





Variability-Aware Memristive Crossbars With ImageSplit Neural Architecture

Aswani Radhakrishnan , Graduate Student Member, IEEE, Anitha Gopi , Graduate Student Member, IEEE, Chithra Reghuvaran , Member, IEEE, and Alex James , Senior Member, IEEE

Abstract—The errors in the memristive crossbar arrays due to device variations will impact the overall accuracy of neural networks or in-memory systems developed. For ensuring reliable use of memristive crossbar arrays, variability compensation techniques are essential to be part of the neural network design. In this paper, we present an input regulated variability compensation technique for memristive crossbar arrays. In the proposed method, the input image is split into non-overlapping blocks to be processed individually by small sized neural network blocks, which is referred to as imageSplit architecture. The memristive crossbar based Artificial Neural Network (ANN) blocks are used for building the proposed imageSplit. Circuit level analysis and integration is carried out to validate the proposed architecture. We test this approach on different datasets using various deep neural network architectures. The paper considers various device variations including R_{OFF}/R_{ON} variations and aging using imageSplit. Along with hardware compensation techniques, algorithmic modifications like pruning and dropouts are also considered for analysis. The results show that splitting the input and independently training the smaller neural networks performs better in terms of output probabilistic values even with the presence of the significant amount of hardware variability.

Index Terms—Image splitting, artificial neural network, memristive crossbar, variability compensation, intel image, CIFAR-10.

I. INTRODUCTION

CROSSBAR memristive arrangement is a popular approach for emulating the Multiply and Accumulate (MAC) operations. The growing applications of memristive crossbar arrays includes modeling neural networks, analog/digital computing, and in-memory computing. Errors in memristive crossbar arrays due to device variations, device agings [1] and noise affect the overall accuracy of the system. To ensure the reliable use of memristive crossbar arrays, variability compensation techniques must be incorporated to reduce the performance variation for a specific application. This paper proposes a memristive crossbar variability compensation technique using ImageSplit.

Manuscript received 9 September 2023; revised 18 February 2024; accepted 5 March 2024. Date of publication 8 March 2024; date of current version 29 March 2024. This work was supported by the Ministry of Electronics and IT, Government of India. The review of this paper was arranged by Associate Editor A. Acharyya. (Aswani Radhakrishnan and Anitha Gopi contributed equally to this work.) (Corresponding author: Alex James.)

Aswani Radhakrishnan, Anitha Gopi, and Alex James are with Digital University Kerala, Thiruvananthapuram 695317, India (e-mail: apj@ieee.org).

Chithra Reghuvaran is with University College Dublin, D04 V1W8 Dublin, Ireland.

Digital Object Identifier 10.1109/TNANO.2024.3375125

The neural network performance and the overall architecture design of MAC implementation depend on the limitations of the weight resolution [2]. Apart this resolution limitation, this memristor is sensitive to various factors such as cycle-to-cycle changes, process variations, and device-to-device variability which leads to conductance variability. The reliability of the crossbar architecture depends on the endurance and ageing of the device [2]. Crossbar computation stability and robustness are increased by adding redundancy to the nodes of the crossbars [3]. The redundancy in the memristor crossbar is introduced by combining memristors and is important for applications with accurate computing [3]. These types of nodes are called super-resolution nodes; they can create stable conductance levels per crossbar node for accurate analog computing in large numbers.

The crossbar arrangement of memristive systems is a popular and effective method for matrix multiplication in the field of neuromorphic computing. Input values are fed through row lines, and output current values are obtained from the bit or column lines. Each intersection point in a crossbar represents the synaptic junctions which can be physically identified as conductance values. Any change in the conductance value can change the entire column current value, and hence, the accumulated current error. Each crossbar column line represents the output values of each node from the neural network layer. The relative current error from a crossbar reflects the error in the neural network layer computation. While multiple crossbars are connected to create a complete neural network, these accumulated current errors influence the output layer current value. Hence the accuracy decreases. The hardware variability associated with this conductance drift for memristive crossbars that drive electron movement when an external force is applied. Changes in the material stack owing to non uniform manufacturing process can alter the read-write cycles. Another key element used for selective write and read operations is the selector device associated with the memristor unit. Hence, the most popular node structures are 1T1M, 2T1M where T represents the CMOS based switching device and M represents the memristor unit. The Leakage currents and parasitic effects associated with the node structure can change the read-write value. Along with crossbar variability, non-idealities from the peripheral circuit also affect the current error. Appropriate input control and readout circuits eliminates the relative current error.

