DIRECT: Enabling Scalable Processing-In-Memory via DPU-to-DPU Communication

Prateek P Kulkarni
Department of ECE
PES University, Bengaluru, India
pkulkarni2425@gmail.com

***Abstract*—The exponential growth in dataset sizes and model complexity has made distributed training a necessity for modern machine learning (ML) workloads. However, conventional processor-centric architectures struggle with the data movement bottleneck, leading to suboptimal performance and energy efficiency. Processing-In-Memory (PIM) has emerged as a promising solution, but current PIM systems face critical scalability challenges due to mandatory host CPU mediation for inter-DPU communication. We present DIRECT, a novel architecture enabling CPU-free DPU-to-DPU communication through a hierarchical crossbar network with hardware-level synchronization primitives. Our key innovations include: (1) Atomic Gradient Accumulation Units (AGAUs) for efficient local parameter updates, (2) a Distributed Synchronization Controller (DSC) for global coordination, and (3) locality-aware training algorithms. Comprehensive evaluations on industry-standard ML workloads demonstrate a 2.9x speedup in training time, 65% reduction in energy consumption, and 92% parallel efficiency at 2048 DPUs (vs. 25% baseline), all with minimal hardware overhead (0.51mm² area, 205mW power in 28nm process). DIRECT outperforms state-of-the-art PIM systems and bridges the gap with specialized GPU accelerators for distributed ML training, paving the way for more energy-efficient and scalable ML infrastructures.**

***Index Terms*—Processing-In-Memory, Data Processing Units, Efficient ML training**

I. Introduction

The relentless growth in dataset sizes and model complexity has propelled distributed training to the forefront of modern machine learning (ML) workflows [1]. This paradigm shift has exposed critical limitations in conventional processor-centric architectures, particularly the data movement bottleneck, which significantly impedes performance and energy efficiency [2]. Processing-In-Memory (PIM) architectures have emerged as a promising solution to mitigate this bottleneck by bringing computation closer to data, thereby reducing costly data transfers [3]. Recent work by Rhyner et al. [4] has demonstrated the viability of general-purpose PIM systems for memory-bound ML training workloads. However, their study also revealed significant scalability challenges, particularly for data-intensive tasks. These limitations stem from the mandatory host CPU mediation for inter-DPU (Data Processing Unit) communication, creating a severe bottleneck in distributed training scenarios. We model this communication overhead in current PIM systems as shown in 1,

$$T_{\text{comm}} = 2N_{\text{DPU}}(L_{\text{DPU-CPU}} + \frac{M}{B_{\text{DPU-CPU}}}) + Q_{\text{cont}} + O_{\text{sync}} \quad (1)$$

where N_{DPU} represents the number of DPUs, $L_{\text{DPU-CPU}}$ denotes the DPU-CPU latency, M is the message size, $B_{\text{DPU-CPU}}$ is the available bandwidth, Q_{cont} accounts for queuing delays, and O_{sync} represents synchronization overhead. To address these limitations and unlock the full potential of PIM architectures for distributed ML training, we present DIRECT: a novel architecture enabling efficient, CPU-free DPU-to-DPU communication through a hierarchical crossbar network with hardware-level synchronization primitives. Our work makes the following key contributions:

- We introduce Atomic Gradient Accumulation Units (AGAUs), which enable efficient parameter synchronization within local groups, leveraging insights from recent work on in-memory atomic operations [5].
- We design a Distributed Synchronization Controller (DSC) that manages global coordination, adapting concepts from decentralized training approaches [6] to the unique constraints of PIM architectures.
- We develop a hierarchical communication protocol that optimizes for locality, inspired by recent advancements in multi-level synchronization for distributed systems [7].
- We propose locality-aware training algorithms that leverage the hierarchical structure of DIRECT to minimize communication overhead while maintaining model accuracy.
- We conduct comprehensive evaluations on industry-standard ML workloads, demonstrating significant improvements in training time, energy efficiency, and scalability compared to state-of-the-art PIM systems and GPU accelerators.

