Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 14
TPCEE 2026

Accelerated Computing for Deep Learning with Computing-in-Memory Technologies

Jianyuan Ma [1, *]

[1] North London Collegiate School Jeju, South Korea

* Corresponding Author Email: maxma9909@gmail.com

Abstract. Due to the rapid growth of deep learning, computational and memory demands in artificial intelligence systems have also increased. In conventional von Neumann architecture systems, the physical separation between memory and computational units causes latency, poor energy efficiency, and limited bandwidth. Computing-in-Memory (CIM) addresses this issue by performing operations directly within or next to memory modules, reducing latency and improving energy efficiency through reduced data movement. This paper reviews the operating principles, memory technologies, recent optimisation strategies, limitations, and future prospects of CIM technology for deep learning. Current-based, charge-based, time-based, and digital-based circuits provide different methods of implementing the multiplication and accumulation operations required by neural networks, while technologies including Static Random Access Memory (SRAM), Dynamic Random Access Memory (DRAM), Resistive Random Access Memory (RRAM), Spin-Transfer Torque Magnetic Random Access Memory (STT-MRAM), and Phase Change Memory (PCM) offer different storage mechanisms. Recent research demonstrates that epitome-based model compression can reduce memory and crossbar requirements, pipeline optimisation can improve computational efficiency, and multicore weight-mapping strategies can increase throughput and hardware utilisation. However, hardware non-idealities, peripheral–circuit overhead, and limited scalability continue to restrict the accuracy and efficiency of CIM implementations. Continued improvements in device stability, hardware-aware training, peripheral circuits, memory density, three-dimensional integration, model compression, and weight mapping may enable more practical large-scale systems. Overall, CIM provides a promising path towards more energy-efficient and scalable hardware for increasingly demanding deep-learning applications.

Keywords: Computing-in-Memory; Deep Learning; Accelerated Computing.

1. Introduction

Deep learning technology has progressed at a remarkable rate in recent years. Increasingly large amounts of available data, as well as improved computational hardware and learning algorithms, have allowed for increasingly complex neural networks. Applications of these networks include areas such as computer vision and natural language processing. However, as neural networks grow in depth and capacity, the amount of computational resources required in order to operate these systems also grows sharply [1].

The most common architecture employed in conventional computer systems is the von Neumann architecture. In this configuration, memory and the processing unit are physically separate, connected to each other by a data bus. When large amounts of continuous data transfer between the two units is required, this configuration suffers from high energy consumption and latency. This is especially problematic for deep-learning workloads, which require a large amount of data and parameters. Performance bottlenecks can arise from repeated data transfers due to limited bandwidth. This restricts the system's overall performance even if the processor is capable of achieving higher speeds. This is known as the von Neumann bottleneck [2].

Due to existing computer architecture limitations, alternative configurations have been developed. A prominent alternative is computing-in-memory (CIM) technology. CIM technology alleviates bandwidth limitations and reduces data movement by moving computational processes to the same

Content from this work may be used under the terms of CC BY-NC 4.0 licence (https://creativecommons.org/licenses/by-nc/4.0/).
Published by Warwick Evans Publishing.

area as the memory. Because of these factors, CIM has the potential to improve computer performance by acting as a computing accelerator for applications such as artificial intelligence [3].

The earliest proposals of CIMs date back to the 1960s, a prominent example being the Logic-in-Memory computer developed by Harold S. Stone [4]. Stone's design combined processing elements and memory, allowing computation to be performed within the memory array itself. In the following decades, many new designs were introduced, including inserting extra logic within Dynamic Random Access Memory (DRAM) and incorporating more adaptable computer logic in order to increase the flexibility of processing hardware. However, progress was hindered due to limitations of memory technology at that time [4]. New developments in the 1990s introduced the concept of near-memory computing, where compute units were placed closer to the memory. This brought reductions in memory access time. Later improvements in die stacking technology enabled advanced memory modules such as Hybrid Memory Cube and High Bandwidth Memory. Despite these advances, the memory and computation remained physically separate [5]. More recent research on Analog In-Memory Computing (AIMC) demonstrates results in neural network inference despite the dependence on external hardware and software. Subsequent single- and few-core AIMC chips integrated data conversion and activation functions, albeit being limited to smaller scale neural networks of fewer than 100,000 weights. Recent multicore chips can support networks exceeding 1 million weights while achieving greater inference accuracy in image recognition tasks [6].

