A Review of In-Memory Computing Architectures for Machine Learning Applications

Sathwika Bavikadi*[1], Purab Ranjan Sutradhar*[2], Khaled N. Khasawneh[1],
Amlan Ganguly[2], Sai Manoj Pudukotai Dinakarrao[1*]
[1]Department of Electrical and Computer Engineering, George Mason University, Fairfax, VA.
[2]Department of Computer Engineering, Rochester Institute of Technology, Rochester, NY.
Email:{sbavikadi,kkhasawn,spudukot}@gmu.edu,{ps9525,axgeec}@rit.edu

ABSTRACT

The state-of-the-art traditional computing hardware is struggling to meet the extensive computational load presented by the rapidly growing Machine Learning (ML) and Artificial Intelligence (AI) algorithms such as Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs). In order to obtain hardware solutions to meet the low-latency and high-throughput computational demands from these algorithms, Non-Von Neumann computing architectures such as In-memory Computing (IMC)/ Processing-in-memory (PIM) are being extensively researched and experimented with. In this survey paper, we analyze and review pioneer IMC/PIM works designed to accelerate ML algorithms such as DNNs and CNNs. We investigate different architectural aspects and dimensions of these works and provide our comparative evaluations. Furthermore, we discuss challenges and limitations in IMC research and also present feasible directions based on our observations and insight.

CCS CONCEPTS

• **Computing methodologies** → *Machine learning*; • **Hardware** → **Hardware accelerators**; *Programmable logic elements.*

KEYWORDS

In-memory Computing, Processing-in-memory, Non Von-Neumann Architectures, Machine learning, CNN, DNN, Artificial Intelligence

ACM Reference Format:

Sathwika Bavikadi, Purab Ranjan Sutradhar, Khaled Khasawneh, Amlan Ganguly, Sai Manoj Pudukotai Dinakarrao. 2020. A Review of In-Memory Computing Architectures for Machine Learning Applications. In *Proceedings of the Great Lakes Symposium on VLSI 2020 (GLSVLSI'20), September 7–9, 2020, Virtual Event, China. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3386263.3407649*

1 INTRODUCTION

Machine learning (ML) algorithms are finding their way into every single branch of scientific research lately. Deep Neural Networks (DNN), particularly the Convolutional Neural Networks (CNN) are by far the most widely used genre of ML algorithms and are widely used in the fields of computer vision, pattern recognition, voice recognition, natural language processing. In recent years, CNNs have achieved revolutionary accuracy at recognizing images that surpasses human capability [20][10]. However, such a level of accuracy also comes with substantial computation workloads. On the other hand, the state-of-the-art processing hardware is suffering from performance bottleneck due to low-bandwidth and high-latency data communication with off-chip memory which is popularly addressed as the 'memory wall' [22]. Therefore, an increasing amount of research in alternative Non-Von Neumann computer architectures such as in-memory and near-memory computing is being carried out in order to overcome the existing challenges of traditional Von Neumann architecture. In-Memory Computing (IMC) or Processing-in-Memory (PIM) bridges the separation between memory and processor by implementing processing units inside the memory chip itself [6][25]. Due to fabrication limitations of the memory chips, these processing elements can achieve only limited functionalities [7]. However, they can leverage the bitline level parallelism inside memory chip to construct a massively parallel SIMD-like processing architecture that also enjoys the very high-bandwidth and low-latency data communication through the memory bitlines. In-memory computing makes a viable candidate for implementing ML applications, most importantly the CNN algorithms in which convolution operations alone accounts for the largest fraction of a CNN computation time when implemented in GPUs and CPUs [20][16]. In-memory Computing architectures can achieve dramatic performance optimization at unmatched power efficiency in CNN/DNN applications.

A vast amount of research has been carried out over the past decade in designing CNN/DNN inference and training machines on different memory platforms including the traditional memory platforms of Static and Dynamic Random Access Memory (SRAM & DRAM) as well as novel non-volatile Resistive RAM (ReRAM), Phase-changing Memory (PCM) and Magnetic RAMs such as Spin Transfer Torque MRAM (STT-MRAM) and Spin Orbit Torque MRAM (SOT-MRAM) technologies. It has been found that a satisfactory level of accuracy can be retained even despite performing various levels of quantization/down-scaling of data parameters in CNN algorithms [1, 9, 11, 12, 33]. This opens up an exploration space for high performance and low power CNN implementations for IoT and Mobile applications. The IMC paradigm is also capitalizing on this data down scaling technique [12, 13, 15, 20, 21, 23, 26, 27, 30, 31].

In this work, we endeavour to present a comprehensive overview of the research into ML-oriented IMC/PIM designs. We analyze and classify several ML-application oriented IMC/PIM papers based

**Both authors contributed equally to this research.

GLSVLSI '20, September 7–9, 2020, Virtual Event, China

ACM ISBN 978-1-4503-7944-1/20/09…$15.00
https://doi.org/10.1145/3386263.3407649

on their architectural properties, functionalities and their impacts on the system performance and overheads. Lastly, we present the existing challenges and the future directions for this field of research.