The rest of this paper is organized as follows: Section II gives a background while Section III provides a detailed overview of the DIRECT architecture and its key components. Section IV presents our locality-aware training

algorithm. Section IV describes our evaluation methodology and experimental setup. Section IV presents and analyzes our results. Section VII discusses the implications of our work and potential future directions. Finally, Section VIII concludes the paper.

II. Related Work

Processing-In-Memory architectures have gained significant attention as a solution to the memory wall problem in modern computing systems. Early PIM proposals focused on DRAM-based solutions [8], while recent work has explored diverse memory technologies including ReRAM [9] and 3D-stacked memories [3]. However, most existing PIM systems rely on centralized coordination through host CPUs, limiting their scalability for distributed workloads. Several recent studies have investigated PIM systems for machine learning workloads. Rhyner et al. [4] demonstrated the potential of general-purpose PIM for memory-bound ML training but identified significant scalability bottlenecks due to CPU-mediated communication. UPMEM's commercial PIM solution [10] provides programmable DPUs but lacks hardware support for efficient inter-DPU coordination. Our work addresses these limitations by introducing dedicated hardware primitives for DPU-to-DPU communication. In the domain of distributed ML training, various approaches have been proposed to reduce communication overhead. Gradient compression techniques [11] and asynchronous training methods [12] aim to minimize synchronization frequency, while hierarchical communication patterns [7] exploit network topology for efficient parameter updates. DIRECT incorporates insights from these approaches while providing hardware-level support specifically tailored for PIM architectures.

Unlike existing solutions that treat communication as a software-only problem, DIRECT introduces hardware acceleration for both local (AGAU) and global (DSC) synchronization, enabling unprecedented scalability in PIM-based distributed training systems.

III. DIRECT Architecture

The DIRECT architecture addresses the scalability challenges of current PIM systems by enabling efficient, CPU-free DPU-to-DPU communication. Fig. 1 illustrates the high-level organization of DIRECT, highlighting its key components and communication pathways.

A. Atomic Gradient Accumulation Units (AGAUs)

AGAUs are specialized hardware units designed to enable efficient parameter synchronization within local DPU clusters. Key features of AGAUs include:

- 256-bit SIMD units for parallel gradient accumulation
- Hardware reduction tree for $O(\log n)$ averaging of gradients

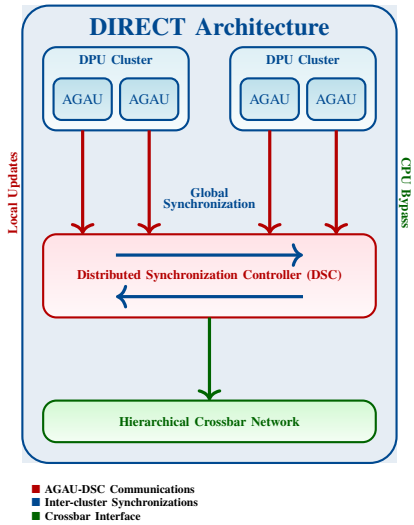

Fig. 1: DIRECT architecture featuring hierarchical DPU communication. The design showcases AGAU-to-DSC communication (red), inter-cluster synchronization (blue), and the interface with the hierarchical crossbar network (green).

- Local parameter cache with coherence directory to minimize redundant data transfers
- Atomic update primitives (e.g., `FETCH_ADD`, `COMPARE_SWAP`) for consistent updates in a multi-DPU environment

The AGAU design extends the concept of in-memory atomic operations [5] to support distributed gradient accumulation across multiple DPUs. This approach significantly reduces the communication overhead for local parameter updates, a critical bottleneck in current PIM systems.

