V_{TE} used in ISPVA. IGVVA-10 provides a more accurate programming than IGVVA-100 as a result of its finer voltage step.

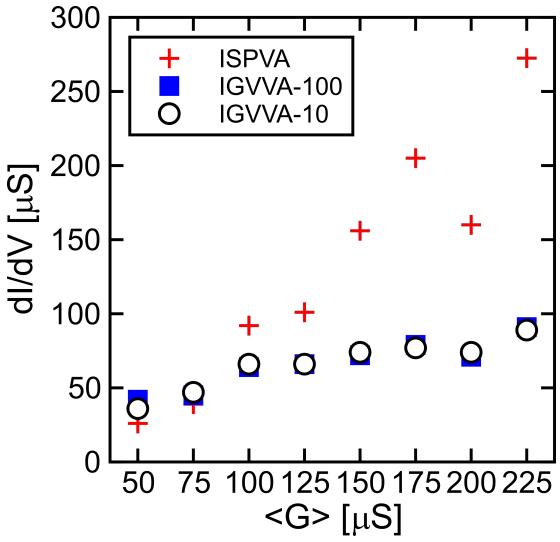

Fig. 6. Maximum slope of $I-V$ characteristics in Fig. 5 as a function of the median conductance <G>, which supports the increasing abruptness of set transition for increasing level in ISPVA compared with the smoother conductance increase in IGVVA-100 and IGVVA-10.

amplitude $V_{r,G}$ = 1.7 V and $V_{r,TE}$ = 0.2 V, respectively, to verify the programmed device state.

Fig. 3 shows the cumulative distributions (CDFs) of device conductances for ISPVA, IGVVA-100, and IGVVA-10 programming of the 4 kbit RRAM array. Among the 5 levels L1∼L5 read at 0.2 V, IGVVA-100 exhibits a larger variability, expecially for L2, followed by ISPVA and IGVVA-10. The latter is the most accurate algorithm in that it provides CDFs with the minimum D2D variability. Note that we programmed each LRS level using a quarter of the total array. To better assess the D2D variations of 5 levels, Fig. 4 shows the standard deviation σ_G of the CDFs as a function of the median conductance <G> for the various algorithms. While σ_G for ISPVA increases at low <G> and decreases at high <G>, IGVVA-100 and IGVVA-10 show a gradual decreasing behavior of σ_G for increasing levels. In particular, IGVVA-100, which has the highest variability, becomes closer to ISPVA from the intermediate levels. On the other hand, IGVVA-10 shows the smallest σ_G for all the levels, achieving a minimum D2D variation slightly larger than 2 μS.

To understand the dependence of D2D variation on the algorithm, Fig. 5 shows the median programming characteristics of levels L2∼L5 for ISPVA, IGVVA-100 and IGVVA-10. ISPVA generally shows an abrupt set transition in correspondence of V_{set}, which makes it difficult to finely tune the conductance. On the other hand, IGVVA shows a smoother increase of conductance thanks to the tight control of I_C by V_G tuning. Additionally, IGVVA-10 shows the best accuracy thanks to the finer ΔV_G allowing a more accurate I_C modulation. In particular, note that L2 with <G> = 50 μS shows the ISPVA curve slightly smoother than IGVVA curves. This behavior results from the low V_G of 1 V applied to set the I_C to 10 μA, which hinders the set transition in RRAM device until the voltage across the RRAM becomes larger enough than the voltage across the less conductive MOS according to the voltage divider rule.

To gain more insight about the algorithm programming accuracy, Fig. 6 displays the slope dI/dV of median characteristics for ISPVA, IGVVA-100 and IGVVA-10 as a function of the programmed level. In agreement with Fig. 5, ISPVA exhibits a large slope with linear increase for increasing level whereas IGVVA-100 and IGVVA-10 show a smaller slope with negligible dependence on ΔV_G. Based on the dI/dV and σ_G behavior for each algorithm, clearly IGVVA-10 appears as the best algorithm for MLC programming in the 4-kbit RRAM array.

