Neuron Deactivation Scheme for Defect-Tolerant Memristor Neural Networks

Seokjin Oh, Jiyong An, and Kyeong-Sik Min
School of Electrical Engineering, Kookmin University, Seoul, Korea
E-mail) mks@kookmin.ac.kr

Abstract— **As amounts of data generated from countless and ubiquitous IoT sensors are increased very sharply, memristor crossbars can be considered very suitable to edge intelligence hardware due to high energy efficiency of computing, dense and 3D integration, non-volatility, multi-state memory, CMOS compatible fabrication etc. But, due to the limits of immature fabrication technology, the memristor crossbars can have defects such as stuck-at-faults. To compensate for malfunction of neural networks caused from the fabrication-related defects, in this paper, a simple neuron deactivation scheme is reviewed and analyzed for maximizing its capability to compensate for the neural network's performance degradation due to the memristor defects. The column deactivation scheme can be particularly useful for the edge intelligence hardware, because the defect map occupying a large amount of memory is not needed during the training. Moreover, the direct mapping from the calculated synaptic weights to the memristor crossbar can save the re-training time required for the defect-aware training scheme.**

Keywords- neuron deactivation scheme, defect-tolerant memristor neural networks, defective memristor crossbars, memristor defects, edge intelligence

I. INTRODUCTION

Traditional CMOS-based computing platforms using Von Neumann architecture such as CPUs, GPUs, etc. are no longer able to handle huge amounts of real-time unstructured data obtained ubiquitously from countless Internet of Things (IoT) devices and sensors [1]. The bottleneck of Von Neumann architecture is caused from that memory units are separated from computing ones in the architecture. For processing the large amounts of unstructured data from IoT sensors, the computing units should access the memory ones very frequently, resulting in increasing the power consumption of memory access to an unacceptably high level in the Von Neumann computing circuits.

To overcome the memory bottleneck of the Von Neumann architecture, emerging computing circuits such as memristor crossbars have been suggested as a new hardware solution for realizing energy-efficient next-generation computing systems, which can be suitable to Artificial Intelligence (AI) applications [2], [3].

Specifically, Vector Matrix Multiplication (VMM) can be achieved by multiplying vectors and matrices using the memristor's voltage-current relationship based on Ohm's law [3]–[5]. If the VMM calculation is processed on the memristor crossbars, the multiplication performance can be enhanced easily by adding extra columns in parallel to the crossbars. In addition to the parallelism, the memristor crossbars can be nonvolatile, stacked layer by layer, used to store multi-states, and fabricated with CMOS devices. All these features make the memristor crossbars suitable to various applications of neural networks processing huge amounts of unstructured data.

Unfortunately, however, due to the limitation of immature fabrication technology, a manufacturing yield of memristor crossbars is still not high enough for implementing next generation computing systems based on emerging computing devices such as memristors. Despite providing higher computing performance and energy efficiency than traditional CMOS computing systems, the memristor crossbars have a variety of fabrication-related difficulties, such as stuck-at-fault defects and variations.

Fig. 1 shows schematics of ideal crossbar and real crossbar. Here the ideal crossbar is assumed no defects. On the contrary, defects such as stuck-at-faults can be found in the real crossbar. The faulty cells get stuck in the High Resistance State (HRS) or the Low Resistance State (LRS) in the real crossbar. The black dot and empty circle represent the normal LRS cell and HRS one, respectively, in the ideal crossbar. In Fig. 1b, the red dot and empty circle represent the stuck-at-LRS fault and struck-at-HRS one, respectively. The stuck-at-fault cells cannot be programmed because their resistance states get stuck at HRS or LRS. In Fig. 1, I_{1+} and I_{1-} represent plus column current and minus one of Column #1, respectively. The both plus and minus columns are needed to calculate the positive and negative synaptic weights. The plus current and minus one of Column #1, I_{1+} and I_{1-} are delivered to the hidden neuron Y_1. In this paper, ternary synaptic weights of -1, +1, and 0 are used in the neural networks.

Now let's consider how faulty memristor devices may affect the neural network's performance implemented with defective memristor crossbars. Unfortunately, even a single memristor defect can have a significant influence on the VMM calculation, since the stuck-at-LRS faults in Fig. 1 can dramatically increase the column current in the crossbar. The training and inference accuracy of neural networks implemented with the defective memristor crossbars may be lowered drastically due to erroneous neuron activation caused by the defective cells [5]–[9].