2 MULTI-DIMENSIONAL DESIGN-SPACE EXPLORATION FOR IMC

Before delving into the analysis of specific IMCs with ML applications, we first introduce different design space variables that play a pivotal role in IMC designs *i.e.,* classification w.r.t. the memory platform, in-memory processing method, data mapping scheme, hardware interfacing and the degree of quantization of data here. Figure 1 presents a categorization of the IMCs based on memory platform, the computing architecture and hardware interfacing method for the IMCs discussed in this paper.

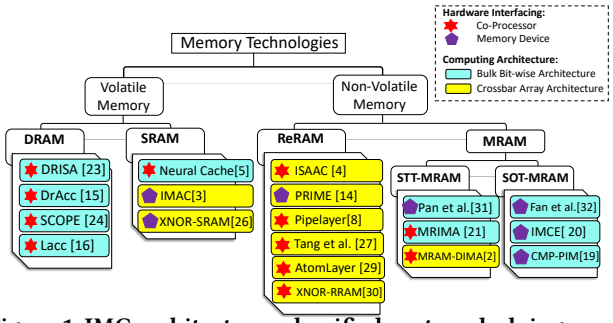


Figure 1: IMC architectures classified w.r.t. underlying memory platform and other design parameters

2.1 Memory Platforms for IMC

Given the importance of memory technology in the design of IMC systems, we first categorize IMC architectures w.r.t. the memory technology that the IMCs are built on. The memory technologies can be classified into: Conventional memory (SRAM & DRAM) and Emerging memory (ReRAM, STT-MRAM, SOT-MRAM).

2.1.1 ***Conventional Memory Technologies:*** Dynamic random access memory (DRAM) and Static random access memory (SRAM) feature charge-storage based cells and therefore are volatile in nature. Among the IMCs reviewed in this paper, [15, 16, 23, 24] are built on DRAM platform. The SRAM based ones are [3, 5, 26].

2.1.2 ***Emerging Memory Technologies:*** ReRAM, MRAMs (STT-MRAM) & Spin orbit torque MRAM (SOT-MRAM) etc. classify as novel memory technologies as they feature non-volatile resistive memory cells. They have multiple resistance states that may be leveraged to store or process binary (MRAMs) or analog (Re-RAM) data and therefore are non-volatile in nature. Among the IMCs reviewed in this paper, [4, 8, 14, 27, 29, 30] are built on ReRAM. STT-MRAM based IMCs are [2, 21, 31] and SOT-MRAM based ones are [19, 20, 32].

2.2 In-Memory Computing Architectures

2.2.1 ***Bulk bitwise Computing:*** Bulk bitwise computing is the most dominant IMC architecture [5, 15, 19–21, 23, 24, 31], shown in Figure 2(a). The logic operations are fundamentally bitwise: both the operands and outputs after the operations are placed along the same bitline (column). Memory rows containing operands are activated concurrently so that they can interact at the sense-amplifiers. The bitline sense-amplifiers, which may feature creative modifications, are leveraged for executing the logic operations. Since all the bitlines in a memory bank/subarray are activated in-parallel they a massive bitwise parallel engine. For example, 22nm DRAM technology features 2048 columns per subarray that can operate in parallel is adopted in [24] for PIM design.

2.2.2 ***Crossbar Array Computing:*** Mostly ReRAM devices [4, 8, 14, 27, 29, 30] (and a few others [2, 3, 26]) leverage this design which is presented in Figure 2(b). It is capable of performing massively parallel analog multiplication-accumulation (MAC) operations by leveraging 2D cell array of analog resistive cells. Analog input signals from the digital-to-analog converters (DACs) are driven through the word lines and are essentially multiplied with the resistance values of the cells before they reach the bitlines. Bitlines accumulate the analog products from the connected cells and forward those through the analog-to-digital converters (ADCs) to the digital domain. This proposed crossbar structure of ReRAM that has displayed a capability to accelerate efficiently matrix – vector multiplications in Neural Networks [4, 8, 14, 29].

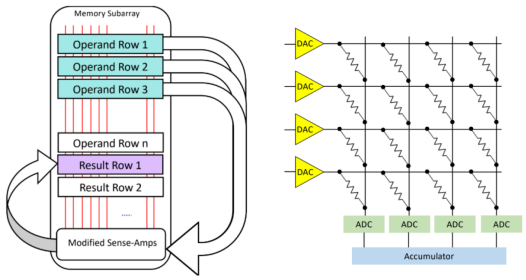


Figure 2: (a) Bitwise Parallel Computing Architecture (b) Crossbar Array Computing Architecture

2.3 Large Data Mapping on IMC

Since the processing elements of the IMC architectures are inherently bitwise, the processing architectures can be classified based on the mapping mechanisms. IMC architectures can be classified into three categories based on data mapping as follows.

Bit-parallel Mapping: The most common mapping scheme where the bits of an operand are placed on different bitlines. Two operands are placed in different rows such that the corresponding bits of the operands are on the same bitlines. Latency is independent of data-width but the degree of parallelism is inversely proportional to the data-width.

Bit-serial Mapping: All the bits of both operands are placed on the same bitline [5]. The operation starts with the LSBs of the operands and advances in bitwise steps, up to the MSBs. Buffer/ latche(s) facilitate the propagation of the output of a bit-step to the next. The degree of parallelism is same as the number of bitlines in a bank/ sub-array but latency per operation is proportional to the data-width.