Changing the device material properties, circuit, and architectural level modifications are the main key solutions that come under variability compensation techniques. Adding

TABLE I
COMPARISON WITH EXISTING APPROACHES, ANN=ARTIFICIAL NEURAL NETWORK, CNN=CONVOLUTIONAL NEURAL NETWORK

No	Structure	Architecture	Algorithm	Accuracy	Variability	Sparsity
1	Parallel [4]	Parallel connected memristor crossbar	CNN, CIFAR-10 CNN, MNIST	90.5% 94%	✗	✗
2	Super-resolution [2]	Multiple rows are tied to a single input	CNN, CIFAR10	91.2% 8 Memristor	✓	✗
3	Bridge [5]	Single memristor node with 5 memristor in bridge structure	ANN, MNIST CNN, MNIST	96% 98%	✓	✗
4	Tiled [6]	Large crossbar array splitted into smaller arrays	ANN, MNIST	91.9%	✗	✗
5	Pseudo-crossbar array [7]	A dummy column to avoid the leaky current	ANN, MNIST	95%	✗	✓
6	3D modelling [8]	3D memristor crossbar architecture	ANN, MNIST	96%	✗	✗
7	ImageSplit	Input image is split into non-overlapping blocks	ANN, CIFAR10 ANN, INTEL IMAGE DATA	69%(8 split) 89%(8 split)	✓	✓

different modulation layers such as an ultra-thin ALD-TiN buffer layer [9], threading dislocation technology [10], and optimizing redox reactions at the metal-oxide interface, can control the dynamics of the switching parameters. In circuit-level compensation techniques, various architectures have been tested to reduce the error in circuitry along with different programming strategies. Another compensation technique is use architectural modifications to reduce the current drop and control switching behavior. Cell structures using multiple memristors can be combined into a single node, and pseudo crossbar arrays and peripheral circuit compensation are novel architectures. Cascading the resistance at the device level can result in different equivalent resistance combinations, thereby increasing the available conductance levels for better mapping. For example the parallel and series combination [2] of memristors in a single node can improve the robustness. However, multiple memristor cell structures increase area overhead [11], [12]. In another approach, learning algorithms can be modified to accommodate variability compensation using pruning and dropouts in trained neural networks.

A comparison of the proposed technique with existing approaches for compensating hardware variability is shown in Table I. Here architectural modification of crossbar array for reducing the variability is listed. This include crossbar as well as node arrangement for neuromemristive systems. Parallel [4] crossbar structure explains the idea of dynamic crossbar node arrangement where number of memristor in a crossbar node varies. Super-resolution [2] approach introduced a solution for limited conductance states or resolution. Multiple rows are tied to a single input to create a crossbar node. These limited resolution problem was addressed by a bridge [5] node configuration where each single node is represented as a 2T5M structure. Large sized arrays are limited by sneak path issues. Hence Tiled architecture [6] or splitted crossbar structure reduces the leaky current issues. A dummy column [7] of crossbar can be used to replace aged one for these leaky current issues. Along with node arrangement of crossbar 3 dimensional [8] crossbar arrangement also experimented in the literature to reduce the computational energy. Only a few architectural modifications analysed the performance under variability constraints. Neural Network architectures are highly sparse and eliminating those sparse values can helps to minimise the computational power. For any circuit component variability increases with increase in

number of modules. Reducing those number of computational module using pruning or dropout method can decrease the variability in general. Here in this work splitting the architecture along with hardware pruning reduced the hardware complexity without reducing the performance.

The concept of imageSplit is introduced in [13] as a neural network tiling solution to reduce the computational complexity of deep neural network. In [13], the input image in the dataset is split into smaller units and each units are processed using small sized neural networks. This paper extends the use of the ImageSplit as a variability compensation technique in memristive crossbar arrays. This paper presents the complete hardware architecture of imageSplit using Artificial Neural Network (ANN). An integration block for combining the output from individual neural networks is designed and evaluated. We consider different device variations including R_{OFF}/R_{ON} variations and aging using imageSplit. Along with hardware compensation techniques, algorithmic modifications like pruning and dropouts are also considered for analysis. The results show that imageSplit can reduce the effect of variability by processing the data on smaller neural networks. Splitting the input and independently training the smaller neural networks performs better in terms of output probabilistic values even with the presence of the significant amount of hardware variability.

The paper is organized in into 5 sections. Section II discusses the ImageSplit Implementation using Memristive crossbar arrays. The compensation techniques using ImageSplit is presented in Section III, Section IV presents the Results and Discussion and the concluding remarks in Section V.

II. IMAGE SPLIT IMPLEMENTATION USING MEMRISTIVE CROSSBAR ARRAYS

For the ImageSplit technique, the input image is split into smaller units and can be visualized as multiple sub-datasets of spatially separated block images [13]. Each split images was processed using an independent neural network model of having smaller sizes. An integration block is used to combine each model output to obtain the classify of the image [13].

Fig. 1 shows the complete architecture implementation of ImageSplit using an ANN. The memristive crossbar implementation of the nosplit case is shown in Fig. 1(a). Consider an image dataset $\mathbb{X} = \{x_1, x_2, \dots, x_P\}$ with each image $x_p, \forall p =$

