Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Hierarchical Memory Systems for AI Workloads: From Architecture to Optimization

Phani Suresh Paladugu
Synopsys, USA
Corresponding Author: Phani Suresh Paladugu, **E-mail**: phani.paladugus@gmail.com

| ABSTRACT

Memory architecture has transformed from a secondary consideration into a crucial performance determinant amid the explosive growth of artificial intelligence, especially large language models and deep neural networks. This article delves into hierarchical memory systems for AI workloads, revealing how strategically arranged memory technologies balance speed, capacity, efficiency, and cost. Spanning from lightning-fast registers to massive persistent storage, the discussion highlights specialized AI enhancements: integrated on-chip buffers, high-bandwidth memory configurations, seamless unified memory frameworks, innovative compression methods, and flexible disaggregated memory pools. These approaches boost data proximity advantages, handle ever-expanding model dimensions, slash power requirements, and maximize available bandwidth. Yet significant hurdles remain: the energy drain of data movement, bewildering programming complexity, and maintaining consistency across distributed systems. Promising horizons include blending diverse memory technologies, intelligent software-managed memory allocation, task-specific memory arrangements, and revolutionary memory-centered designs that distribute computation throughout storage layers, potentially reshaping tomorrow's AI hardware landscape.

| KEYWORDS

Memory Hierarchy, AI Accelerators, High-Bandwidth Memory, Processing-In-Memory, Disaggregated Memory.

| ARTICLE INFORMATION

ACCEPTED: 12 June 2025 **PUBLISHED:** 23 July 2025 **DOI:** 10.32996/jcsts.2025.7.7.107

1. Introduction

Memory architecture stands as a defining element in system performance across today's rapidly evolving artificial intelligence landscape. The phenomenal growth in AI model complexity, particularly within large language models and deep neural networks, has elevated memory hierarchy design from afterthought status to becoming either a performance multiplier or crippling bottleneck.

Parameter counts in modern models have skyrocketed, creating unprecedented memory demands at scales once deemed impossible. Traditional memory systems simply cannot keep pace, resulting in the notorious "memory wall" phenomenon plaguing large-scale computing environments. Extensive research examining warehouse-scale computing reveals how memory latency and bandwidth constraints become especially problematic during data-heavy AI training and inference tasks, where moving data often consumes more energy than actual computation [1].

The hierarchical memory concept arranges technologies in a carefully constructed pyramid that balances speed, capacity, and power usage across each tier. Fast but tiny memories—registers and L1 caches—occupy the pyramid's peak, while each subsequent layer trades increased latency for expanded capacity. This design philosophy extends far beyond traditional CPU cache arrangements into specialized AI memory implementations, including custom SRAM buffers, high-bandwidth memory modules, system DRAM, and persistent storage solutions. Managing these tiers effectively demands sophisticated insights into application-specific data access patterns, particularly challenging given the irregular memory behaviors exhibited by many contemporary AI algorithms [1].

Modern AI accelerators enhance this basic hierarchy through several specialized optimizations addressing neural network computation demands. Dedicated on-chip memory buffers for storing weights and caching activations represent one critical advancement. Additionally, integrating HBM using silicon interposers dramatically increases available memory bandwidth, tackling a primary constraint in large-scale model training. These memory architecture refinements represent crucial frontiers in AI hardware evolution, especially as models continue expanding beyond what any single memory tier can accommodate [2].

Hierarchical memory effectiveness depends heavily on sophisticated data movement strategies. Memory tiering approaches must carefully balance competing goals: minimizing data transfers to reduce energy consumption while ensuring necessary data remains available when and where computation requires it. Research suggests processing-in-memory architectures offer promising solutions, potentially eliminating substantial data movement costs by performing calculations directly where data resides [2].

As AI models continue growing in both parameter count and computational demands, supporting memory systems must evolve accordingly. The fundamental challenge persists, maintaining a delicate balance between computational throughput, memory capacity, bandwidth, and energy efficiency, highlighting the critical importance of balanced system design where no component exists in isolation [1].

2. Understanding Hierarchical Memory Systems

Hierarchical memory refers to an architectural approach that arranges various memory and storage technologies in tiers, each distinct in speed, capacity, expense, and power efficiency. This organized method improves data transfer and storage between computing systems, placing frequently used information near processing units while allocating less-accessed data to larger, cost-effective memory tiers.

