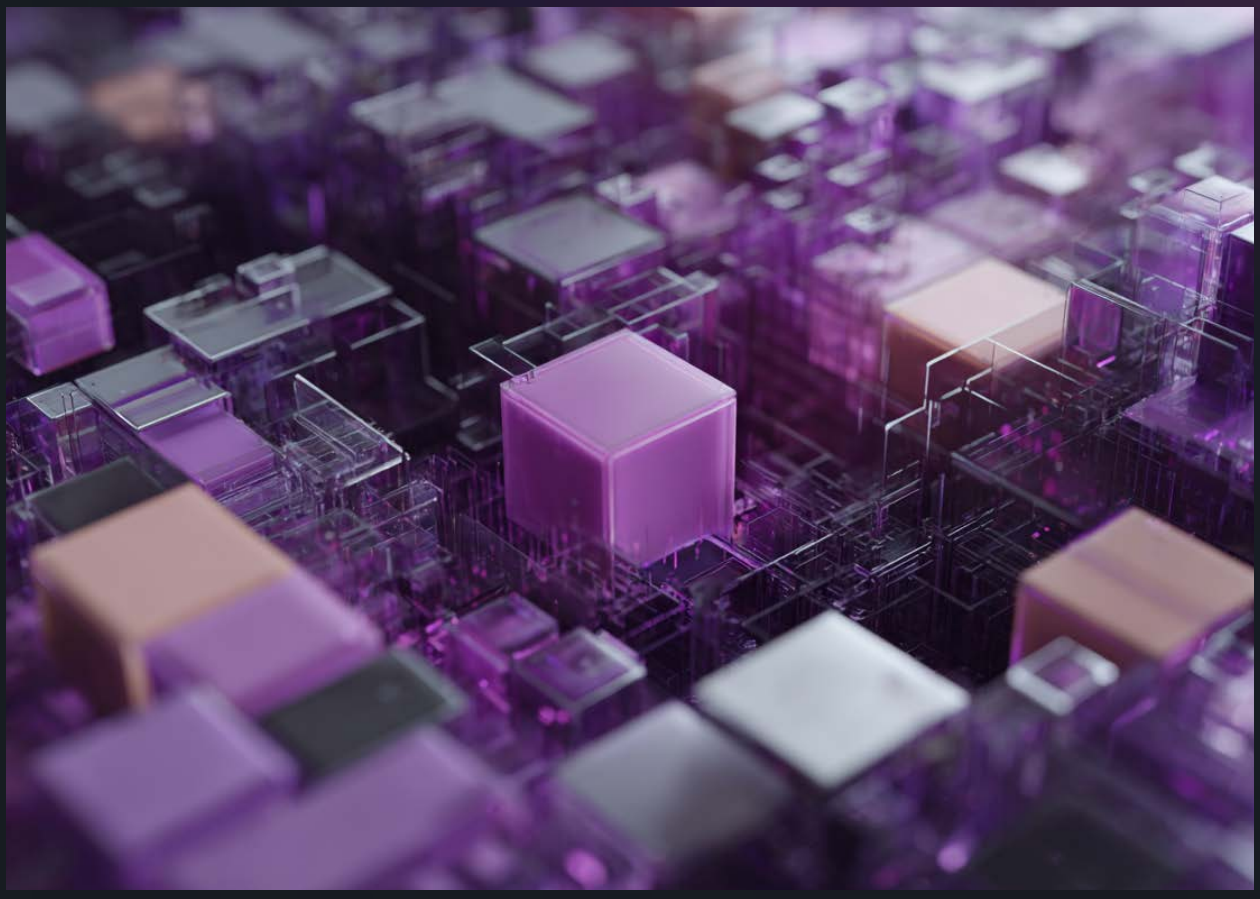
# The Impact of Storage on the AI Lifecycle



WRITTEN BY
Steve McDowell

NAND RESEARCH

COMMISSIONED BY
WEKA®

## Table of Contents


The Impact of Storage Architecture On the AI Lifecycle

How Storage Can Cause AI Pipelines to Slowdown

Storage Requirements within the AI Lifecycle

Storage Capabilities Needed to Meet AI Demands

How NeuralMesh Addresses AI Storage Bottlenecks

Breaking the Memory Wall: WEKA's Augmented Memory Grid

Conclusion: Storage for Production AI at Scale


Click the page number to return here

**DEPLOYING AI SOLUTIONS** is rapidly becoming a priority for enterprises across nearly every industry, challenging organizations of all types, from traditional IT organizations to neo-cloud providers, who struggle to adapt legacy infrastructure to meet the unique needs of the AI lifecycle. While AI impacts almost every element of the stack, modern AI pipelines are most directly impacted by the need to consume and generate data at a scale and speed not typically required by traditional enterprise workloads.

Today's enterprise AI agenda is increasingly driven by inference workloads — the production deployment of AI models to deliver real-time business value. Unlike the training phase, inference represents the operational heart of enterprise AI, where models must deliver consistent, low-latency responses to end-users and business applications.

Whether on-premises or in the cloud, legacy enterprise storage architectures often fail to meet the varying needs of each phase of the AI lifecycle. These approaches are particularly ill-suited for the demands of distributed training and modern inference workloads, where keeping expensive AI infrastructure idle – often due to storage bottlenecks – has a real economic impact on the enterprise.

Production AI fundamentally changes the requirements of the underlying storage infrastructure, with inference now demanding storage architectures that can support high concurrency, microsecond-sensitive operations while maintaining cost-effective token processing, or "tokenonomics", at an enterprise scale.

Furthermore, as AI models grow larger and more sophisticated, particularly with the emergence of agentic AI and large reasoning models (LRMs), the economics of token processing has become a critical factor in determining the viability and scalability of AI deployments.

As AI moves into the enterprise, inference performance has become a primary concern for AI model providers and enterprise IT leaders. While training workloads may run periodically, inference workloads must operate continuously, serving potentially thousands of concurrent users with strict latency requirements. This operational reality demands storage architectures that can deliver consistent, predictable performance while scaling seamlessly to meet fluctuating demand patterns.

Let's look at the unique demands placed on data infrastructure throughout the entire AI lifecycle including data ingestion, training, inferencing, and lifecycle management, with a particular emphasis on the inference challenges facing today's enterprise AI initiatives. Storage architectures must evolve to meet these challenges, and we'll look at how WEKA does just that with its purpose-built storage solution, NeuralMesh™ and its revolutionary Augmented Memory Grid™ technology.

# How Storage Can Cause AI Pipelines to Slowdown

Using a storage system that isn't designed for the rigors of deep learning can cause AI pipelines to stall, which can significantly hinder the efficiency and progress of the training process. This can lead to the underutilization of expensive AI training clusters.

