Construct Convolutional Neural Networks Using Low-yield Binary Memristor Crossbars

Sheng-Yang Sun, Hui Xu, Jiwei Li, Qingjiang Li, Hongqi Yu, Haijun Liu*
College of Electronic Science and Technology
National University of Defense Technology
Changsha 410073, Hunan, China
{sunshengyang13, xuhui, lijiwei10, liqingjiang, yhq, liuhaijun}@nudt.edu.cn

***Abstract*—Recent years, a number of multi-level memristor-based neural networks have been proposed, these architectures use multi-level memristors to complete analog vector-matrix multiplication calculations which to improve computing efficiency, and also increase power consumption and implementation complexity. In addition, low yield of multi-level memristor crossbars is still a major issue in memristor-based neuromorphic computing. In this paper, we proposed a method which constructs CNNs by using ternary neural networks on binary memristor crossbars, and a training strategy which can hold on the high neural network accuracy is utilized under low-yield conditions. To demonstrate the performance of the training strategy, two neural networks architectures are applied for simulation experiments. Extensive experiments show that convolutional neural networks can still obtain more than 95% accuracy even at 80% yield.**

***Index Terms*—binary memristor, low-yield crossbars, convolutional neural networks, neuromorphic computing**

I. Introduction

Convolutional neural networks (CNNs) [1], [2], excellent methods of deep learning applications to visual problems, have attracted significant attention in recent years due to its remarkable performance in computer vision tasks, such as in image classification [3] [4] and object detection [5]. These advantages motivate interests to transplant deep CNNs models to embedded terminals such as smart phones. However, deep CNNs have an exploding number of synapses and use high-precision weights such as 2^N-bit which leads to the complicated computations and much more power consumption [6], [7].

Memristor [8], a low-consumption and non-volatile device, has earned remarkable attention as synapses for embedded neuromorphic processing [9]. The resistance of memristors can be tuned to a specific level by applying voltages pulses, so the device can store the parameter of neural networks. Memristor crossbar arrays can quickly perform vector-matrix multiplications which are costly calculation for neural networks [10]. Fig. 1 shows the basic structure of a memristor crossbar array [11]. Some researches have shown that analog vector-matrix multiplications with memristor crossbars can be orders of magnitude that are more efficient than GPUs or CPUs based computations [12].

Some memristor-based neural networks architectures have been proposed [13] [14], these architectures use multi-level memristors to complete analog vector-matrix multiplication calculations which to improve computing efficiency also increase power consumption and implementation complexity [9]. On the other hand, manufacturing yield of multi-level devices is still a major issue in memristor fabrication [15] [16], and the device state usually can only be stable in a high resistance state (HRS) or low resistance state (LRS) [17]. Therefore, using binary memristor crossbars to implement neural networks is particularly important to promote the practical application of hardware CNNs.

Several non-full-precision neural networks have been studied nowadays [18]–[20], but only few papers [7] studied neural networks on binary memristor crossbars. In this paper, we proposed a method that constructs CNNs by using ternary neural networks (TWNs) [21] on binary memristor crossbars, and a training strategy which can hold on the accuracy is utilized under low-yield conditions. To demonstrate the performance of this method, we carry out the experiments on a three-layer CNNs architecture [22] which proposed in our prior work.

This paper is organized as follows. Section II introduces the proposed method and the specific training strategy. Section III exhibits the experimental results. And the final Section IV concludes this paper.

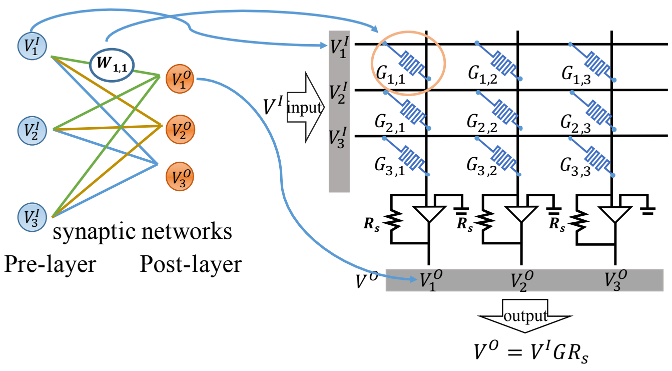


Fig. 1. A 3×3 memristor crossbar array demo. Mapping of a one-layer neural network on the crossbar array, i.e. the input of pre-layer, adaptable synaptic weights and weighted sum output of post-layer maps to the pulse input from V^I, memristor conductance and current output through V^O [23].

