A Novel Transpose 2T-DRAM based Computing-in-Memory Architecture for On-chip DNN Training and Inference

Yuansheng Zhao*, Zixuan Shen*, Jiarui Xu*, Kevin C.T. Chai†, Yanqing Wu‡, Chao Wang*
Email: chao_wang_me@hust.edu.cn
*School of Optical and Electronic Information, Huazhong University of Science and Technology, Wuhan, Hubei, China.
†School of Integrated Circuits, Peking University, Beijing, Beijing, China.
‡Institute of Microelectronics, Agency for Science, Technology and Research, Singapore.

2023 IEEE 5th International Conference on Artificial Intelligence Circuits and Systems (AICAS) | 979-8-3503-3267-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/AICAS57966.2023.10168641

***Abstract*—Recently, DRAM-based Computing-in-Memory (CIM) has emerged as one of the potential CIM solutions due to its unique advantages of high bit-cell density, large memory capacity and CMOS compatibility. This paper proposes a 2T-DRAM based CIM architecture, which can perform both CIM inference and training for deep neural networks (DNNs) efficiently. The proposed CIM architecture employs 2T-DRAM based transpose circuitry to implement transpose weight memory array and uses digital logic in the array peripheral to implement digital DNN computation in memory. A novel mapping method is proposed to map the convolutional and full-connection computation of the forward propagation and back propagation process into the transpose 2T-DRAM CIM array to achieve digital weight multiplexing and parallel computing. Simulation results show that the computing power of proposed transpose 2T-DRAM based CIM architecture is estimated to 11.26 GOPS by a 16K DRAM array to accelerate 4CONV+3FC @100 MHz and has an 82.15% accuracy on CIFAR-10 dataset, which are much higher than the state-of-the-art DRAM-based CIM accelerators without CIM learning capability. Preliminary evaluation of retention time in DRAM CIM also shows that a refresh-less training-inference process of lightweight networks can be realized by a suitable scale of CIM array through the proposed mapping strategy with negligible refresh-induced performance loss or power increase.**

Keywords—Computing in Memory, back propagation algorithm, transpose matrix, DRAM, Deep Neural Network

I. INTRODUCTION

As number of parameters and amount of computation in Deep Neural Networks (DNNs) processing in Artificial Intelligence (AI) applications continues to increase, traditional von Neumann architectures suffers high latency and high power consumption of data movement between computation and memory units [1]. By embedding some of the intensive computational tasks into the memory units to reduce the data communication between the computation and memory units, Computing-in-Memory (CIM) is an effective approach to significantly improve the energy efficiency of DNN computations, which makes it suitable for edge devices.

Recently, CIM architectures, based digital domain and analog domain computation methods, are realized by various embedded memory technologies, e.g., Static Random Access Memory (SRAM), Dynamic Random Access Memory (DRAM) [2-5], and Non-Volatile Memory (NVM) [6]. Against the emerging memory technologies with unmatured process and degraded endurance [6], SRAM and DRAM based CIM have much longer lifetimes and is compatible for mature Complementary Metal Oxide Semiconductor (CMOS) processes. As compared to SRAM that has a large cell area, DRAM has significantly higher storage density, which is very suitable for large-scale CIM computation in edge applications. In addition, digital domain CIM has higher noise margin and higher accuracy than analog domain CIM method. Therefore, this work focuses on the research of digital domain DRAM-based CIM for edge devices.