Another development of Resistive Random Access Memory (ReRAM) based CIM has been the improvement of efficiency and flexibility. While early systems relied on off-chip software and hardware to perform specific functions, later designs became more integrated. However, maintaining a trade-off between accuracy and efficiency remained an issue. In order to address this issue, Wan et al. developed NeuRRAM, a 48-core RRAM-CIM chip capable of performing a variety of AI applications. NeuRRAM offers twice the energy efficiency of previous RRAM-CIM chips while achieving accuracies nearly identical to previous quantised models [7].

As AI and deep-learning become a more integral part of everyday life, CIM technology is essential for the functionality and implementation of more hardware demanding applications. Therefore, this paper aims to examine the existing technologies and progress of storage-computing integration technologies for deep learning, including its current development, limitations, and future outlook.

2. Principles and Memory Technologies of Computing-in-Memory

Modern deep neural networks used in AI applications, such as autonomous driving and intelligent manufacturing, contain a large number of parameters, as well as addition and multiplication operations. In traditional Von Neumann architecture, where the memory and processing units are kept separate, high latency and energy costs are incurred due to continuous data movement.

CIM technology performs computation directly within the memory, which improves energy efficiency by reducing data movement. At the circuit level, CIM carries out multiplication and accumulation through four major approaches: Current-based, charge-based, time-based, and digital-based. Current-based CIM circuits utilize electrical current, where currents are summed on a bitline, and naturally perform Multiply-Accumulate (MAC) calculations due to Kirchhoff's Law. Charge-based CIM controls charge stored in capacitors and accumulates through charge redistribution. Time-based CIM performs computation through controlling signal latency and delay, while digital-based CIM uses digital logic to perform multiplication and accumulation [8].

In artificial neural networks, neurons in different layers are interconnected through weights, which can be organized into matrices. The output of neurons in the subsequent layer is determined by the activations from the preceding layer and the synaptic weights connecting the neurons. These operations require a massive number of Matrix Vector Multiplication (MVM) for processes such as forward-back propagation. MVM calculations that consist mostly of MACs are a major part of neural network computation. CIM architectures map the weights of a neural network onto memory arrays

and perform operations directly on the memory hardware itself. This reduces the bottleneck issues of typical Von Neumann-based architectures [9].

CIM can be developed using a range of memory technologies, which may depend on its application. These technologies include conventional approaches such as DRAM and Static Random Access Memory (SRAM), as well as DRAM derivatives such as HBM and HMC, which utilise 3D-stacking to achieve increased bandwidth compared to regular DRAM. Emerging non-volatile technologies are also used, such as Spin-Transfer Torque Magnetic Random Access Memory (STT-MRAM), ReRAM, and Phase Change Memory (PCM). STT-MRAM stores data using magnetic states, ReRAM represents data using high and low resistance states, and PCM utilises electrodes separated by a phase change material, which can switch between crystalline and amorphous states, for example, which have distinct electrical properties [10].

3. Research status of CIM

3.1. Epitome-Based Model Compression for CIM Accelerators

As AI computation becomes increasingly integrated in the modern world, AI models are continuously increasing in complexity. Therefore, the number of parameters required to be stored also increases. Modern deep neural networks (DNNs) therefore can place high demands on memory technologies. However, storage density limitations on current CIM devices mean that applying CIM technology to large-scale neural network problems is still a challenge [9].

Model compression algorithms aim to reduce the burden of Convolutional Neural Networks (CNNs). These algorithms typically utilise compression techniques such as quantisation, which uses parameters with reduced precision, or neural architecture search, where the most efficient combination of neural operators is chosen. However, existing compression methods do not take into account specific specifications of CIM accelerators. This led Wang et al. to develop Epitome, as shown in Fig. 1. This EPIM framework replaces traditional convolution operators with compact epitomes before applying quantisation and adapting the data path for CIM hardware [11]. As shown in Fig. 1(a), the framework converts a conventional CNN into an epitome-based network and prepares it for deployment on a CIM accelerator. Fig. 1(b) shows how the modified data path selects the required input features and memory rows for each epitome patch, while Fig. 1(c) shows how frequently reused regions are given greater importance during quantisation to preserve accuracy at low bit widths. An epitome contains a compact set of parameters which is significantly smaller than the full convolution weights. A sampler repeatedly extracts small sections from the epitome, and this process is continued until the sampled sections are able to be reconstructed into the weight tensor required for the convolution. This allows the CNN to maintain convolution-like functionality while reducing the number of parameters needed to be stored, lowering the memory requirements of CIM hardware [11]. Experimental results showed that the 3-bit quantized EPIM achieved 71.59% top-1 accuracy on ImageNet while reducing crossbar area by 30.65 times, outperforming existing pruning-based CIM methods [11].