*Corresponding author.

II. Low-Yield Binary Memristor Crossbars based Convolutional Neural Networks

A. Convolutional Neural Networks with Binary Memristors

Ternary weight networks (TWNs) [21] which seek to make a balance between the binary precison counterparts and full precision counterparts are applied to obtain the synapse weights of the neural network. TWNs constrain the weights to be ternary-valued: -1, 0 and +1. Assuming a full precision weights W^f, and a ternary weights W^t. The optimization issue can be described as follows,

$$\widetilde{\alpha}, \widetilde{W^t} = \underset{\alpha, W^t}{\arg\min} \, \mathrm{J}(\alpha, W^t) = \|W^f - \alpha W^t\|_2^2 \tag{1}$$

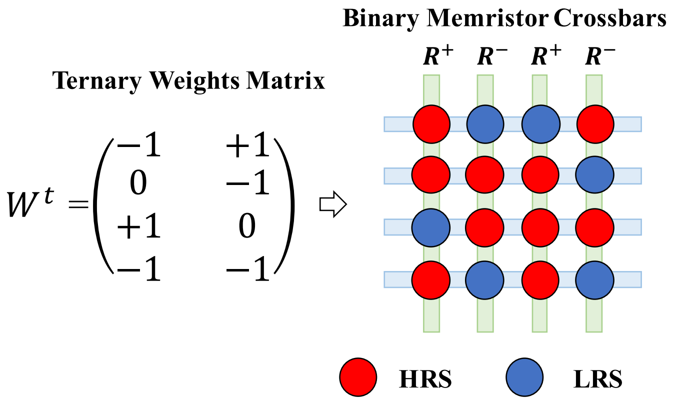
where α is a nonnegative scaling factor, $W_i^t \in \{-1, 0, +1\}$, $i = 1, 2, ..., n$, here n is the number of parameters. With the approximation $\alpha W^t \approx W^f$, the forward propagation can be described as

$$\mathbb{Z} = \delta(X * W^f) \approx \delta(X * (\alpha W^t)) = \delta((\alpha X) \oplus W^t) \tag{2}$$

where $\mathbb{Z}$ is the layer output, X is the layer input, δ is a nonlinear activation function, $*$ indicates the vector-matrix calculation or a convolutional operation, $\oplus$ indicates the vector-matrix calculation or a convolutional operation without multiplication.

In TWNs scheme, 2-bit storage space is needed for one synapse weight. Therefore, we use two memristors to indicate the positive and negative weights in the ternary weights matrix, and use the differential output to represent the results of the analog computation [22] [24].

Fig. 2 shows how to use a binary memristor crossbar to indicate a ternary weights matrix. Here assuming W^t is a 4×2 ternary weights matrix, a size of 4×4 memristor crossbars is utilized for implementation. if the value of W_i^t is -1, the resistance of corresponding position R^+ in crossbars would be HRS, and the R^- would be LRS. In contrast, the R^+(LRS) and R^-(HRS) are utilized to represent the value of +1 in W_i^t. In addition, two HRS memristors are used to indicate the value of zero.



Binary Memristor Crossbars
Ternary Weights Matrix
$W^t = \begin{pmatrix} -1 & +1 \\ 0 & -1 \\ +1 & 0 \\ -1 & -1 \end{pmatrix}$
R^+ R^- R^+ R^-
HRS
LRS

Fig. 2. Diagram of using binary memristor crossbars to indicate a ternary weights matrix.

