Processing-in-Memory Designs Based on Emerging Technology for Efficient Machine Learning Acceleration

Bokyung Kim
Duke University
Electrical and Computer Engineering
Durham, NC, USA
bokyung.kim828@duke.edu

Hai "Helen" Li
Duke University
Electrical and Computer Engineering
Durham, NC, USA
hai.li@duke.edu

Yiran Chen
Duke University
Electrical and Computer Engineering
Durham, NC, USA
yiran.chen@duke.edu

ABSTRACT

The unprecedented success of artificial intelligence (AI) enriches machine learning (ML)-based applications. The availability of big data and compute-intensive algorithms empowers versatility and high accuracy in ML approaches. However, the data processing and innumerable computations burden conventional hardware systems with high power consumption and low performance. Breaking away from the traditional hardware design, non-conventional accelerators exploiting emerging technology have gained significant attention with a leap forward since the emerging devices enable processing-in-memory (PIM) designs of dramatic improvement in efficiency. This paper presents a summary of state-of-the-art PIM accelerators over a decade. The PIM accelerators have been implemented for diverse models and advanced algorithm techniques across diverse neural networks in language processing and image recognition to expedite inference and training. We will provide the implemented designs, methodologies, and results, following the development in the past years. The promising direction of the PIM accelerators, vertically stacking for More than Moore, is also discussed.

CCS CONCEPTS

• **Hardware** → **Emerging architectures**; **Memory and dense storage**; • **Computer systems organization** → *Data flow architectures*; • **Computing methodologies** → **Artificial intelligence**; **Machine learning**.

KEYWORDS

Processing-in-Memory, Accelerators, Emerging Technology, Memristor, Deep Learning

ACM Reference Format:
Bokyung Kim, Hai "Helen" Li, and Yiran Chen. 2024. Processing-in-Memory Designs Based on Emerging Technology for Efficient Machine Learning Acceleration. In *Great Lakes Symposium on VLSI 2024 (GLSVLSI '24), June 12–14, 2024, Clearwater, FL, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649476.3660367

GLSVLSI '24, June 12–14, 2024, Clearwater, FL, USA

ACM ISBN 979-8-4007-0605-9/24/06
https://doi.org/10.1145/3649476.3660367

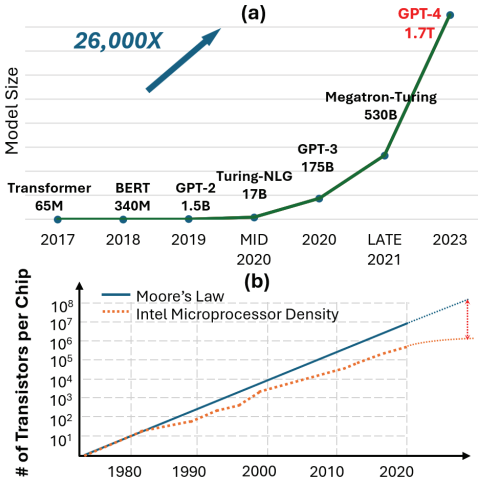

Figure 1: (a) Dramatically increasing model size in recent years and (b) the end of Moore's law

1 INTRODUCTION

Numberless decent applications on edges stem from artificial intelligence (AI)-based technology. Machine learning (ML) techniques leveraging deep neural networks (DNNs) are considered the key to the bright future of AI, as they have been verified through diverse applications from spatial processing (e.g., image recognition) to sequential processing (language and bioinformatics), even both like video-based applications. However, the model size is unprecedentedly increasing under the development of algorithms. As shown in Figure 1(a), the emergence of large language models (LLMs) expedites the model size increase, and accordingly state-of-the-art deep learning needs trillions of parameters.

The increasing size of DNN models, however, is not affordable in hardware due to two fundamental limitations: the end of Moore's law and memory wall—also known as von Neumann bottleneck. Moore's law was found by empirical observation by a co-founder of Intel, Gordon E. Moore: a doubling in transistor density every two years. The law stood firm for many decades, indicating remarkable technology node down-scaling. Yet a gap between the projection Moore made and industry capabilities at the time began to form around 2000, as Figure 1(b) displays. Researchers around the globe reached a consensus that the end of Moore's law was approaching after the 2000s. Furthermore, the other limitation, memory wall, was caught in the existing computer architecture, which is based on von Neumann architecture separating the memory and computing units. The heterogeneity in the von Neumann architecture

Table 1: Characteristics according to different memory types, including DRAM, SRAM, RRAM, PCM, and MRAM [16].