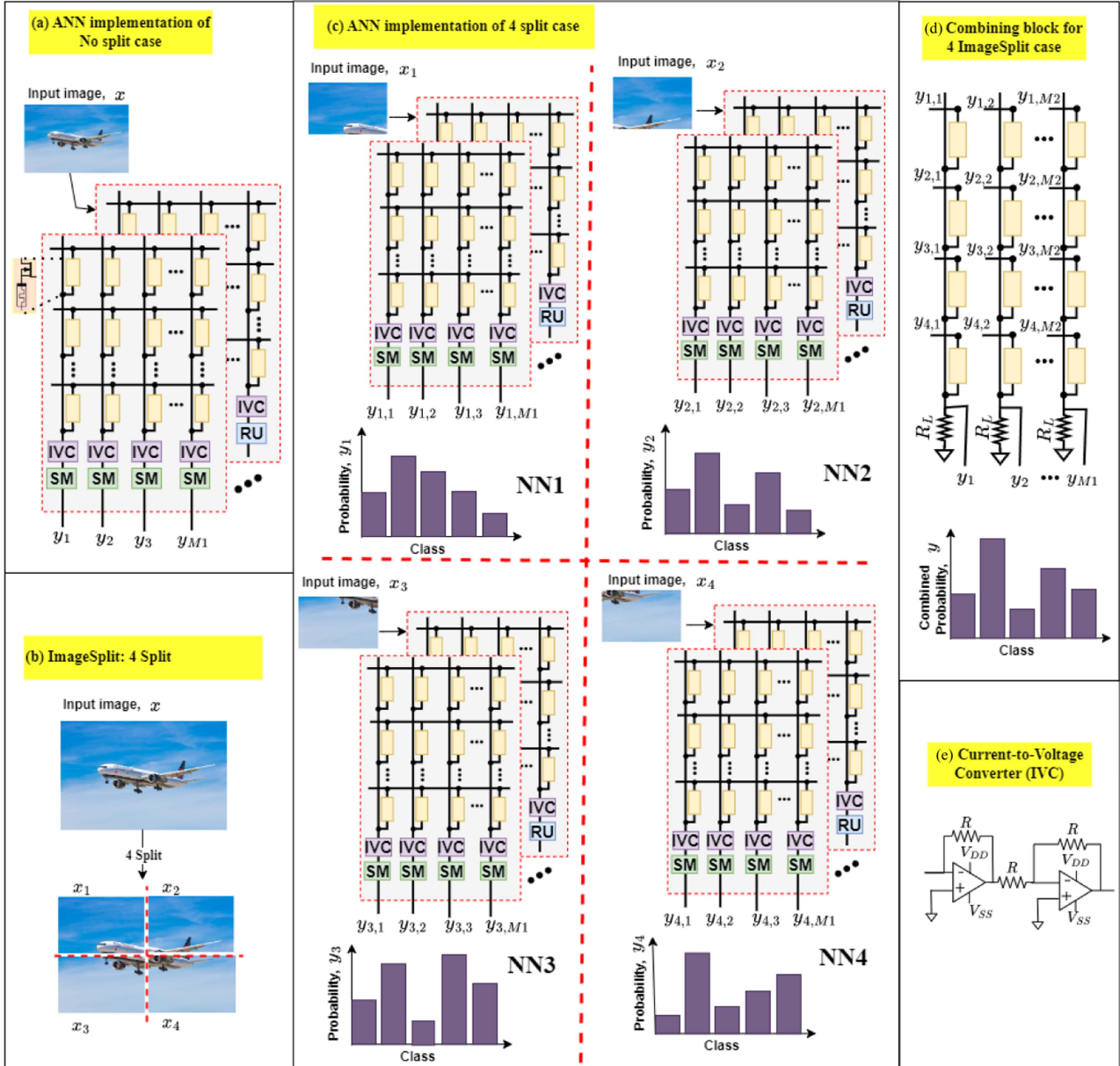


Fig. 1. ANN implementation of ImageSplit Methodology (a) ANN implementation of no split case, (b) ImageSplit: 4 split case Processing, (c) ANN Implementation of 4 split case (d) Integration block for 4 Imagesplit case and (e) Circuit implementation of Current-to-Voltage converter (IVC). SM: Softmax Activation function, RU: Relu Activation Function, NN: Neural Network.

$\{1, 2, 3, \dots, P\}$ of size $A \times B$. For the nosplit case, $A \times B$ and M_1 are the sizes of input and output neurons respectively with M hidden layers, Fig. 1(a). Now, we consider splitting the input image x_p into L sub-parts. In the image-splitting case, each image was split into sub-units of size $A1 \times B1$. Fig. 1(b) shows the imageSplit for $L = 4$, that is, the original image is divided into four multiple non-overlapping blocks. These blocks can be of various sizes, Fig. 2 shows the blocks with same sizes. The splitting generates L separate datasets from \mathbb{X} , for four splits, we have four smaller datasets from \mathbb{X} , \mathbb{X}_1 dataset containing the upper-left half, \mathbb{X}_2 containing the upper-right half, \mathbb{X}_3 containing the lower-left half and \mathbb{X}_4 containing the lower-right half. These subsets of separate datasets from imageSplit were processed

separately using smaller neural network architectures. Fig. 1(c) shows the ANN implementation for the 4 split case. Each dataset was processed using smaller neural networks NN1, NN2, NN3 and NN4. These smaller individual networks process data in parallel and classify them. The neural architecture uses a softmax activation function for the output layer and all the other layers use the ReLU activation function. The circuit implementation of the activation functions is presented in Fig. 2.