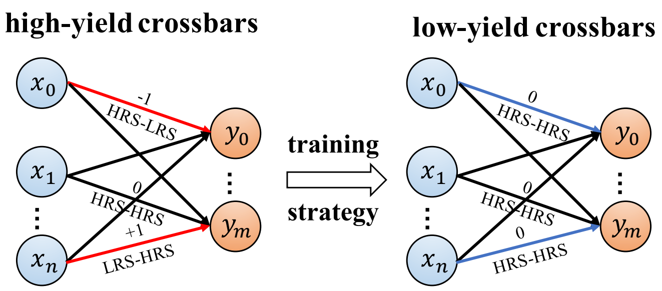


Fig. 3. The schematic of training strategy under low-yield memristor crossbars. The red synapse corresponds to the damaged device in the low-yield array, and the blue synapse represents the weight adjusted after using the training strategy.

From the TWNs algorithm, we can know that a positive threshold parameter Δ is computed to determine the W^t. And the threshold Δ can be approximated to

$$\Delta \approx \frac{0.7}{n} \sum_{i=1}^{n} |W_i| \tag{3}$$

where n is the size of weights set, and W_i is one of the elements of weights set W. The resistance value in the memristor crossbars can be determined by Δ, as shown in Table I.

TABLE I
THE CORRESPONDING RESISTANCE OF THE VALUES IN THE MEMRISTOR CROSSBARS.

Ternary-valued	R^+	R^-	**Conditions**
-1	HRS	LRS	$W_i < -\Delta$
0	HRS	HRS	$\|W_i\| \leq \Delta$
+1	LRS	HRS	$W_i > \Delta$

The resistance of the devices in the array can be determined through Table I. Therefore, binary memristors can be used to implement the convolutional neural network architecture.

B. Training under Low-yield Memristor Crossbars

The mini-batch stochastic gradient descent (SGD) method is used to train the neural network, and the ternary-valued weights are not used during the parameters update but during the forward and backward propagations [25] [26]. To reduce the complexity of hardware implementation, batch normalization (BN) is not used for training [27].

Although the memristor state can be tuned to HRS or LRS, the device may be damaged by aggressive programming and testing cycles [23]. The single-bit failure (SBF) means that a memristor device frozen at LRS or HRS [28], and high SBF rate (low-yield) would degrade the neural networks accuracy remarkably.

Aiming at the issue of improving the accuracy of the neural networks under low-yield conditions, we considered the position of the damaged device in the crossbars during training.

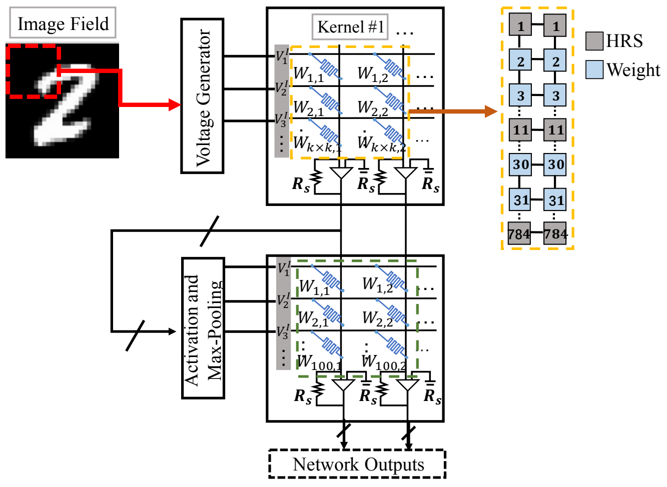


Fig. 4. The implementation of memristor-based convolutional neural networks in the crossbar array.

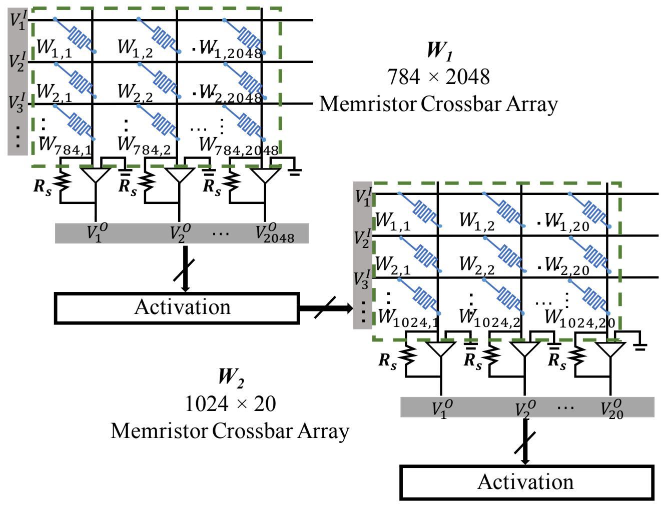


Fig. 5. The implementation of memristor-based multi-layer perceptron in the crossbar array.