Crossbar Mapping: A two-dimensional data-mapping scheme specifically utilized by crossbar array computing architectures discussed in sub-section 2.2. The weights are stored in the cell(s) along each bitline in analog form and the inputs are driven through different word-lines in analog form.

2.4 Hardware Interfacing

IMC devices can be interfaced either as a memory device [3, 14, 19, 20, 31] with in-memory processing capabilities or as a co-processor through PCI/PCIe port like an FPGA/GPU [5, 15, 16, 21, 23, 24]. In

the latter case, the device does not retain the host memory functionality and acts entirely as an accelerator co-processor with its sophisticated ISA, compiler and programming framework as described in [21, 23, 24]. Since these devices do not have to comply with the memory device (*i.e.,* DIMM) specifications, they feature extensive modifications such as resizing and redesigning of memory organization [23], including additional units such as digital processing units (DPU) [19–21], latches [5], registers & LUTs [24], and ADC/DAC [4, 27, 30] to name a few.

2.5 Quantized CNN Inference

In addition to the underlying hardware platforms, IMC architectures can also be classified based on the precision of data stored or used for the inference. Large (32-bit or above) floating-point precision of the inputs and weights provides maximum accuracy from deep learning algorithms,but at the same time incurs outrageous computational load from convolution operations. Fortunately, quantizing the data does not significantly affect the performance of ML algorithms [17, 18, 26, 30], thus attracting a lot of attention. We observe the following dominant trends in data precision of IMC.

2.5.1 ***16/8-bit fixed point precision:*** Both inputs and weights are 16/8-bit fixed point numbers, as can be seen in recent works such as [4, 5, 8, 16, 19, 24, 29].

2.5.2 ***BWNN & TWNN:.*** Input kernel has fixed-point precision (8/16/32 bit) and the weights are downscaled to Binary (single bit) & Ternary (2-bit) Weighted Neural Networks (BWNN & TWNN) [9][33]. Convolutions are reduced to accumulation operations in this scheme. The works such as [15, 21, 23] follow this scheme.

2.5.3 ***BCNN/TCNN:.*** Even further scaling down leads to Binary [17, 20, 26, 27, 31] or Ternary [26] CNNs where both the inputs and weights have binary (1-bit) or ternary (2-bit) precisions. Convolutions are effectively reduced to bitwise operations (bitwise AND [20, 31], XNOR [1, 11, 26, 30]), followed by accumulation.

3 IMC ARCHITECTURES

Here, we delve into further details of the ML-application oriented IMC works and present brief reviews for each of them.

3.1 *DRAM-based IMCs*

AMBIT: AMBIT [28] is an IMC that performs bulk bitwise AND & OR operations based on charge sharing across bitlines by simply activating three operand rows (wordlines) concurrently. Additionally, AMBIT performs NOT operations by a novel mechanism which is facilitated by enhancing one row of cells in a subarray with two access transistors (2TC1) formation. Although AMBIT does not have a Machine Learning aimed application domain, it establishes the basis for two other CNN accelerators DRISA[23] DRACC[15].

DRISA: DRISA [23] is a DRAM-based in-situ CNN accelerator that performs CNN inference on 8-bit BWNN. DRISA leverages a 3T1C memory cell-formation from older DRAMs for implementing NOR logic and also the regular 1T1C DRAM cell-formation for AND, OR & NOT operations, whose mechanisms are borrowed from AMBIT. Additionally DRISA features a shifting layer, which together with the logic layer forms a re-configurable in-memory processing layer that can implement selection, ripple carry addition, carry-save addition and complete multiplication on large data. The DRAM memory organization is completely revamped through the resizing of parallel-operable banks and subarrays and also an additional hierarchical stage above banks called a Group.

DrAcc: DrAcc [15] is a DRAM-based CNN accelerator that performs TWNN inference where the convolution operations are reduced to accumulation (addition). It too follows AMBIT's mechanism for bitwise AND/OR logic operations but its key feature is the carry look-ahead adder implemented by modifying the sense-amplifier circuit. With an additional shifting layer, DrAcc can implement all CNN layers such as pooling, quantization-normalization and activation inside the memory. Furthermore, three data partitioning and mapping strategies as well as three different performance modes are proposed which offer maximum throughput, minimum latency and maximum power efficiency respectively.

SCOPE: SCOPE [24] is an in-situ DNN (CNN/RNN) accelerator that utilizes stochastic computing (H2D algorithm) to simplify computation-intensive multiplication operations into bitwise AND operations. It's a bulk bitwise IMC/PIM architecture that can perform AND/OR operations. The subarray sense-amplifiers are modified to facilitate logic circuitry, register, shifter and Stochastic Number Generator unit (SNG) for stochastic conversion. It can perform carry-save addition by combining the bitwise logic operations and capable of both inference and training of CNN with 8-bit fixed point precision.