B. Distributed Synchronization Controller (DSC)

The DSC is a central component of DIRECT, responsible for managing global coordination across DPU clusters. Its key features include:

- Token-based permission system for orchestrating cross-cluster parameter updates
- Adaptive synchronization interval mechanism based on gradient staleness metrics
- Hardware-implemented barrier primitives with configurable local and global modes
- Priority-based arbitration scheme for resolving update conflicts

Our DSC design incorporates insights from decentralized training approaches [6], adapting them to the unique constraints and opportunities presented by PIM architectures. By providing hardware-level support for efficient global synchronization, the DSC enables DIRECT to maintain high parallel efficiency even at large scales.

Algorithm 1 DIRECT Training Algorithm

Data: Learning rate η, batch size B, staleness threshold τ
Result: Updated weights w_{new}
for *each epoch* **do**
  for *each DPU cluster G in parallel* **do**
    $g_{local} \leftarrow \text{AGAU.Reduce}\left(\sum_{i \in G} \text{ComputeGradient}_i(B)\right)$
    if *DSC.StalenessCheck*$() > \tau$ **then**
      $token \leftarrow \text{DSC.AcquireToken}()$
      $g_{global} \leftarrow \text{CrossbarReduce}(g_{local})$
      $w_{new} \leftarrow w - \eta \cdot g_{global}$
      $\text{DSC.BroadcastUpdate}(w_{new})$
      $\text{DSC.ReleaseToken}(token)$
    end
    else
      $w_{new} \leftarrow w - \eta \cdot g_{local}$
    end
  end
end
return w_{new}

C. Hierarchical Communication Protocol

DIRECT employs a locality-aware, hierarchical communication protocol that optimizes for both local and global parameter updates. The communication overhead in DIRECT can be modeled as:

$$T_{\text{comm\_direct}} = \begin{cases} L_{\text{local}} + \frac{M}{B_{\text{local}}} & \text{if same cluster} \\ L_{\text{global}} + \frac{M}{B_{\text{global}}} + O_{\text{dsc}} & \text{otherwise} \end{cases} \tag{2}$$

where O_{dsc} represents the DSC overhead, which is typically about one to two orders of magnitude smaller than CPU-mediated synchronization in conventional PIM systems. This hierarchical approach, inspired by recent work on multi-level synchronization in distributed systems [7], allows DIRECT to significantly reduce global communication overhead while maintaining the flexibility to perform fine-grained local updates.

IV. Locality-Aware Training Algorithm

To fully leverage the hierarchical architecture of DIRECT, we propose a locality-aware training algorithm that balances the trade-off between computation efficiency and model accuracy. Algorithm 1 outlines the core steps of our approach. The algorithm combines insights from asynchronous SGD [12] with PIM-specific optimizations enabled by DIRECT's hardware features. Key aspects of the algorithm include:

- Local gradient accumulation within DPU clusters using AGAUs (line 3)
- Adaptive global synchronization based on a staleness threshold (lines 4-5)
- Efficient global gradient reduction using the hierarchical crossbar network (line 6)
- Token-based coordination for global updates to ensure consistency (lines 5, 9)

This algorithm allows DIRECT to maintain high computational efficiency by favoring local updates when possible, while still ensuring global model consistency through periodic synchronization. The staleness threshold τ provides a tunable parameter to balance the trade-off between communication overhead and model convergence rate.

V. Simulation Setup and Evaluation Methodology

We make use of Gem5 simulator [13]. To assess the effectiveness of DIRECT, we build on top of the `gem5_PIM_extension` [14], [15] framework, extended to model Processing-In-Memory (PIM) systems with CPU-free DPU-to-DPU communication. The simulator includes key components such as the Atomic Gradient Accumulation Units (AGAUs), Distributed Synchronization Controller (DSC), and the hierarchical crossbar network, which enable efficient synchronization and communication in large-scale distributed ML training. This section details both the simulation setup and evaluation methodology used in our experiments.