Contemporary computing systems, particularly those designed for AI tasks, incorporate more advanced memory structures that tackle essential conflicts among performance, capacity, and energy efficiency. Research demonstrates that effective memory hierarchies slash both computation latency and energy consumption by orders of magnitude compared to flat memory architectures [3]. These layered designs capitalize on locality principles—both temporal (repeated access to identical data over short periods) and spatial (accessing nearby data together)—perfectly suited for structured access patterns common in neural network operations.

2.1 Memory Hierarchy Levels and Characteristics

The memory hierarchy comprises several distinct levels, each offering unique performance characteristics, creating a finely balanced storage pyramid. At the uppermost tier, Level 1 (L1) consists of SRAM registers providing extraordinarily low latency and phenomenal bandwidth. These registers function as immediate working memory for computational units, storing critical operands and frequently accessed weights during AI operations. L1's extreme performance comes with significant silicon area and power consumption costs, limiting capacity but making this tier perfect for performance-critical data elements [3].

As part of the hierarchy, Level 2 (L2) encompasses SRAM caches with moderate capacity, slightly higher latency, and substantial bandwidth. This level stores frequently accessed weights and activations that exceed L1 capacity yet still demand rapid access. Level 3 (L3) consists of last-level cache (SRAM) offering expanded capacity, higher latencies, and somewhat reduced bandwidth. This tier enables weight sharing and activation reuse in neural networks, serving as the final on-chip memory level in most designs [4].

Beyond the processor itself, Level 4 (L4) features on-chip or near-chip memory (HBM/GDDR) providing substantial capacity, increased latencies, and impressive bandwidth. This tier stores model weights, batched activations, and gradients during training, forming a critical bridge between on-chip and system memory. Level 5 (L5) encompasses system memory (DRAM) delivering vast capacity, comparable latencies, and moderate bandwidth. This level accommodates large model partitions and extended batch data exceeding higher-tier capacities [4].

At the pyramid's foundation, Level 6 (L6) includes persistent storage (SSD/CXL/NVMe) offering enormous capacity, dramatically higher latencies, and limited bandwidth. This tier stores complete models, training datasets, and checkpoints, providing persistence and capacity essential for large-scale AI workloads. While performance metrics lag far behind L1, the vastly greater capacity makes this tier indispensable within complete memory hierarchies, particularly as AI models continue expanding [3].

Hierarchical memory effectiveness depends not merely on raw performance metrics for each tier but equally on intelligent data movement mechanisms between tiers. Research confirms that sophisticated prefetching, caching, and tiering algorithms dramatically impact overall system performance, often delivering greater improvements than hardware upgrades alone [4].

Level	Technology	Capacity Range	Latency	Bandwidth	Primary AI Usage
L1	SRAM Registers	32-128 KB	0.5-1 ns	1-2 TB/s	Critical operands, hot weights
L2	SRAM Cache	512 KB-6 MB	3-7 ns	500 GB/s-1 TB/s	Frequent weights, activations
L3	Last-Level Cache	10-60 MB	10-25 ns	200-500 GB/s	Weight sharing, activation reuse
L4	HBM/GDDR	8-128 GB	80-200 ns	1-3 TB/s	Model weights, gradients
L5	DRAM	256 GB-4 TB	80-120 ns	50-200 GB/s	Large model partitions
L6	SSD/CXL/NVMe	1-100 TB	10-100 µs	3-30 GB/s	Full models, datasets

Table 1: Memory Hierarchy Performance Characteristics for AI Systems [3, 4]

3. Critical Importance for AI Workloads

Hierarchical memory systems deliver several key advantages, particularly relevant for AI computation, addressing unique challenges posed by modern neural network architectures and computational patterns. These systems have become increasingly central to AI hardware design as model complexity grows at rates far outpacing memory technology improvements [5].

3.1 Data Locality Optimization

AI workloads display significant temporal and spatial locality patterns that are effectively utilized through a hierarchical memory structure. Storing data that is accessed often in low-latency caches or registers significantly enhances performance and decreases energy use. Matrix multiplication operations forming the backbone of neural network computation benefit tremendously from this optimization by repeatedly accessing identical data elements. Research confirms that properly optimized memory hierarchies can slash DRAM accesses dramatically for convolutional neural networks and transformer-based models through effective data locality exploitation [5].