For a dataset with M_2 number of classes, each network has M_2 outputs, that is, from NN1 the outputs are $\{y_{1,1}, x_{1,2}, \dots, y_{1,M_2}\}$ and similarly from NN2, NN3 and NN4. These four outputs from the individual networks may be the same or different. These outputs are fed into an integration block for the final output.

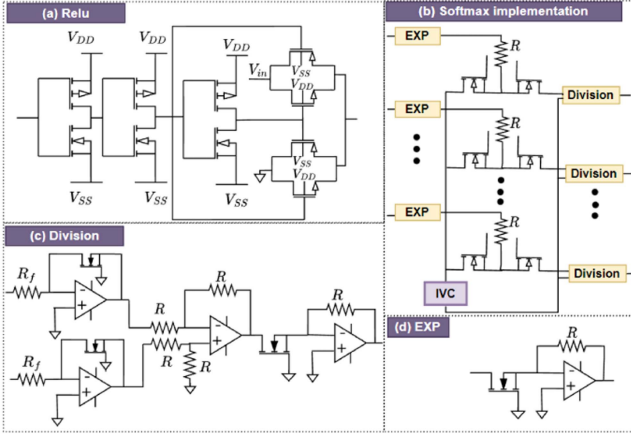


Fig. 2. Circuit Implementation of Activation functions. (a) Circuit implementation of ReLU [14], (b) Circuit implementation of Softmax [15], (c) Circuit implementation of Division block [15] and (d) Circuit implementation of EXP block [15].

The circuit implementation of integration block is illustrated in Fig. 1(d). The memristor integration block was a memristor crossbar structure of size $L \times M_2$. The cells in the first column takes the output of Class 1 from NN1, NN2, NN3 and NN4. Second column for Class 2 and so on. Thus the crossbar output $\{y_1, x_2, \dots, y_{M_2}\}$ gives the integrated probabilities of NN1, NN2, NN3 and NN4. The final output y_j is obtained across the load resistance R_L . The y_j can be computed as

$$y_j = \frac{\sum_{i=1}^L (y_{i,j}/R_i)}{\frac{1}{R_L} + \sum_{i=1}^L (1/R_i)} \quad (1)$$

where R_i is the memristor resistance. From (1), the class with the maximum y_j value is detected as the final prediction, that is, the class most often detected by smaller networks is chosen as the final prediction.

III. COMPENSATION TECHNIQUE USING IMAGESPLIT

Improving device material characteristics and modifying both the circuit and architectural level [16] neural structures are widely used approaches for reducing the impact of hardware variability challenges. Along with hardware compensation techniques, algorithm-level modifications, such as like pruning and dropouts can significantly reduce aging-related issues by omitting unwanted weight values. Using this approach both hardware-level and algorithmic-level compensation were carried out.

A. Modular Arrangement

Recent architectural modifications include modular structure arrangements, a series-parallel [2] combination of memristive nodes [17], and a bridge super-resolution approach. However the multiple memristor cell structure increases the area overhead. These approaches help improve the single node resolution whereas the scalar or modular arrangement of crossbar structures are reduces the area power-related [11], [12] crossbar challenges. Splitting a large neural network can produce a relative readout

error; however, splitting the input and independently training smaller neural networks performs better in terms of output probabilistic values.

B. Aging Challenges

Owing to the repeated supply of voltage during the programming stage of the memristor, the filament inside the material stack undergoes aging [18] thereby changing the effective conductance range available for mapping. Along with quantization errors, this change in the conductance range leads to an inappropriate mapping of the weight values [19]. The change in the desired conductance value from the actual programmed value can be classified into the following categories;

Case 1:

$$G_{ON} + AG_{ON} \text{ and } G_{OFF} + AG_{OFF} \quad (2)$$

Case 2:

$$G_{ON} + AG_{ON} \text{ and } G_{OFF} - AG_{OFF} \quad (3)$$

Case 3:

$$G_{ON} - AG_{ON} \text{ and } G_{OFF} + AG_{OFF} \quad (4)$$

Case 4:

$$G_{ON} - AG_{ON} \text{ and } G_{OFF} - AG_{OFF} \quad (5)$$

G_{ON} and G_{OFF} are the maximum and minimum conductance values, respectively. The conductance variability ratio is denoted by "A". Ideally, the peak values affect the conductance variability more than other conductance states. However, the other conductance states also diminished when the variability exceeded the threshold level. In the case of failure type 2(case 2) there is no conductance loss due to aging. Hence, these studies considered only three different types of failures. A change in the conductance causes a relative current error which in turn reduces the performance degradation of the proceeding layers. These cumulative current errors were directly proportional to the parameter size. The Independent training and validation of the neural networks showed a degradation in the current errors. In addition pruning and dropout reduce the parameter count without affecting performance accuracy.

C. Pruning and Dropout

Pruning and dropouts are two popular methods for removing weight values in a neural network. Dropout removes random nodes whereas pruning eliminates weight values that have lower magnitude values. The level of pruning [20] or dropout was determined by measuring the algorithm performance with hardware variability. A highly pruned network can be easily affected by hardware variability. Analyzing hardware variability with sparse updates can minimize the area, power and computational complexity.