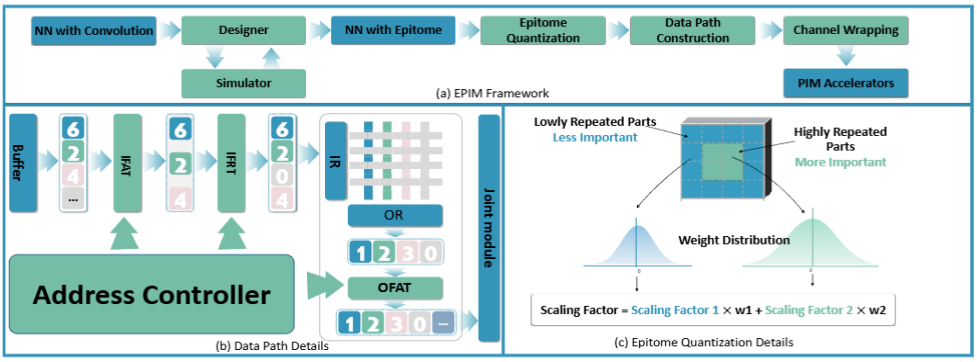

Fig. 1 Overview of the EPIM framework: (a) design workflow, (b) modified data path, and (c) epitome-aware quantization [11].

3.2. Pipeline Optimisation for CIM-Based CNN Accelerators

Pipeline design improves the efficiency of CIM-Based CNNs by allowing multiple layers to be processed simultaneously. However, these designs face difficulties due to differences in computational load across different processes, which require different amounts of time to process, causing bottlenecks and reducing the efficiency of the pipeline. To resolve this, Han et al. proposed a pipeline design for nonvolatile-based CIM, which arranges the CNN into multiple pipeline stages. For bottlenecked layers of the pipeline, weights are duplicated in order to increase speed and reduce latency by half. Non-bottlenecked layers utilise activation recomputation to reduce hardware area by recomputing some activations instead of directly storing them [12].

As shown in Fig. 2, the baseline design processes layers sequentially and leaves periods of hardware inactivity, whereas a pipeline-depth arrangement allows several inputs to move through different pipeline stages simultaneously. Weight duplication accelerates bottlenecked layers, while activation recomputation reduces the required SRAM area by regenerating selected activations instead of storing them [12].

Experimental results using DenseNet 121 showcase the progressive improvements in efficiency from the proposed pipeline strategies. The inverse of the computational efficiency(1/CE) value decreased from 30 to 4.7 after using pipeline depth arrangement, then to a further reduced value of 1.4 after implementing weight duplication, and finally to a value of 1.0 after using the activation recomputing strategy. This reduction corresponds to a 30x decrease in 1/CE [12].

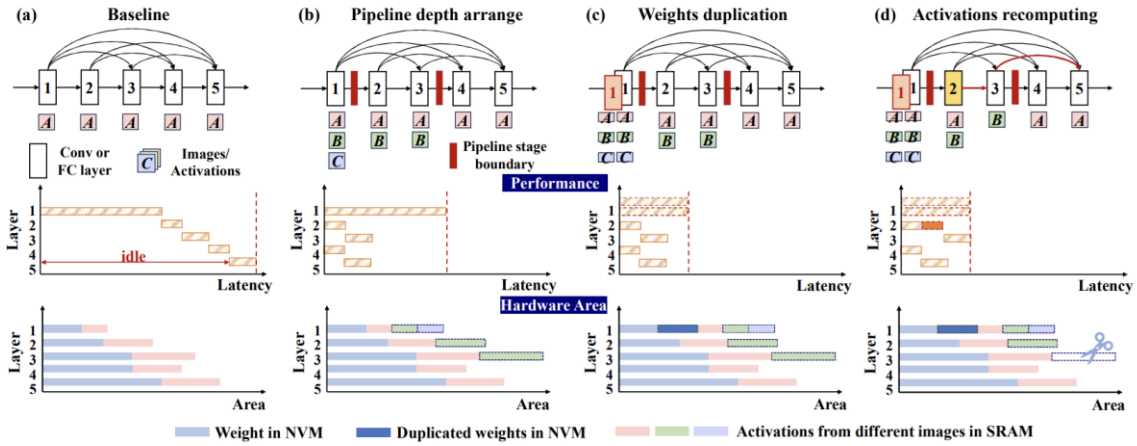

Fig. 2 Overview of the proposed pipeline design for nvCIM-based CNN inference accelerators [12].