III. Neural Network Implementation

Levels L2∼L5 programmed by IGVVA-10 in Figs. 3-6 were used to implement the 2-layer fully-connected neural network of Fig. 7(a). The network, which consists of 197 input neurons, 20 hidden neurons, and 10 output neurons, was developed to implement the recognition task on the handwritten digit images of Modified National Institute of Standards and Technology (MNIST) dataset. In particular, we trained and tested the network using MNIST images of size 14x14 to ensure that the number of weights was consistent with the array size.

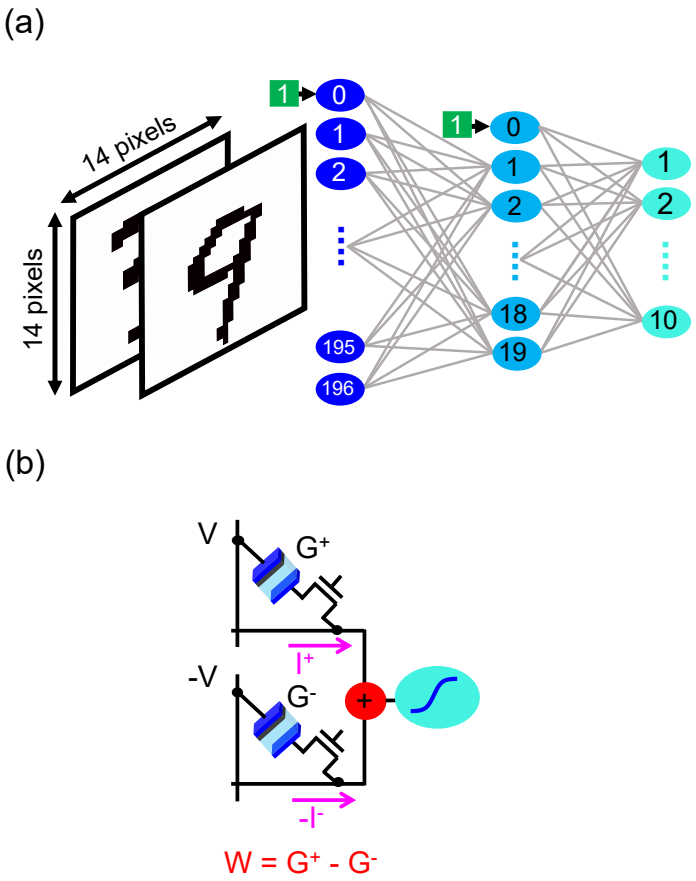

Fig. 7. (a) Sketch of 2-layer fully-connected neural network implemented for MNIST recognition task. (b) Schematic of encoding scheme used to map the weights of the neural network into the RRAM array. Each weight W is achieved by a pair of 1T1R RRAM devices as difference of their conductances G^+ and G^-, respectively.

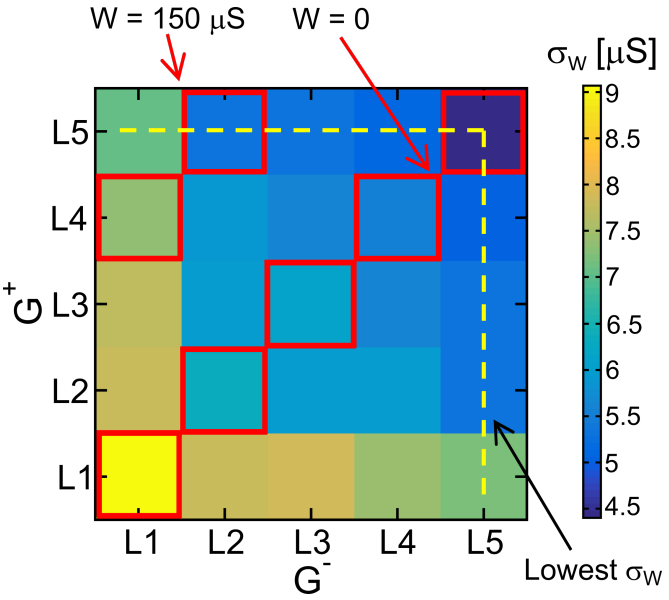

Fig. 8. Color plot of standard deviation σ_W of 9 differential weights obtained by the difference of the 5 IGVVA-10 conductance CDFs in Fig. 3. Among all the combinations for each weight, the best configuration is the one exhibiting the lowest σ_W (dashed lines).