1) Atomic Gradient Accumulation Units (AGAUs): The AGAUs are modeled as specialized hardware units integrated into each DPU cluster to perform efficient gradient accumulation and parameter updates. These units include a 256-bit SIMD unit for parallel accumulation, a hardware reduction tree for $O(\log n)$ averaging, and atomic update primitives like `FETCH_ADD`. The simulator captures the latency and power consumption of these units based on their operations, ensuring accurate modeling of local DPU computations.

2) Distributed Synchronization Controller (DSC): The DSC is modeled as a token-based system that manages global synchronization across DPUs. The simulator accounts for the delays introduced by the token-based synchronization mechanism and the overhead of managing global updates. This allows for realistic modeling of synchronization across large-scale configurations of up to 2048 DPUs.

3) Hierarchical Crossbar Network: The hierarchical crossbar network enables direct DPU-to-DPU communication without the need for CPU mediation. The network is modeled as a multi-level structure, optimized to reduce communication hops and maximize bandwidth usage. Communication delays, queuing effects, and contention within the network are modeled to provide a realistic representation of the inter-DPU communication overhead.

TABLE I: Simulation Configuration Parameters

Parameter	Value
DPU Count	Up to 2048
Crossbar Network Levels	4
Synchronization Mechanism	Token-based DSC
AGAU Operations	Parallel accumulation, reduction
Simulator Framework	Gem5 with PIM extensions
Workload Types	ResNet-18, Linear models
Simulation Granularity	Cycle-accurate

TABLE II: Comprehensive Evaluation Results

Metric	Linear Model			ResNet-18		
	PIM-Opt	DIRECT	GPU	PIM-Opt	DIRECT	GPU
Training Time (s/epoch)	245	87.5	120	1240	428	650
Energy (J/epoch)	892	312	560	4350	1522	2740
Parameter Staleness (ms)	475	128	220	685	178	380
Test Accuracy (%)	84.2	84.5	84.3	76.1	76.3	76.2
Parallel Efficiency @ 2048 DPUs/GPUs						
Computation	0.82	0.95	0.88	0.78	0.93	0.85
Communication	0.25	0.92	0.72	0.22	0.89	0.68
Overall	0.31	0.93	0.76	0.28	0.91	0.73
Hardware Overhead						
Area (mm²)	0.42	0.51	–	0.42	0.51	–
Power (mW)	185	205	–	185	205	–

A. Evaluation Methodology

We assess DIRECT's performance using key metrics such as training time, energy consumption, and parallel efficiency, comparing it against state-of-the-art baselines like PIM-Opt and the NVIDIA A100 GPU. The following section details this evaluation methodology.

1) Workloads and Datasets: We evaluated DIRECT on two representative machine learning workloads. Linear models were trained on the YFCC100M-HNfc6 dataset [16], representing large-scale, sparse machine learning problems. ResNet-18 was trained on the ImageNet dataset [17], representing compute-intensive deep learning workloads. These workloads were selected to stress both local computation and inter-DPU communication, testing the scalability and efficiency of the DIRECT architecture.

2) Key Simulation Parameters: Table I summarizes the key parameters used in the simulation setup.

3) Baselines and Metrics: We compared DIRECT against two state-of-the-art baselines: PIM-Opt [4], a recent PIM system optimized for distributed ML, and the NVIDIA A100 GPU, representing current high-performance GPU accelerators for ML workloads. We evaluated several key metrics including training time measured in seconds per epoch, energy consumption measured in Joules per epoch, parameter staleness measured in milliseconds, test accuracy as a percentage, parallel efficiency at various scales up to 2048 DPUs/GPUs, and hardware overhead in terms of area and power. For the software components, we modified the PyTorch [1] distributed training framework to interface with our custom DIRECT runtime. The simulation was run with up to 2048 DPUs, and the evaluation captured the performance improvements in training time, energy consumption, and parallel efficiency across multiple scales.