To compensate for the neural network's performance degradation due to the memristor defects, the synapse-aware training methods are used widely [10], [11]. The re-training method considering memristor defects can compensate for the accuracy degradation very effectively. However, a large amount of memory should be used during the defect-aware re-training time for storing the memristor defect map [12], [13]. Moreover, the re-training time can be very long to consider all the memristor defects one by one. The memory overhead of the defect map and long re-training time of the synapse-based defect-aware scheme can impose a big burden on the edge intelligence hardware such as IoT sensors, etc.

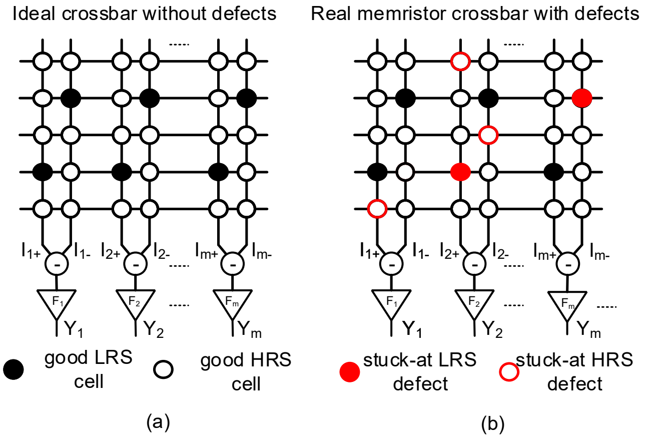

Fig. 1. The schematics of (a) the ideal crossbar without defects and (b) the real crossbar with stuck-at-fault defects.

Unlike the defect-aware training method, we can consider to exclude defective columns from the neural network's training and inference [12]. If the defective columns are deactivated during the training and inference of the neural networks, only the normal columns in the defective memristor crossbar can participate in the neural network's operations [12]. By doing so, the accuracy degradation due to the memristor defects can be improved even though the neural networks are realized with the defective crossbars.

In next Section 2, we review the neuron deactivation scheme that excludes the severely-defective columns, which contain a large number of defect cells, from the neural network's operation. Moreover, we propose a simple method that can detect the severely-faulty columns easily without measuring all the stuck-at-faults cell by cell. In section 3, we compare the neural network's performance of the real memristor crossbars without and with the defective column deactivation for various defect percentages. Section 3 considers the accuracy degradation due to not only the memristor defects, but also the memristor's conductance variation. Finally, in Section 4, this paper is summarized.

II. Neuron Deactivation Scheme for Defect-Tolerant Memristor Neural Networks

Fig. 2(a) shows a flowchart of the neuron deactivation and the training/inference of the defective crossbar for realizing defect-tolerant memristor neural networks. First, we program all the cells in the crossbar to HRS. After programming all the cells to HRS, we measure each column's current one by one. The measured column current is compared to I_{REF}. If the column's current is larger than I_{REF}, the column can be defined as severely-defective. The stuck-at-LRS faults contained in the defective column can increase its column current significantly, because they are not able to be programmed to HRS. Once the column is defined as severely-defective, a neuron connected to the column should be deactivated during the neural network's operations such as training and inference to prohibit the defective column from being involved in the neuron activation.

One thing to note here is that the neuron deactivation scheme explained in this paper does not need the memristor defect map during the training, which needs a large amount of memory to store the memristor defect information such as defect types and locations of the entire memristor crossbar [12]. Moreover, the neuron deactivation scheme does not need the re-training which is needed for considering the memristor defects during the training. Unlike the defect-aware training scheme, the neural deactivation scheme are able to transfer the calculated synaptic weights directly to the memristor crossbar. This is because the memristor crossbar after the neural deactivation can be assumed that it is composed of only normal cells.

Fig. 2(b) shows a schematic of memristor crossbar circuit with the neuron deactivation scheme and the current comparator. In Fig. 2(b), the red box represents the severely-defective columns. Here the other columns not surrounded by red box can be considered normal. If a column has small number of defects, the column can be considered normal. I_{1+} and I_{1-} represent positive column current and negative one, respectively. Y_1, Y_2 and etc. represent the output voltages from the activation function circuit. For detecting the defective column, the column current is compared with the reference current by the comparator, C, as shown in Fig. 2(b). The current comparison is performed column by column and the result is stored at the latch, L that is connected to the activation circuit, F. If the column is defective severely, the latch, L can disable the activation function circuit, F, regardless of the column current.