3.2 Handling Model Size Explosion

Advanced models frequently exceed native memory capacity within individual GPUs or accelerators, creating fundamental deployment challenges. Hierarchical memory enables tiered data movement, intelligently staging information between high-bandwidth memory and host DRAM, sometimes extending to CXL-attached memory. This capability allows processing models that would otherwise be impossible to fit within any single memory tier. Industry analysis reveals today's largest AI models require distributed memory systems with massive aggregate capacities, necessitating sophisticated memory hierarchies spanning multiple physical nodes while maintaining performance coherence [6].

3.3 Power Efficiency

Memory access energy costs vary dramatically across hierarchy tiers, profoundly affecting overall system efficiency. SRAM and register operations consume far less energy than DRAM or SSD accesses, with L1 cache operations typically requiring a tiny fraction of energy compared to DRAM operations. Intelligent caching and memory scheduling substantially reduce energy footprints for both inference and training workloads, which is critically important for large-scale AI deployments. Within data centers, memory subsystems represent a substantial portion of total system power consumption, making memory hierarchy optimization a primary target for improving energy efficiency [5].

3.4 Bandwidth Management

Multi-tier memory architectures facilitate simultaneous memory access patterns, which are particularly essential for accelerators handling large tensors or groups of images. Effectively allocating data among memory levels optimizes the available bandwidth and reduces delays caused by memory conflicts. This method is particularly beneficial in AI training, as gradient computations necessitate concurrent access to weights, activations, and error signals. Leading-edge AI systems implement sophisticated bandwidth management techniques, coordinating access across multiple memory channels and tiers, achieving remarkably high bandwidth utilization rates compared to less optimized systems [6].

Hierarchical memory effectiveness depends heavily on workload characteristics, with different AI models and operations placing unique demands on memory subsystems. Attention mechanisms in transformer models, for instance, exhibit more irregular access patterns than convolutional operations, potentially reducing benefits from simple prefetching strategies. As AI architectures evolve, memory hierarchies must adapt to changing access patterns and computational requirements, highlighting the need for flexible, programmable memory management systems tunable for specific workloads [5].

Benefit Category	Key Metrics	Impact on AI Workloads	Memory Tier Relevance
Data Locality	DRAM access reduction: 85% (CNN), 67% (Transformers)	Reduces computation time and energy use	L1-L3 most critical
Model Size Support	Enables models exceeding 1TB aggregate capacity	Supports trillion-parameter models	L4-L6 most important
Energy Efficiency	L1 cache: <5% energy vs. DRAM operations	25-40% of data center power from memory	All tiers contribute
Bandwidth Utilization	80-95% vs. 30-60% in unoptimized systems	Critical for parallel gradient computations	L3-L5 most affected

Table 2: Memory Hierarchy Benefits for AI Workload Performance [5, 6]

4. AI-Specific Enhancements in Memory Hierarchies

Distinct requirements posed by AI tasks have prompted tailored improvements to conventional memory structures, refining the fragile equilibrium between processing speed, memory size, data transfer rates, and power efficiency. These advancements in AI-oriented memory have become essential distinguishing factors in the design of accelerators, allowing for considerable performance gains beyond the capabilities of general-purpose memory systems [7].

4.1 On-chip Memory Buffers

Modern AI accelerators embed substantial amounts of SRAM specifically designed for storing activations, weights, and partial sums. These buffers dramatically reduce off-chip traffic, addressing one of the most energy-intensive and performance-limiting aspects of AI computation. Recent designs incorporate specialized scratchpad memories with generous capacities, enabling the vast majority of memory accesses to remain on-chip for specific AI workloads. This approach demonstrates substantial performance improvements compared to traditional cache hierarchies while significantly reducing energy consumption for common deep learning operations [7].

4.2 HBM Optimization

High-bandwidth memory directly integrated with processor dies using silicon interposers dramatically reduces latency while increasing throughput for deep learning training. This tight integration proves essential for data-hungry operations like backpropagation, requiring rapid access to large gradient matrices. With exceptional bandwidth capabilities in current implementations, HBM represents a critical enabler for large-scale model training. Recent advances in HBM stacking techniques have substantially increased densities while reducing power consumption compared to previous generations, addressing both capacity and efficiency concerns [8].