	DRAM	SRAM	RRAM	PCM	MRAM
Cell structure	1T1C	6T	1T1R	1T1R	1T1R
Cell size	$6F^2$	$>100F^2$	4-$12F^2$	4-$30F^2$	6-$50F^2$
Write voltage	<1V	<1V	<3V	<3V	<1.5V
W/R latency	<10ns	0.3ns	10ns	50ns	20ns
Write energy	~5fJ	~0.7fJ	~0.1pJ	~10pJ	~0.1pJ
Endurance	$> 10^{16}$	$> 10^{16}$	$< 10^{12}$	$< 10^{9}$	$> 10^{15}$

limits the computational performance improvement because of the speed mismatch between the computing and data transfer [9]. To overcome the memory wall in the post-Moore era, we need a new paradigm in hardware.

Fortunately, the emerging memory technology, discovered under the name of *memristor*, has expedited the development of a novel hardware paradigm. The memristor is considered a prominent solution as a memory device for its brilliant properties, such as non-volatility. Unlike the volatile conventional memories, the non-volatility remarkably reduces power consumption by holding written data without a power supply. Also, the memristor is advantageous in area efficiency and multi-level storage capability. Moreover, the metal-insulator-metal (MIM) structure enables the form of a crossbar architecture, which is actively exploited in process-in-memory (PIM) accelerators.

PIM is an effective solution to tackle the memory wall by integrating memory and computing models into one unit. The memristor-based PIM particularly empowers the efficient execution of matrix processing because of its crossbar architecture parallelizing computations. Correspondingly, memristor-based PIM accelerators have been actively researched for DNN models to alleviate the movement of numerous parameters across heterogeneous units with high-performance computing through high parallelism.

In this paper, we present a summary of PIM hardware designs based on emerging technology for ML acceleration in the past decade. Memristor-based PIM accelerators have made progress by chasing the development of algorithms. Its potential in ML acceleration was demonstrated with traditional models and has been extended to various DNN models with inference and training processes. The advanced algorithm techniques to leverage the inherent sparsity of DNNs started being considered for accelerator designs. Then, different dataflow was first introduced and considered in a recent year. Beyond such a trace of memristor-based PIM accelerators, we discuss one of the most promising technologies in memory-based accelerators for the future, vertically stacked 3D, and its status quo.

2 EMERGING NONVOLATILE TECHNOLOGY

Memristor was defined in the early 1970s as a fourth element, which had been missing for a long [4, 5]. Then, a memristor device was first realized by the nano-scale thin-film structure in 2008 [19]. The device remembers data as resistance (memristance), even without power, which could reduce remarkable power consumption. As a result, it has been considered a promising technology for the future, named emerging nonvolatile memory (eNVM). Based on the potential, various eNVMs have been introduced with different

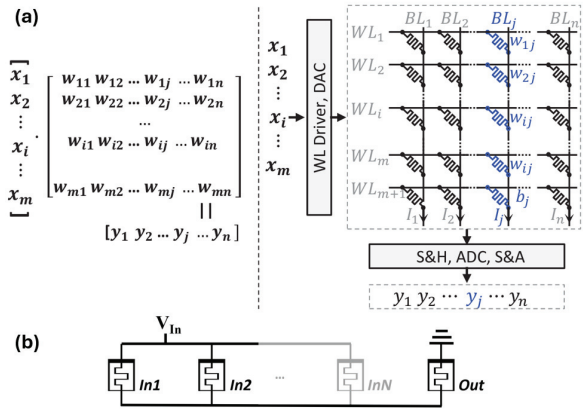

Figure 2: (a) VMM and its implementation in RRAM-based crossbar hardware [1] and (b) digitally implemented logic computation methodology [10], *MAGIC* [14].

materials and properties, including resistive, phase-change, and magnetoresistive RAM (RRAM, PCRAM, and MRAM). Along with the non-volatility, eNVMs have desirable properties as memory technology. It can be confirmed in Table 1, which compares the conventional memories and eNVMs. Conventional memories sacrifice resources noticeably. Specifically, DRAM slowly reads and writes data, and SRAM needs large areas. On the other hand, eNVMs show small area occupation, high efficiency, and even multi-bit storage [2]. Among eNVM devices, RRAM is one of the particularly compelling future devices because of its device excellence (Table 1).