The weights of the neural network were initially trained on the 60000 images of MNIST train dataset with 64-bit floating point (FP-64) precision by application of the backpropagation rule [19]. After the off-line training in software, the weights were reduced to 9 discrete levels by using an incremental quantization procedure similar to the one proposed in [20]. This method consists of the iterative execution of 3 operations, namely (i) weight partition, (ii) group-wise quantization, and (iii) re-training. Weight partition equally divides the weights of each fully-connected layer into two groups based on the error due to the level quantization. The first group with larger quantization error is subject to quantization to 9 levels, while the second group with the remaining weights is re-trained by backpropagation rule to compensate for the accuracy loss caused by weight quantization. Note that the re-training operation is performed by keeping the quantized weights of the first group fixed. By repeating this procedure on the re-trained FP-valued weights, we gradually implemented an incremental quantization of all the neural network weights, thus achieving a neural network with 9-level discrete weights.

To map the 9-level weights into the hardware array, we adopted the conventional differential scheme [10] shown in Fig. 7(b). Each weight W was mapped into a differential pair of 1T1R RRAM devices, where the sum of positive and negative currents activated by read voltages with opposite polarity, respectively, results in a current proportional to $W = G^+ - G^-$. Then, this current is fed as input of a neuron of the next layer which converts it into a voltage by application of a sigmoid activation function.

The 9 discrete levels of the weights W were obtained as differences among the 5 conductance levels in Fig. 3. To optimize the accuracy of 9 differential conductance weights within our neural network, we assumed device programming by IGVVA-10 with the corresponding σ_G in Fig. 4. The color plot in Fig. 8 shows the standard deviation of W, σ_W, of all the possible configurations to obtain a certain $W = G^+ - G^-$, *e.g.* W = 0 can be obtained from $G^+ = G^-$ corresponding to the five cases along the diagonal while W = 150 μS can be obtained as either L4-L1 or L5-L2. Thanks to the low σ_G of L5 reported in Fig. 4, the lowest σ_W is found along the upper row (G^+ = L5) and the rightmost column (G^- = L5). Note that this weight mapping approach minimizes imprecision, however it is not optimal from the viewpoint of the current minimization. To mitigate the current-induced IR drop in the array, the interconnect wire resistances should be much lower than the device resistances [21].

These optimized combinations were selected for implementing the 9 differential weights of the neural network for MNIST recognition, whose CDFs and corresponding σ_W are shown in Fig. 9(a) and Fig. 9(b), respectively. According to the color plot in Fig. 8, the level CDFs for W = $\pm$ 200 μS show the highest variability as they can be only obtained by the difference between L5 and L1 which is the level with the