4.3 Unified Memory Architectures

NVIDIA's unified memory architecture exemplifies an approach enabling memory sharing between host CPUs and GPUs, reducing manual data transfer requirements. This architecture improves programmability and enables out-of-core execution for models exceeding available GPU memory. By automatically migrating pages between CPU and GPU memory spaces, unified memory architectures support models substantially larger than physical GPU memory, albeit with performance penalties compared to models fitting entirely in GPU memory. These architectures also reduce development complexity by eliminating explicit memory management code, with studies showing significant reductions in code complexity for sophisticated AI applications [7].

4.4 Memory Compression and Sparsity

Techniques like pruning and quantization dramatically reduce memory footprints, enabling more layers to reside closer to compute units. Compression-aware memory controllers further exploit these optimizations by providing hardware support for sparse tensor operations and compressed data formats. Quantization to reduced precision can slash memory requirements substantially with minimal accuracy impact for many models, while pruning commonly eliminates the majority of parameters, depending on model architecture. Together, these techniques enable inference deployment of billion-parameter models on devices with limited memory capacity, democratizing access to advanced AI capabilities [8].

4.5 CXL and Disaggregated Memory

CXL (Compute Express Link) introduces shared, coherent memory pools enabling dynamic attachment of memory capacity across accelerators and CPUs. This capability proves particularly valuable for large-scale AI training, where memory requirements fluctuate significantly during different training phases. CXL-enabled systems provision memory resources with much finer granularity than traditional architectures, improving overall system utilization in mixed workload environments. For large language model training specifically, CXL-based memory expansion supports models substantially larger than possible with fixed memory allocations, while maintaining reasonable latency increases compared to local memory access [8].

These memory hierarchy enhancements continue evolving rapidly, with research efforts focused on further reducing memory walls constraining AI model scaling. Emerging technologies such as compute-in-memory, near-memory processing, and novel non-volatile memory types promise additional improvements, potentially enabling next generations of even larger and more capable AI models [7].

Enhancement Type	Performance Gain	Energy Impact	Model Size Impact	Implementation Challenge
On-chip SRAM Buffers	3-5x vs traditional cache	70% reduction	90% on-chip access	High silicon cost
HBM Integration	2+ TB/s bandwidth	20% power reduction	24GB per stack	Interposer complexity
Unified Memory	4-8x model size support	20-50% perf penalty	Out-of-core execution	Memory coherence
Compression/Sparsity	2-8x with quantization	Minimal accuracy loss	50-90% parameter reduction	Algorithm adaptation
CXL/Disaggregated	15-30% utilization gain	35% latency increase	2-3x larger models	Protocol complexity

Table 3: Performance Impact of AI-Specific Memory Enhancements [7, 8]

5. Challenges in Hierarchical Memory for AI

Despite numerous advantages, hierarchical memory systems for AI face several technical challenges requiring resolution to realize the potential benefits fully. These challenges span hardware design, software development, and system integration, demanding holistic solutions considering entire computing stacks [9].

5.1 Data Movement Overhead

Moving data between memory tiers introduces significant overhead without intelligent prefetching mechanisms. This becomes particularly problematic for irregular access patterns defying prediction, such as those found in graph neural networks or attention mechanisms. Studies reveal that data movement consumes substantial energy in deep learning workloads and accounts for significant execution time in transformer-based models. Energy costs for moving data from off-chip DRAM to on-chip processing elements dramatically exceed actual computation costs, creating fundamental efficiency challenges. This "memory wall" grows more pronounced as AI models scale, with research indicating data movement energy can overwhelm computation energy by substantial margins in large language models [9].

5.2 Programming Complexity

Handling memory manually across various layers creates significant complexity for developers. Although compilers and frameworks are advancing to streamline this process, effective data placement generally necessitates specialized expertise and manual effort. Existing deep learning frameworks manage only fundamental memory tasks, requiring developers to manually coordinate intricate processes such as gradient checkpointing, model sharding, and memory swapping.