There are multiple reasons that a storage system not designed for AI may stall:

- **Insufficient I/O Bandwidth:** When the storage system's bandwidth is inadequate to handle the volume of data being read or written, it can cause bottlenecks. This delays data loading and checkpointing, resulting in idle times for GPUs and CPUs.
- **High Latency:** High latency in accessing data from storage can significantly slow down the data retrieval process. Increased latency causes delays in feeding data to the training units, interrupting the training flow and reducing overall efficiency.
- **Limited Storage Throughput:** The storage system may not be able to sustain the required throughput for continuous data read/write operations. This can lead to slower data access speeds, causing training processes to stall while waiting for data.
- **Storage Contention:** Contention occurs when multiple processes or applications compete to access the same storage resources. This can lead to delays and inefficiencies as the storage system tries to manage concurrent access requests, resulting in slower data transfer rates.
- **Fragmentation:** Data fragmentation within the storage system can cause delays in accessing contiguous data blocks, increasing the time required to read/write data and interrupting the training process.
- **Insufficient Memory Buffering:** If the memory buffers used for caching data between storage and computation units are inadequate, it can cause frequent stalls. This results in frequent data swapping between memory and storage, which increases latency and reduces throughput.
- **Concurrency Issues:** Managing concurrent data access across multiple nodes in distributed training environments can be challenging. Poor coordination and synchronization can lead to stalls as nodes wait for data to be available or consistent.

Deploying a storage platform based on a storage architecture designed for the entire AI lifecycle can reduce storage stalls during both generative AI training and inference cycles, resulting in more efficient and effective AI deployments.

Unlike training workloads that can tolerate occasional performance variations, inference workloads directly impact user experience and business operations, making storage performance a critical success factor for enterprise AI deployments.

# Storage Requirements within the AI Lifecycle

The AI lifecycle is made up of a series of stages, each with unique performance requirements for storage systems. These stages include data collection and preparation, model training, inference and deployment, and subsequent lifecycle management. Interestingly, in today's AI-driven environment, inference operations dominate storage requirements due to their continuous and production-critical nature, as well as the need to serve real-time business applications.

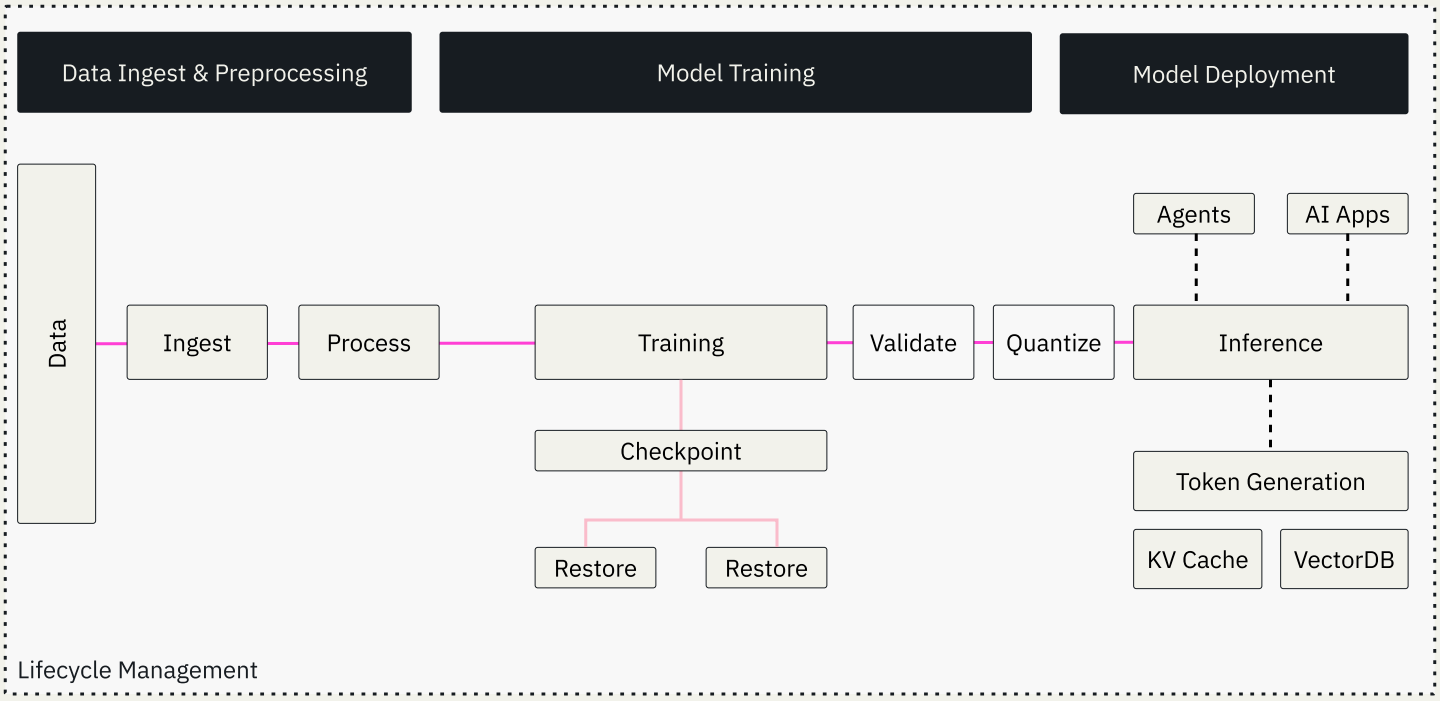



Figure 1: AI Lifecycle

Deep learning for generative AI is characterized by intensive data read and write operations, frequent access to large datasets, and substantial intermediate data handling. While these I/O patterns impact both training and inference processes, the performance characteristics required for enterprise inference are fundamentally different—demanding microsecond-level responsiveness, high concurrency support, and predictable latency under varying load conditions.