As memristive devices, RRAMs have a metal-insulator-metal (MIM) structure, where a metal oxide layer is sandwiched between a top electrode (TE) and a bottom electrode (BE). Voltages with a proper pulse width or magnitude over a threshold program the device resistance (memristance). The resistance level accordingly programmed represents data in a cell. It is noteworthy that the MIM structure has motivated brain-inspired hardware design through the crossbar array form. As illustrated in Figure 2(a), the array is structured by placing devices at every cross-point. Thanks to Ohm's and Kirchhoff's Law, RRAM crossbar arrays accelerate vector-matrix multiplication (VMM) operations, which are dominant in neural network models.

While multi-level representation is available in RRAM for efficient computing, the analog implementation could deteriorate the efficiency because of reliability issues from non-ideal device properties. Enough margin by using RRAM in digital can guarantee reliable results, which can be achieved by reading the data with two states, low and high resistant states (LRS and HRS) expressing '1' and '0', respectively. For read operations, a voltage under the writing threshold is applied across two electrodes of the device, and the detected current obtained from the device indicates the device state, that is, the stored logic value in the cell. This digital usage of RRAM devices can be extended to logic computing in RRAM, as described in Figure 2(b). Although the device is interpreted digitally, the multiplied result at each cross-point is accumulated with the current from other cells and it requires power-hungry peripheries like converters. That's why logic computation with RRAMs has been investigated as one of the implementation methodologies,

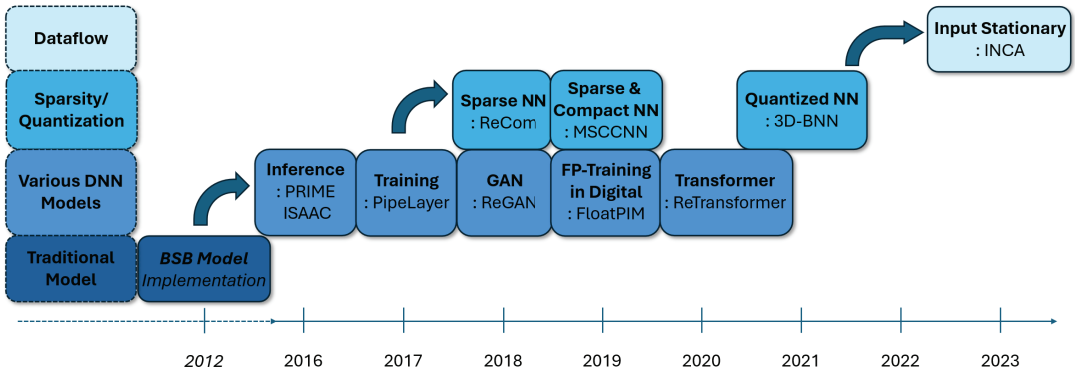

Figure 3: The development of RRAM-based PIM accelerators.

under the name of memristor-aided logic (MAGIC) [14] based on the voltage divider rule.

3 PROPOSED ACCELERATOR DESIGNS

Before the DNNs boom, traditional models ignited machine learning research. There have been efforts to implement the traditional models in hardware accordingly. Specifically, M. Hu et al. [6] explored the brain-state-in-a-box (BSB) model by realizing the synaptic weighting function of RRAM. Because of the intrinsic positiveness of devices, the design implements two memristive arrays to express real numbers. Parallel computing through two RRAM arrays expedites VMM and the calculated results are passed to the subsequent components to process the BSB model computations. RRAM-based hardware implementation with PIM is continually researched for diverse learning models, including unsupervised learning [11]. Beyond the traditional and relatively simple algorithms, the early exploration became the foundation of DNN accelerators. According to the explosive DNN algorithms, RRAM-based PIM accelerators have been proposed with diverse design objectives, from inference/training to the consideration of dataflow, as described in Figure 3.

3.1 Various models in inference/training

The successful examples of DNN implementation led to RRAM-based PIM accelerator designs for various models. Convolution neural network (CNN) has been one of the most successful algorithms because of its performance in image classification since the convolution operations of LeNet-5 successfully classified the hand-written dataset. The effectiveness of convolution operations in image feature extraction brings about diversified CNN architectures, which outperform even human-level accuracy.