LAcc: LAcc [16] is a DRAM based PIM CNN accelerator that leverages the memory cells in a DRAM subarray to implement Look-Up-Tables (LUT) which store pre-calculated outputs of multiplication operations. Together with an XOR-based adder implemented on modified sense amplifiers, LAcc can perform 16-bit convolution operations. Since LUTs expand quadratically with the operands' size, LAcc leverages matrix decomposition to keep LUTs smaller. For further LUT-reduction, it proposes fixing of either weights or inputs. LAcc performs convolution with 16-bit precision with impressive power and performance figures.

3.2 *SRAM-based IMCs*

Neural Cache: Neural Cache [5] is a SRAM based in-memory DNN inference (primarily focused at CNNs) engine made by re-purposing cache memory. It can implement convolution, pooling, quantization and fully-connected layers at 8-bit data-precision. Its base functionalities are bitwise AND & NOR operations which are performed simply by concurrently activating the operand rows. It can perform bit-serial addition, subtraction, multiplication, comparison, search and copy on larger data, aided by carry latches appended to the sense-amplifiers. A Transpose Memory Unit provides hardware support to reorganize data in bit-serial format inside the memory.

IMAC: IMAC [3] is a CNN inference engine that leverages ReRAM like analog multiplication and accumulation (MAC) to perform convolution of multi-bit data. Memory cells in a row contain multi-bit weights. Analog version of the input signals are driven through the wordlines of that row and the corresponding analog voltage signals on the bitlines are aggregated to perform MAC operations. Peripheral converters (DAC/ADC) perform digital-to-analog conversion and vice versa. It can implement fully connected layers and activation (ReLU) function too.

XNOR-SRAM: XNOR-SRAM [26] is an in-memory binary/ ternary convolution macro for accelerating neural networks. Binary/ ternary CNN reduces convolution operations to a XNOR operations [11] followed by accumulation (XAC). XAC is performed by driving binary/ternary input signals through the word-lines in analog form which causes the 8T-SRAM cells that contain the binary weights to

Table 1: Prominent IMC Architectures and their characteristics

Name	Memory	Interfacing	ML Algorithm	Data Precision	Large Data Mapping	In-memory Bitwise Op.	In-memory Large Data Op.	Dataset, Neural Network Architecture
DRISA [23]	DRAM	Co-Processor	CNN,RNN (Inf.)	BWNN	Bit-parallel	AND, OR, NOR, Shift	Add(CSA* & RCA*), Mul.*	AlexNet , VGG-16, VGG-19 & ResNet-152
DrAcc [15]		Co-Processor	CNN (Inf.)	TWNN	Bit-parallel	AND, OR, NOT	Add (CLA)	Mnist, Cifar10, AlexNet, VGG-16 & VGG-19
LAcc [16]		Co-Processor	CNN (Inf.)	16-bit	LUT	N/A	Addition	Lenet-5, AlexNet, VGG16, VGG19 & ResNet-152
SCOPE [24]		Co-Processor	CNN/RNN (Inf. & Train)	8-bit (INT8)	Bit-parallel	AND/OR	Add*, Mul.(Stochastic)*	AlexNet(train.), VGG-16, ResNet-152 & Vanilla RNN
Neural Cache [5]	Cache (SRAM)	Co-processor	CNN (Inf.)	8-bit	Bit-serial	AND, NOR	Add/Sub*, Mul*, Comp*	Inception V3
IMAC [3]	SRAM	Memory Device	CNN (Inf.)	5-bit	Crossbar (Analog)	N/A (Analog)	MAC	Lenet-5, VGG
XNOR-SRAM [26]		Memory Device	CNN(Inf.)	BCNN/ TCNN	Crossbar(Analog)	XNOR	N/A	Custom MLP, VGG-like CNN & ResNet-14
ISAAC [4]	ReRAM	Co-Processor	CNN/DNN (Inf.)	16-bit Fixed	Crossbar (Analog)	N/A (Analog)	MAC (Analog)	VGG-(1,2,3,4), MSRA-(1,2,3), Deepface
PRIME [14]		Memory Device	CNN/ MLP (Inf.)	6-bit (In.) & 8-bit (Wt.)	Crossbar (Analog)	N/A (Analog)	MAC (Analog)	VGG-D, CNN-(1,2) & MLP-S/M/L (3 Layers)
Pipelayer [8]		Co-processor	CNN (Inf. & Train)	16-bit	Bitline-wise Kernel/ X-bar	N/A	MAC (Analog)	AlexNet, VGG-(A,B,C,D,E)
Tang et al. [27]		Co-processor	CNN (Inf.)	BCNN (Analog)	Crossbar (Analog)	MAC (Analog)	N/A	LeNet, AlexNet
Atomlayer [29]		Co-Processor	CNN Inf.	16-bit	Disjoint row Kernel/ X-bar	N/A (Analog)	MAC(Analog)	VGG-19 & DCGAN(Inf.),VGG-19 & ResNet-152(Train.)
XNOR-RRAM [30]		Co-processor	CNN/BNN (Inf.)	BNN(1-bit)	Crossbar	XNOR	N/A	Custom MLP & CNN
Pan et al. [31]	STT-MRAM	Memory Device	CNN (Inf.)	BCNN	N/A	AND/NAND, OR/NOR, XOR*	ADD (RCA)	XNOR-NET
MRIMA [21]		Co-processor	DNN/CNN (Inf.)	Low-width/ BWNN	Bit-Parallel	(N)AND,(N)OR (2/3), X(N)OR	ADD(RCA), MAC*, Shift	CNN(ISCAS85)
MRAM-DIMA [2]		Co-processor	DNN(Inf.)	4-bit(In.) & 5-bit(Wt.)	Crossbar(Col. Major)	N/A	MAC	LeNet-300-100 on MNIST, 9 layer CNN on CIFAR-10
Fan et al. [32]	SOT-MRAM	Memory Device	CNN (Inf.)	BCNN	Bit-Parallel	AND/OR	Binary MAC	AlexNet-BCNN
IMCE [20]		Memory Device	CNN (Inf. & Train)	BCNN/ Var. width	Bit-parallel	AND/OR	Bin/Multi-bit MAC*	AlexNet
CMP-PIM [19]		Memory Device	DNN (Inf.)	8-bit/ Var. Width	Bit-Parallel	AND/OR/XOR	Comparison*, Mac*	CMP-NET (ResNet Based)

