Performance of Pulse-Programmed Memristive Crossbar Array with Bimodally Distributed Stochastic Synaptic Weights

Nadine Dersch[1,2], Eduardo Perez[3,4], Christian Wenger[3,4], Christian Roemer[1,2], Mike Schwarz[1], Benjamin Iniguez[2], Alexander Kloes[1]

[1]NanoP, THM University of Applied Sciences, Giessen, Germany, [2]DEEEA, Universitat Rovira i Virgili, Tarragona, Spain, [3]IHP-Leibniz-Institut für innovative Mikroelektronik, Frankfurt (Oder), Germany, [4]BTU Cottbus-Senftenberg, Cottbus, Germany

***Abstract*—In this paper, we present a method of implementing memristive crossbar array with bimodally distributed weights. The bimodal distribution is a result of pulse-based programming. The memristive devices are used for the weights and can only have an ON (logical "1") or an OFF (logical "0") state. The state of the memristive device after programming is determined by the bimodal distribution. The highly efficient noise-based variability approach is used to simulate this stochasticity. The memristive crossbar array is used to classify the MNIST data set and comprises more than 15,000 weights. The interpretation of these weights is investigated. In addition, the influence of the stochasticity of the weights and the accuracy of the weights on the classification results is considered.**

Keywords- artificial neural networks, memristive crossbar array, bimodal distribution, noise-based simulation, pulse-programming, variability

I. Introduction

Memristive devices (MDs) are non-volatile memories and are considered promising candidates for the development of hardware-based artificial neural networks (ANNs) [1][2]. The MDs can be in one of the two: states low-resistive-state (LRS) and high-resistive-state (HRS) [3]. During the SET process, the MD is switched to LRS, which corresponds to a logical "1". The RESET process switches the MD to HRS, which corresponds to a logical "0". The MDs exhibit stochastic fluctuations which result in device-to-device and cycle-to-cycle variability [4]. This stochasticity variability can be simulated using the Noise Based Variability Approach (NOVA) [5]. ANNs can be implemented as a memristive crossbar array, whereby a single cell, consisting of two MDs, functions as a weight with possible values from -1 to +1 [6].

II. Setup of the Memristive Crossbar Array and Programming Scheme

To classify the MNIST data set (images consisting of 28x28 pixels), the memristive crossbar array consists of 784 inputs, 10 outputs and 15,680 memristive cells (as in [5]). In the simulation, the MDs are considered as simple fluctuating resistors for simplicity. Two memristive cells are required to design a weight W between -1 and +1: one G^+ and one G^- cell (see Eq. 1 for calculation).

$$W = G^+ - G^- \qquad \text{Eq.1}$$

The required weight values come from the software training. The memristive cells are programmed by applying pulses, which can be changed in terms of amplitude, pulse width and number of pulses [7]. The starting point is that the weights are set via "probabilities". The MDs can only become the logical values "0" and "1" and their state changes with a certain "probability" depending on the pulses applied. Accordingly, their conductivity follows a bimodal distribution. For each applied pulse, the state of the memristive cell can be represented via a bimodal distribution (how many devices are statistically in the HRS and in the LRS) [8]. In [9] it is shown that the statistical variation resulting from a superposition of many bimodal distribution functions can be represented by the superposition of Gaussian distribution functions. This allows the replacement of the bimodal distributions with Gaussian distributions for the usage of NOVA to simulate the fluctuations of the MDs as in [5]. The simulations are carried out with the Spectre simulator Cadence Virtuoso. After the simulation, the winner is determined according to the winner-takes-all principle as in [10].

III. Interpretation of the Weight Definition from Pulse Programming

A pulse with a pulse width of 1 µs and an amplitude of 0.8V is defined for programming the devices. This pulse is sent 100 times to 128 different cells that are in the HRS before the first pulse. A MD can therefore be in the HRS state or in one of the 100 programming states depending on the number of applied pulses (measurement data from [7]). To be able to use NOVA, an average value and a standard deviation are calculated for each programming state using the 128 measured curves. According to Eq. 1, two MDs are required for a weight W, whereby a weight W can be composed according to figure 1 (a).

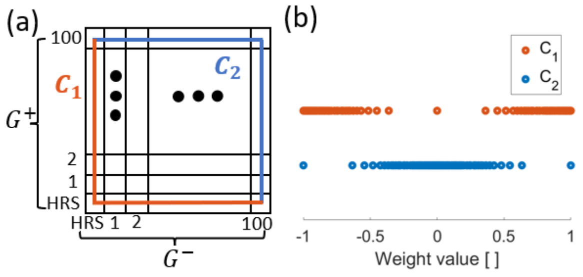

Figure 1 (a) Representation of the 10,201 possible weights within the -1 to +1 range in the 101x101 matrix via paths. (b) Possible discrete weight values within the -1 to +1 range depending on the selected path.

Paths (e.g. C_1 and C_2) can be formed within this matrix, which must be set for the range from -1 to +1 for G^+ and G^-. However, it is noticeable that the values of G^+/G^- do not increase continuously from HRS to the 100th pulse, which means that the values from -1 to +1 can also be set with different accuracy. This is illustrated in figure 1 (b). Here the path C_1 covers weight values in the range from -1 to -0.5 and +0.5 to +1 with high accuracy, whereby C_2 provides a higher resolution in the range from -0.5 to +0.5.

Figure 2 shows the standard deviation of each possible weight W. Here it can be seen that path C_1 has lowest standard deviations and C_2 has the highest standard deviations.