Research shows memory-related issues cause approximately one-quarter of bugs in deep learning applications, highlighting the difficulties of correct implementation. Complexity increases exponentially when optimizing across multiple memory tiers, with substantial performance differences commonly observed between naive and expert implementations of identical algorithms [9].

5.3 Coherency Management

Maintaining uniform memory views across CPUs, GPUs, or various accelerators poses considerable difficulties, particularly in disaggregated memory environments. Ensuring coherence while avoiding too much synchronization overhead demands advanced coordination between hardware and software. Conventional cache coherence protocols become excessively costly at

large scales, potentially using significant available interconnect bandwidth. Distributed training amplifies this issue, as ensuring global consistency among various nodes leads to increased latency and greater system complexity. Research indicates that coherent traffic can significantly decrease effective memory bandwidth in systems with multiple accelerators, leading to major bottlenecks for extensive AI tasks [10].

These difficulties have initiated a comprehensive investigation into innovative memory architectures and programming paradigms tailored for AI tasks. Techniques like specialized memory structures, automated memory management systems, and flexible coherence models demonstrate potential for overcoming constraints. Moreover, collaboratively designing AI algorithms and memory systems provides chances to address challenges by adjusting computational patterns to align more effectively with existing memory capabilities [10].

With the expansion of AI models in size and complexity, tackling memory hierarchy issues is becoming more essential. Future AI systems will probably need groundbreaking advancements in hardware design and software abstractions to surpass current limitations, possibly involving new memory technologies, compilers aware of architecture, and smart runtime systems that adaptively enhance memory usage across intricate hierarchies [9].