Known as described above, two memristors in crossbars are utilized to denote a synapse of neural network, so we analyze it in three cases below.

HRS-LRS: This combined approach is defined in this paper as the "HL" model, meaning that R^+ is set to HRS and R^- is tuned to LRS. If the device at the R^+ is frozen at HRS and the device at the R^- can be tuned, or the device at R^- is frozen at LRS and the device at R^+ can be tuned, then we set to "HL" mode.

HRS-HRS or LRS-LRS: Both devices state are HRS or LRS is defined as "HH" or "LL" mode. This mode will be applied if both the devices at R^+ and R^- are frozen at HRS or LRS.

LRS-HRS: One of the device at R^+ is set to LRS and the other at R^- is set to HRS, this combined method is defined as the "LH" mode. In contrast to the "HL" model, if the device at the R^+ is frozen at LRS and the device at the R^- can be tuned, or the device at the R^- is frozen at HRS and the device at the R^+ can be tuned, the "LH" mode is set.

It can be known from the above three modes that the synaptic weight value at which SBF occurs can be described by Eq. 4.

$$\widetilde{W_i^{t*}} = \begin{cases} -1 & \text{“HL” mode} \\ 0 & \text{“HH” or “LL” mode} \\ +1 & \text{“LH” mode} \end{cases} \tag{4}$$

During the training of the neural networks, the $\widetilde{W_i^{t*}}$ is not affected by weights updates, only used during the forward and backward propagations.

Fig. 3 shows a demo that uses the training strategy under low-yield crossbars conditions. As shown in the figure, supposed that the devices that implement the red synapse is abnormal, for example, for a weight value of -1, the resistance at the R^+ position is fixed at HRS, for the "+1" synapse, the resistance at R^- is frozen at LRS. After applying training method, the weights of "-1"(HRS-LRS) and "+1"(LRS-HRS) are fixed to "0"(HRS-HRS).

III. Experiments

A. Experimental Settings

Experiments in this study are conducted using a computer with 16GB DDR4, Intel Xeon E7 (2.6 GHz), and a NVIDIA Titan XP graphics card. This work is implemented with the caffe [29] open-source library to train the ternary weight networks. Experiments are conducted with Monte-Carlo simulation method in Python.

1) Memristor Model: Experiments are executed based on the $Pt/HfO_2 : Cu/Cu$ devices, which was proposed in our previous work [30]. The LRS and HRS of the device are approximately 10^3 and 10^5 Ω, respectively. During the set(0→1.8V) and reset(0→-2V) process, a stable HRS/LRS ratio of 100 can be obtained.

2) Neural Networks Architecture: A three-layer CNNs architecture [22] which includes five 9×9 convolution kernels and 2×2 max-pooling is implemented in our simulations. As Fig. 4 shown, a 28×28 gray image from MNIST dataset [31] which consists of 60,000 handwritten images for the training and 10,000 images for the testing is converted to voltage vector to the crossbar array. Five convolution kernels are mapped into a 784×4000 array, and each kernel performs 400 convolution computations, so there are 4,000 outputs including negative and nonnegative weights. And the absolute activation function is applied between the convolutional layer and the max-pooling layer. In the same way, the fully connected layer is mapped into a 100×20 crossbar array. It is worth noting that we replaced the non-convolutional regions in the array with HRS.