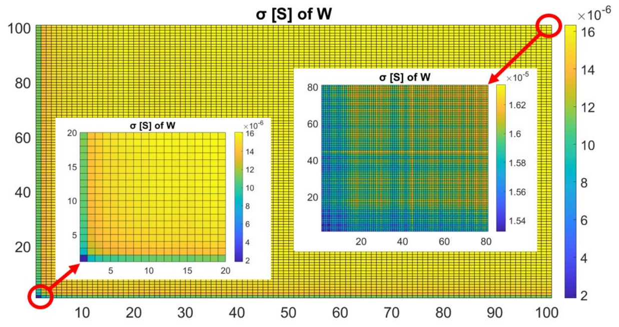

Figure **2** Visualization of the resulting standard deviation for each possible weight to be set.

IV. Simulation Results of the Memristive Crossbar Array

The memristive crossbar array is tested with the same images of a "7" and a "4" as from [5] (programming of weights by conductance level), as well as the image of a "1". For all cases, the weights are trained with a resolution of 0.1 weight stepping. The paths C_1 and C_2 are compared by adjusting the target weight to the closest possible value (see figure 1(b)). The results are shown in Table 1, as percentage of classifying the given number as a winner.

Table 1 Classification results of the images of a "7", "4" and "1" with path C_1 and C_2. The expected result is marked in green and the result with the highest probability is marked in blue.

Results	"7", C1	"4", C1	"1", C1	"7", C2	"4", C2	"1", C2
0	4.65%	23.94%	2.81%	7.73%	9.36%	4.45%
1	1.35%	6.94%	76.74%	6.28%	8.64%	15.48%
2	0.44%	0.36%	0.08%	11.17%	10.79%	9.13%
3	2.57%	1.10%	4.34%	12.11%	7.67%	11.00%
4	1.13%	8.52%	4.60%	8.14%	11.69%	8.57%
5	0.24%	0.88%	0.21%	9.73%	9.62%	9.39%
6	0.84%	16.15%	4.35%	6.00%	10.30%	10.60%
7	87.84%	32.58%	6.23%	20.68%	11.20%	12.00%
8	0.92%	9.39%	0.34%	9.74%	12.30%	10.26%
9	0.02%	0.14%	0.30%	8.42%	8.43%	9.12%

Table 1 shows that path C_2 delivers significantly worse classification results than path C_1. The reason for this is that the variability in the weights is very large for path C_2, which means that no precise classification is possible. The three digits are all classified with a similar probability. In contrast, in path C_1 the "7" is classified correctly with 87.84% (in [5] in the worst case 99.4% and best case 100%) and the "1" with 76.74%. The "4" is misclassified in most cases and is most frequently identified as a 7. In [5], this "4" is correctly classified in the best case with 50.44%.

Two factors play a role in the result of the programming: 1) The possible fluctuation of the desired weight value and 2) the number of adjustable weights or the accuracy of how finely resolved they should be set.

Path C_1 was selected for testing the "1" image, with weights set to increments of 0.1, 0.05 and 0.025 and noise levels of 100% (referred to the statistical variations as observed in the measurements), 85%, 67%, 25% and 0%. The results are shown in figure 3.

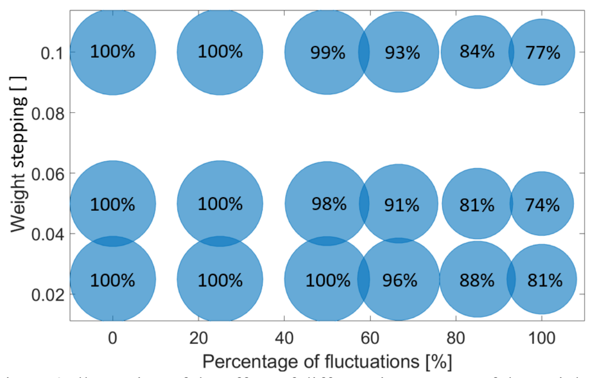

Figure **3** Illustration of the effect of different increments of the weights and a reduction in the variability of the weights. The percentage for the correct classification of "1" is given.

Figure 3 shows that the classification with a step of 0.025 is better than with 0.1, but the result only improves slightly. However, the 0.05 step is worse than with 0.1, as the finer discretization leads to weight combinations with an increased σ. It is noticeable that a reduction of the variability provides significantly better classification results. Even with a reduction of 1/3, the results are in the 90% range for all step sizes.

V. Conclusion

The programming via pulses shows strong variability in the weight values. As a result, the classification accuracy of the MNIST dataset is influenced by the variability depending on the G^+/G^- settings. In addition, regarding a high probability for a correct classification, it is more important that the weights fluctuate less than whether they can be set precisely. A reduction of the fluctuations observed in measurements by 33% already shows significant improvements. However, even without fluctuations a precise setting of conductance states is important to achieve the correct classification results.

References

[1] C. Zambelli et al., International Memory Workshop, pp. 1-4, 2014.
[2] P. Huang et al., IEEE Trans. Electron Devices, vol. 64, no. 2, 2017.
[3] Z. Jiang et al., IEEE TED., vol. 63, no. 5, pp. 1884–1892, 2016.
[4] A. Kloes et al., Solid-State Electronics, vol. 201, p. 108606, 2022.
[5] N. Dersch et al., Solid-State Electronics, vol. 209, p. 108760, 2023.
[6] C. Zambelli et al., ICMTS, pp. 27-31, 2014.
[7] E. Perez et al., JJAP, vol. 61, no. SM, 2022.
[8] C. Wenger et al., IEEE EDL, vol. 40, no 4, 2019.
[9] N. Dersch et al., submitted to IEEE LAEDC, 2024.
[10] N. Bogun et al., MIXDES, pp. 83-88, 2022.