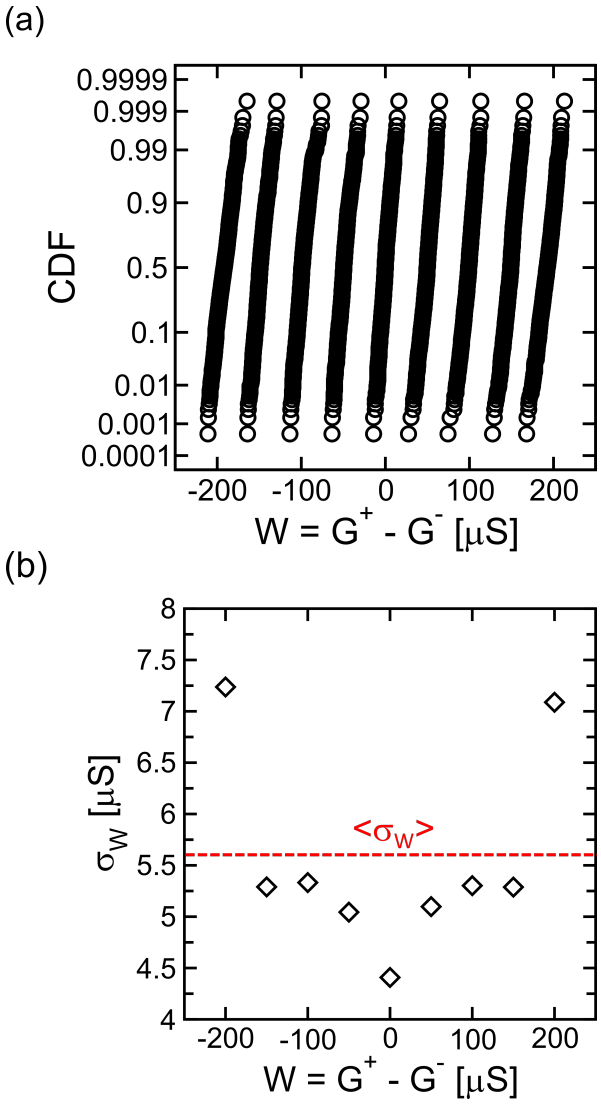

Fig. 9. (a) CDFs of 9 differential weights calculated by the 5 IGVVA-10 conductance CDFs by using the best weight configuration with the lowest σ_W shown in Fig. 8. (b) Standard deviation σ_W of 9 differential weight CDFs in (a) as a function of the weight conductance W. σ_W is minimum for W = 0 because it is obtained by the difference of L5 CDFs exhibiting the smallest σ_G as shown in Fig. 4.

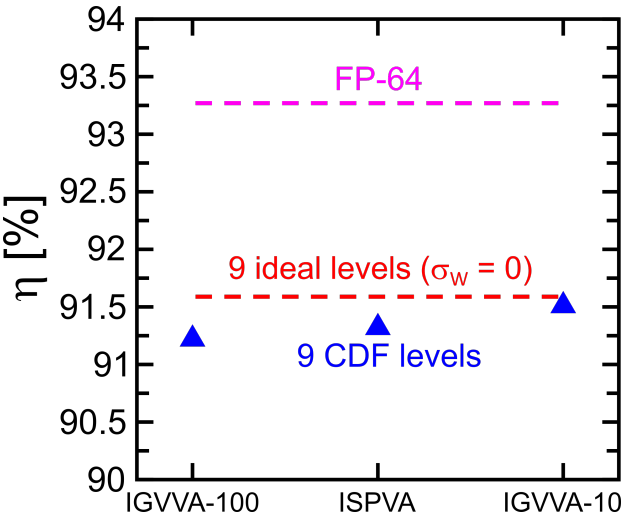

Fig. 10. Testing accuracy η on MNIST test dataset using 9 weight levels implemented by 3 programming algorithms. IGVVA-10 precision is comparable with 9 ideal levels and FP-64 performance.