*Operations formulated by combining bitwise functionalities of the IMC

produce XNOR output signals on the read bitlines (RBL). Analog accumulation of XNOR outputs takes place on the RBL, followed by digitization in the ADC.

3.3 *ReRAM-based IMCs*

ISAAC: ISAAC [4] is a ReRAM crossbar array structured CNN/DNN in-situ accelerator that maps synaptic weights in the form of analog resistance values in the memory cells. In ISAAC, input bits are driven in serial through the word-lines in the form of analog voltage pulses. The corresponding current magnitudes in the bitlines represent the accumulated products of the input bits with all the weights (cells) connected along the respective bitlines. These partial products are digitized, shifted and then accumulated for complete MAC operations. Weights are represented in 2's complement to account for both positive and negative ones. ISAAC has a sophisticated hierarchical architecture including MAC, ADC/DAC, sigmoid and max pool units in each processing 'tile'. ISAAC maps all the layers of a CNN in different tiles and adopts a pipelined data-flow.

PRIME: PRIME [14] is an in-memory computing architecture on ReRAM platform. It is created by modifying a portion of ReRAM crossbar array that can either act as host memory or a Neural Network (NN) accelerator. It performs analog MAC in the crossbar array in a mechanism resembling that of ISAAC, except that separate crossbars are dedicated to positive and negative analog weights. It modifies default ReRAM micro-architecture heavily to also facilitate sigmoid, ReLU, max pooling, ADC/DAC & precision control unit and also a buffer subarray. PRIME features bank-level parallelism and offers software and API support for programming and data-mapping.

PipeLayer: Pipelayer [8] is a ReRAM based accelerator which leverages spike-based signals and introduces batch level intra-layer parallelism through an inter-layer pipeline implementation. To improve throughput, this accelerator supports tiled architecture which combines computation and data encoding together in order to process data in parallel in the in-ReRAM crossbar array. The accelerator analyzes data dependencies between layers to improve the effectiveness of pipelining and inter-layer parallelism.

Tang et al.: Tang et al. [27] presents a BCNN accelerator that focuses heavily on optimizing CNN inference on the ReRAM crossbar platform. It considers a binary data-storage in analog ReRAM cells to ensure maximum robustness from process variations. An elaborate matrix splitting scheme is presented for mapping large CNNs across multiple crossbar units. Different layers of the CNN including convolutional, fully-connected and max-pooling layers are mapped to different ReRAM crossbar blocks and the execution of the layers is pipelined. A sophisticated buffer (Line-Buffer) ensures the fastest and the most efficient transition of data from one layer to the next.

Atomlayer: Atomlayer [29] is a ReRAM based accelerator that leverages atomic layer computation in which one layer of a NN is processed at a time to eliminate pipelining. It addresses ISSAC's [4] limitation in performing NN training and the low power-efficiency issue of Pipelayer [8]. In order to reduce the pipeline bubbles, this design utilizes a large DRAM main memory to store initial and intermediate data generated in the neural network inference or training. It uses rotating crossbars as the key components to perform atomic layer computations. It also uses row disjoint filter mapping to distribute filter rows across multiple its multiple processing elements to ensure maximum data reuse and reduces the DRAM bandwidth.

XNOR-RRAM: XNOR-RRAM [30] is a Binary Neural Network (MLP) inference engine implemented on ReRAM. Since the inputs and weights are binary (1-bit), the convolution operations are reduced to bitwise XNOR operations. A Dual Complementary Cell setup is used to perform the XNORs. A worldline switch matrix replaces the row decoder in the ReRAM synaptic architecture that enables parallelism across memory rows. Operand matrices are split and distributed across bitlines. Partial sums are generated at multi-level sense amplifiers (MSLA) on each bitline and then quantized, accumulated and undergo binary activation to obtain output.