In addition, a 784×1024×10 multi-layer perceptron (MLP) is applied in our simulations. Fig. 5 demonstrates the MLP implementation on two memristor crossbars. The matrix W_1 which consists of 802,816 synapses is mapped to a 784×2048 array. Similarly, matrix W_2 is mapped to a 1024×20 crossbar array, and the output of the first array is fed into the second array after activated by absolute activation function (Abs).

3) Training Details: We use the L2 regularization method, learning rate scaling procedure and optimization method (SGD with momentum) except any pre-processing approach and data augmentation. The detailed parameter settings are as follows.

TABLE II
NEURAL NETWORKS PARAMETERS SETTING FOR TRAINING.

Parameters	**Value**
weight decay	1e-4
mini-batch size	128
initial learning rate	0.01
training iterations	30,000
momentum	0.9

4) Device Defects Generation Method: Device defects generation includes variations and SBF in our simulations. The device is frozen at HRS with a probability of 50% and the probability of 50% at LRS when the simulated device occurs a SBF issue.

Devices variations is also in our consideration. The devices variation mainly includes two types: parametric variation [32] and switching variation [33]. The imperfect fabrication such as line-edge roughness, random dopants and oxide thickness causes the parametric variation. The switching variation is caused because of driving circuit during the reading or writing cycles, and small programmed pulses would lead to a large variation of memristor resistance. When the device occurs variation, the resistance has changed from R_{ij} to $\widetilde{R_{ij}}$, it can be described as follows:

$$\widetilde{R_{ij}} \leftarrow R_{ij} \cdot e^{\theta_{ij}}; \theta_{ij} \sim N(0, \sigma^2) \quad (5)$$

where the θ_{ij} represents the resistance variation, which follows the lognormal distribution [34], and the parameter σ denotes the extent of the variation in our simulation.

B. Simulation Results

Fig. 6 shows the MNIST dataset recognition accuracy with varying the yield of crossbar arrays from 80% to 98%. The 2%-20% SBF errors in crossbar arrays (yield from 80% to 98%) were randomly generated which means that the positions of the damaged devices in the array generated randomly. Twenty experiments are executed in each yield, and the accuracy of each corresponding point is the average recognition rate. The blue line in the figure indicates the accuracy of the CNNs without any defects, which can reach 98.08%. The red line in the figure represents the recognition rate of the CNNs after using the training strategy, compared with the accuracy without the training strategy indicated by the yellow line. It can be seen from the figure that the lower the yield, the lower the network recognition rate without the training strategy is. And at 80% array yield, the accuracy is only 52.32%. After adopting the training strategy, the recognition rate of the network can still remain above 96% in the range of 80% to 98% yield. For example, at 90% yield, the network without

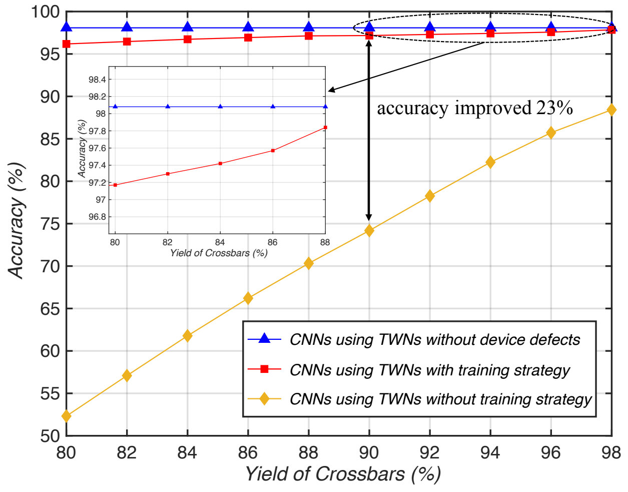


Fig. 6. MNIST recognition accuracy of CNNs using TWNs algorithm with varying the yield of crossbars from 80% to 98%.

the training strategy can only achieve 74.18% accuracy, and after using the training strategy, the accuracy of network can reach 97.17%, and the recognition rate improves nearly 23%.

We executed device variations experiments on MLP and the three-layer CNNs architectures. The yield of the array is fixed at 90%, and the performance of the network is shown in Table III.