3.3. Weight Mapping in Multicore CIM Architectures

Efficient weight mapping is crucial for CIM systems as the size of neural-network weight matrices may exceed the capacity of a single CIM memory array. NeuRRAM utilises a multi-core configuration containing 48 CIM cores working in parallel, with each core containing a 256x256 array. The chip employs several different strategies to distribute different parts of a neural-network layer across these cores. Computationally intensive layers can be duplicated across several cores in order to improve throughput. Different layers can be allocated to separate cores, while larger weight matrices which exceed the capacity of one RRAM array can be divided and distributed across multiple cores [7].

As shown in Fig. 3, NeuRRAM's layer-weight mapper can assign one layer to one core, duplicate demanding layers across multiple cores, combine smaller layers within a core, rearrange weights to improve core utilisation, or divide large layers across several cores for parallel execution. These strategies improve hardware utilisation and allow layers of different sizes and computational demands to be mapped efficiently [7].

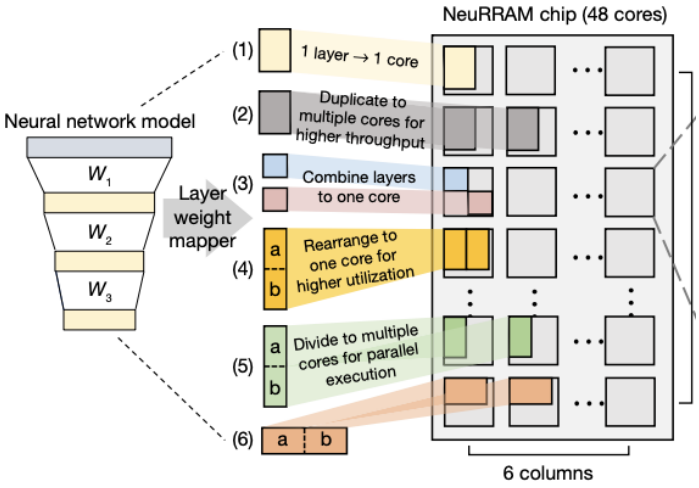

Fig. 3 Weight-mapping strategies across the 48-core NeuRRAM chip [7].

4. Limitation and Outlook

A significant issue of CIM technology is the loss of accuracy due to the influence of hardware non-idealities on memory devices. Conductance is used to represent weights of a neural network in analogue CIM. These values may drift over time, while device variations, analogue noise, and hardware non-linearities may affect the accuracy of MAC calculations [1]. Due to these imperfections, the accuracy of hardware implementations is therefore lower than software values. The direction of future developments could be towards increasing the stability of NVM devices, as well as utilising hardware-aware training techniques to increase the resistance of CNNs to hardware errors [1].

Another limitation of CIM technology is the overhead of supporting hardware and circuits next to the memory array. Analogue systems require Analog-to-Digital Converters (ADCs) and Digital-to-Analog Converters (DACs) in order to convert signals between digital and analogue forms. Therefore, even though the memory array itself can carry out efficient MAC operations, ADC and DAC chips consume a significant amount of energy and physical space, which reduces the overall efficiency of the system. Future developments could focus on reducing peripheral circuit overhead by allowing the sharing of ADCs across multiple columns. This reduces total occupied area as well as power consumption while maintaining performance [1].

The scalability of CIM technology becomes a major challenge due to the increase in size of modern neural networks. The increase in complexity of neural networks comes with a larger number of required stored parameters. This requires an increase in chip area and a larger number of memory arrays. In cases where the neural-network model cannot fit on a single chip, weights may be required to be stored on multiple chips or in off-chip memory. This reduces the power consumption and latency benefits brought by CIM [1]. Future development directions include increasing memory density and 3D integration, which allows for more arrays to be placed in the same area. This increases storage density, allowing more weights to be stored in the same area. Improvements in weight compression and mapping techniques are also beneficial for further increasing the efficiency of CIM hardware.