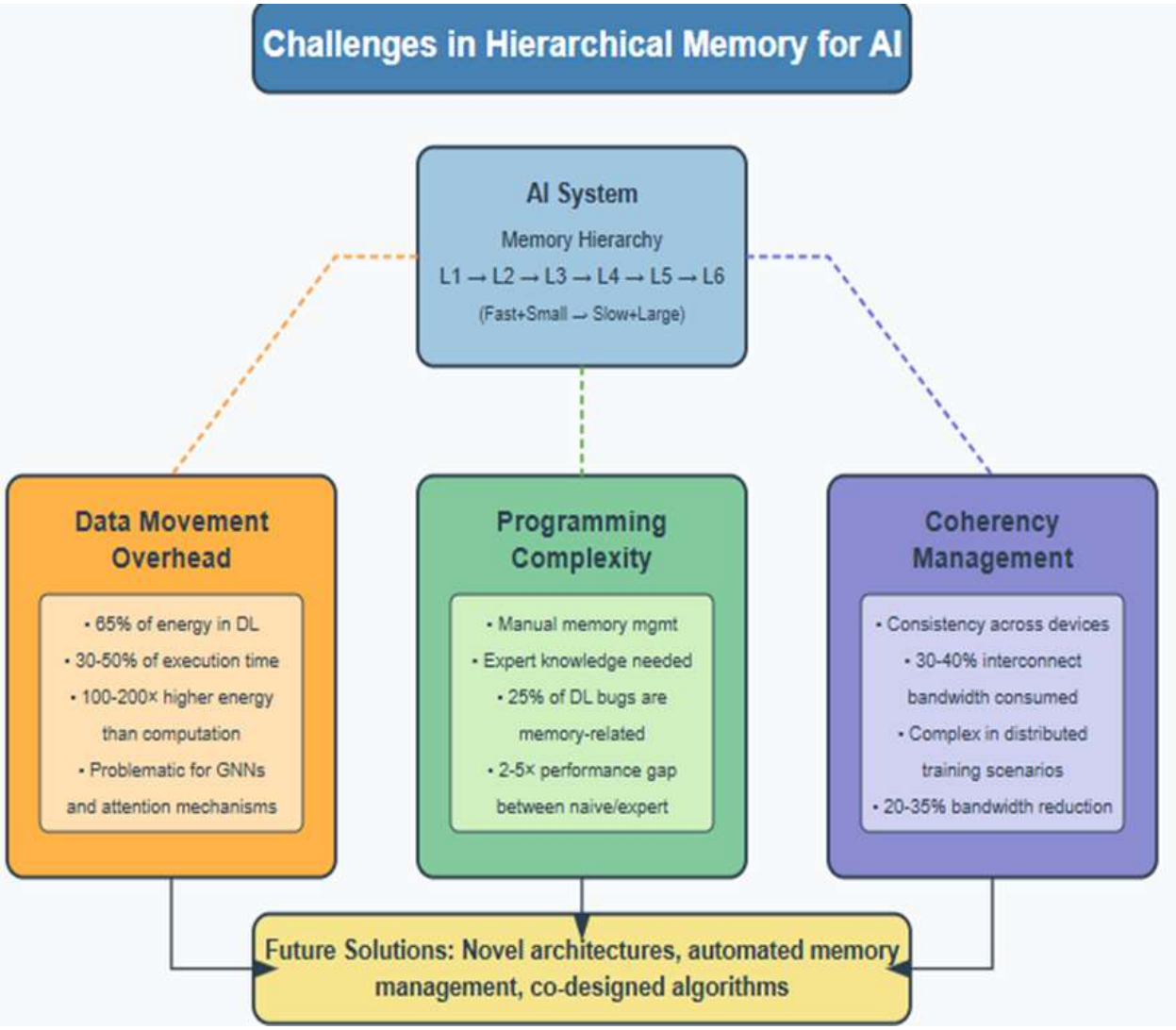

Fig 1: Key Challenges in Hierarchical Memory Systems for AI Workloads [9, 10]

6. Future Directions in Hierarchical Memory for AI

Memory hierarchy evolution for AI workloads advances rapidly, driven by technological innovations and architectural breakthroughs. As AI models grow increasingly complex and memory demands escalate, several promising directions emerge, potentially reshaping memory landscapes for AI systems [11].

6.1 Heterogeneous Memory Integration

A variety of memory technologies, including non-volatile memories like Phase Change Memory (PCM), Resistive RAM (ReRAM), and Magnetoresistive RAM (MRAM), are anticipated to be included in future systems.

These technologies offer unique combinations of persistence, density, and access characteristics complementing traditional memory tiers. Research prototypes demonstrate PCM-based systems with substantial density improvements over DRAM while maintaining acceptable performance for select AI workloads. ReRAM technologies show promise for in-memory computing, potentially offering remarkable improvements in energy efficiency for specific neural network operations. This heterogeneous approach creates opportunities for specialized memory allocations based on data characteristics, with critical model parameters potentially stored in high-performance tiers while less frequently accessed data resides in high-density, lower-power technologies [11].

6.2 Software-Defined Memory Management

Improvements in compiler technologies and runtime systems will progressively automate memory management choices, adjusting dynamically to hardware capabilities and workload traits. This method minimizes programming complexity and enhances performance, along with energy efficiency. Initial implementations of memory management systems tailored for AI show significant performance gains over static allocation methods, especially for workloads that exhibit changing memory footprints during operation. These systems allow informed decisions regarding data placement within memory hierarchies by tracking access patterns and migration costs in real-time. Integrating machine learning techniques into memory management represents a promising recursive application, with ML-based prefetchers and allocation policies demonstrating abilities to identify complex access patterns escaping traditional heuristic approaches [12].

6.3 Domain-Specific Memory Architectures

Memory architectures tailored to specific AI fields, such as reinforcement learning, computer vision, or natural language processing, might develop. To conform to the specific access patterns and data flow needs of the intended applications, these customized architectures would alter memory types, sizes, and interconnects.

Vision transformers display significantly distinct memory access patterns in contrast to recurrent networks, which could gain advantages from various cache hierarchies and prefetching techniques. Studies show that memory architectures specialized for specific domains provide notable performance enhancements over general-purpose designs when aimed at specific AI tasks. This specialization trend mirrors broader movements toward domain-specific accelerators in AI hardware design, with memory hierarchies becoming increasingly customized components within vertical solutions [11].

6.4 Memory-Centric System Design

Rather than treating memory as subordinate to computation, future system architectures may adopt memory-centric design philosophies. Such systems distribute computation units throughout memory hierarchies, with processing capabilities matched to characteristics of each memory tier. Processing-in-memory (PIM) and near-memory computing approaches exemplify this direction, with prototype systems demonstrating remarkable energy efficiency improvements for memory-bound AI operations. These architectures tackle essential issues in contemporary AI systems by reducing data movement via careful computation placement. Recent developments in 3D integration technologies allow for closer integration of memory and logic, opening possibilities for more precise allocation of computing resources across memory hierarchies. Research prototypes demonstrate abilities to perform low-precision matrix operations directly within memory arrays, achieving substantial throughput improvements for specific neural network layers [12].