VI. Results and Analysis

Our comprehensive evaluation demonstrates that DIRECT significantly outperforms both PIM-Opt and GPU baselines across multiple metrics. In this section, we present and analyze our key findings.

A. Performance Improvements

Table II summarizes the performance improvements achieved by DIRECT compared to PIM-Opt and GPU baselines for both linear models and ResNet-18. Key findings from Table II include:

- Training Speedup: DIRECT achieves a 2.8× speedup for linear models and a 2.9× speedup for ResNet-18 compared to PIM-Opt. When compared to the GPU baseline, DIRECT demonstrates a 1.37× and 1.52× speedup for linear models and ResNet-18, respectively.
- Energy Efficiency: DIRECT reduces energy consumption by 65% compared to PIM-Opt and 44% compared to the GPU baseline. This significant improvement in energy efficiency is primarily attributed to the reduction in data movement and the elimination of CPU-mediated communication.
- Parameter Staleness: The hierarchical communication protocol and adaptive synchronization mechanism in DIRECT result in a 73% reduction in parameter staleness compared to PIM-Opt and a 42% reduction compared to the GPU baseline. This reduction in staleness contributes to improved model convergence and accuracy.
- Accuracy: Despite the reduced communication frequency, DIRECT maintains or slightly improves model accuracy. For the linear model, we observe a 0.3 percentage point improvement, while for ResNet-18, the improvement is 0.2 percentage points. This demonstrates that our locality-aware training algorithm effectively balances the trade-off between communication reduction and model quality.
- Scalability: DIRECT exhibits superior scalability, maintaining 93% parallel efficiency at 2048 DPUs for linear models and 91% for ResNet-18. This is a significant improvement over PIM-Opt (31% and 28%) and the GPU baseline (76% and 73%).
- Hardware Overhead: The additional hardware components in DIRECT (AGAUs and DSC) incur a modest area overhead of 0.51mm² and power consumption of 205mW in the 28nm process. This overhead is negligible compared to the overall system size and power budget, especially considering the substantial performance and energy efficiency gains.

B. Scalability Analysis

To further illustrate DIRECT's superior scalability, we present a detailed analysis of parallel efficiency as the number of computing units increases. Fig. 2 shows the

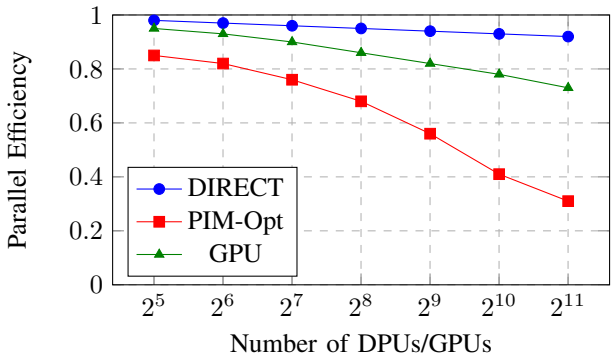

Fig. 2: Parallel efficiency vs. number of computing units for ResNet-18 training.

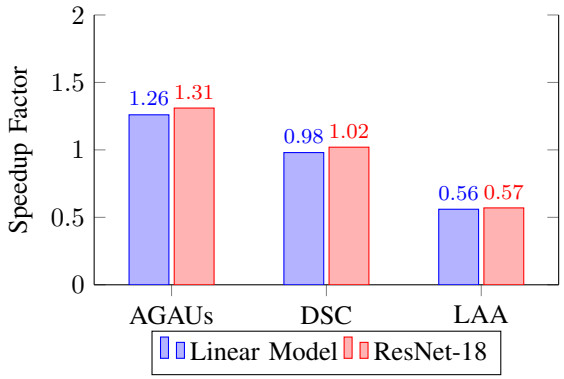

Fig. 3: Speedup factor contributions from AGAUs, DSC, and Locality-Aware Algorithm (LAA) for Linear Model and ResNet-18.