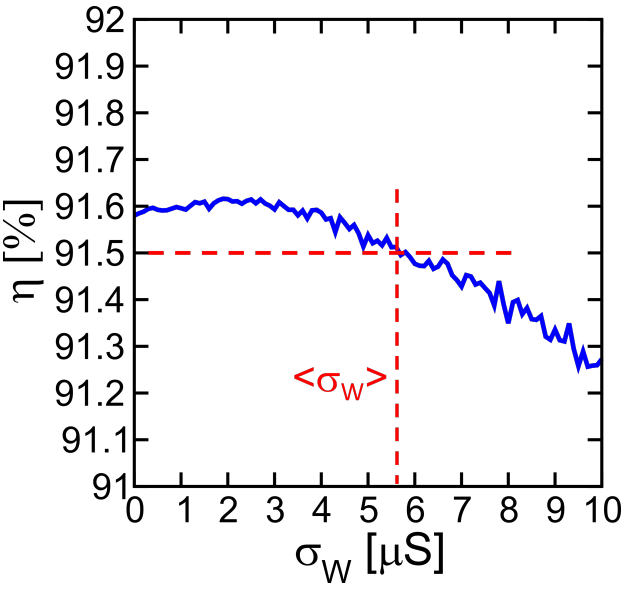

Fig. 11. Testing accuracy η for increasing σ_W of 9 differential weights obtained by experimental IGVVA-10 CDFs. Note that η decreases from 91.6% to 91.5% indicated in Fig. 10 as σ_W reaches $<\sigma_W>$ shown in Fig. 9(b).

highest σ_G. On the contrary, the central level W = 0 displays the minimum σ_W since it is achieved by the difference of L5 levels with minimum σ_G.

IV. Neural Network Accuracy

After achieving the neural network with 9 differential conductance weights using IGVVA-10, we tested the accuracy of the neural network on MNIST classification task by presentation of the 10000 handwritten digits of MNIST test dataset.

Fig. 10 shows the testing accuracy η of MNIST recognition for the 2-layer fully-connected neural network with the differential weights of Fig. 9(a). The inference accuracy achieved by IGVVA-10 conductance weight levels ($\eta = 91.5\%$) well compares with the software neural network with FP-64 weights ($\eta = 93.27\%$) and after application of the incremental quantization scheme ($\eta = 91.6\%$). A similar optimization as in Fig. 8 was carried out for ISPVA and IGVVA-100, which show a lower accuracy. Note that the impact of additional non-idealities such as device non-linearity and interconnect wire resistances was not addressed in the array-level network calculations.

Fig. 11 shows the calculated η as a function of σ_W, confirming the accuracy loss with increasing variation under the assumption that all the 9 differential weights are affected by the same amount of D2D variability. Note that the inference accuracy reaches the best calculated performance indicated

in Fig. 10 as σ_W approximates the median of σ_W corresponding to the differential weight CDFs of Fig. 9(a). Also, note that even assuming σ_W = 10 μS, further accuracy drop from no variation (about 1.67% from FP-64 representation) to with variation is only 0.35%. These results support the co-optimization of quantized-weight training and IGVVA-10 as valuable tool to achieve accurate RRAM-based neural networks.

V. Conclusions

We studied the MLC programming algorithms of RRAM for neural network accelerators. IGVVA-10 shows the lowest variability thanks to the smoother programming characteristics. A 2-layer neural network is implemented in a 4-kbit RRAM array with 5 levels programmed with IGVVA-10, showing 91.5% inference accuracy comparable to software accuracy. The results support the quantized-weight training scheme with IGVVA-10 and variation optimization for accurate RRAM-based neural networks.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[2] G. Indiveri and S.-C. Liu, "Memory and information processing in neuromorphic systems," *Proc. IEEE*, vol. 103, no. 8, pp. 1379–1397, 2015.

[3] D. Ielmini and H.-S. P. Wong, "In-memory computing with resistive switching devices," *Nat. Electron.*, vol. 1, pp. 333–343, 2018.

[4] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, "Memory devices and applications for in-memory computing," *Nat. Nanotechnol.*, vol. 15, pp. 529–544, 2020.

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

[6] S. Balatti, S. Larentis, D. Gilmer, and D. Ielmini, "Multiple memory states in resistive switching devices through controlled size and orientation of the conductive filament," *Adv. Mater.*, vol. 25, pp. 1474–1478, 2013.

[7] L. Zhao, H.-Y. Chen, S.-C. Wu, Z. Jiang, S. Yu, T.-H. Hou, H.-S. P. Wong, and Y. Nishi, "Multi-level control of conductive nano-filament evolution in HfO_2 ReRAM by pulse-train operations," *Nanoscale*, vol. 6, pp. 5698–5702, 2014.