Inference: The need for hardware accelerators for CNN models also ignites the PIM designs for high performance. PRIME [3] and ISAAC [17] are representative designs to accelerate the inference of CNNs. By unrolling the 2D weight parameters in one kernel in CNNs, RRAM arrays efficiently perform the convolution operations. Both accelerators implemented other functions by proper peripheral components. Specifically, PRIME presents a full-stack solution including hardware architecture and software interface to support in-situ computation of neural networks in main memory, resulting in four orders of magnitude improvement over conventional CPU in speedup and energy saving. In ISAAC, the read-out, sample and hold, analog-digital conversion, and post-processing are scheduled in a pipeline manner in each compute tile to provide higher throughput. ISAAC outperforms a previous accelerator, which reported 450× speedup and 150× lower energy than NVIDIA K20M GPU, with a 14.8× higher throughput on average.

Training: In addition to the feed-forward process in inference, training further involves backpropagation and weight update procedures. Correspondingly, training is more compute-intensive than inference, and the witness of the successful hardware implementation of the inference process also stimulates training accelerators. The call for training accelerators brought attention to PipeLayer [18], which realizes the essential computations in training. PipeLayer analyzes the training requirements and then introduces the pipelining scheme for efficient DNN training, as displayed in Figure 4, by adding weight update into the computation pipeline and resolving the dependencies in training. The RRAM arrays are designed as morphable subarrays and memory subarrays to perform two functions, computation and storage, respectively. Subarrays configured into memory mode store the activations for calculating the errors in the backpropagation. For batch computing of multiple input samples, PipeLayer executes an intra-pipeline-fashioned process with the samples. The intra-pipelining is feasible because weight update is conducted at the end of each batch process and the samples during the batch process show no dependence. PipeLayer shows 42.45× speedup and 7.17× energy saving compared with a GPU platform.

GAN training: The competitive results from DNN models devised a fortified training methodology through an unsupervised paradigm in *generative adversarial networks (GANs)*. As illustrated in Figure 5(a), GAN is based on an adversarial game between a generator (G) and a discriminator (D), which are typically modeled by DNNs and trained simultaneously. Generally, G produces high-quality samples from noise to simulate the distribution of authentic data samples, and D is an optimized binary classifier trained by distinguishing between real and generated samples. While GANs are powerful, the two DNNs impose a main challenge in GAN acceleration due to the high demand for computing resources and frequent data transfer. Furthermore, GANs exploit more computing phases through highly complex operations, when compared to traditional supervised deep learning. From the challenges, ReGAN

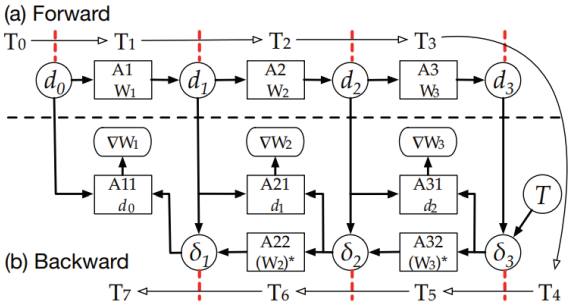

Figure 4: The pipelined training in PipeLayer [18].

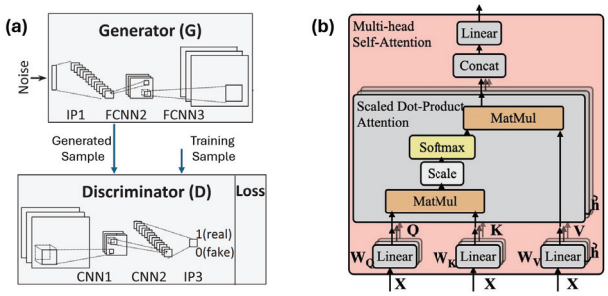

Figure 5: Various compute- and memory-intensive algorithms, such as (a) GAN based on CNN architectures [1] and (b) multi-head-attention structure enabling LLMs [22].

[1] leverages RRAM PIM effectiveness in VMM high-performance computing. ReGAN partitions RRAM memory arrays into three regions. One is memory subarrays to purely save data, another is full-function subarrays for on-site VMM and data storage, and the last one is buffer subarrays for intermediates between layers, including generated images, data required to compute partial derivatives, etc. To avoid wasting the limited bandwidth of memory subarrays by buffer accesses, ReGAN connects the buffer and full-function arrays by high-throughput private local data ports. Also, ReGAN adopts the spike-based scheme for smooth communications across RRAM subarrays and peripheries to minimize the energy overhead by power-hungry analog-digital converters (ADC). The non-linear activation function is conducted by utilizing a look-up table, and other peripheral units, such as subtractors and shift-and-adds, are implemented to support various computations in training. ReGAN reports 240× speedup and 94× energy saving than GPU on average.