5. Conclusion

Overall, CIM provides a promising approach to overcoming the latency, bandwidth, and energy limitations caused by repeated data transfers in conventional von Neumann architectures. This review showed that CIM performs neural-network computations within or close to memory and can implement multiplication and accumulation using current-based, charge-based, time-based, or digital-based circuits across conventional and emerging memory technologies. Recent optimisation methods also demonstrate how the limitations of CIM hardware can be addressed: the EPIM framework reduces the parameters and crossbar area required by convolutional neural networks, pipeline-depth arrangement, weight duplication, and activation recomputation improve computational efficiency and

balance hardware use, and NeuRRAM's flexible weight-mapping strategies distribute neural-network layers across multiple cores to increase throughput and utilisation. These developments demonstrate that effective CIM systems depend not only on the characteristics of the memory device, but also on the co-design of neural-network algorithms, data paths, mapping strategies, and supporting circuits. Nevertheless, analogue noise, device variation, conductance drift, hardware non-linearities, ADC and DAC overhead, and limited storage density continue to restrict the accuracy, efficiency, and scalability of current implementations. Future research should therefore improve the stability of non-volatile memory devices, reduce peripheral-circuit overhead, and use hardware-aware training to make neural networks more resistant to physical errors. Increasing memory density through improved devices and three-dimensional integration, together with more effective model-compression and weight-mapping methods, may also allow larger networks to operate on fewer chips and preserve the advantages of CIM at scale across a wider range of demanding real-world computing environments. By bringing memory and computation together, CIM has the potential to support more efficient deep-learning accelerators for increasingly complex artificial-intelligence applications. The continued development of this technology is therefore significant not only for improving individual accelerator designs, but also for reducing the broader computational and energy costs associated with the expanding use of deep learning.

References

[1] Haensch W, Raghunathan A, Roy K, et al. Compute in-memory with non-volatile elements for neural networks: A review from a co-design perspective. Advanced Materials, 2023, 35(37): e2204944.

[2] Lepri N, Glukhov A, Cattaneo L, et al. In-memory computing for machine learning and deep learning. IEEE Journal of the Electron Devices Society, 2023, 11: 587-601.

[3] Sun Z, Kvatinsky S, Si X, et al. A full spectrum of computing-in-memory technologies. Nature Electronics, 2023, 6(11): 823-835.

[4] Ghose S, Boroumand A, Kim J S, et al. Processing-in-memory: A workload-driven perspective. IBM Journal of Research and Development, 2019, 63(6): 3:1-3:19.

[5] Sebastian A, Le Gallo M, Khaddam-Aljameh R, et al. Memory devices and applications for in-memory computing. Nature Nanotechnology, 2020, 15(7): 529-544.

[6] Le Gallo M, Khaddam-Aljameh R, Stanisavljevic M, et al. A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference. Nature Electronics, 2023, 6: 680-693.

[7] Wan W, Kubendran R, Schaefer C, et al. A compute-in-memory chip based on resistive random-access memory. Nature, 2022, 608(7923): 504-512.

[8] Sun W, Yue J, He Y, et al. A survey of computing-in-memory processor: From circuit to application. IEEE Open Journal of the Solid-State Circuits Society, 2024, 4: 25-42.

[9] Yu K, Kim S, Choi J R. Trends and challenges in computing-in-memory for neural network models: A review from device design to application-side optimization. IEEE Access, 2024, 12: 186679-186702.

[10] Asifuzzaman K, Miniskar N R, Young A R, et al. A survey on processing-in-memory techniques: Advances and challenges. Memories – Materials, Devices, Circuits and Systems, 2023, 4: 100022.

[11] Wang C, Dong Z, Zhou D, et al. EPIM: Efficient processing-in-memory accelerators based on Epitome. Proceedings of the 61st ACM/IEEE Design Automation Conference (DAC '24), San Francisco, USA, 2024, 1-6.

[12] Han L, Huang P, Zhou Z, et al. Pipeline design of nonvolatile-based computing in memory for convolutional neural networks inference accelerators. 2024 Design, Automation & Test in Europe Conference & Exhibition (DATE), 2024: 1-2.