For AI Internet of Things (AIoT) applications, on-chip training is essential to enable re-training to both improve CIM accuracy and protect user privacy [2-3]. In the literature, some transpose SRAM-based CIM arrays are reported to achieve on-chip training [2,3]. However, there are limitations in the existing SRAM-based CIM solutions as the SRAM array-based transpose depends on decoupled read transistors [2] or additional two-way transpose multiplication cells [3] that result in significant hardware overhead. In addition, SRAM fundamentally limits the size of the CIM array due to its CMOS technology yield issue. For the DRAM-based CIM methods, the existing CIM architectures [4-5] cannot support Back Propagation (BP) computation as required by the on-chip DNN training. This is because these existing architectures cannot support both row-wise and column-wise calculations while keeping the weight matrix unchanged within the CIM arrays. Hence, it is required for these conventional DRAM-based CIM architectures to conduct off-chip training or reorganize the weight matrix in the memory array after each Forward Propagation (FP) operation for on-chip training, resulting in significantly extra power consumption and latency. Therefore, efficient in-situ transpose of weight matrix in the memory array is of paramount importance for on-chip BP-based learning in DRAM CIM.

To solve the aforementioned issues, this paper proposes a transpose 2T-DRAM based CIM architecture which can efficiently implement on-chip learning while keeping both memory array and peripheral circuit minimally changed. The proposed CIM architecture supports not only DNN inference but also on-chip BP-based training. There are three major contributions as follows:1) At array level, a 2T-DRAM based transpose memory is proposed to support BP's in-memory realization without extra circuit, which can also achieve higher weight reuse and eliminate weight updating in the memory array to save the energy consumption of data movement during the transpose operation in BP; 2) At circuit level, digital-domain CIM arithmetic logic based on the transpose memory is adopted to achieve high accuracy and robustness; 3) At architecture level, novel weight mapping methods for Fully-Connected (FC) layers and Convolutional (CONV) layers are proposed to implement efficient in-memory BP acceleration with weight reuse and parallel computation.

II. Proposed Training-Inference Transpose DRAM CIM Macro

A. Algorithm analysis of FP and BP in DNN learning

In the FP process of DNN learning, CONV layers perform convolutional computation and extract the high-dimensional features from input data, while FC layers are used do perform classification based on the extracted features. During the BP process, the weight matrix of both CONV and FC layers are updated for achieving a higher classification performance. However, different from the FC layers in FP process, the weight matrix of FC layers needs to be transposed during the BP computation, as shown in Fig.1.

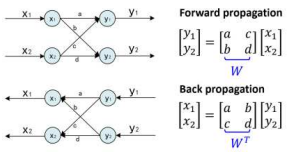


Fig. 1. Matrix transpose: the key operation of BP algorithm in DNN learning.

In the FP process of the CONV layers, the input X is computed with different convolution kernels W to calculate feature maps. For a weight matrix size of $m \times m$, an input matrix size of $k \times k$, and a stride of 1, the computation of CONV layers in FP process is described by:

$$\begin{bmatrix} X_{1,k} & \cdots & X_{1,k} \\ \vdots & \ddots & \vdots \\ X_{k,1} & \cdots & X_{k,k} \end{bmatrix} conv \begin{bmatrix} W_{1,1} & \cdots & W_{1,n} \\ \vdots & \ddots & \vdots \\ W_{n,1} & \cdots & W_{n,n} \end{bmatrix} = \begin{bmatrix} Y_{1,1} & \cdots & Y_{1,k-n+1} \\ \vdots & \ddots & \vdots \\ Y_{k-n+1,1} & \cdots & Y_{k-n+1,k-n+1} \end{bmatrix} \quad (1)$$

In the FP process of the FC layers, for a weight matrix size of $i \times j$, an input matrix size of $1 \times i$, the computation of FC layers is depicted by:

$$\begin{bmatrix} W_{1,1} & \cdots & W_{i,1} \\ \vdots & \ddots & \vdots \\ W_{1,j} & \cdots & W_{i,j} \end{bmatrix} \times \begin{bmatrix} X_1 \\ \vdots \\ X_i \end{bmatrix} = \begin{bmatrix} Y_1 \\ \vdots \\ Y_j \end{bmatrix} \quad (2)$$

In the FP process, weight matrix is multiplied with input matrix X to calculate the output Y. In the BP process, the error between the predicted output and the actual output is back propagated to the input layer. During the backpropagation across the network, the output error ΔY of each layer is fed as the input error ΔX of the subsequent next layer. The output error ΔY in each layer is also used to calculate the weight gradient for the weight update during the training.