IV. RESULTS AND DISCUSSION

The circuit implementation of the ANN architecture using the proposed imageSplit method was evaluated. The ANN was

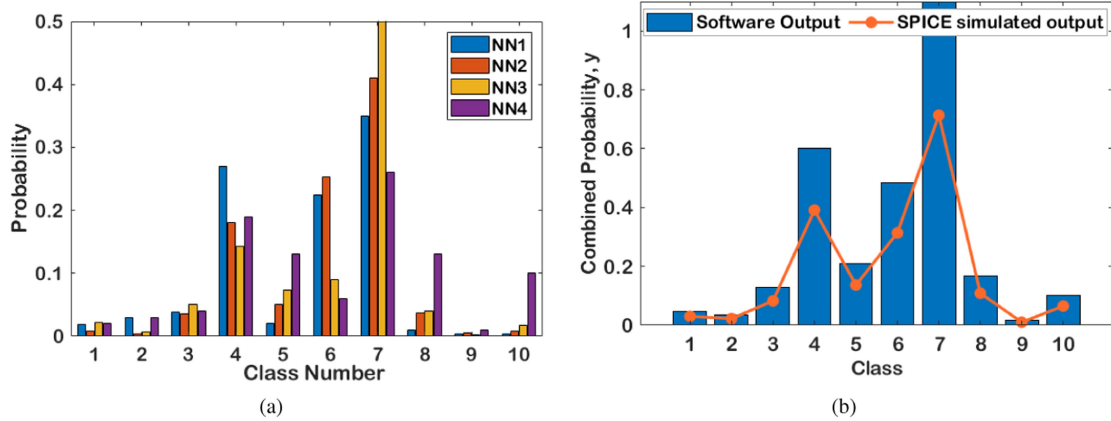


Fig. 3. Performance plot of 4 ImageSplit (a) Output of Neural Network1 NN1, NN2, NN3, NN4 (b) Combining block: Comparison between software and Circuit simulated output in SPICE.

trained using Intel Image Classification and CIFAR-10 data-sets. Python coding was performed for the training of the neural network architectures, and the trained weights were mapped to the memristors crossbar arrays. The memristor model considered is Known Multi-Stable Switch (MSS) with WO_x device parameters. The trained weights are mapped according to $R_{ON} = 1 \text{ K}\Omega$ and $R_{OFF} = 100 \text{ K}\Omega$. The 1T1M memristor cell are implemented using 22 nm high-k PTM models [21].

The proposed variability compensation using the imagesplit methodology was analyzed using both hardware and software level parameters. Two ANN architectures were used for the performance comparison of different splitting counts. Upto 16 level split counts were carried out for the analysis. The Single-layer Artificial Neural Network(SANN) architecture consists of two sequential layers with node size of m, l where m denotes the image size and l denotes the output class size. For example, in a split count of 2, the image is partitioned into two halves with 16×32 pixels for the CIFAR10 dataset. Hence the input layer will has a node size equal to $16 * 32$. Similarly, when we increase the split count from 2 to 16, each partition is separately taken out and fed into a separate SANN structure. For the multi-layer artificial neural network (MANN), additional 2 hidden layers were added for training. Node size of 100 and 50 were used. The weight values from the trained models were mapped to the appropriate conductance value range. The level of quantization determines the mapping accuracy and a performance analysis for different quantization levels is carried out.

Each neural network was trained separately using the split images. For example, for a split count of 4, four neural networks were trained with an image size of 16×16 . 4 different trained weight values are mapped onto conductance values and calculated softmax probabilistic values from the outer layer. An integration block accumulates the corresponding probability values from each network. The final decision can be made by taking the maximum probability value from the accumulated values. Fig. 3 shows the performance plots of the four ImageSplit images. Fig. 3(a) represents the output of Neural Network1 (NN1), NN2, NN3, NN4, and (d) shows the performance of the integration block: comparison between software and circuit simulated output in SPICE.

A. Pruning and Dropout

Pruning minimizes the sparsity. Both the SANN and MANN architectures were pruned for different percentage values from 0.05. Table II shows the performance accuracy for both the SANN and MANN with the CIFAR-10 and Intel image classification datasets. When we moved from two split counts to eight, the performance accuracy increased. Compared with eight, splitting count 16 shows a reduction in accuracy. Table III shows the performance accuracy after dropout for both the SANN and MANN with CIFAR-10 and Intel image classification datasets. The network was trained with different dropout percentages and the performance accuracy for different split counts was calculated.

B. Aging Analysis

Table IV shows the output layer Relative Current Error(RCE) values. The performance analysis under different variability conditions for the SANN with the CIFAR10 dataset is shown in this table. Two variability percentage values were considered 5 and 10. Split counts with 2, 4, 16, and 32 levels and trained weights with different quantization levels for mapping were used for analyze the impact of variability. The quantization levels range from 2 to 16. In the case of failure type 2(case 2) there is conductance loss due to aging. Both upper and lower conductance values moved in the upper and lower directions, respectively. This reduced the current error. The current error also increases with an increase in the variability percentage. In the case of number of split, the variability or current error decreased with an increase in the number of splits. Large crossbar layers have more hardware variability, and hence, more relative current errors. The splitted crossbar architecture reduces hardware complexity and reduced current error values.