One thing to note here is that it is important to find a proper reference current, I_{REF} that distinguishes the severely-defective columns from the normal ones in the crossbar. The reference current that determines the number of deactivated columns in the crossbar can affect the neural network's performance. For example, if the reference current is set too high, the recognition rate can be degraded significantly, because a large number of LRS defect cells can take part in the neural network's operations. On the other hand, too many defective columns are deactivated, which means the reference current is set too low, the recognition rate can be degraded, too. This is because too many columns are prohibited from being involved in the neuron activation.

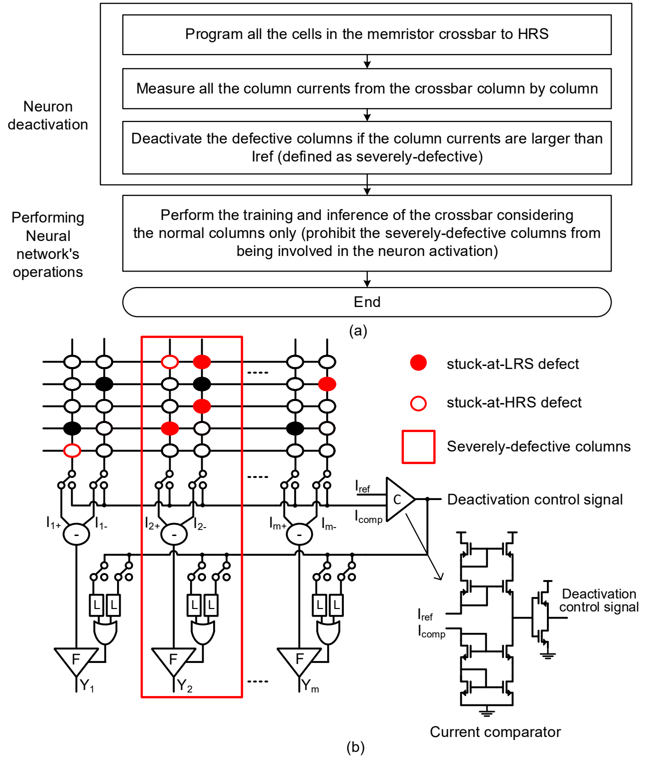
Program all the cells in the memristor crossbar to HRS
Measure all the column currents from the crossbar column by column
Neuron deactivation
Deactivate the defective columns if the column currents are larger than Iref (defined as severely-defective)
Performing Neural network's operations
Perform the training and inference of the crossbar considering the normal columns only (prohibit the severely-defective columns from being involved in the neuron activation)
End
(a)
stuck-at-LRS defect
stuck-at-HRS defect
Severely-defective columns
Iref
Icomp
C
Deactivation control signal
I1+
I1-
I2+
I2-
Im+
Im-
L
F
Y1
Y2
Ym
Iref
Icomp
Deactivation control signal
Current comparator
(b)

Fig. 2. (a) The flowchart of neuron deactivation and training/inference (b) The schematic of memristor crossbar with neuron deactivation scheme and the current comparator

III. Simulation Results

Fig. 3 shows the percentage of LRS defects per column ranked in descending order when the entire crossbar's defect percentage is assumed as 20%. In Fig. 3, the neural network implemented with the defective crossbar is composed of only fully-connected layers. The network with fully-connected layers is tested for MNIST data set. Here, the number of input neurons is decided as 784 for testing MNIST. For the tested neural networks, three cases of the number of hidden and output neurons are assumed in the simulation. The three cases are 213, 414 and 615 for Fig. 3(a), (b), and (c), respectively. The vertical dashed lines in Fig. 3(a), (b), and (c) indicate the percentage of LRS defects per column, at which the reference current is defined. The reference current can be found at the percentage of LRS defects per column showing the best recognition rate of the neural network implemented with the defective crossbar. During the training and inference, the columns to the left of the vertical dashed line that have a higher rank are deactivated.