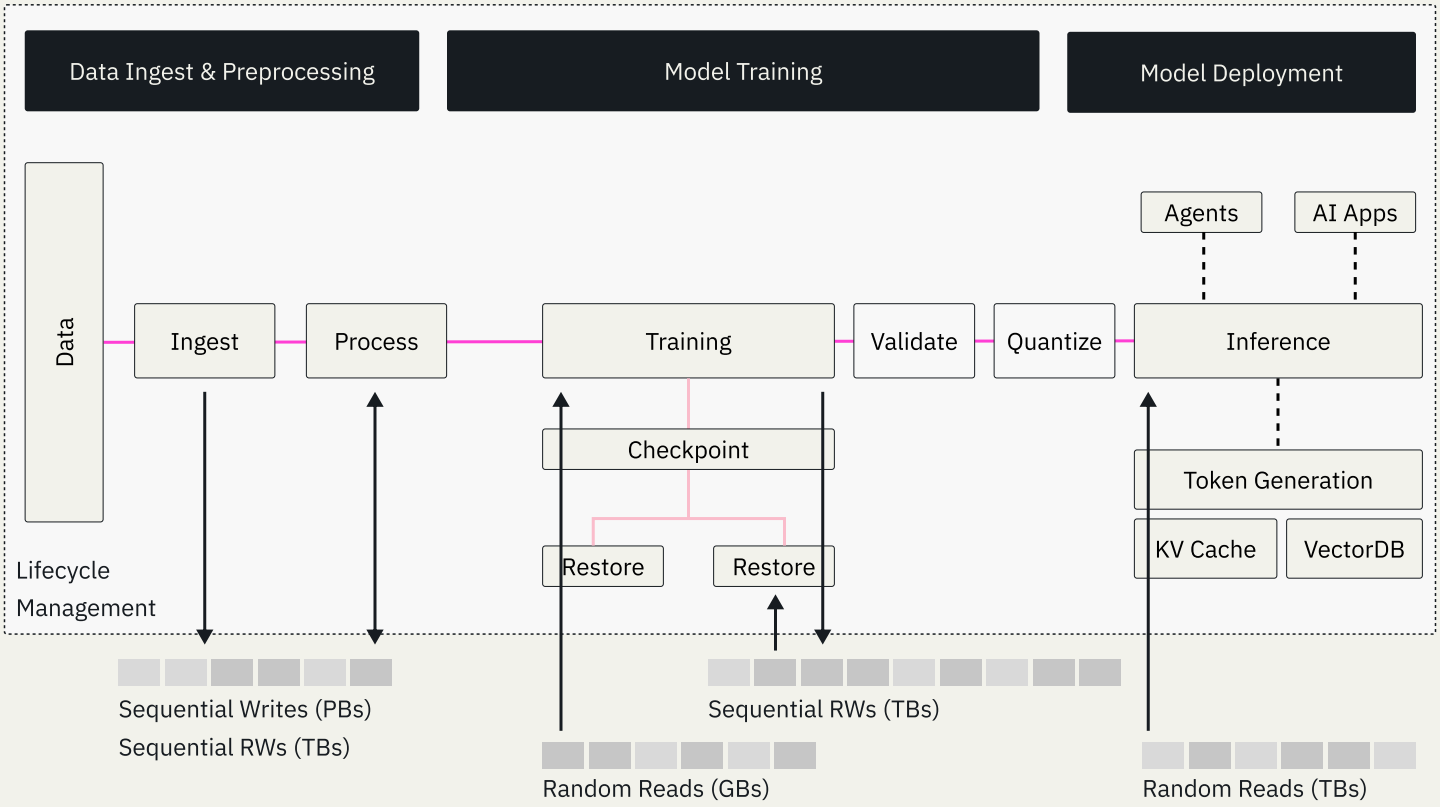



Figure 2: Impact of Storage on the AI Lifecycle

# Storage Requirements for Data Ingest and Preprocessing

In the first phase of the AI pipeline, data used for training and inference is collected from various sources. The raw data is then moved into a centralized storage system or data lake. Increasingly, enterprises are also implementing real-time data pipelines to support live inference applications that require access to continuously updated datasets and streaming data sources.

Once collected, the ingestion process uses ETL (Extract, Transform, Load) or other preprocessing techniques to ensure the data is correctly formatted and stored. This requires the system to move large amounts of raw data from storage into the servers performing the ETL functions.

For inference-focused deployments, this phase must also support real-time data transformation and feature engineering to ensure models receive properly formatted input data with minimal latency.

This stage involves mixed read and write operations from a storage system perspective. Raw data is read from storage and processed in memory. The preprocessed data is written back to storage or held in memory. High read-and-write IOPS are necessary to handle frequent small file operations, particularly for inference workloads that may require rapid access to diverse data sources and feature stores.

High sequential read throughput is crucial for efficiently loading large datasets. ETL pipelines often read data in batches to optimize memory usage and computational efficiency.

# Why Does Storage Throughput Matter for Model Training?

Training is where machine learning happens. At a high level, training is an iterative process where data that has been ingested and cleaned is delivered to a cluster of GPUs. Deep learning algorithms iteratively feed data into a neural network model in batches until the desired result is achieved.

While training remains critical for model development, enterprises are increasingly focused on optimizing training workflows to support rapid model iteration and deployment cycles.

The ability to quickly train, validate, and deploy models to production inference environments has become a competitive advantage, requiring storage architectures that can seamlessly support the transition from training to production deployment.

Efficient data handling, high computational power, and effective optimization techniques are critical to the success of deep learning training. The performance of the underlying storage architecture has the most significant impact here.

Training includes multiple tasks, each of which places differing demands on the storage system:

| Training Task | Description | I/O Characteristics |
|---|---|---|
| **Training Data Loading** | During training, batches of data are continuously loaded from storage into the training framework. | Sustained high throughput and low latency read operations are crucial. Data loading is often parallelized to keep GPUs/CPUs fully utilized. |
| **Checkpointing** | Periodic saving of model states (parameters, gradients, weights, biases, etc.) to storage to enable recovery and continuity in case of interruptions. | Efficient checkpointing requires high write throughput and low latency. The process involves writing large files at regular intervals, making sequential write performance important. |
| **Intermediate Data Handling** | Intermediate results, such as forward and backward pass computations, are temporarily stored during training. | High memory bandwidth and low latency are critical for handling the frequent read/write operations of intermediate data |
| **Model Evaluation & Validation** | Like training data loading, batches of validation data are read from storage to evaluate model performance. | High read throughput and low latency are needed to load validation data efficiently, ensuring timely evaluation without interrupting the training process. |

Table 1: I/O Characteristics For AI Training

Four primary processes impact storage infrastructure across training epochs:

**1. Data Storage:**
Training generative AI requires access to vast datasets, often comprising terabytes or more of data. High-performance storage systems provide faster data access and retrieval, reducing data loading times and improving overall training speed. For enterprise deployments, the same storage infrastructure must also support inference workloads, requiring architecture that can handle both batch training operations and real-time inference requests simultaneously.

**2. Checkpointing:**
Regular checkpointing (saving model states) is essential to prevent data loss and facilitate model recovery. Fast storage systems enable quick writing and reading of checkpoints, minimizing downtime and allowing for efficient training continuation. Checkpoints are crucial when training is distributed across multiple GPUs.

**3. Data Throughput:**
The rate at which data can be read from or written to storage is crucial for feeding data into the training pipeline. High-throughput storage solutions ensure that data is delivered to the GPU/CPU without delays, maximizing resource utilization and training efficiency. Enterprise-grade storage must maintain this throughput performance even when concurrently serving inference workloads that require predictable, low-latency access to model artifacts and feature data.

**4. Parallel Access with a Single Namespace:**
Training involves distributed systems with multiple GPUs/CPUs accessing data simultaneously. Storage systems with high parallel access capabilities support multiple data streams concurrently, reducing contention and ensuring smooth data flow.

# How Does Checkpointing Impact Model Progress?

Checkpointing during training is a critical process that involves periodically saving the model's state to storage. This practice ensures that training can resume from the last saved state in case of interruptions, such as hardware failures or power outages. The storage system's performance and capabilities significantly impact the efficiency and reliability of checkpointing.

| Storage Attribute | Checkpointing Impact |
|---|---|
| **Write Throughput & Latency** | High write throughput and low latency are crucial for fast and efficient checkpointing. A storage system with low write speeds or high latency can result in prolonged checkpointing times, leading to delays in the training process. |
| **Checkpoint Frequency** | With fast and reliable storage, more frequent checkpoints can be created without significantly affecting the training cycle. This provides finer-grained recovery points, minimizing data loss in case of a failure. Conversely, slow storage may necessitate less frequent checkpoints, increasing the risk of losing more progress during interruptions. |
| **Storage Capacity** | Insufficient storage capacity can limit the number of checkpoints that can be stored, leading to the overwriting or deletion of older checkpoints. This reduces the ability to revert to earlier states if needed. Adequate storage ensures that multiple checkpoints can be retained for better recovery options. |
| **Scalability** | Scalable storage solutions can accommodate growing data needs without performance degradation. This is important for maintaining efficient checkpointing as model sizes and training complexity increase. Non-scalable storage can become a bottleneck, slowing the checkpointing process and impacting overall training performance. |
| **Parallel Access** | Storage systems that support high parallel access can efficiently handle simultaneous checkpointing operations from multiple nodes, reducing contention and latency. This ensures smooth and synchronized checkpointing across the distributed training setup. Storage systems with poor parallel access capabilities can lead to contention and delays, disrupting the training workflow. |

Table 2: Impact of Checkpointing on AI Performance

The performance and characteristics of the storage system profoundly impact the efficiency and reliability of checkpointing during training. High-performance, reliable, and scalable storage solutions enable faster and more frequent checkpoints, reduce the risk of data loss, and ensure smooth training progress. By optimizing storage systems and implementing appropriate strategies, the overall training process can be made more resilient and efficient.