C. Area and Power

The area and power requirements of the imagesplit neural architecture trained for the CIFAR10 dataset are presented in Table V. The table provides different split cases for the SANN architecture. The split cases are bench marked with the no

TABLE II
PERFORMANCE ACCURACY FOR 2 DIFFERENT DATASETS AND 2 ANN ARCHITECTURES FOR PRUNING

Percentage (%)	CIFAR10 dataset								Intel image classification dataset							
	SANN				MANN				SANN				MANN			
	0	4	8	16	0	4	8	16	0	4	8	16	0	4	8	16
0	45	52	62	55	49	52	69	60	28	69.4	75.5	71	34	80.8	89.6	72
5	41	50	55	45	45	52	55	49	29	68.8	75.1	70.9	32	75.6	82.3	68.6
10	39	49	51	44	42	50	53	46	25	62.3	70.3	67.2	30	71.2	76.2	66.4
30	37	47.5	48.5	43.3	47	49	49	45	26.8	59.9	68	62.3	31.2	67.2	73.6	64.5
50	43.3	45.3	48.2	41	45.2	47	48.8	44	18.9	51.7	65	59.7	29.9	63	70.8	59.9
70	39	42.2	42	38	43	45.6	47	43.6	12.3	46.8	51	50.8	21.3	58.6	65.6	52.3
90	31	38	39	30.3	38	41.2	42.8	41.8	7.6	32.7	41.9	42.3	18.2	48.4	50.3	48.6

Percentage denotes pruning percentage and 0, 4, 8, and 16 denote splitting count.

TABLE III
PERFORMANCE ACCURACY FOR 2 DIFFERENT DATASETS AND 2 ANN ARCHITECTURES FOR DROPOUT

Percentage (%)	CIFAR10 dataset										Intel image classification dataset									
	SANN					MANN					SANN					MANN				
	0	2	4	8	16	0	2	4	8	16	0	2	4	8	16	0	2	4	8	16
0	42	49	52	62	58	50.5	52	52	69	60	32.3	69.4	75.5	75.5	71	39.6	39.6	70	89	72
10	32	40	41	58.5	50	49	40	42.6	40	57.4	25	25	62.3	71.3	68	34.2	34.2	68	84	63
30	25	38.1	37.5	51	47.5	39	44	39	42	51	18	18	58	63	60.4	28	28	62	78	51
50	21	23	30.3	43	38	31	34.5	34	39	37	11	11	35	59	55	20	20	50	60	40

Percentage denotes dropout percentage and 0, 2, 4, 8, and 16 denote splitting count.

TABLE IV
RCE DEPICTING EFFECT OF R_{ON} AND R_{OFF} VARIABILITY ON SANN WITH CIFAR10 DATASET

Percentage A (%)	RCE(%) for SANN with cifar10 dataset																
	Count=2				Count=4				Count=8				Count=16				
	L=2	L=4	L=8	L=16	L=2	L=4	L=8	L=16	L=2	L=4	L=8	L=16	L=2	L=4	L=8	L=16	
5	Case 1	1.6	1.4	1.0	0.8	1.2	1.22	0.8	0.6	0.9	0.8	0.65	0.4	0.65	0.54	0.42	0.36
	Case 2	1.4	1.12	0.9	0.7	1.0	1.09	0.5	0.56	0.2	0.11	0.025	0.024	0.019	0.014	0.01	0.001
	Case 3	2.0	1.99	1.2	1.10	1.90	1.4	0.78	1.09	1.3	0.78	0.63	0.5	0.69	0.59	0.52	0.42
	Case 4	2.6	2.68	1.33	1.23	2.3	2.0	1.0	1.19	0.8	0.6	0.4	0.24	0.53	0.48	0.34	0.12
10	Case 1	5.8	4.8	3.9	2.8	4.2	3.8	2.3	2.4	1.8	1.6	1.25	1.0	1.3	1.5	1.10	0.96
	Case 2	2.3	2.0	1.9	1.7	1.6	1.9	1.59	1.8	2.6	1.9	1.4	0.9	1.79	1.4	1.2	1.2
	Case 3	6.3	5.9	4.3	4.0	5.1	4.6	3.8	3.7	3.5	2.8	2.0	1.8	2.3	2.0	1.98	1.6
	Case 4	10.2	8.6	8.2	7.4	8.9	7.6	6.8	6.9	7.8	6.6	5.3	3.9	5.9	6.0	5.0	2.96

L denotes quantization level.