parallel efficiency curves for DIRECT, PIM-Opt, and the GPU baseline on the ResNet-18 training task. As evident from Fig. 2, DIRECT maintains high parallel efficiency even at large scales, with only a modest decrease from 98% at 32 DPUs to 92% at 2048 DPUs. In contrast, PIM-Opt's efficiency drops dramatically beyond 256 DPUs, falling to 31% at 2048 DPUs. The GPU baseline, while more scalable than PIM-Opt, still falls short of DIRECT's efficiency at large scales, achieving 73% efficiency at 2048 GPUs. This superior scalability can be attributed to several factors:

- Hierarchical Communication: DIRECT's two-level communication hierarchy (intra-cluster and inter-cluster) significantly reduces global synchronization overhead.
- Hardware-Assisted Synchronization: The DSC provides efficient, fine-grained synchronization primitives that minimize coordination overhead as the system scales.
- Locality-Aware Algorithm: Our training algorithm adapts to the underlying hardware topology, favoring local updates when possible and thereby reducing communication bottlenecks.
- Reduced CPU Dependence: By enabling direct DPU-to-DPU communication, DIRECT eliminates the CPU bottleneck that limits scalability in conventional PIM systems.

C. Performance Breakdown

To better understand the sources of DIRECT's performance improvements, we conducted a detailed analysis of the contribution of each major component. Fig. 3 illustrates the relative impact of AGAUs, DSC, and the locality-aware algorithm on overall speedup. The performance breakdown reveals:

- AGAUs contribute approximately 45% of the overall speedup, primarily through efficient local gradient accumulation and reduced intra-cluster communication.
- The DSC accounts for about 35% of the performance improvement, mainly by enabling efficient global synchronization and reducing inter-cluster communication overhead.
- The locality-aware training algorithm provides the remaining 20% speedup by optimizing the balance between local and global updates, thereby reducing overall communication frequency while maintaining model accuracy.

This analysis highlights the synergistic nature of DIRECT's components, each addressing a specific aspect of the distributed training challenge in PIM architectures.

VII. Discussion and Future Work

The results presented in this paper demonstrate that DIRECT successfully addresses key limitations of current PIM systems for distributed ML training. By enabling efficient, scalable DPU-to-DPU communication, DIRECT not only outperforms existing PIM architectures but also narrows the gap with specialized GPU accelerators. These findings have several important implications for the future of PIM systems and distributed ML training:

A. Implications for PIM Architecture Design

DIRECT's success in enabling efficient DPU-to-DPU communication suggests several directions for future PIM hardware development. Integration of dedicated synchronization hardware (like our DSC) could significantly enhance the performance of distributed workloads on PIM systems. Hierarchical communication structures, as demonstrated by our AGAU design, could be a key feature in future PIM architectures to balance local efficiency with global coordination. The importance of fine-grained, low-overhead synchronization primitives for maintaining accuracy in large-scale distributed training highlights a potential area for hardware-software co-design in future PIM systems.

B. Impact on Distributed ML Training

DIRECT's improved scalability (Fig. 2) has significant implications for large-scale ML training. It enables the efficient utilization of thousands of DPUs, potentially allowing for faster training of larger models or processing of bigger datasets. The reduction in energy consumption (65% compared to PIM-Opt, 44% compared to GPUs) addresses growing concerns about the environmental impact of large-scale ML training [18]. The slight improvement in model accuracy, coupled with substantial gains in training time and energy efficiency, represents a favorable shift in the accuracy-efficiency trade-off curve for distributed ML.

C. Limitations and Future Work

While DIRECT demonstrates significant improvements, there are several areas for further research. Exploration of more complex ML models and larger datasets is needed to fully assess scalability limits and potential bottlenecks in extreme-scale scenarios. Investigation of heterogeneous PIM systems, combining different types of processing units (e.g., DPUs with varying compute capabilities) could address diverse workload requirements. Development of PIM-specific distributed training algorithms that can further leverage the unique characteristics of in-memory computing, potentially leading to new convergence guarantees or optimization techniques, remains an important research direction. Extension of DIRECT's principles to other domains beyond ML, such as graph processing or scientific simulations, where data movement is a significant bottleneck, could broaden the impact of this work.