In enterprise environments, optimizing storage systems also directly impacts the speed at which new models can be deployed to production inference systems, making checkpoint performance a critical factor in competitive advantage and time-to-market for AI-driven business capabilities.

# What Makes Inference Storage Needs Different From Training?

When deployed, the model reads new input data to generate outputs (e.g., text generation, image synthesis). Inference is all about speed, efficiency, and cost efficiency. It requires ultra-low latency and high IOPs to quickly access the model in order to optimize response times. For enterprise inference applications, storage systems must have ultra-low latency while also supporting random access patterns to quickly retrieve specific data subsets required for individual inference requests.

This phase is increasingly characterized by the critical importance of tokenomics — the economics of token processing that directly impacts the cost-effectiveness and scalability of AI inference operations. The cost of an inference request, particularly with large language models, is often tied to token costs which are based on the number of tokens in both the input prompt and the generated output. Since inference is often performed at scale (millions or billions of times), even small improvements in efficiency can lead to significant cost savings.

AI inference faces unprecedented challenges as models grow larger and context windows expand. The emergence of agentic AI and large reasoning models has intensified the demand for optimizing real-time requests, creating what industry experts call the “memory wall”—a fundamental limitation where GPU memory constraints become the primary bottleneck in inference performance.

## What is the “Memory Wall” Challenge in Inference?

A fundamental limitation in modern AI inference is the amount of memory available. GPUs process vast amounts of data in parallel, but the memory available per GPU is fixed. As models grow in complexity and require longer contexts, their memory footprint expands beyond what a single GPU can handle. This results in inefficiencies where GPUs are memory-starved, causing significant bottlenecks in AI token generation.

Additionally, GPUs cannot directly scale their memory independently of compute. When additional memory is needed, the only solution is to add more GPUs, which increases costs without proportionally improving performance. Many AI workloads exhibit a mismatch in resource utilization, often resulting in up to 70% idle GPU time as excess computational power goes unused because more memory is required.

## Why Are Token Economics Now a Core Business Factor?

Tokenomics has emerged as a crucial factor in the success of AI deployments. The cost per token, influenced by factors such as time to first token (TTFT), overall throughput, and GPU utilization, directly impacts the economic viability of AI applications. Poor tokenomics can render even the most sophisticated AI models commercially unviable.

Key tokenomics challenges include:

- **Prefill Latency**: Large context windows can require 20+ seconds just for prefill operations, creating unacceptable user experiences
- **Memory Overhead:** KV cache requirements grows linearly with sequence length, forcing expensive GPU over-provisioning
- **Resource Waste:** Traditional architectures discard computed KV cache data after use, resulting in redundant computation on subsequent requests

## How Does Extending KV Cache Improve Performance?

The KV cache is a crucial component in transformer-based models, storing previously computed attention keys and values. This cache allows models to avoid recomputing these values for tokens that have already been processed, significantly improving inference efficiency. However, traditional approaches face severe limitations:

1. **Limited GPU Memory:** KV cache must fit within GPU memory constraints, limiting context length and concurrent requests
2. **Cache Disposal:** Most systems discard KV cache data after inference completion, wasting valuable computation
3. **Memory Contention:** Competition between model weights and KV cache for limited GPU memory reduces overall efficiency

The solution lies in extending KV cache capabilities beyond traditional GPU memory limitations while maintaining the microsecond-level access speeds required for efficient inference.

# How Does Lifecycle Management Drive Ongoing Storage Demand?

Once deployed, models need continuous monitoring for performance, drift, and anomalies, which involves collecting and storing new data. Periodically, models are retrained with updated data to improve accuracy. This stage requires efficient data ingestion, fast read/write operations, and scalability to handle growing data volume, demanding that the storage system delivers high throughput, low latency, high IOPS, and sufficient scalability.

In enterprise production environments, lifecycle management has become the dominant storage consumer, as inference workloads generate massive volumes of operational data, including request logs, response metrics, model performance telemetry, and user feedback. This operational data must be processed in real-time to detect model drift, performance degradation, and potential security issues. The storage infrastructure must simultaneously support high-throughput log ingestion, real-time analytics for monitoring dashboards, and rapid access to historical data for model retraining and compliance reporting.

Models and training data are also often archived, requiring the long-term storage of raw, processed, and generated data for future use, compliance, or audit purposes. Efficient archiving requires balanced read and write throughput. Compression and deduplication techniques are often employed to optimize storage space and reduce I/O load.

For enterprises operating at scale, the volume of inference-generated data often exceeds training data by orders of magnitude, requiring storage architectures that can efficiently tier data based on access patterns while maintaining rapid retrieval capabilities for compliance and audit requirements.

# Storage Capabilities Needed to Meet AI Demands

The AI pipeline requires a storage architecture that can consistently and concurrently meet the I/O demands for all stages of an AI pipeline. This task is often challenging for legacy storage architectures, which are designed for the less stringent requirements of traditional enterprise workloads. At the same time, parallel file systems, such as the open-source Lustre file system, are designed for classical HPC environments and lack the optimizations required to support today's AI lifecycle.

As discussed above, both training and inference workloads are challenges for traditional storage systems. While the high-throughput requirement of training requires AI-optimized storage, the rise of inference as the primary enterprise AI workload have even more fundamentally shifted storage requirements. Unlike training workloads, which can be scheduled and optimized for batch processing, inference workloads must deliver consistent and predictable performance under highly variable load conditions.

Enterprise inference applications may experience sudden traffic spikes, require sub-millisecond response times, and must maintain high availability standards that directly impact business operations and customer experience.

A storage architecture designed for the specific needs of the AI pipeline has to be optimized in several areas. These include its approach to handling metadata, its ability to manage distributed storage, its capability to extend beyond traditional memory limitations for inference optimization, and its ability to overcome common causes of slowdowns in the AI pipeline.

Most critically, for enterprise deployments, the storage architecture must excel at accelerating every part of the AI lifecycle by serving the concurrent, latency-sensitive access patterns that characterize production inference workloads while maintaining the high-throughput capabilities required for training and data processing operations.