3.4 *STT-MRAM based IMCs*

Binary CNN: Pan et al. [31] is a Binary Convolutional Neural Network (BCNN) accelerator implemented on STT-MRAM. It uses multi-level STT-MRAM cell structure (MLC-STT) where two asymmetric cells are fused together to contain two bits. It is capable of bulk bitwise AND/NAND, OR/NOR through multi-row activation and also XOR and ripple-carry addition through modified sense amplifiers. Auxiliary processing unit (APU) peripheral to the memory bank performs other CNN operations such as Batch normalization, Pooling etc. This work is evaluated on XNOR-NET [11].

MRIMA: MRIMA [21] is an IMC device implemented on STT-MRAM platform and is aimed at two different application domains: CNN inference & AES data encryption. Both applications are facilitated by in-memory compute arrays that can either act as memory

devices or perform bulk bitwise AND, NAND, NOR (on either 2 or 3 operands) & XOR/XNOR operations by leveraging multi-row activation and using different reference currents for the sensing of each the logic outputs. MRIMA can perform BWNN and low bit-width CNN inferences. Convolutions are simplified by performing column decomposition of input and kernel matrices, followed by generating partial products by bitwise parallel AND operations between input & and kernel columns and finally shifting & accumulating those partial products. MRIMA has its own ISA and can be connected as a co-processor through the PCIe port.

MRAM-DIMA: MRAM-DIMA [2] is an MRAM-based crossbar array multiplication-accumulation engine aimed at DNN applications. Weights are mapped in column major format across bitlines and inputs are provided through the wordlines in pulse-width modulated (PWM) form. Analog vector multiplication takes place in the MRAM cells in parallel and partial products are generated on the bitlines; later to be accumulated by a digital processor. MRAM-DIMA is also capable of performing clipped ReLU activation.

3.5 *SOT-MRAM based IMCs*

Binary CNN: Fan et al. [32] is a Binary CNN accelerator based on SOT-MRAM platform. BCNN inference reduces convolutions to simple in-memory bitwise AND operations, followed by accumulation in peripheral bit-counter. A DPU (Digital Processing Unit) performs Batch Normalization, scaling, multiplier & pooling operations.

IMCE: IMCE [20] is a SOT-MRAM based IMC device that can perform CNN inference & training at different data precision combinations by leveraging a Binary convolution mechanism. This mechanism involves column decomposition of the input f-map and the kernel (weight matrix), bitwise convolution of those columns and finally shifting and accumulation of those partial products through peripheral counter, shifter & adder to obtain the output f-map. An in-memory DPU executes Quantization, Batch Normalization, Pooling and Activation Filters.

CMP-PIM: CMP-PIM [19] is a SOT-MRAM IMC that performs comparator-based DNN inference. Here the pixel-wise convolution operations in a convolutional layer are replaced by depth-wise separable convolutions which are a combination of binary pattern feature extraction (depth-wise convolution) and binary point-wise (bitwise) convolutions. The depth-wise convolution leverages fixed ternary kernel which is implemented through multi-bit comparison and accumulation in this paper. The bulk bitwise IMC architecture of CMP-PIM can perform AND/NAND, OR/NOR, XOR/XNOR operations.The point-wise convolution stage is implemented through bitwise AND, followed by bit-count, shifting and accumulation operations. Quantization, batch normalization and activation takes place in in-memory DPU.

4 ANALYSIS OF IMC ARCHITECTURES

Figure 3a and 3b present the comparison of area of the IMCs based on traditional and emerging memory platforms respectively. DRAM CNN/DNN accelerators are centered around a baseline area of 60 mm^2 except for SCOPE whose area is more than four times that of a commodity DRAM chip. SRAM based Neural Cache's smaller footprint is due to its very low memory capacity (35MB). ISAAC and PRIME have the highest areas among the ReRAM IMCs. This can be attributed to their large data precision (16-bit). However, AtomLayer also has 16-bit data-precision and yet it has a significantly smaller

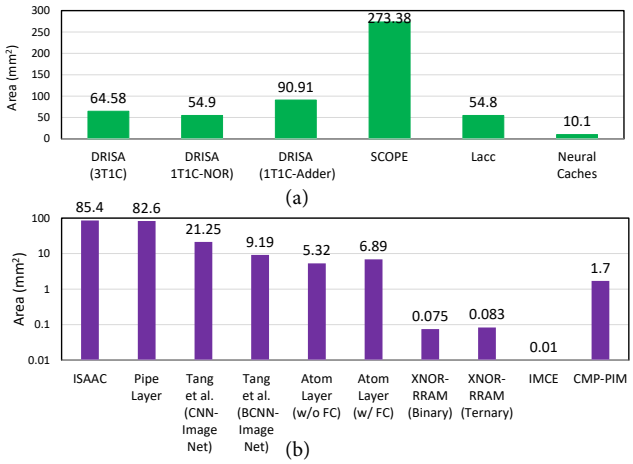
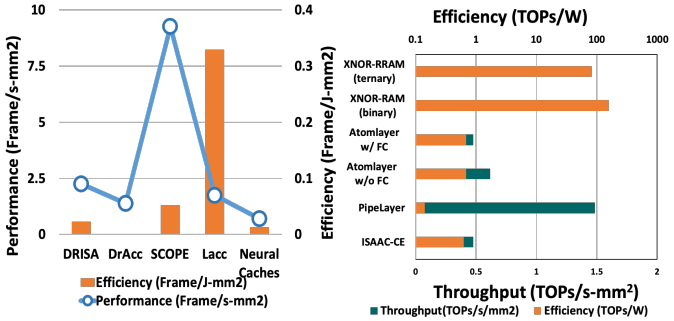


Figure 3: Area for IMC architectures with: (a) traditional and (b) emerging memory technologies

area because it employs massive resource re-utilization aimed at reducing processing area. The two data precision configurations of Tang et al. (CNN & BCNN) and the Binary precision of XNOR-RRAM also reflect the same pattern of area being proportional to data-precision. Similar observations can also be made for MRAM (STT-MRAM & SOT-MRAM) based works - IMCE with Binary data precision and CMP-PIM with 8-bit data precision where CMP-PIM has a larger footprint.