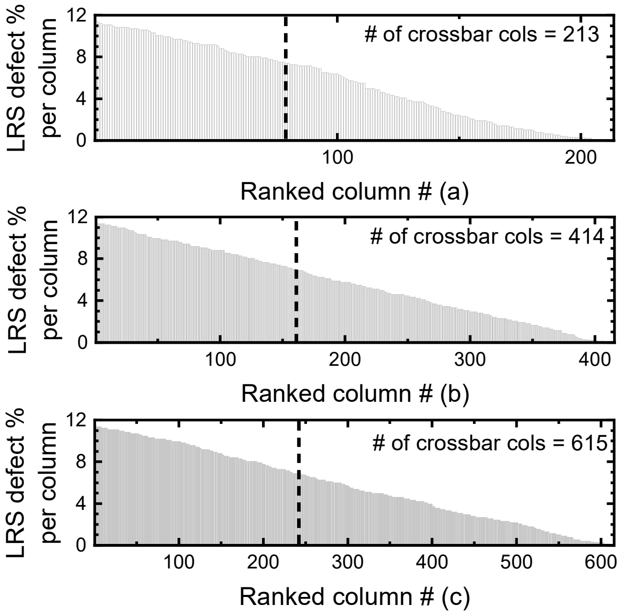

Fig. 3. The LRS defect percentage per column versus the ranked column # in descending orfer for (a) the number of crossbar's columns=213, (b) the number of crossbar's columns=414, and (c) the number of crossbar's columns=615

Fig. 4(a) compares the MNIST recognition rate between the ideal crossbar with no defects and the real crossbar with the defect percentage=5%. Here we tested three crossbars having different numbers of crossbar's columns. They are 213, 414, and 615, respectively, as indicated in Figure 3. For each case, the first bar is for the ideal crossbar. The second bar is for the defective crossbar with the neuron deactivation. The third bar is for the defective crossbar without the neuron deactivation. In Fig. 4(a), when the number of crossbar's column=213, the ideal crossbar has the recognition rate as high as 96.2%. If we consider the real crossbar with the defect percentage=5%, the recognition rate is lowered to 90.8%, when the neuron deactivation scheme is not used. If the severely-defective columns are deactivated, the rate can be recovered from 90.8% to 92%, for the defect percentage=5%. As the number of crossbar's columns is increased, the recognition rate is improved much better, as indicated in Fig. 4(a). The rate loss of the neuron deactivation scheme is only about 1%, compared to the ideal crossbar. Fig. 4(b) compares the MNIST recognition rate between the ideal crossbar with the defect percentage=0% and the real one with the defect percentage=20%.

Similarly with Figure 4(a), Figure 5(a) compares the CIFAR-10 recognition rate between the ideal memristor crossbar without defect and the real crossbar with the defect percentage=5%. Here, we tested 3 cases of the number of columns in the crossbar. They are 213, 414, and 615, respectively. The neural network's architecture for testing CIFAR-10 is ResNet [14], [15]. In this simulation, it is assumed that only the fully connected layers are implemented with the memristor crossbars. The convolution layers are assumed to be calculated by the traditional CMOS digital circuits. Figure 5(b) compares the CIFAR-10 rate between the ideal and real crossbars for the defect percentage=20%.

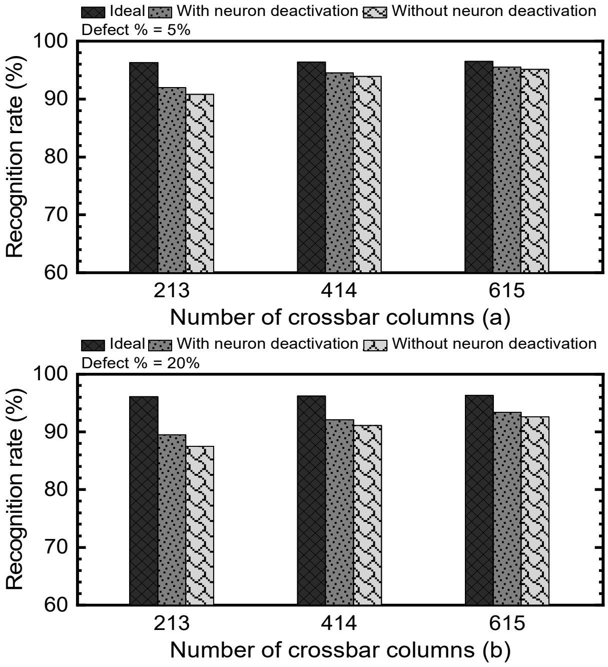

Fig. 4. The MNIST recognition rate with and without the deactivation scheme with increasing the number of crossbar's columns when (a) the crossbar's defect percentage=5% and (b) the crossbar's defect percentage=20%.

One more thing to note here is another fabrication-related factor that degrades the neural net is memristor's conductance variation of HRS and LRS. Fig. 6 compares the CIFAR-10 recognition rate with the memristor's conductance variation=0%, 5%, and 10% for different numbers of crossbar's columns. Here, the crossbar's defect percentage is assumed (a) 5%, (b) 10%, (c) 15%, and (d) 20%, respectively. Fig. 6 indicates that the recognition rate is improved better as the number of crossbar's columns is increased.