## Why Does Metadata Efficiency Matter for AI Pipelines?

Efficient handling of storage metadata is critical throughout the AI lifecycle, from data ingestion and preprocessing to model training, deployment, and monitoring. Metadata operations, which include actions like opening files, fetching attributes, and managing directories, can significantly impact performance if not handled properly.

| Storage Attribute | Checkpointing Impact |
|---|---|
| **Data Ingest & Preprocessing** | **High Metadata Overhead:** During data ingestion and preprocessing, numerous files are often read, created, and modified. Each of these operations involves metadata transactions.<br>**Latency:** Poor metadata handling can lead to high latency, slowing down the rate at which data is ingested and preprocessed.<br>**Efficiency:** Efficient metadata management ensures that these operations are performed quickly, allowing for faster data ingestion and preprocessing, which is crucial for timely AI model training.<br>**Inference Support:** For real-time inference applications, metadata efficiency directly impacts the ability to access feature stores and input data with the microsecond-level responsiveness required for production workloads. |
| **Model Training** | **Frequent Checkpoints:** Training AI models, especially deep learning models, involves frequent checkpointing to save the model's state. Each checkpoint operation involves metadata transactions to create and manage checkpoint files.<br>**Performance Bottlenecks:** Inefficient metadata handling can lead to performance bottlenecks, where the training process stalls while waiting for metadata operations to complete.<br>**Scalability:** Efficient metadata handling is crucial for scaling up training processes, particularly in distributed training environments where multiple nodes concurrently access and modify data.<br>**Inference Integration:** Enterprise training workflows increasingly require seamless integration with deployment pipelines, where metadata operations must support both training checkpoints and inference model artifact management. |

Table 3: Impact of Metadata Handling on the AI Pipeline (continued on the next page)

| Storage Attribute | Checkpointing Impact |
|---|---|
| **Inference** | **Configuration Management:** Deploying AI models involves reading configuration files and loading model binaries, requiring metadata operations.<br><br>**Start-up Time:** Efficient metadata handling reduces the start-up time for model deployment, ensuring that models are quickly available for inference tasks.<br><br>**Reliability:** Poor metadata handling can lead to configuration errors and delays, affecting the reliability and responsiveness of AI services.<br><br>**Inference:** In production inference environments, metadata operations occur continuously as models access configuration data, feature definitions, and model artifacts, making metadata performance critical to maintaining SLA compliance and user experience. |
| **Lifecycle Management** | **Log Management:** Monitoring AI models involves generating and accessing numerous log files, including metadata operations for creation, modification, and deletion.<br><br>**Data Analysis:** Analyzing performance logs and other monitoring data requires efficient access to metadata to retrieve relevant files quickly.<br><br>**Retraining Efficiency:** Efficient metadata handling ensures that data needed for retraining models can be quickly accessed, facilitating timely updates and improvements.<br><br>**Managing Inference:** Enterprise inference workloads generate massive volumes of operational metadata that must be processed in real-time for monitoring, compliance, and continuous model improvement, making metadata performance a primary determinant of overall system efficiency. |

Table 3: Impact of Metadata Handling on the AI Pipeline

Efficient metadata handling is crucial for optimizing the performance and scalability of the AI lifecycle. From data ingestion and preprocessing to model training, deployment, and monitoring, each phase involves numerous metadata operations that can become bottlenecks if not appropriately managed.

In enterprise environments where inference workloads dominate storage operations, metadata performance has a direct impact on business-critical applications and user-facing services, making it a primary factor in overall system success and competitive advantage.

## How Does Distributed Storage with a Single Namespace Improve AI Workflows?

A distributed storage solution with a single namespace provides a unified view of the entire storage infrastructure, allowing users to access data across multiple storage systems and locations as if it were a single, coherent file system.

Deploying a storage solution that leverages a single namespace, and where distributed storage delivers a level of scalability, performance, collaboration, and efficiency that storage systems designed for the more traditional needs of the enterprise struggle to match. This addresses many of the challenges of handling large datasets and complex AI training and inference workflows.

A distributed storage architecture brings several efficiencies to any storage solution:

- **High Performance:** Data is distributed across multiple nodes, allowing parallel access and processing. This enhances read and write speeds, reduces latency, and improves overall performance crucial for AI training and inference tasks.
- **Fault Tolerance and Redundancy:** Data replication across different nodes ensures that no single point of failure can lead to data loss. This increases data reliability and availability, ensuring continuous operation even in case of hardware failures.

- **Scalability:** Storage can scale quickly by adding more nodes to the system, supporting the growing data storage needs of AI projects and allowing for expansion without significant reconfiguration.
- **Geographical Distribution:** Data can be stored across multiple geographic locations, reducing the latency for globally distributed teams and providing disaster recovery options by replicating data across regions.
- **Cost Efficiency:** Distributed storage systems can reduce costs by leveraging commodity hardware and cloud resources, using cost-effective solutions, and optimizing resource utilization.

Beyond these intrinsic benefits, a distributed storage architecture also brings specific benefits to the AI pipeline:

**Distributed System Impact**

| Phase | Single Namespace | Distributed Storage |
|---|---|---|
| **Data Ingest & Preprocessing** | Simplifies data aggregation from various sources, providing a unified view for preprocessing tasks.<br>For enterprise inference pipelines, enables seamless access to real-time data feeds and feature stores from multiple sources without complex data movement operations. | Handles large volumes of data efficiently, ensuring quick data ingestion and preprocessing through parallel processing.<br>Supports the high-concurrency data access patterns typical of production inference workloads. |
| **Model Training** | Ensures seamless access to training data, regardless of its physical location, simplifying the training setup.<br>Enables enterprise MLOps workflows where training and inference share the same data infrastructure.<br>Facilitates easy deployment of models by providing a consistent data access layer across different environments. | It provides high-throughput and low-latency data access, which is critical for feeding data to GPUs/CPUs during training. It supports distributed training across multiple nodes.<br>Maintains performance even when concurrent inference workloads are accessing the same storage infrastructure. |
| **Model Deployment** | Facilitates easy deployment of models by providing a consistent data access layer across different environments.<br>Critical for enterprise deployments where models must be accessible across multiple inference endpoints and geographic locations. | Ensures models and inference data are available with low latency, supporting real-time and batch inference needs.<br>Provides the scalability needed to serve thousands of concurrent inference requests with consistent performance. |
| **Lifecycle Management** | This namespace simplifies the collection and access of monitoring data from various sources, aiding in efficient retraining cycles.<br>Enables unified monitoring and observability across all inference deployments and geographic regions. | Manages storing and retrieving large volumes of monitoring data, ensuring efficient retraining processes.<br>Handles the massive data volumes generated by production inference workloads while maintaining rapid access for real-time monitoring and alerting. |

Table 4: Impact of Distributed Storage with a Single Namespace on the AI Pipeline

Combining a single namespace and distributed storage provides significant value across the AI lifecycle by enhancing data accessibility, improving performance, facilitating collaboration, and ensuring scalability and reliability. These technologies address the critical needs of modern AI workflows, enabling efficient data management and processing, which are essential for successful AI training, deployment, and continuous improvement.

For inference deployments, these capabilities are fundamental to achieving the scale, performance, and reliability required for business-critical AI applications that serve millions of users and process thousands of requests per second.