(a) Benchmarked on AlexNet for traditional memory IMCs **(b) Emerging memory IMCs**

Figure 4: Performance Comparison of traditional and emerging IMC architectures

Figure 4a presents performance (throughput per unit chip area) and efficiency (output per unit energy consumption) benchmark on AlexNet for volatile memory-based (DRAM and SRAM) IMCs. SCOPE presents splendid performance figures, albeit at a rather low efficiency which can be attributed to its stochastic method for multiplication operations. LAcc, thanks to its LUT implementation, ensures the maximum efficiency among these works. Neural Cache's lowest efficiency and lowest performance figure can be traced back to its comparatively inefficient SRAM platform and smaller capacity (35 MB), respectively.

Figure 4b presents throughput (as TOPs/s for unit area) and efficiency (as TOPs per unit power) comparison for Non-volatile Memory IMCs. Pipelayer has dramatically higher throughput but lowest energy efficiency compared to ISAAC & Atomlayer because it leverages massive intra-layer data parallelism to maximize pipeline

throughput. Although Atomlayer has the advantage of training capability, it does not outperform ISAAC in the field of efficiency. Due to very low data-precision (binary/ ternary precision), XNOR-RRAM is the most efficient ReRAM IMC under comparison.

5 CHALLENGES IN IMC IMPLEMENTATION

Since IMCs leverage the memory architecture for processing, these are highly susceptible to the design restrictions imposed by the architecture and circuit level design of memory platform that it is built on. For example, DRAM stores data capacitively which requires frequent refreshing, making the logic operations timing-sensitive. SRAM is limited by the comparatively lower cell density and energy efficiency. ReRAM suffers from high write latency and STT-MRAM suffers from high error rate due to fluctuations in ohmic properties of the cells. Each memory technology has its own features which present unique implementation challenges. For example, DRAM's single bitline cell [23] vs. SRAM's complementary bitline pair cell [5] necessitates different mechanisms for bitwise logic implementations. Therefore, the key challenge in designing an IMC is to perform minimal modifications to baseline structure of the memory chip in order to facilitate desired functionalities, while at the same time meeting the fabrication limitations & restrictions and also obtain reliable performance.

6 FUTURE PROSPECTS

Mobile devices are the best candidates for future IMC applications since these are constrained by their limited processing capabilities and the efficiency demands originating from the battery limitations. We expect to see an increasing use of IMC in mobile and edge devices and for IoT applications as well as real-time applications such as facial/voice recognition, pattern matching, object detection and digital assistance. IMCs can be predicted to leverage emerging low bit-precision inference and training algorithms of DNN and CNN such as [1, 11, 18] more extensively. We also expect to see Recurrent Neural Networks (RNNs) like LSTMs to be implemented on IMC platforms. More LUT-based IMC implementations like LAcc [16] can be expected on DRAM and possibly on non-volatile platforms as well since LUTs offer very high efficiency and flexibility by replacing computations with pre-calculated outputs. 3-D stacked DRAM can be expected to be exploited for IMC models such as bulk bitwise processing and crossbar multiplication, alongside the more conventional near-memory computing (NMC) model where the processing elements are located on a separate layer in the 3-D stack. Magnetic memories (STT-MRAM/SOT-MRAM), especially multi-level cell MRAMs (MLC-STT) can be expected to achieve more popularity as IMC platform due to their non-volatility as well as a wider functionality range compared to DRAM and SRAM. Although we do not expect dramatic innovation in crossbar array architecture of ReRAM, improvement can be expected from the architectural side of ReRAM based IMCs. Overall, we expect a massive growth in the research of IMCs which will introduce broader and newer dimensions of enhancements to baseline memory structures for facilitating a wider range of in-memory operations with more attractive performance and efficiency figures.

REFERENCES

[1] M Courbariaux and Yoshua Bengio. 2016. BinaryNet: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1. *ArXiv* (2016).

[2] A. D. Patil et al. 2019. An MRAM-Based Deep In-Memory Architecture for Deep Neural Networks. In *IEEE International Symposium on Circuits and Systems.*

[3] A Mustafa et al. 2020. IMAC: In-Memory Multi-Bit Multiplication and ACcumulation in 6T SRAM Array. *IEEE Transactions on Circuits and Systems I: Regular Papers* (2020).