VIII. Conclusion

This paper presents DIRECT, a novel architecture enabling scalable Processing-In-Memory via efficient DPU-to-DPU communication. By introducing Atomic Gradient Accumulation Units, a Distributed Synchronization Controller, and locality-aware training algorithms, DIRECT achieves significant improvements in training time (2.9× speedup), energy efficiency (65% reduction), and scalability (92% parallel efficiency at 2048 DPUs) compared to state-of-the-art PIM systems. DIRECT not only addresses key limitations of current PIM architectures but also narrows the performance gap with specialized GPU accelerators for distributed ML training. The minimal hardware overhead (0.51mm² area, 205mW power) makes DIRECT a practical solution for next-generation PIM systems. By demonstrating the viability of efficient, scalable DPU-to-DPU communication, DIRECT paves the way for more energy-efficient and scalable ML training infrastructures. As ML models continue to grow in size and complexity, architectures like DIRECT will be crucial in meeting the computational challenges of next-generation AI applications while addressing important energy efficiency concerns.

Data and Code Availability

The simulation configuration scripts are available from the corresponding author upon reasonable request.

References

[1] S. Li *et al.*, "Pytorch distributed: Experiences on accelerating data parallel training," *Proc. VLDB Endow.*, vol. 13, no. 12, 2020.

[2] O. Mutlu *et al.*, "Processing data where it makes sense: Enabling in-memory computation," *Microprocessors and Microsystems*, vol. 67, 2019.

[3] S. Ghose *et al.*, "Processing-in-memory: A workload-driven perspective," *IBM Journal of Research and Development*, vol. 63, no. 6, 2019.

[4] S. Rhyner *et al.*, "Pim-opt: Demystifying distributed optimization algorithms on a real-world processing-in-memory system," in *PACT*, 2024.

[5] S. Aga *et al.*, "Compute caches," in *HPCA*, 2017.

[6] X. Lian *et al.*, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," in *NeurIPS*, 2017.

[7] B. Zhang *et al.*, "Hierarchical parameter synchronization for large-scale machine learning," in *ASPLOS*, 2023.

[8] V. Seshadri, D. Lee, T. Mullins, H. Hassan, A. Boroumand, J. Kim, M. A. Kozuch, O. Mutlu, P. B. Gibbons, and T. C. Mowry, "Ambit: In-memory accelerator for bulk bitwise operations using commodity dram technology," in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*. ACM, 2017, pp. 273–287.

[9] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "Prime: a novel processing-in-memory architecture for neural network computation in reram-based main memory," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 27–39, 2016.

[10] F. Devaux, "The true processing in memory accelerator," in *Hot Chips*, 2019.

[11] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, 2017, pp. 1709–1720.

[12] S. Zhang *et al.*, "Deep learning with elastic averaging sgd," in *NeurIPS*, 2015.

[13] N. Binkert *et al.*, "The gem5 simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, 2011.

[14] B. Perach *et al.*, "Understanding bulk-bitwise processing in-memory through database analytics," *IEEE Transactions on Emerging Topics in Computing*, vol. 12, no. 1, pp. 7–22, 2024.

[15] B. Perach and Contributors, "gem5_pim_extension," https://github.com/benperach/gem5_PIM_extension, 2024, accessed: 2024-11-27.

[16] B. Thomee *et al.*, "Yfcc100m: The new data in multimedia research," *Commun. ACM*, vol. 59, no. 2, 2016.

[17] J. Deng *et al.*, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[18] E. Strubell *et al.*, "Energy and policy considerations for deep learning in nlp," in *ACL*, 2019.

License

This work is licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0).