# How NeuralMesh Addresses AI Storage Bottlenecks

It's clear that traditional monolithic storage systems, designed for predictable workloads and fixed infrastructure, simply cannot meet the scale, speed, and complexity requirements of today's AI pipelines. The spiky I/O patterns, massive scale, multi-tenancy demands, and cloud-native deployment models inherent in enterprise AI environments expose the limitations of legacy approaches, leading to performance bottlenecks, operational rigidity, and resource underutilization.

Recognizing this challenge, WEKA developed NeuralMesh, a containerized microservices storage architecture built from the ground up for the AI era. NeuralMesh is a complete paradigm shift from traditional storage designs, embracing the microservices-based approach that has already transformed compute, networking, and application infrastructure throughout the modern data center.

NeuralMesh addresses the complex and demanding requirements of the AI lifecycle. It is an integrated, high-performance, scalable, and resilient storage system that effectively supports various stages of AI workflows, from data ingestion and preprocessing to model training, deployment, inference, and monitoring.


NeuralMesh delivers the five core capabilities essential for modern AI infrastructure:

- **Modularity:** Storage services, including protocol handling, data protection, and telemetry, are decoupled and run independently in orchestrated containers. This separation enables each service to be optimized for its specific function, allowing for independent scaling and management.
- **Elasticity:** Individual services can scale up or down dynamically based on demand without disrupting the rest of the system. This capability is crucial for AI workloads that experience dramatic variations in resource requirements across different phases of the lifecycle.

- **Isolation:** Failures in one component cannot cascade across the platform, providing fault tolerance and protection from noisy neighbors in multi-tenant environments. This isolation is particularly significant for enterprise AI deployments, where multiple projects and teams share a common infrastructure.
- **Agility:** Upgrades and changes can be rolled out with minimal impact, allowing for continuous operations and the rapid adoption of new capabilities. This agility supports the fast-moving nature of AI development, where new techniques and optimizations are constantly emerging.
- **Portability:** Container-based services can run anywhere — on bare metal, in the cloud, or in hybrid environments — without requiring custom infrastructure. This portability enables consistent performance and management across diverse deployment scenarios.

## Flexible Storage for AI

NeuralMesh allows AI teams to define and deploy storage resources with the same precision and automation they apply to compute and networking. Through orchestration and API-driven workflows, storage configurations can be version-controlled, automated, and integrated into CI/CD pipelines, aligning storage management with modern DevOps and MLOps practices.

This approach is particularly valuable for AI workloads where storage requirements can vary dramatically between training, validation, and inference phases. Teams can programmatically configure storage policies, performance characteristics, and resource allocations to match the specific needs of each workload, ensuring optimal performance and cost efficiency.

## Multi-Tenant by Design

Enterprise AI environments typically support multiple teams, projects, and workloads sharing common infrastructure. NeuralMesh is designed for multi-tenancy, offering isolated service execution, robust QoS controls, and granular resource management. This ensures that high-priority inference workloads receive guaranteed performance levels while development and training activities operate without interference.

The architecture's isolation capabilities also provide security benefits, ensuring that data and operations from different tenants remain separate while sharing the underlying infrastructure efficiency.

## Dynamic Scaling Without Downtime

Traditional storage systems require careful capacity planning and often disruptive upgrade procedures. NeuralMesh's microservices architecture enables dynamic scaling, allowing new capacity and services to be added without downtime or service interruption. This capability is essential for AI environments where data volumes can grow exponentially, and performance requirements can shift rapidly as models evolve, and user demands change.

The NeuralMesh architecture supports both scale-up and scale-out scenarios, enabling organizations to right-size their storage infrastructure according to current needs while maintaining the flexibility to expand seamlessly as requirements evolve.

## Microsecond Latency Through Optimized Design

NeuralMesh achieves microsecond latency by eliminating the performance overhead inherent in legacy storage architectures. Traditional systems suffer from layering inefficiencies, involuntary kernel context switches, and hardware bottlenecks that can add milliseconds of latency to each I/O operation. For AI inference workloads where response times directly impact user experience and business outcomes, these delays are unacceptable.

The service-oriented architecture of NeuralMesh eliminates these bottlenecks through optimized data paths, intelligent caching, and direct hardware integration, delivering the consistent and predictable performance required for production AI applications.

# NeuralMesh Advantages

NeuralMesh offers numerous advantages to alternative approaches:

| Attribute | NeuralMesh Advantage |
|---|---|
| **High Performance/ Low Latency** | **Parallel Data Access:** The NeuralMesh architecture supports parallel access to data, enabling multiple GPUs and CPUs to read and write data concurrently. This parallelism is crucial for maintaining high throughput during intensive AI training sessions.<br><br>**Optimized for Small I/O Operations:** NeuralMesh efficiently handles numerous small read and write operations, which are typical in AI workloads, especially during the training and inference phases.<br><br>**High Throughput:** NeuralMesh delivers high throughput and low latency, ensuring that data is rapidly available for AI processes and minimizing idle times for computational resources. |
| **Scalability** | **Horizontal Scaling:** NeuralMesh's distributed architecture allows for seamless horizontal scaling. As data volumes increase, additional nodes can be added to the storage system without compromising performance.<br><br>**Support for Large Datasets:** NeuralMesh can manage extensive datasets commonly found in AI projects, ensuring that the storage infrastructure scales with the organization's data needs. |
| **Metadata Handling** | **Efficient Metadata Operations:** NeuralMesh is designed to handle metadata operations efficiently, reducing latency associated with file access, directory listings, and attribute fetching. This efficiency is critical for maintaining performance in metadata-intensive AI workflows.<br><br>**Distributed Metadata Management:** By distributing metadata responsibilities across multiple nodes, NeuralMesh avoids bottlenecks and ensures consistent performance even under heavy metadata workloads. |
| **Fault Resilience** | **Data Replication:** NeuralMesh ensures high availability and reliability through robust data replication mechanisms. This fault tolerance is crucial for maintaining continuous AI operations, even in the event of hardware failures.<br><br>**High Availability:** The architecture ensures high availability, preventing storage system outages from disrupting AI workflows. |
| **Hybrid Cloud Support** | **Seamless Integration:** NeuralMesh integrates seamlessly with both on-premises and cloud environments, offering flexibility in deployment models. This hybrid cloud support enables organizations to leverage the scalability and cost advantages of the cloud while maintaining control over critical data.<br><br>**Data Mobility:** NeuralMesh facilitates easy data movement between on-premises and cloud storage, supporting dynamic AI workloads that may span multiple environments. |
| **GPU Memory Extension** | **Augmented Memory Grid:** NeuralMesh's revolutionary capability extends GPU memory to petabyte scale, breaking traditional memory barriers for inference workloads.<br><br>**KV Cache Persistence:** Enables retention and reuse of computed KV cache data, eliminating redundant computation and dramatically improving tokenomics. |
| **Service-Oriented Architecture** | **NeuralMesh Microservices:** NeuralMesh's container-native architecture delivers modularity, elasticity, isolation, agility, and portability through independent, orchestrated services.<br><br>**Storage-as-Code:** Enables programmatic definition and deployment of storage resources through API-driven workflows, integrating seamlessly with DevOps and MLOps practices.<br><br>**Dynamic Scaling:** Add capacity and services without downtime, supporting the rapidly changing requirements of AI workloads across training, validation, and inference phases.<br><br>**Multi-Tenant Isolation:** Provides strong QoS controls and resource management, ensuring high-priority inference workloads maintain guaranteed performance while supporting concurrent development activities. |

Table 5: Attributes of NeuralMesh

These attributes strategically map well to the needs of today's AI infrastructure:

**Model Training:** During model training, large volumes of data are continuously loaded into GPUs for processing. The high throughput and low latency NeuralMesh offers ensure that data is delivered efficiently, keeping GPUs fully utilized and reducing training times.