TABLE III
MNIST DATASET ACCURACY COMPARISON OF DIFFERENT ARCHITECTURE WITH THE PARAMETER σ OF VARIATION. (**90%** YIELD)

Accuracy(%) / **Parameter(σ)** / **Methods**	**0.6**	**0.8**	**1.0**	**1.2**
MLP using TWNs without strategy	64.48	62.73	58.33	53.93
MLP using TWNs with strategy	**93.89**	**92.75**	**91.62**	**90.71**
CNNs using TWNs without strategy	73.68	71.25	68.91	63.59
CNNs using TWNs with strategy	**96.72**	**95.17**	**94.02**	**92.16**

As can be seen from the data, the recognition rate of the neural networks decreases as the variation of the device increases. In the worst case, the lowest recognition rate of MLP and CNNs without training strategy is only 53.93% and 63.59%, respectively. After adopting the training strategy, the recognition rate of MLP and CNNs has been significantly improved. For example, when $\sigma = 0.8$, the recognition rate of MLP without training strategy is only 62.73%, and the recognition rate can achieve 92.75% after using training strategy. Similarly, when $\sigma = 1.0$, the recognition rate of CNNs using the training strategy can reach 94.02%, which is up to 25.11% compared with the 68.91% accuracy while the training strategy is not used. By using the training strategy, when the device fluctuates greatly ($\sigma = 1.2$), the recognition rates of MLP and CNNs can achieve 90.71% and 92.16% respectively, indicating that the training strategy works well.

IV. Conclusion

In this paper, a training strategy is proposed for constructing convolutional neural networks with low-yield binary memristor crossbars. By considering the position of the damaged device in the training process, the synaptic weight in the neural network is fixed to ensure the recognition rate. Two neural networks architectures are mapped into two memristor crossbar arrays for experiments. From the simulation results above, it can be seen that the recognition rate of the neural network can still reach a high recognition rate after using the training strategy under low-yield crossbars, and CNNs can still obtain more than 95% accuracy even at 80% yield. Considering the device variations, MLP and CNNs can still maintain accuracy of more than 90%. For future work, we consider to evaluate the power consumption of the CNNs using low-yield binary memristor crossbars.

Acknowledgment

This work is supported by National Natural Science Foundation of China under grant 61471377, 61604177, 61704191 and 61804181.

References

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[2] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.

[3] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3639–3655, 2017.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[5] J. Bernal, K. Kushibar, D. S. Asfaw, S. Valverde, A. Oliver, R. Martí, and X. Lladó, "Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review," *Artificial intelligence in medicine*, vol. 95, pp. 64–81, 2019.

[6] S.-Y. Sun, H. Xu, J. Li, Q. Li, and H. Liu, "Cascaded architecture for memristor crossbar array based larger-scale neuromorphic computing," *IEEE Access*, vol. 7, pp. 61 679–61 688, 2019.

[7] K. Van Pham, T. Van Nguyen, S. B. Tran, H. Nam, M. J. Lee, B. J. Choi, S. N. Truong, and K. Min, "Memristor binarized neural networks," *J. Semicond. Technol. Sci*, vol. 18, no. 5, pp. 568–588, 2018.

[8] L. Chua, "Memristor-the missing circuit element," *IEEE Transactions on circuit theory*, vol. 18, no. 5, pp. 507–519, 1971.

[9] S.-Y. Sun, H. Xu, J. Li, H. Liu, and Q. Li, "Cascaded neural network for memristor based neuromorphic computing," in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–6.

[10] L. Xia, T. Tang, W. Huangfu, M. Cheng, X. Yin, B. Li, Y. Wang, and H. Yang, "Switched by input: Power efficient structure for rram-based convolutional neural network," in *Proceedings of the 53rd Annual Design Automation Conference*, 2016, pp. 1–6.

[11] S. Yu, "Neuro-inspired computing with emerging nonvolatile memorys," *Proceedings of the IEEE*, vol. 106, no. 2, pp. 260–285, 2018.

[12] M. Hu, J. P. Strachan, Z. Li, E. M. Grafals, N. Davila, C. Graves, S. Lam, N. Ge, J. J. Yang, and R. S. Williams, "Dot-product engine for neuromorphic computing: Programming 1t1m crossbar to accelerate matrix-vector multiplication," in *Proceedings of the 53rd annual design automation conference*. ACM, 2016, p. 19.