In the BP process of CONV layers, in order to get the error of the input matrix ΔX, the output error matrix ΔY needs to be padded and then calculated with the convolution kernel matrix that is rotated by 180°. The computation of CONV layers in BP process is described by:

$$\begin{bmatrix} 0 & \cdots & & \cdots & 0 \\ \cdots & & & & \cdots \\ & \Delta Y_{1,1} & \cdots & \Delta Y_{1,k-n+1} & \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ & \Delta Y_{k-n+1,1} & \cdots & \Delta Y_{k-n+1,k-n+1} & \\ \cdots & & & & \cdots \\ 0 & \cdots & & \cdots & 0 \end{bmatrix} conv \begin{bmatrix} W_{n,n} & \cdots & W_{n,1} \\ \vdots & \ddots & \vdots \\ W_{1,n} & \cdots & W_{1,1} \end{bmatrix} = \begin{bmatrix} \Delta X_{1,1} & \cdots & \Delta X_{1,k} \\ \vdots & \ddots & \vdots \\ \Delta X_{k,1} & \cdots & \Delta X_{k,k} \end{bmatrix} \quad (3)$$

In the BP process of the FC layers, the input error matrix ΔX is calculated from the transpose weight matrix W^T. The computation of FC layers in BP process is described by:

$$\begin{bmatrix} W_{0,0} & \cdots & W_{1,j} \\ \vdots & \ddots & \vdots \\ W_{i,1} & \cdots & W_{i,j} \end{bmatrix} \times \begin{bmatrix} \Delta Y_1 \\ \vdots \\ \Delta Y_j \end{bmatrix} = \begin{bmatrix} \Delta X_1 \\ \vdots \\ \Delta X_i \end{bmatrix} \quad (4)$$

Inevitably, it is of high importance to perform the in-situ 180° rotation and transpose of the weight matrix to enable efficient on-chip learning for CIM acceleration of DNN.

B. Proposed transpose 2T-DRAM for on-chip learning

Fig.2 shows several common DRAM memory cells. Comparing with the other DRAM cells, 2T-DRAM cell is particularly suitable for providing non-destructive and transposable reads due to its functional symmetry of Read Word Line (RWL) and Read Bit Line (RBL). Since the RWL and RBL of 2T-DRAM is interchangeable, it is possible to implement transpose operation within the DRAM array by only using peripheral selector switches controlled by transpose enabling signal, to exchange the readout direction from RWL to RBL. In addition, 2T-DRAM cells eliminate the need for special capacitors in 1T1C cells, while providing higher density and longer retention time than 3T-DRAM cells [7].

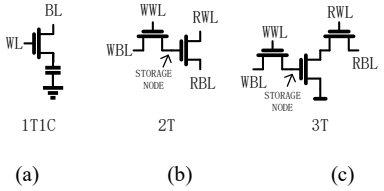


Fig. 2. (a) 1T1C DRAM cell; (b) 2T-DRAM cell; (c) 3T-DRAM cell.

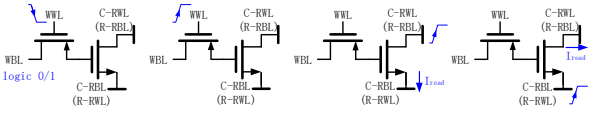


Fig. 3. Circuit configurations of 2T-DRAM based transpose operations: (a) Write mode; (b) Standby mode; (c) Normal read mode; (d) Transpose read mode.