**Model Deployment and Inference:** During inference, models require quick access to data for real-time inference. NeuralMesh optimized metadata handling and intelligent caching provide the necessary speed and reliability, ensuring deployed models can operate effectively without latency issues. This is particularly critical for enterprise inference workloads that must serve thousands of concurrent requests while maintaining sub-millisecond response times and strict SLA compliance.

**Monitoring and Retraining:** Continuous monitoring and periodic retraining of AI models generate significant I/O activity. The scalable architecture and efficient data management capabilities support these processes by providing quick access to monitoring data and facilitating seamless data ingestion for retraining.

**Mixed Training & Inference Environments:** In production environments, the volume of inference-generated operational data often exceeds training data by orders of magnitude, requiring storage systems that can efficiently handle massive write throughput while maintaining rapid read access for real-time monitoring and analytics.

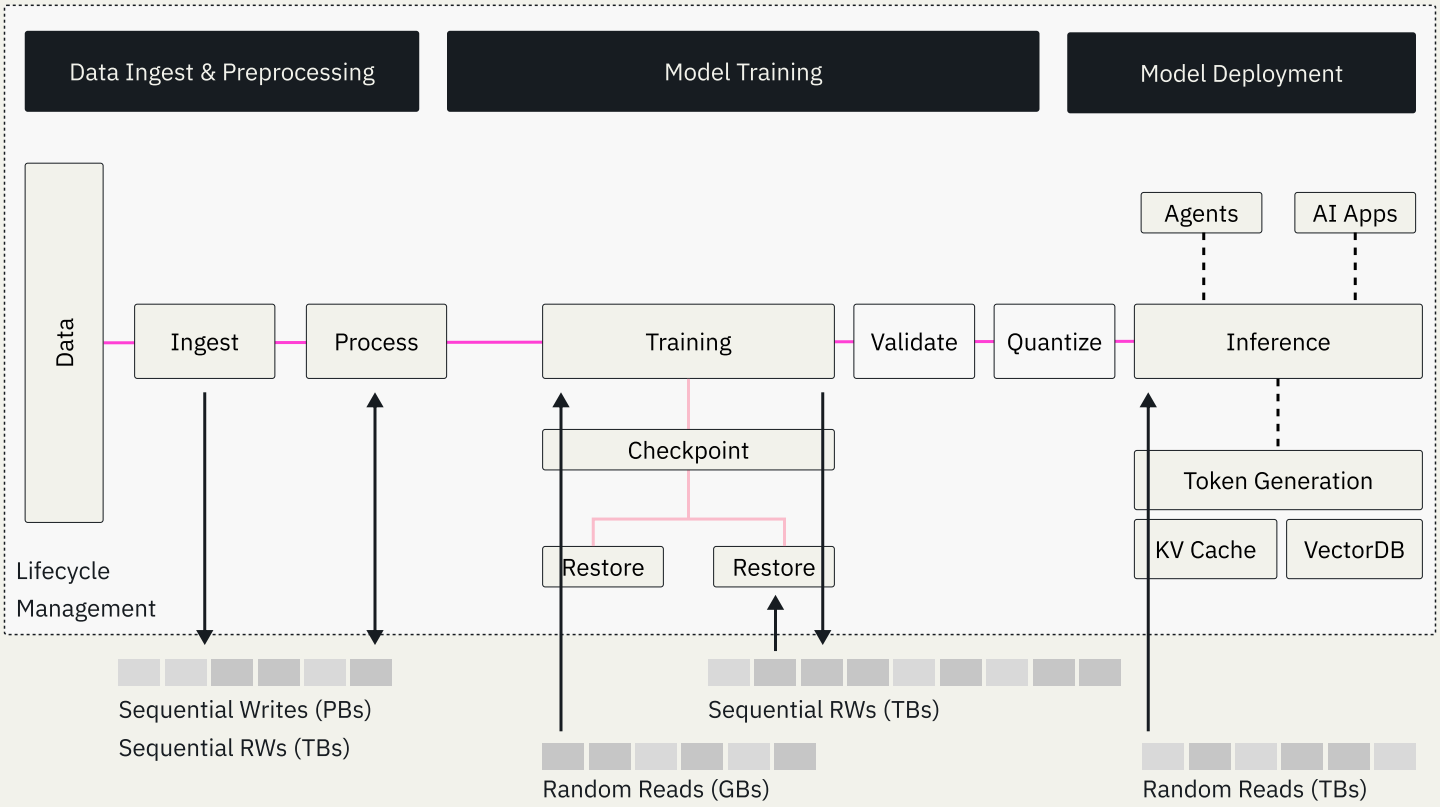



Figure 2: Impact of Storage on the AI Lifecycle

WEKA's approach is achieving demonstrable real-world benefits. Stability AI (the team behind Stable Diffusion), for example, revolutionized its infrastructure strategy to significantly enhance resource utilization. Using NeuralMesh, Stability AI was able to reduce data storage expenses by 95% per terabyte while simultaneously increasing GPU utilization and shortening model training times. The approach has the added benefit of propelling Stability AI towards its sustainability objectives.

Stability AI found that, on average, training epoch times were reduced by three weeks, with a faster storage solution that eliminated manual data management tasks. The company also achieved 93% GPU utilization by eliminating storage bottlenecks associated with small file handling and metadata lookups. As a result, its researchers spend less time manually loading and preprocessing data.

# Breaking the Memory Wall: WEKA's Augmented Memory Grid

To address the critical challenges of tokenomics and GPU memory limitations, WEKA has introduced its revolutionary Augmented Memory Grid, a groundbreaking capability that extends GPU memory to NeuralMesh, enabling the caching of prefixes or key-value (KV) pairs.

WEKA's Augmented Memory Grid offers large, persistent memory that integrates with the NVIDIA Triton Inference Server, enabling AI model builders to overcome traditional memory limitations.