[13] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 14–26, 2016.

[14] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory," in *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3. IEEE Press, 2016, pp. 27–39.

[15] C. Soell, M. Reichenbach, J. Roeber, A. Hagelauer, R. Weigel, and D. Fey, "Case study on memristor-based multilevel memories," *International Journal of Circuit Theory and Applications*, vol. 46, no. 1, pp. 99–112, 2018.

[16] X. Zhu, C. Wu, Y. Tang, J. Wu, and X. Yi, "Multi-level programming of memristor in nanocrossbar," *IEICE Electronics Express*, vol. 10, no. 5, pp. 20 130 013–20 130 013, 2013.

[17] S. N. Truong, S. Shin, S.-D. Byeon, J. Song, H.-S. Mo, and K.-S. Min, "Comparative study on statistical-variation tolerance between complementary crossbar and twin crossbar of binary nano-scale memristors for pattern recognition," *Nanoscale research letters*, vol. 10, no. 1, p. 405, 2015.

[18] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1," *arXiv preprint arXiv:1602.02830*, 2016.

[19] S.-Y. Sun, J. Li, Z. Li, H. Liu, H. Liu, and Q. Li, "Quaternary synapses network for memristor-based spiking convolutional neural networks," *IEICE Electronics Express*, pp. 16–20 190 004, 2019.

[20] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6869–6898, 2017.

[21] F. Li, B. Zhang, and B. Liu, "Ternary weight networks," *arXiv preprint arXiv:1605.04711*, 2016.

[22] S.-Y. Sun, Z. Li, J. Li, H. Liu, H. Liu, and Q. Li, "A memristor-based convolutional neural network with full parallelization architecture," *IEICE Electronics Express*, pp. 16–20 181 034, 2019.

[23] S.-Y. Sun, H. Xu, J. Li, Z. Li, Y. Sun, Q. Li, and H. Liu, "Cases study of inputs split based calibration method for rram crossbar," *IEEE Access*, vol. 7, pp. 141 792–141 800, 2019.

[24] S. Sun, J. Li, Z. Li, H. Liu, Q. Li, and H. Xu, "Low-consumption neuromorphic memristor architecture based on convolutional neural networks," in *2018 International Joint Conference on Neural Networks (IJCNN)*, July 2018, pp. 1–6.

[25] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Advances in neural information processing systems*, 2015, pp. 3123–3131.

[26] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *European conference on computer vision*. Springer, 2016, pp. 525–542.

[27] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[28] C. Liu, M. Hu, J. P. Strachan, and H. Li, "Rescuing memristor-based neuromorphic design with high defects," in *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2017, pp. 1–6.

[29] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[30] S. Liu, W. Wang, Q. Li, X. Zhao, N. Li, H. Xu, Q. Liu, and M. Liu, "Highly improved resistive switching performances of the self-doped pt/hfo 2: Cu/cu devices by atomic layer deposition," *SCIENCE CHINA Physics, Mechanics & Astronomy*, vol. 59, no. 12, p. 127311, 2016.

[31] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[32] D. Niu, Y. Chen, C. Xu, and Y. Xie, "Impact of process variations on emerging memristor," in *Proceedings of the 47th Design Automation Conference*. ACM, 2010, pp. 877–882.

[33] S. Yu, Y. Wu, and H. S. P. Wong, "Investigating the switching dynamics and multilevel capability of bipolar metal oxide resistive switching memory," *Applied Physics Letters*, vol. 98, no. 10, p. 2237, 2011.

[34] S. R. Lee, Y.-B. Kim, M. Chang, K. M. Kim, C. B. Lee, J. H. Hur, G.-S. Park, D. Lee, M.-J. Lee, C. J. Kim *et al.*, "Multi-level switching of triple-layered TaOx RRAM with excellent reliability for storage class memory," in *2012 Symposium on VLSI Technology (VLSIT)*. IEEE, 2012, pp. 71–72.