FP training in digital: However, the efficiency of RRAM arrays in VMM computing improves fixed-point computing but is hardly realizable in floating-point computations. Moreover, previous designs heavily relying on analog VMM computing necessitate power-hungry components converting digital to analog or vice versa, as the accumulated currents from an RRAM array should be interpreted in digital to communicate with other units. FloatPIM [7] executes DNNs with floating-point (FP) computing for the first time without ADC/DACs by leveraging digital implementation methodology in PIM. The MAGIC scheme removes the need for converters for required computations and operations by propagating them digitally. More importantly, the logic-based implementation enables floating-point computing of complex operations.

Transformer: Beyond the image applications, the huge success of large language models (LLMs) makes fast progress by producing superior algorithms like GPT and BERT model series. The multi-head-attention model (Figure 5(b)) becoming the basis of the LLM success shows its effectiveness in language processing, and ReTransformer [22] is a design for the transformer in a pipeline fashion by reordering the data generation and the computing process in scaled dot-product attention layer between W_Q and W_K. The inference starts with the initialization of two RRAM arrays with W_Q and W_K, respectively. Following the original flow, two weight matrices (W_Q and W_K) are computed with input matrices to generate Q and K, which will multiply the Q and K^T intermediate matrices. RRAM crossbars, therefore, hold one of the results Q and K for the dot product operation between Q and K.

Instead, ReTransformer proposes to reorder the original computation flow into two cascaded steps. After computing the Q matrix first, it is multiplied with the weight matrix (W_K), rather than computing the K matrix separatively. Then, the generated matrix is multiplied by the input matrix. To prevent frequent writing operations of intermediate values into RRAM arrays, ReTansformer performs the optimized computing flow in a pipeline fashion. Initialization of RRAM arrays with weight matrices at the beginning is followed by two stages, to multiply weights and inputs for Q and to compute intermediate results by multiplying the generated Q and weights. In the meantime, other RRAM crossbar arrays are initialized with inputs so that the intermediate results can be performed with the input matrix to shorten the computing time. The evaluation displays the improved efficiency of ReTransformer with the proposed reordering and pipelining scheme by up to 80%.

3.2 Hardware support for compressed ML

Because of the increasing model size, advanced algorithm techniques have emerged, i.e., sparsification [21] and quantization [15]. DNN algorithms have inherent sparsity, referring to the radio of zeros, in activations and weights because of non-linear functions, etc. However, the irregular distribution of the zero values cannot be exploited to accelerate the learning process for its unpredictability. Consequently, intended sparsification through pruning insignificant values.

Sparse NN: ReCom [8] is an RRAM-based efficient PIM architecture to better resource utilization in compressed NN by leveraging the sparsity in weights and activations. Weights are compressed through structured sparsification, and non-zero activations corresponding to the non-zero weights are fetched through an optimized unit, structurally weight-oriented fetching (SWOF). In-layer pipeline for memory and computation (IMPC) further speeds up the process, while skipping redundant computations in case of all-zero activations. Experiments verify that ReCom can shorten the computation time by up to 10.66× and improve energy efficiency by up to 9.43×, compared to the early introduced accelerator, PipeLayer. Memristor-based sparse compact CNN (MSCCNN) [20] further saves resource consumption by compact models and sparsification through pruning. Point-wise and depth-wise convolutions, which make compact models, are utilized to replace fully-connected and vanilla convolution layers, respectively. Then, MSCCNN applied pruning to remove redundancy and pursue further compact

network structures. The evaluation results prove excellent hardware efficiency improvement and even superior accuracy levels by MSCCNN.

Quantized NN: In addition, low-precision computing is induced by the quantization of high-precision activations/parameters to reduce the overhead in the hardware and learning process. As a result, quantized DNNs preserve accuracy with much smaller overheads in hardware Kim et al. proposed a 3D RRAM design for a binarized neural network (3D-BNN), an outcome of an extreme quantization of weight parameters [10]. The authors noticed that MAGIC-based computing consumes numerous RRAM devices and cycles for intermediate values because multiplication and addition need several logic-computing despite one-bit operations. Driven by the motivation, the authors propose simplified logic-based BNN computations and their efficient implementation in 3D RRAM hardware.