TABLE V
POWER CONSUMPTION AND AREA COMPARISON

Split	Power		Area	
	Crossbar (μ W)	Amplifier (mW)	Crossbar (μ m ²)	Amplifier (μ m ²)
SANN				
No split	122.88	11.71	15.36	18.05
2	61.44	11.71	7.68	18.05
4	30.72	5.86	3.84	9.02
8	15.36	5.86	1.92	9.02
16	7.68	2.92	0.96	4.51
MANN				
No split	1106.88	66.61	138.36	102.65
2	1045.44	66.61	130.68	102.65
4	822.72	60.76	102.84	93.62
8	807.36	60.76	100.92	93.62
16	703.68	57.82	87.96	89.11

SANN: single layer ANN.

splitting case. The amplifier circuits were designed with a three stage amplifier circuit using a 22 nm CMOS node [15]. The results in Table V shows that there is considerable reduction in area and power requirements with image splitting. The output node size is the same for the no split case and 2 and similarly 4 & 8; hence, the same amplifier power is used for these cases in the SANN. The effect of reduction in area and power is more

prominent for the MANN neural network architecture than for the SANN, as shown in Table V. Clearly too many splits need not necessarily increase accuracy. There should be a trade-off between the area, power and accuracy when choosing the proper split.

V. CONCLUSION

In this paper, we presented an imagesplit technique for compensating variability-related challenges in neuro-memristive crossbar systems. Along with hardware compensation techniques, algorithm-level modifications such as pruning and dropouts significantly reduce aging-related issues. The performance of the split structure varies from two split counts to the eight number of splits. Later, a 16-number split counted shows the degradation of performance considering both single-layer and multi-layered ANN architectures. Pruning and dropout reduce the complexity by reducing the number of parameters required for computation. This, in turn helps reduce cumulative aging issues in a structure. Splitting the input and independently training smaller neural networks performs better in terms of output probabilistic values, even in the presence of a significant amount of hardware variability. Memristor crossbar arrays are

difficult to scale in hardware owing to the presence of a large amount of variability. The use of multiple tiled small crossbar units is more feasible for hardware [8] applications. Imagesplit is one of the solutions to keep the crossbar size small and for simple architectural design. Imagesplit is one of the solutions to keep the crossbar size small and for simple hardware implementation.

REFERENCES

- [1] 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.*, 2019, pp. 1–5.
- [2] A. P. James and L. O. Chua, "Analog neural computing with super-resolution memristor crossbars," *IEEE Trans. Circuits Syst. I: Regular Papers*, vol. 68, no. 11, pp. 4470–4481, Nov. 2021.
- [3] A. P. James and L. O. Chua, "Variability-aware memristive crossbars: a tutorial," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 69, no. 6, pp. 2570–2574, Jun. 2022.
- [4] J. Lee, J. K. Eshraghian, K. Cho, and K. Eshraghian, "Adaptive precision CNN accelerator using radix-x parallel connected memristor crossbars," 2019, *arXiv:1906.09395*.
- [5] A. Radhakrishnan, S. Pallathuvalappil, B. Choubey, and A. James, "Bridge memristor super-resolution crossbars," *IEEE Trans. Emerg. Sel. Topics Circuits Syst.*, vol. 12, no. 4, pp. 944–951, Dec. 2022.
- [6] D. Mikhailenko, C. Liyanagedera, A. P. James, and K. Roy, "M₂ CA: Modular memristive crossbar arrays," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2018, pp. 1–5.
- [7] P.-Y. Chen et al., "Mitigating effects of non-ideal synaptic device characteristics for on-chip learning," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2015, pp. 194–199.
- [8] B. R. Fernando, Y. Qi, C. Yakopcic, and T. M. Taha, "3D memristor crossbar architecture for a multicore neuromorphic system," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2020, pp. 1–8.
- [9] Y. Fang et al., "Improvement of HfO₂-based RRAM device variation by inserting ALD TiN buffer layer," *IEEE Electron Device Lett.*, vol. 39, no. 6, pp. 819–822, Jun. 2018.
- [10] T. Tanikawa, K. Ohnishi, M. Kanoh, T. Mukai, and T. Matsuoka, "Three-dimensional imaging of threading dislocations in GaN crystals using two-photon excitation photoluminescence," *Appl. Phys. Exp.*, vol. 11, no. 3, Feb. 2018, Art. no. 031004, doi: [10.7567/APEX.11.031004](https://doi.org/10.7567/APEX.11.031004).
- [11] Y. Cassuto, S. Kvatinisky, and E. Yaakobi, "Sneak-path constraints in memristor crossbar arrays," in *Proc. IEEE Int. Symp. Inf. Theory*, 2013, pp. 156–160.
- [12] L. Shi, G. Zheng, B. Tian, B. Dkhil, and C. Duan, "Research progress on solutions to the sneak path issue in memristor crossbar arrays," *Nanoscale Adv.*, vol. 2, no. 5, pp. 1811–1827, 2020.
- [13] E. M., R. Chithra, and A. James, "ImageSplit: Modular memristive neural network," in *Proc. IEEE 22nd Int. Conf. Nanotechnol.*, 2022, pp. 535–538.
- [14] O. Krestinskaya, B. Choubey, and A. James, "Memristive GaN in analog," *Sci. Rep.*, vol. 10, no. 1, 2020, Art. no. 5838.
- [15] R. Chithra, A. R. Aswani, and A. P. James, "TMS-crossbars with tactile sensing," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 69, no. 3, pp. 1842–1846, Mar. 2022.
- [16] I. Yourkas, D. Stathis, G. C. Sirakoulis, and S. Hamdioui, "Alternative architectures toward reliable memristive crossbar memories," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 24, no. 1, pp. 206–217, Jan. 2016.
- [17] C. Yakopcic, M. Z. Alom, and T. M. Taha, "Extremely parallel memristor crossbar architecture for convolutional neural network implementation," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2017, pp. 1696–1703.
- [18] S. Zhang, G. L. Zhang, B. Li, H. H. Li, and U. Schlichtmann, "Aging-aware lifetime enhancement for memristor-based neuromorphic computing," in *Proc. Des., Automat. Test Euro. Conf. Exhib.*, 2019, pp. 1751–1756.
- [19] R. Kumar, A. Chordia, A. Aswani, A. James, and J. N. Tripathi, "Uncertainty quantification of memristor crossbar array for vector matrix multiplication," in *Proc. IEEE 25th Workshop Signal Power Integrity*, 2021, pp. 1–4.
- [20] C.-Y. Chen and K. Chakrabarty, "Pruning of deep neural networks for fault-tolerant memristor-based accelerators," in *Proc. 58th ACM/IEEE Des. Automat. Conf.*, 2021, pp. 889–894.
- [21] 22 nm PTM model for metal gate high-k CMOS: V2.0, 2007. [Online]. Available: <http://ptm.asu.edu/>