Fig.3 shows four key operation modes of 2T-DRAM cell, where the logic 1 is for VDD and the logic 0 is for GND: 1) Write mode: 1-bit weight is written to the storage node, as shown in Fig.3 (a); 2) Standby mode: the storage node keeps the 1-bit information, as shown in Fig.3 (b); 3) Normal read mode: the 1-bit weight is read by a signal pulse from Column-RWL(C-RWL) to Column-RBL(C-RBL), as shown in Fig.3(c) ;4) Transpose read mode: the 1-bit weight is read from C-RBL to C-RWL, which are re-named as Row-RWL(R-RWL) and Row-RBL(R-RBL), respectively, as shown in Fig.3(d). Based on the above operations, the input data can be fed into the 2T-DRAM array from the C-RWL or R-RWL signal, implementing bit-wise AND operation (i.e. multiplication) with the stored weight data.

Fig.4 shows the array behavior of normal read mode and transpose read mode. The INPUT signal is applied to global RWL (GRWL) with blue color, and the bit-wise multiplication result is read through the sense amplifier (SA) connected to global RBL (GRBL) with red color. Note that the G-RWLs are connected to RWLs, (i.e., C-RWLs in normal read operation and R-RWLs in transpose read operation, as shown in Fig.4 (a) and Fig.4 (b) respectively), through the aforementioned selector switches controlled by signal TEN. Similarly, the G-RBLs are connected to RBLs. As the transpose of the weight matrix can be achieved by the row-wise read enabled by the signal TEN, BP involving the FC-layer multiplication with transposed weight matrix can be done within 2T-DRAM array without extra peripheral circuit overhead or any additional weight data movement.

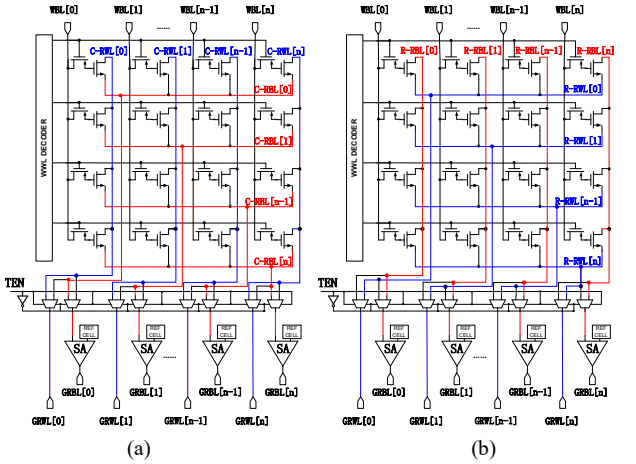


Fig. 4. Proposed transpose DRAM array: behaviors of (a) normal read mode and (b) transpose read mode.

C. *Proposed mapping method of CONV layers for 2T-DRAM CIM*

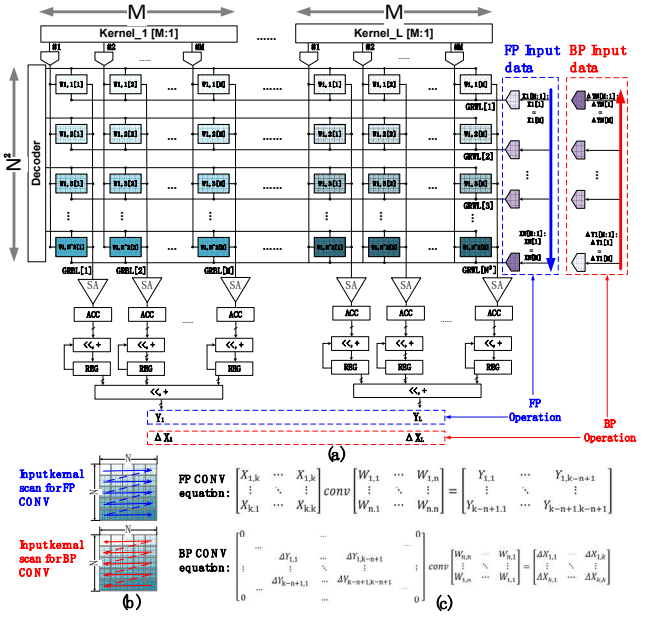


Fig. 5. Proposed mapping method of CONV layers in the 2T-DRAM CIM array.