BNN maintains high accuracy with great savings in parameters at the algorithm level. Moreover, the adoption of binarized weights eliminates the requirement of multiplication and necessitates only additions. However, the original logic-based addition demands numerous intermediates and corresponding devices. Therefore, the authors propose a simplified logic-based full-adder scheme to shorten the computing latency and device consumption, recognizing that the outcome of a logic-based full-adder depends on a specific term in the equation. After that, its implementation is performed with a 3D RRAM structure for parallel computing. Half of the operands are saved into stacked layers with different assumptions regarding carry-in. According to the real carry-out value, the corresponding layer is selected for the computation propagation. In such a design, 3D-BNN shows no accuracy loss from the original BNN but also saves time and the number of RRAMs in PIM by up to 6×.

3.3 Dataflow

The fact that data movements to off-chip components are expensive leads to an in-depth discussion on dataflow in other types of accelerators based on systolic arrays, GPUs, etc. In contrast, the previous RRAM-based PIM accelerators have been built on one dataflow, *weight-stationary (WS) dataflow*, by keeping weights in PIM RRAM arrays and fetching/saving inputs (activations) to external memory units. It has been adopted because weight parameters are static values while activations are dynamic. However, catching that dataflow is one of the most critical factors determining the accelerator efficiency in PIM also, INCA [13] investigated and raised a question about the efficiency of WS for the first time. According to the observations in INCA, WS has four fundamental limitations in hardware. First, as shown in Figure 6(a), WS needs to move the same data according to the layer propagation in DNN because of a unique feature of NN—activations of a layer became inputs of its next layer. Besides, dataflow affects the number of required devices, RRAM array utilization, and accuracy, following insights provided by INCA. WS needs more RRAM cells than IS because of the larger weight parameter size in inference and the unable recycling of weight parameters in training. Also, when executing compact DNN models, coarsely embedded RRAM arrays in WS show low utilization. Accuracy is dropped in WS because of RRAM non-ideal properties because it turned out that accuracy leans on weight values, which are impacted by the real effects of RRAM devices in WS.

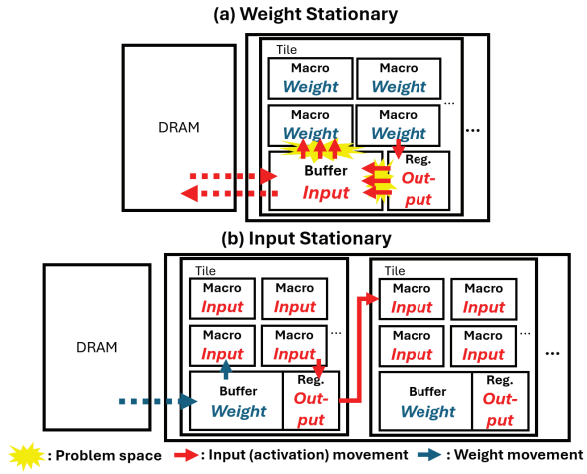

Figure 6: Comparison between (a) WS and (b) IS dataflow proposed in INCA [13].

Input Stationary: Instead, INCA enforces an input-stationary dataflow by placing inputs (activations) in PIM modules and transferring weight parameters from/to external memories. Accordingly, activations can be immediately consumed for the next layer like Figure 6(b), without saving and fetching them into external memories; that is, INCA reduces redundant accesses needed by activations in DNN. The less input data and recycling capability save a remarkable number of RRAM; embedded RRAM arrays in INCA are fully utilized regardless of model size; and accuracy is meagerly affected by input (activation) variation due to device noise. As the prevalent RRAM array design increases the number of redundant device cells, INCA proposes a two-transistor-one-RRAM (2T1R)-based array structure for direct (one-shot) convolution, as illustrated in Figure 7(a). The proposed design conducts one convolution per operation, losing high parallelism. Therefore, the authors stacked 64 crossbar layers horizontally to preserve high parallel computing. The first IS-implemented hardware shows noticeable improvement in energy efficiency (20.6× and 260× in inference and training on average), and speedup (4.8× and 18.6×)—even three orders of magnitude improved in case of compact models.