On the other hand, the recognition rate is degraded with increasing the memristor's conductance variation. To minimize the programmed conductance variation, we can use the fine memristor programming scheme, where the conductance variation of HRS and LRS can be suppressed less than 5% [16]. If the memristors are programmed with the moderate programming scheme, the programmed conductance variation can be as large as 10% [16]. From Fig. 6(d), the rate loss due to the conductance variation can be controlled as small as 0.6% for the number of crossbar's columns=615, the conductance variation=10%, and the memristor defect percentage=20%.

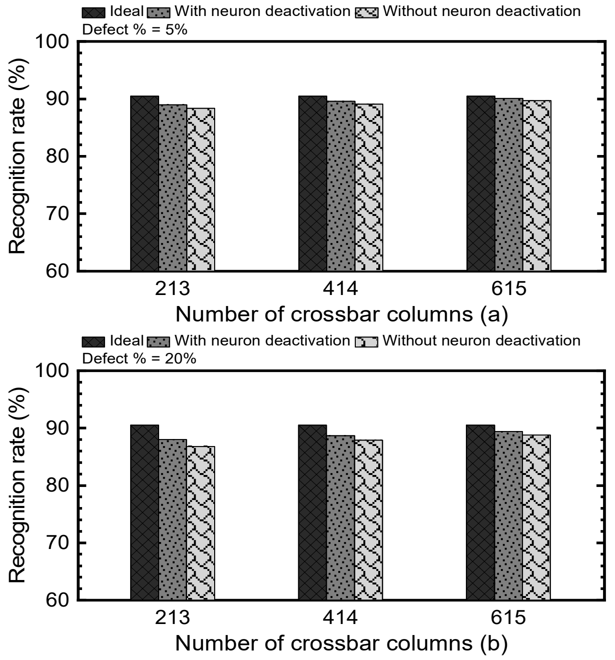

Fig. 5. The CIFAR-10 recognition rate with and without the deactivation scheme with increasing the number of crossbar's columns, when (a) the crossbar's defect percentage=5% and (b) the crossbar's defect percentage=20%. The first bar is for the ideal crossbar. The second bar is for the defective crossbar with the neuron deactivation. The third bar is for the defective crossbar without the neuron deactivation.

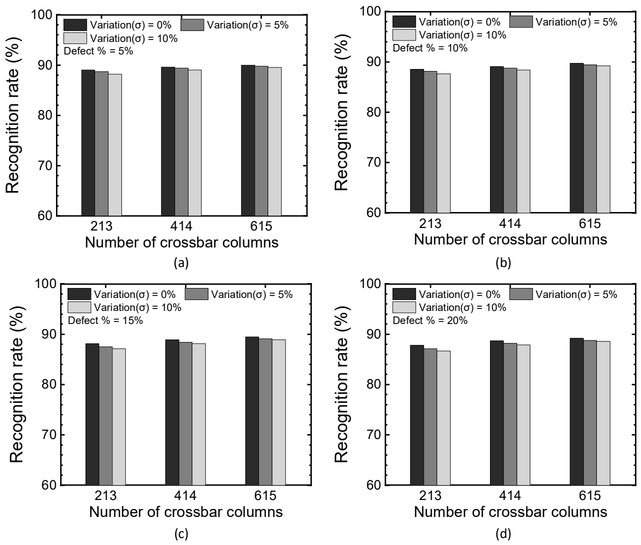

Fig. 6. The CIFAR-10 recognition rate with the memristance variation=0%, 5%, and 10% for different numbers of crossbar columns, when (a) the defect percentage of memristor crossbar=5%, (b) the defect percentage of memristor crossbar=10%, (c) the defect percentage of memristor crossbar=15%, and (d) the defect percentage of memristor crossbar=20%.

IV. Conclusions

To compensate for malfunction of neural networks caused from the fabrication-related defects, in this paper, the simple neuron deactivation scheme was reviewed and analyzed for various numbers of crossbar's columns, various defect percentages, and various memristor's conductance variation.

The column deactivation scheme reviewed in this paper could be particularly useful for the edge intelligence hardware. The column deactivation scheme does not need the memristor defect map occupying a large amount of memory. Moreover, the direct mapping from the calculated synaptic weights to the crossbar can save the re-training time required for the defect-aware training scheme significantly.