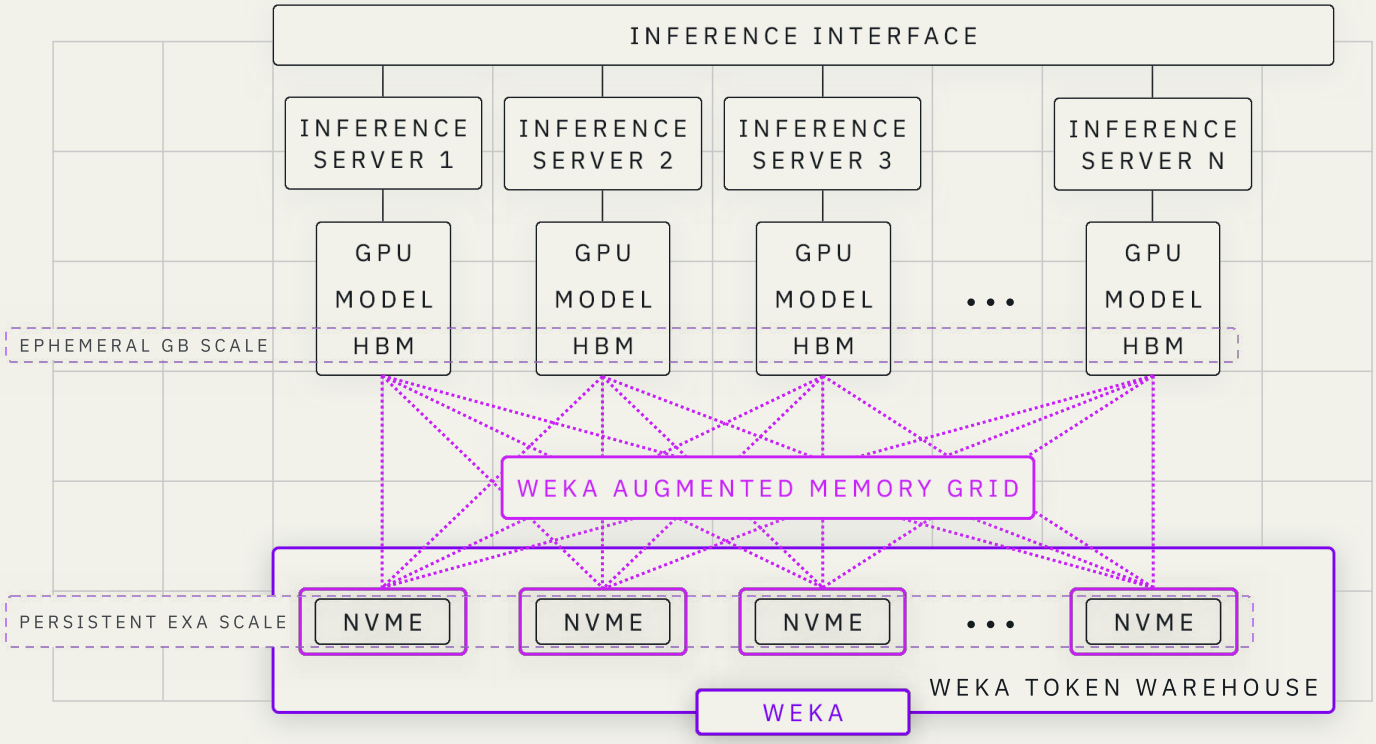



Figure 3: WEKA's Augmented Memory Grid

This innovation is uniquely architected to bring:

- **Petabytes of Persistent Storage for KV Cache:** For the first time, the memory for large AI model inferencing extends to petabytes of memory—approximately three orders of magnitude, or 1,000 times more than today's fixed DRAM increments of single terabytes.
- **Optimizations for Inference Infrastructure:** The addition of persistent memory eliminates the need to over-provision GPUs when memory is full, thereby mitigating the trade-offs between speed, accuracy, and cost.
- **Dynamic Resource Reallocations:** By offloading KV Cache data from high-bandwidth memory (HBM) DRAM, GPUs can focus on the most critical tasks, which improves performance across the inference system.

WEKA's testing with Augmented Memory Grid has demonstrated unprecedented results in tokenomics optimization:

- **41x reduction in time to first token (TTFT):** Prefill time dropped from 23.97 seconds to just 0.58 seconds for a 105,000-token input sequence
- **24% reduction in token throughput costs** for the entire inference system
- **Microsecond latencies** with multi-terabyte-per-second bandwidth
- **No compression or quantization is required**, maintaining full model accuracy

These results were achieved using:

- NVIDIA DGX H100
- 8-node WEKApod with PCIe Gen 5
- NVIDIA Quantum-2 QM9700 64-port NDR 400Gb/s InfiniBand switches

The approach enables enterprises to retain, retrieve, and reuse KV cache data with microsecond latency, dramatically improving the economics of AI inference operations while maintaining the highest levels of accuracy.

# Conclusion: Storage for Production AI at Scale

Modern AI pipelines are directly impacted by the storage system's ability to deliver high throughput, low latency, and high IOPS while supporting the demanding data operations of training and inference. As the industry evolves toward agentic AI and large-scale reasoning models, the economics of token processing have become an increasingly critical factor in determining the viability and scalability of AI deployments.

The enterprise shift from experimental AI to production inference has fundamentally changed storage requirements. While training workloads remains important for model development, today's enterprise AI agenda is dominated by inference workloads that must deliver consistent, real-time performance to support business-critical applications and customer-facing services. This operational reality demands storage architectures that excel at serving concurrent, latency-sensitive access patterns while maintaining the high-throughput capabilities required for training and data processing operations.

Traditional storage solutions, whether on-premises or in the cloud, often fail to meet the varying needs of each phase of the AI lifecycle. These legacy approaches are particularly ill-suited for the demands of distributed training and modern inference workloads, where storage performance has a direct impact on user experience, business operations, and revenue generation. The consequences of storage performance issues extend far beyond development timelines—they immediately affect production systems, economics, and customer satisfaction.

The emergence of the memory wall as a fundamental bottleneck in AI inference has created an urgent need for innovative solutions that can extend GPU memory capabilities while maintaining the microsecond-level performance required for efficient token processing. The ability to persist and reuse KV cache data represents a paradigm shift in inference efficiency, potentially reducing costs by orders of magnitude while dramatically improving user experience. WEKA solves many of these challenges with its NeuralMesh and revolutionary Augmented Memory Grid technology. Its advanced features, including high performance, scalability, efficient metadata handling, intelligent caching, fault tolerance, and memory extension capabilities, make it exceptionally well-suited for the modern AI lifecycle. The ability of WEKA's Augmented Memory Grid to extend GPU memory to petabyte scale while maintaining microsecond latencies represents a breakthrough in addressing the tokenomics challenges facing the industry. This breakthrough is particularly critical for enterprises seeking to deploy large-scale inference applications that can serve thousands of concurrent users with strict latency requirements.

For enterprises deploying AI at scale, NeuralMesh delivers the infrastructure foundation needed to support both experimental AI development and production inference workloads on a unified platform. This capability is essential for organizations seeking to maximize their AI infrastructure investments while ensuring that production applications can deliver the performance and reliability required for business success.

Organizations adopting NeuralMesh can expect improved efficiency, reduced latency, enhanced scalability, and dramatically improved tokenomics, all of which are critical for successful AI implementations in the era of agentic AI and large reasoning models. Most importantly, they can confidently deploy inference applications that meet the demanding performance requirements of modern enterprise users while maintaining the economic efficiency needed for sustainable AI operations at scale.


## By Steve McDowell, Chief Analyst and Founder NAND Research Company

Steve McDowell is a technologist with over 25 years of deep industry experience in a variety of strategy, engineering, and strategic marketing roles, all with the unifying theme of delivering data, storage and systems technologies into the enterprise market.