4 STACKING FOR FUTURE

The ever-sizing models and their complexity need much-shortened signal distance to prevent data transfer delay and signal loss as well as to pursue the maximized area density. The limitation in integrating more transistors per unit chip area turns the attention in academia and industry to stacking technologies. As proposed in previous designs [10, 13] (Figure 7), 3D stacking technology has been applied for RRAM accelerators. The most straightforward methodology for stacking, namely heterogeneous 3D stacking technology (H3D), is using through-silicon vias (TSVs). The previously proposed 3D PIM designs are also based on H3D, where individual dies manufactured separately are bonded by TSVs after a punching process. However, H3D does not minimize the area density because it should secure the space for thick TSVs and micro-bumps for bonding. Moreover, the heterogeneously stacked planes need accurate and precise alignments.

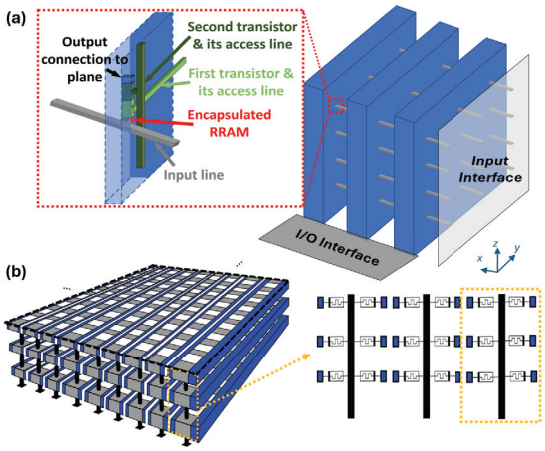

Figure 7: Diverse 3D RRAM architectures proposed in (a) INCA [13] and (b) efficient 3D RRAM for BNN [10].

Alternatively, monolithic 3D (M3D) design has emerged as a breakthrough for future 3D technologies [12]. Distinguished from the separate fabrication of each tier, dies in M3D integration are sequentially fabricated. Sequential stacking employs monolithic inter-tier vias (MIVs) whose dimensions are similar to local vias; therefore, it maximizes the area density. Also, the use of MIVs in M3D instead of TSVs mitigates the alignment issues in H3D. RRAM is also proved for the M3D architecture; Tang and Xu et al. successfully manufactured a 3D PIM system using stacked 1T1R arrays, taking advantage of M3D [23]. Each of the three stacks in the design serves different roles: standard CMOS logic functions in the first die and the following two 1T1R arrays for computation (second die) and data storage (third die). Upper stacks are fabricated with a low-temperature process because high temperatures would degrade the functionality of lower dies. Likewise, a tacked architecture can have more layers in the M3D technology. Additionally, different RRAM materials tailored for corresponding goals of different tiers are used in the design, i.e., hafnium oxide (second layer) and tantalum oxide (third) The M3D PIM chip was evaluated with representative DNN algorithms like ResNets in CNNs. While the classification results were comparable to a GPU, the M3D RRAM PIM chip spent around 49× less time and 39× less energy than the baseline GPU.

5 CONCLUSION

This paper presented an overview of RRAM-based PIM accelerators in the past decade. Behind the great success of ML, increasing model size, resource-restrained edges, and limitations in hardware call for a new paradigm, PIM. RRAM-based PIM acceleration has been researched to support advanced ML model designs and techniques, and we specifically summarized RRAM-PIM designs for inference and training of various models, compressed ML, and dataflow. The development of PIM accelerators is in progress with advanced hardware architecture with 3D stacking technology for the future.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation (NSF) 2328805 and 2112562.