[4] A. Shafiee et al. 2016. ISAAC: A Convolutional Neural Network Accelerator with In-Situ Analog Arithmetic in Crossbars. In *ACM/IEEE International Symposium on Computer Architecture (ISCA).*

[5] C Eckert et al. 2018. Neural Cache: Bit-Serial In-Cache Acceleration of Deep Neural Networks.

[6] D. Patterson et al. 1997. A case for intelligent RAM. *IEEE Micro* (1997).

[7] Kim et al. 1999. Assessing Merged DRAM/Logic Technology. *Integr. VLSI J.* (1999).

[8] L. Song et al. 2017. PipeLayer: A Pipelined ReRAM-Based Accelerator for Deep Learning. In *IEEE Int. Symp. on High Performance Computer Architecture (HPCA).*

[9] M Courbariaux et al. 2015. BinaryConnect: Training Deep Neural Networks with Binary Weights during Propagations. In *Proceedings of the 28th International Conference on Neural Information Processing Systems.*

[10] M. Lechner et al. 2019. ResCoNN: Resource-Efficient FPGA-Accelerated CNN for Traffic Sign Classification. In *Int. Green and Sustainable Computing Conf.*

[11] M Rastegari et al. 2016. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks.

[12] M. Wess et al. 2018. Weighted Quantization-Regularization in DNNs for Weight Memory Minimization Toward HW Implementation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2018).

[13] Nima Taherinejad et al. 2015. Memristors' Potential for Multi-bit Storage and Pattern Learning. *IEEE European Modelling Symposium (EMS)* (2015).

[14] P. Chi et al. 2016. PRIME: A Novel Processing-in-Memory Architecture for Neural Network Computation in ReRAM-Based Main Memory. In *ACM/IEEE International Symposium on Computer Architecture (ISCA).*

[15] Q. Deng et al. 2018. DrAcc: a DRAM based Accelerator for Accurate CNN Inference. In *ACM/ESDA/IEEE Design Automation Conference (DAC).*

[16] Q. Deng et al. 2019. LAcc: Exploiting Lookup Table-based Fast and Accurate Vector Multiplication in DRAM-based CNN Accelerator. In *ACM/IEEE Design Automation Conference (DAC).*

[17] R. Andri et al. 2016. YodaNN: An Ultra-Low Power Convolutional Neural Network Accelerator Based on Binary Weights. In *IEEE Computer Society Annual Symposium on VLSI (ISVLSI).*

[18] R Zhao et al. 2017. Accelerating Binarized Convolutional Neural Networks with Software-Programmable FPGAs. In *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays.*

[19] S. Angizi et al. 2018. CMP-PIM: An Energy-Efficient Comparator-based Processing-In-Memory Neural Network Accelerator. In *ACM/ESDA/IEEE Design Automation Conference (DAC).*

[20] S. Angizi et al. 2018. IMCE: Energy-efficient bit-wise in-memory convolution engine for deep neural network. In *Asia and South Pacific Design Automation Conference (ASP-DAC).*

[21] S. Angizi et al. 2020. MRIMA: An MRAM-Based In-Memory Accelerator. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2020).

[22] S. Lu et al. 2012. Scaling the "Memory Wall": Designer track. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD).*

[23] S Li et al. 2017. DRISA: A DRAM-Based Reconfigurable In-Situ Accelerator. In *IEEE/ACM International Symposium on Microarchitecture.*

[24] S. Li et al. 2018. SCOPE: A Stochastic Computing Engine for DRAM-Based In-Situ Accelerator. In *IEEE/ACM International Symposium on Microarchitecture (MICRO).*

[25] S. M. P. Dinakarrao et al. 2017. A Scalable Network-on-Chip Microprocessor With 2.5D Integrated Memory and Accelerator. *IEEE Transactions on Circuits and Systems I: Regular Papers* (2017).

[26] S. Yin et al. 2020. XNOR-SRAM: In-Memory Computing SRAM Macro for Binary/Ternary Deep Neural Networks. *IEEE Journal of Solid-State Circuits* (2020).

[27] T. Tang et al. 2017. Binary convolutional neural network on RRAM. In *Asia and South Pacific Design Automation Conference (ASP-DAC).*

[28] V Seshadri et al. 2017. Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology. *IEEE/ACM International Symposium on Microarchitecture (MICRO)* (2017).

[29] X. Qiao et al. 2018. AtomLayer: A Universal ReRAM-Based CNN Accelerator with Atomic Layer Computation. In *ACM/ESDA/IEEE Design Automation Conference (DAC).*

[30] X. Sun et al. 2018. XNOR-RRAM: A scalable and parallel resistive synaptic architecture for binary neural networks. In *Design, Automation Test in Europe Conference Exhibition (DATE).*

[31] Y. Pan et al. 2018. A Multilevel Cell STT-MRAM-Based Computing In-Memory Accelerator for Binary Convolutional Neural Network. *IEEE Transactions on Magnetics* (2018).

[32] D. Fan and S. Angizi. 2017. Energy Efficient In-Memory Binary Deep Neural Network Accelerator with Dual-Mode SOT-MRAM. In *IEEE International Conference on Computer Design (ICCD).*

[33] Fengfu Li and Bin Liu. 2016. Ternary Weight Networks. *CoRR* (2016).