Acknowledgment

The work was financially supported by NRF-2022R1A5A7000765, NRF-2019K1A3A1A25000279, NRF-2021R1A2C1011631, NRF-2021M3F3A2A01037972, and SRFC-TA1903-01. The CAD tools were supported by IDEC, Daejeon, Korea.

References

[1] A. Keshavarzi and W. van den Hoek, "Edge intelligence—On the challenging road to a trillion smart connected IoT devices," *IEEE Des. Test*, vol. 36, no. 2, pp. 41–64, 2019.

[2] B. Li, Y. Shan, M. Hu, Y. Wang, Y. Chen, and H. Yang, "Memristor-based approximated computation," *Proc. Int. Symp. Low Power Electron. Des.*, pp. 242–247, 2013, doi: 10.1109/ISLPED.2013.6629302.

[3] B. Li, P. Gu, Y. Shan, Y. Wang, Y. Chen, and H. Yang, "RRAM-Based Analog Approximate Computing," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 34, no. 12, pp. 1905–1917, 2015, doi: 10.1109/TCAD.2015.2445741.

[4] L. Xia *et al.*, "Technological Exploration of RRAM Crossbar Array for Matrix-Vector Multiplication," *J. Comput. Sci. Technol.*, vol. 31, no. 1, pp. 3–19, 2016, doi: 10.1007/s11390-016-1608-8.

[5] M. Hu, H. Li, Y. Chen, Q. Wu, G. S. Rose, and R. W. Linderman, "Memristor crossbar-based neuromorphic computing system: A case study," *IEEE Trans. neural networks Learn. Syst.*, vol. 25, no. 10, pp. 1864–1878, 2014, doi: 10.1109/TNNLS.2013.2296777.

[6] I. Kataeva, F. Merrikh-Bayat, E. Zamanidoost, and D. Strukov, "Efficient training algorithms for neural networks based on memristive crossbar circuits," *Proc. Int. Jt. Conf. Neural Networks*, vol. 2015-Septe, 2015, doi: 10.1109/IJCNN.2015.7280785.

[7] L. Xia *et al.*, "Stuck-at fault tolerance in RRAM computing systems," *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 8, no. 1, pp. 102–115, 2017, doi: 10.1109/JETCAS.2017.2776980.

[8] C. Liu, M. Hu, J. P. Strachan, and H. H. Li, "Rescuing Memristor-based Neuromorphic Design with High Defects," *Proc. - Des. Autom. Conf.*, vol. Part 12828, 2017, doi: 10.1145/3061639.3062310.

[9] W. Choi *et al.*, "WOx-Based Synapse Device with Excellent Conductance Uniformity for Hardware Neural Networks," *IEEE Trans. Nanotechnol.*, vol. 19, pp. 594–600, 2020, doi: 10.1109/TNANO.2020.3010070.

[10] J. An, S. Oh, T. Van Nguyen, and K. S. Min, "Synapse-Neuron-Aware Training Scheme of Defect-Tolerant Neural Networks with Defective Memristor Crossbars," *Micromachines*, vol. 13, no. 2, 2022, doi: 10.3390/mi13020273.

[11] S. Jin, S. Pei, and Y. Wang, "On Improving Fault Tolerance of Memristor Crossbar Based Neural Network Designs by Target Sparsifying," *Proc. 2020 Des. Autom. Test Eur. Conf. Exhib. DATE 2020*, pp. 91–96, 2020, doi: 10.23919/DATE48585.2020.9116187.

[12] T.-V. Nguyen, K.-V. Pham, and K.-S. Min, "Hybrid Circuit of Memristor and Complementary Metal-Oxide-Semiconductor for Defect-Tolerant Spatial Pooling with Boost-Factor Adjustment," *Materials (Basel).*, 2019.

[13] K. Van Pham, T. Van Nguyen, and K.-S. Min, "Partial-Gated Memristor Crossbar for Fast and Power-Efficient Defect-Tolerant Training.," *Micromachines*, vol. 10, no. 4, Apr. 2019, doi: 10.3390/mi10040245.

[14] A. Krizhevsky, V. Nair, and G. Hinton, "CIFAR-10 and CIFAR-100 Datasets," *Available online https//www.cs.toronto.edu/~kriz/cifar.html (accessed 20 Oct. 2018)*, 2018.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[16] K. Van Pham, S. B. Tran, T. Van Nguyen, and K. S. Min, "Asymmetrical training scheme of binary-memristor-crossbar-based neural networks for energy-efficient edge-computing nanoscale systems," *Micromachines*, vol. 10, no. 2, p. 141, 2019, doi: 10.3390/mi10020141.