REFERENCES

[1] Fan Chen, Linghao Song, and Yiran Chen. 2018. ReGAN: A pipelined ReRAM-based accelerator for generative adversarial networks. In *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 178–183.
[2] Yiran Chen, Xiaobin Wang, Wenzhong Zhu, Hai Li, Zhenyu Sun, Guangyu Sun, and Yuan Xie. 2010. Access scheme of multi-level cell spin-transfer torque random access memory and its optimization. In *2010 53rd IEEE International Midwest Symposium on Circuits and Systems*. IEEE, 1109–1112.
[3] Ping Chi, Shuangchen Li, Cong Xu, Tao Zhang, Jishen Zhao, Yongpan Liu, Yu Wang, and Yuan Xie. 2016. Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory. *ACM SIGARCH Computer Architecture News* 44, 3 (2016), 27–39.
[4] Leon Chua. 1971. Memristor-the missing circuit element. *IEEE Transactions on circuit theory* 18, 5 (1971), 507–519.
[5] Leon O Chua and Sung Mo Kang. 1976. Memristive devices and systems. *Proc. IEEE* 64, 2 (1976), 209–223.
[6] Miao Hu, Hai Li, Qing Wu, Garrett S Rose, and Yiran Chen. 2012. Memristor crossbar based hardware realization of BSB recall function. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–7.
[7] Mohsen Imani, Saransh Gupta, Yeseong Kim, and Tajana Rosing. 2019. Floatpim: In-memory acceleration of deep neural network training with high precision. In *Proceedings of the 46th International Symposium on Computer Architecture*. 802–815.
[8] Houxiang Ji, Linghao Song, Li Jiang, Hai Li, and Yiran Chen. 2018. ReCom: An efficient resistive accelerator for compressed deep neural networks. In *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 237–240.
[9] Bokyung Kim, Zhixu Du, Jingwei Sun, and Yiran Chen. 2023. Towards the Efficiency, Heterogeneity, and Robustness of Edge AI. In *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. IEEE, 1–7.
[10] Bokyung Kim, Edward Hanson, and Hai Li. 2021. An efficient 3d reram convolution processor design for binarized weight networks. *IEEE Transactions on Circuits and Systems II: Express Briefs* 68, 5 (2021), 1600–1604.
[11] Bokyung Kim and Hai Li. 2020. Leveraging 3d vertical rram to developing neuromorphic architecture for pattern classification. In *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE, 258–263.
[12] Bokyung Kim and Hai Li. 2023. Monolithic 3D stacking for neural network acceleration. *Nature Electronics* 6, 12 (2023), 937–938.
[13] Bokyung Kim, Shiyu Li, and Hai Li. 2023. Inca: Input-stationary dataflow at outside-the-box thinking about deep learning accelerators. In *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 29–41.
[14] Shahar Kvatinsky, Dmitry Belousov, Slavik Liman, Guy Satat, Nimrod Wald, Eby G Friedman, Avinoam Kolodny, and Uri C Weiser. 2014. MAGIC—Memristor-aided logic. *IEEE Transactions on Circuits and Systems II: Express Briefs* 61, 11 (2014), 895–899.
[15] Fangxin Liu, Wenbo Zhao, Zongwu Wang, Yilong Zhao, Tao Yang, Yiran Chen, and Li Jiang. 2022. IVQ: In-memory acceleration of DNN inference exploiting varied quantization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41, 12 (2022), 5313–5326.
[16] P Mannocci, M Farronato, N Lepri, L Cattaneo, A Glukhov, Z Sun, and D Ielmini. 2023. In-memory computing with emerging memory devices: Status and outlook. *APL Machine Learning* 1, 1 (2023).
[17] Ali Shafiee, Anirban Nag, Naveen Muralimanohar, Rajeev Balasubramonian, John Paul Strachan, Miao Hu, R Stanley Williams, and Vivek Srikumar. 2016. ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars. *ACM SIGARCH Computer Architecture News* 44, 3 (2016), 14–26.
[18] Linghao Song, Xuehai Qian, Hai Li, and Yiran Chen. 2017. Pipelayer: A pipelined reram-based accelerator for deep learning. In *2017 IEEE international symposium on high performance computer architecture (HPCA)*. IEEE, 541–552.
[19] Dmitri B Strukov, Gregory S Snider, Duncan R Stewart, and R Stanley Williams. 2008. The missing memristor found. *nature* 453, 7191 (2008), 80–83.
[20] Shiping Wen, Huaqiang Wei, Zheng Yan, Zhenyuan Guo, Yin Yang, Tingwen Huang, and Yiran Chen. 2019. Memristor-based design of sparse compact convolutional neural network. *IEEE Transactions on Network Science and Engineering* 7, 3 (2019), 1431–1440.
[21] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems* 29 (2016).
[22] Xiaoxuan Yang, Bonan Yan, Hai Li, and Yiran Chen. 2020. ReTransformer: ReRAM-based processing-in-memory architecture for transformer acceleration. In *Proceedings of the 39th International Conference on Computer-Aided Design*. 1–9.
[23] Yibei Zhang, Yijun Li, Jianshi Tang, Ningfei Gao, Lei Gao, Haitao Xu, Ran An, Qi Qin, Zhengwu Liu, Dong Wu, et al. 2023. 3D Stackable CNTFET/RRAM 1T1R Array with CNT CMOS Peripheral Circuits as BEOL Buffer Macro for Monolithic 3D Integration with Analog RRAM-based Computing-In-Memory. In *2023 International Electron Devices Meeting (IEDM)*. IEEE, 1–4.