Fig.5 presents the proposed mapping method of CONV layers in the transpose 2T-DRAM CIM array for FP and BP processes without array-level transpose operations. For each $N \times N$ size CONV kernel of L channels with M-bit width weight vectors in the same layer, each element is flattened row-wise and stored in $N^2 \times M$ memory cells, as shown in Fig.5(a). Notably, each bit of an element is mapped in M columns from low to high bit. The CONV kernels of different channels can be mapped to different $N^2 \times M$ memory groups to provide parallel computing.

According to the scanning order, as shown in the direction of the red and blue arrows on the two $N \times N$ size CONV kernels in the Fig.5(b), the input data is fed to the GRWL, and passed into the weight matrix in a bit-serial fashion. For the FP process with weight matrix mapping as mentioned above, the input data is fed from the first order of the first row to the last row. As for the BP process, aiming to rotate the weight matrix 180° and keep the weight matrix stored in the memory array unchanged, the input data is changed from the first order of the last row to the first row by the input address decoder. As shown in Fig.5(a), each input data for both FP and BP processes corresponds to a row and is fed into the array in a bit-serial way. The multiplication results of each convolution kernel within the DRAM array are passed to the SAs and summed up by the digital logic in the peripheral circuits to complete a layer computation.

D. *Proposed mapping method of FC layers for 2T-DRAM CIM*

Fig.6 presents the proposed mapping method of FC layers in the transpose 2T-DRAM CIM array. For the FC layers in DNN, in order to reuse the weight matrices stored in DRAM arrays for both FP and BP processes, the transpose characteristics of the weight matrices in FC layers during BP as shown by (2) and (4) can be exploited for in-situ matrix transpose operations. Combined with the proposed transpose circuitry as shown in Fig.4, the weight matrices are split in a manner of bit-serial slices and mapped into the transpose DRAM sub-arrays to achieve parallel computation.

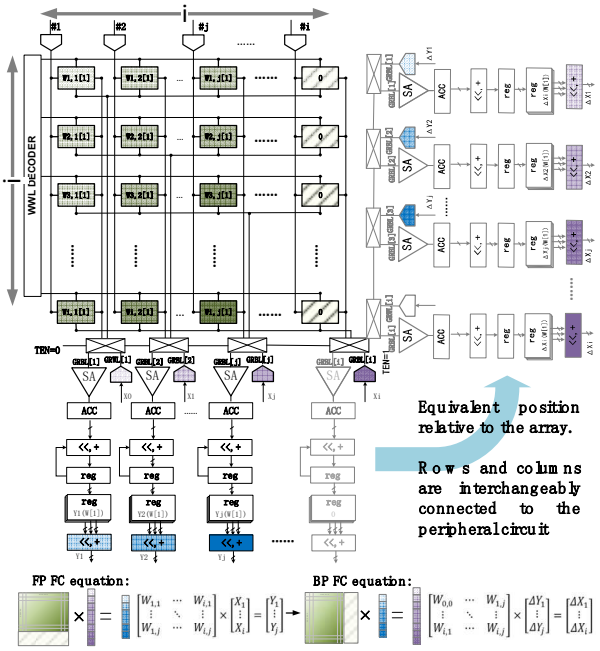


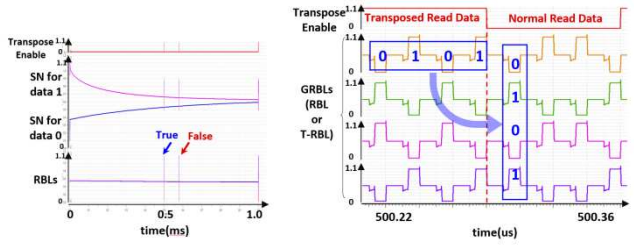
Fig. 6. Proposed mapping method of full-connection layers in the 2T-DRAM CIM array.