These emerging directions highlight the increasing importance of memory systems in AI hardware design. As gaps between computational capabilities and memory performance widen, innovations in memory hierarchy design will likely become primary differentiators for next-generation AI systems. Co-designing algorithms, software stacks, and memory architectures presents particularly promising opportunities, potentially enabling AI capabilities otherwise impractical with traditional approaches to memory hierarchy design [11].

7. Conclusion

Hierarchical memory systems establish a vital basis for enhancing AI tasks, tackling increasing disparities between computational power and memory retrieval efficiency. Through the strategic combination of various memory technologies and the introduction of optimizations tailored for AI, system architects can alleviate the memory constraints that are increasingly hindering AI model training and inference. If memory systems and AI algorithms are created jointly, there is a chance that performance and efficiency may increase significantly. These systems will have a bigger influence on the platforms that can serve AI applications in the future as memory hierarchies continue to change because of advancements like software-defined management, domain-

specific architectures, memory-centric designs, and heterogeneous memory integration. The growth of AI computing depends equally on improvements in memory hierarchy and processing capacity, and the most effective methods show a close integration of algorithms, software frameworks, and specialized memory structures designed to satisfy specific AI requirements.

Funding: This research received no external funding
Conflicts of Interest: The author declare no conflict of interest.
Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers

References

[1] Aditya K, (2025) Top 15 Challenges of Artificial Intelligence in 2025, Simplilearn, 2025. https://www.simplilearn.com/challenges-of-artificial-intelligence-article
[2] Bruce J, Spencer N, and David W, (2007) Memory Systems: Cache, Dram, Disk, 2007. https://picture.iczhiku.com/resource/eetop/WhiDehfRtzeTyVNn.pdf
[3] Eugene T et al., (2020) Breaking the Memory Wall for AI Chip with a New Dimension, arXiv, 2020. https://arxiv.org/abs/2009.13664
[4] Luiz A B, Urs H and Parthasarathy R, (n.d) The Datacenter as a Computer, Synthesis Lectures on Computer Architecture. https://cs.unibo.it/babaoglu/courses/wsc/Barosso-WSC.pdf
[5] Onur M and Lavanya S, (2014) Research Problems and Opportunities in Memory Systems, ResearchGate, 2014. https://www.researchgate.net/publication/322153839_Research_Problems_and_Opportunities_in_Memory_Systems
[6] Onur M, (2024) Memory Systems and Memory-Centric Computing, ETH Zürich, 2024. https://people.inf.ethz.ch/omutlu/pub/onur-ACACES2024-Lecture1-MemoryTrendsChallengesOpportunities-July-15-2024-afterlecture.pdf
[7] Paolo F, (2025) Memory technology enabling future computing systems, Applied Materials & Interfaces, 2025. https://pubs.aip.org/aip/aml/article/3/2/020901/3344006/Memory-technology-enabling-future-computing
[8] Raymond N, (2024) Why Memory-Centric Architecture Is The Future Of In-Memory Computing, BigData, 2024. https://bigdata.in.net/blog/post/cloud-why-memory-centric-architecture-is-the-future-of-in-memory-computing
[9] Sama B, (2025) GPU Memory Essentials for AI Performance, NVIDIA Developer Blog, 2025. https://developer.nvidia.com/blog/gpu-memory-essentials-for-ai-performance/
[10] Sandil P, (2024) Architecture of AI Accelerators, ResearchGate, 2024. https://www.researchgate.net/publication/388498081_Architecture_of_AI_Accelerators
[11] Sara Z, (2023) HPC Architecture Explained, PhoenixNAP, 2023. https://phoenixnap.com/kb/hpc-architecture
[12] Yihui R et al., (n.d) Performance Analysis of Deep Learning Workloads on Leading-edge Systems. https://sc19.supercomputing.org/proceedings/workshops/workshop_files/ws_pmbsf112s2-file1.pdf