Aswani Radhakrishnan (Graduate Student Member, IEEE) received the bachelors's degree in electronics and communication in 2017, and the Master of Technology degree in embedded systems in 2020. She is currently working toward the Ph.D. degree with the School of Electronics Systems and Automation, Digital University of Kerala, Thiruvananthapuram, India. From 2020 to 2021, she was a Research Assistant and is currently involved in a few projects related to hardware based neural network implementation. She has authored or coauthored several papers related to both hardware and software neural network optimization problems including CNN and LSTM. Her research interests include neural network accelerators, neural networks, analog circuits, natural language processing, and neuromorphic computing systems. She is a reviewer and author of several top IEEE journals and conferences.



Anitha Gopi (Graduate Student Member, IEEE) received the bachelor's degree in electronics engineering and the Master of Technology degree. She is currently working toward the Ph.D. degree with the School of Electronics Systems and Automation, Digital University of Kerala, Thiruvananthapuram, India. Her research interests include image sensors and low power AI processing near image sensors. She is currently involved in a few projects related to imaging and low power memristive network implementation. Her research interests include analog circuits and neuro-imaging computing systems.



Chithra Reghuvaran (Member, IEEE) received the Ph.D. degree from the National Institute of Technology Rourkela, Rourkela, India. She is currently the Training Manager with Maker Village, Kochi, Kerala. Before joining Maker Village, she was a Faculty with the Amrita School of Engineering, Bangalore Campus, and Indian Institute of Information Technology Vadodra, Gandhinagar, India. She has authored or coauthored many research articles in referred journals and conferences and has reviewed articles of several international journals and conferences. Her research interests include the Internet of Things, sensor edge intelligent circuits, and hardware realization of neuromorphic systems and related areas. She was the recipient of the Erasmus Mundus India-Europe Action-2 HERITAGE Scholarship funded by the European Union. She is a Member of IEEE Circuits and Systems Society.



Alex James (Senior Member, IEEE) received the Ph.D. degree in electronics engineering from the Queensland Micro and Nanotechnology Center, Griffith University, Brisbane, QLD, Australia. He is currently a Professor of AI hardware with the School of Electronic Systems and Automation and the Dean (Academic), Digital University Kerala, Trivandrum, India. He is a Prof-in-Charge of Maker Village that supports more than 80 hardware startups. He is the Chief Investigator with the Center for Intelligent IoT Sensors and the India Innovation Centre for Graphene, developing new products to the market. His research interests include a broad range of brain-inspired systems, memristive systems, intelligent semiconductor devices, analog circuits, and imaging systems. He was the Founding Chair of IEEE Kerala Section Circuits and Systems Society. He is a Member of the IEEE CASS Technical Committee on Nonlinear Circuits and Systems, IEEE CASS Technical Committee on Cellular Nanoscale Networks and Memristor Array Computing, IEEE Consumer Technology Society Technical Committee on Quantum in Consumer Technology (QCT), Technical Committee on Machine Learning, Deep Learning and AI in CE (MDA), and a Member of BCS Fellows Technical Advisory Group (F-TAG). He was an Editorial Member of the Information Fusion, Elsevier, IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE during 2017–2018 (Guest Associate Editor) and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS 1 during 2018–2023. He has been an Associate EIC of IEEE OPEN JOURNAL OF CIRCUITS AND SYSTEMS since 2024, an Associate Editor for IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS since 2024, an Associate Editor for Frontiers in Neuroscience (Section: Neuromorphic Systems) and IEEE ACCESS. He is also a Life Member of ACM, Senior Fellow of HEA, Fellow of the British Computer Society, and Fellow of IET.