As shown in Fig.6, the input data is transferred in a bit-serial mode from GRWL, and the bit-wise multiplication are performed within the DRAM array. The final accumulation and activation are done by the digital peripheral after the SAs. Thanks to the in-situ transpose operation in the array, the weight matrix is unchanged for both FP and BP processes, as the input data direction and output data direction are simply controlled by the TEN signal. Similar to the in-memory computation of the CONV layers, all data on the same row or column of the weight matrix can be computed with the input data in a high degree of parallelism.

III. Comparison and Discussion

To evaluate the retention time and transpose function of the proposed transpose 2T-DRAM array, a 4×4 transpose DRAM array as a case study is designed and simulated with a 40nm CMOS technology on Cadence Virtuoso. Fig.7 shows the circuit-level simulation results of retention time and

transpose function of the 4×4 transpose DRAM array. The retention time of the 2T-DRAM is up to 500 µs as shown in Fig.7(a). Fig.7(b) shows that the proposed 2T-DRAM array can correctly achieve transpose function within 500 µs. At 100 MHz, 25K times of 1-bit multiplications can be completed within 500 µs, and corresponding addition operations can also be completed in the peripheral computing circuit at the same time. Within the retention time, the macro can complete one FP and one BP operation of a typical layer (e.g., 256×10 FC layer), which enables a refresh-less training process for light-weight DNN networks.



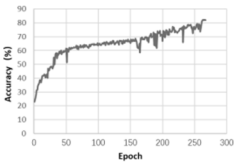
(a) (b)

Fig. 7. Transistor-level simulation waveforms: (a) Retention time of the 2T-DRAM cell; (b) Transposed and normal read function of a 4×4 transpose DRAM array at 500 µs.

In this study, the 4×4 transpose 2T-DRAM array is expanded to16K in function level to evaluate the efficiency of the proposed transpose 2T-DRAM based CIM method, in terms of the computing power and CNN on-chip-training accuracy. During a training cycle of the last FC layer with a size of 256×10 in a typical 4CONV+3FC network, the time of 1280 clock cycles required by 1-bit readout and calculation during FP process only occupy 5.12% of the retention time of the designed 2T-DRAM. Refresh requirement of the weight matrix for the FC layer during the subsequent BP process will interrupt the DNN computation, which introduces performance loss or power overhead. Based on the proposed transpose circuitry and mapping methods, the FC layers' weight matrix can be in-situ transposed without extra operations. Therefore, the weight matrix of FC layers and CONV layers can be directly read out without interruption of the DNN training, as shown in Fig.7(b). By using the proposed mapping methods that reuse weight matrix in the FP and BP process, an interruption-less DNN training can be achieved.

Layers	Filter Size	Input Size	Output Size	Input Channel	Output Channel	Filters
CONV1	3	32	32	3	32	32
CONV2	3	32	32	32	32	32
MaxPooling	2	32	16	32	32	32
CONV3	3	16	16	32	64	64
CONV4	3	16	16	64	64	64
MaxPooling	2	16	8	64	64	64
FC1	-	4096	512	-	-	-
FC2	-	512	256	-	-	-
FC3	-	256	10	-	-	-

(a)



(b)

Fig. 8. (a) Network structure information and (b) the Top-1 validation accuracy curve on CIFAR-10 dataset of an 8-bit 4CONV+3FC neural network.

In order to validate the efficiency of the proposed transpose DRAM-based CIM method, we have evaluated the Top-1 CIFAR-10 classification accuracy by using an 8-bit input and weight neural network with 4 CONV layers and 3 FC layers, as shown in Fig.8 (a). The classification accuracy of the 8-bit model can reach to 82.15%, as shown in Fig.8 (b).