[8] A. Athmanathan, M. Stanisavljevic, N. Papandreou, H. Pozidis, and E. Eleftheriou, "Multilevel-cell phase-change memory: a viable technology," *IEEE Journal of Emerging Topics on Circuits and Systems (JETCAS)*, vol. 6, no. 1, pp. 87–100, 2016.

[9] S. Ambrogio, P. Narayanan, H. Tsai, R. M. Shelby, I. Boybat, C. di Nolfo, S. Sidler, M. Giordano, M. Bodini, N. C. P. Farinha, B. Killeen, C. Cheng, Y. Jaoudi, and G. W. Burr, "Equivalent-accuracy accelerated neural-network training using analogue memory," *Nature*, vol. 558, no. 7708, pp. 60–67, 2018.

[10] G. W. Burr, R. M. Shelby, C. di Nolfo, J. W. Jang, R. S. Shenoy, and P. Narayanan, "Experimental demonstration and tolerancing of a large-scale neural network (165,000 synapses), using phase-change memory as the synaptic weight element," *IEEE IEDM Tech. Dig.*, pp. 697–700, 2014.

[11] P. Yao, H. Wu, B. Gao, S. B. Eryilmaz, X. Huang, W. Zhang, Q. Zhang, N. Deng, L. Shi, H.-S. P. Wong, and H. Qian, "Face classification using electronic synapses," *Nat. Commun.*, vol. 8, p. 15199, 2017.

[12] C. Li, D. Belkin, Y. Li, P. Yan, M. Hu, N. Ge, H. Jiang, E. Montgomery, P. Lin, Z. Wang, W. Song, J. P. Strachan, M. Barnell, Q. Wu, R. S. Williams, J. J. Yang, and Q. Xia, "Efficient and self-adaptive in-situ learning in multilayer memristor neural networks," *Nat. Commun.*, vol. 9, p. 2385, 2018.

[13] C. Li, M. Hu, Y. Li, H. Jiang, N. Ge, E. Montgomery, J. Zhang, W. Song, N. Davila, C. E. Graves, Z. Li, J. P. Strachan, P. Lin, Z. Wang, M. Barnell, Q. Wu, R. S. Williams, J. J. Yang, and Q. Xia, "Analogue signal and image processing with large memristor crossbars," *Nat. Electron.*, vol. 1, pp. 52–59, 2018.

[14] S. Yu, "Neuro-inspired computing with emerging nonvolatile memory," *Proc. IEEE*, vol. 106, no. 2, pp. 260–285, 2018.

[15] T. Gokmen and Y. Vlasov, "Acceleration of deep neural network training with resistive cross-point devices: Design considerations," *Front. Neurosci.*, vol. 10, p. 333, 2016.

[16] E. Pérez, A. Grossi, C. Zambelli, P. Olivo, R. Roelofs, and Ch. Wenger, "Reduction of the cell-to-cell variability in $Hf_{1-x}Al_xO_y$ based RRAM arrays by using program algorithms," *IEEE Electron Device Lett.*, vol. 38, no. 2, pp. 175–178, 2017.

[17] V. Milo, C. Zambelli, P. Olivo, E. Pérez, M. K. Mahadevaiah, O. G. Ossorio, Ch. Wenger, and D. Ielmini, "Multilevel HfO_2-based RRAM devices for low-power neuromorphic networks," *APL Materials*, vol. 7, p. 081120, 2019.

[18] D. Ielmini, "Modeling the universal set/reset characteristics of bipolar RRAM by field- and temperature-driven filament growth," *IEEE Trans. Electron Devices*, vol. 58, no. 12, pp. 4309–4317, 2011.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[20] A. Zhou, A. Yao, Y. Guo, L. Xu, and Y. Chen, "Incremental network quantization: towards lossless CNNs with low-precision weights," *arXiv:1702.03044*, 2017.

[21] D. Ielmini and G. Pedretti, "Device and circuit architectures for in-memory computing," *Adv. Intell. Syst.*, vol. 2, no. 7, p. 2000040, 2020.