Table I compares the computing power and CNN on-chip-training accuracy between the proposed 2T-DRAM array and prior CIM works based on Python simulation. Due to the achieved refresh-less feature and digital-domain computing, the proposed DRAM-based CIM method can implement fixed-point calculation of DNN model. We have trained the 4CONV+3FC fully quantized model by using symmetric affine quantization. The accuracy is better than the analog-domain DRAM-based CIM acceleration [4]. Furthermore, thanks to the proposed mapping methods to obtain the high parallelism, the overall computing power of the proposed 2T-DRAM method can achieve 2.4× GOPS @100 MHz against the method with the same array size in [4].

TABLE I. COMPARISON WITH PRIOR CIM WORKS IN TERMS OF COMPUTING POWER AND CNN ON-CHIP-TRAINING ACCURACY

CIM Solution	6T-SRAM [3]	7T-SRAM [2]	3T-DRAM [1]	1T1C-DRAM [4]	Proposed 2T-DRAM[a]
CIM domain	Digital	Digital	Analog	Analog	Digital
Array Size	80Kb	64Kb	8Kb	16Kb	16Kb
BP Support	Yes	Yes	No	No	Yes
Input Precision	8 bits	16 bits	4 bits	8 bits	8 bits
Weight Precision	8 bits	8 bits	4 bits	8 bits	8 bits
Params.	NA	NA	NA	2.168 M	2.296 M
Model	ResNet-20	4CONV+1FC	VGG16	4CONV+2FC	4CONV+3FC
Dataset	CIFAR-10	LFW	CIFAR-10	CIFAR-10	CIFAR-10
Accuracy	91.94 %	83.9 %	90.6 %	80.1 %	82.15%
GOPS	NA	NA	NA	4.71	11.26

a. Estimation based on Python, Top-1 accuracy.

IV. CONCLUSION

In this paper, a transpose 2T-DRAM based CIM architecture is proposed for CIM on-chip training and inference, which can effectively implement the BP algorithm in the DRAM arrays with minimum changes in peripheral circuits. Circuit simulation results show that the proposed 2T-DRAM CIM solution can achieve high precision and high reliability by efficient digital-domain DNN acceleration with weight reuse and high parallelism. The proposed transpose 2T-DRAM architecture provides an efficient design solution for on-chip training and inference in CIM-based DNN acceleration.

REFERENCES

[1] C. Yu et al., "A logic-compatible eDRAM compute-in-memory with embedded ADCs for processing neural networks," *IEEE Tran. Circuits and Systems I: Regular Papers*, vol. 68, no. 2, pp. 667-679, Feb. 2021.

[2] K. Bong et al., "A low-power convolutional neural network face recognition processor and a CIS integrated with always-on face detector, " *IEEE J. Solid-State Circuits*, vol. 53, no. 1, pp. 115–123, Jan. 2018.

[3] J.-W. Su et al., "Two-way transpose multibit 6T SRAM computing-in memory macro for inference-training AI edge chips, " *IEEE J. Solid-State Circuits*, vol. 57, no. 2, pp. 609–624, Feb. 2022.

[4] S. Xie et al., "eDRAM-CIM: compute-in-memory design with reconfigurable embedded-dynamic-memory array realizing adaptive data converters and charge-comain computing," *Proc. IEEE Intl. Solid-State Circuits Conf.*, pp. 248-250, 2021.

[5] Z. Chen et al., "A 65nm 3T dynamic analog RAM based computing-in-memory macro and CNN accelerator with retention enhancement, adaptive analog sparsity and 44TOPS/W system energy efficiency, " *Proc. IEEE Intl. Solid-State Circuits Conf.*, pp. 240–242, Feb. 2021.

[6] J. Xu et al., "In Situ aging-aware error monitoring scheme for IMPLY-based memristive computing-in-memory systems," *IEEE Trans. Circuits and Systems I: Regular Papers*, vol. 69, no. 1, pp. 309-321, Jan. 2022.

[7] R. Giterman et al., "An 800-MHz mixed-VTH 4T IFGC embedded DRAM in 28-nm CMOS bulk process for approximate storage applications," *IEEE J. Solid-State Circuits*, vol. 53, no. 7, pp. 2136-2148, Jul. 2018.