Technical White Paper

Designing and Building a Data Pipeline for Your AI Workflows

Deploy AI, Machine Learning, and Deep Learning Across Your Enterprise from Edge to Core to Cloud

Santosh Rao, NetApp
March 2018 | WP-7264

Executive Summary

Enterprises are eager to take advantage of artificial intelligence (AI) technologies such as deep learning (DL) to introduce new services and enhance insights from company data. As data science teams move past proof of concept to operationalize deep learning, they must focus on creating a complete data architecture that eliminates bottlenecks to facilitate faster model iteration.

Designing a data architecture involves thinking holistically about the data pipeline from data ingest and edge analytics to data prep and training in the core data center to archiving in the cloud. It is critical to understand performance requirements, datasets, and data services needed. However, you should also consider future extensibility and supportability as deep learning hardware and cloud approaches evolve over time.

This white paper discusses AI infrastructure challenges and how NetApp can help you build a data pipeline for your deep learning workflows today, while future-proofing investments in your AI infrastructure. Careful infrastructure planning can smooth the flow of data through your deep learning pipeline, lead to faster time to deployment, and maximize competitive differentiation.

NetApp®

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction: Is Your Infrastructure Ready to Support AI Workflows in Production?

Organizations across all industries are eager to take advantage of artificial intelligence (AI) technologies to introduce new services and gain new insights from company data. However, as data science teams move past proof of concept projects and begin to operationalize AI technology, they often experience issues with data management. For example, it can be a struggle to move or copy data across multiple data repositories. It has also been a challenge to meet production-quality service levels for performance and protection across large and dynamic datasets.

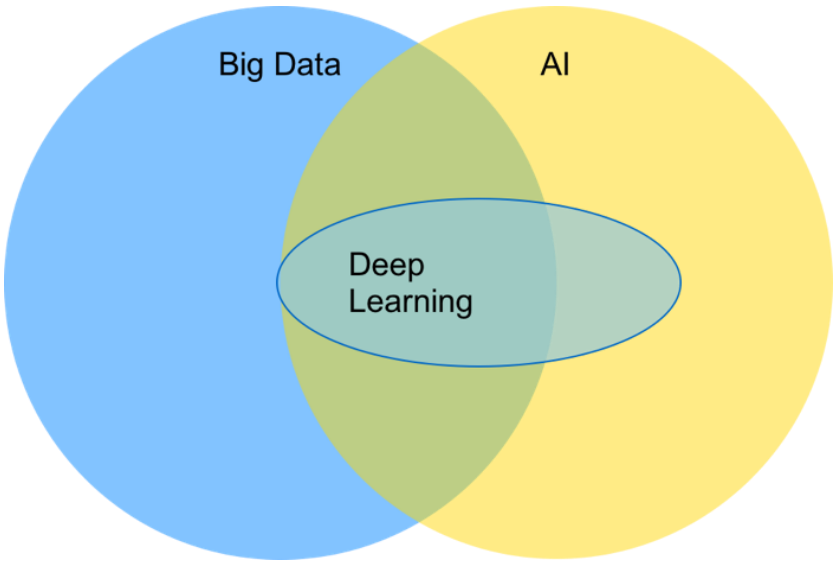
Part of the challenge is that the data flow necessary for successful AI isn't isolated to the data center. As enterprises of all types embrace Internet of Things (IoT) and AI technologies, they face data challenges from edge to core to cloud.

For example, many automotive companies have begun gathering data from a growing number of vehicles (the edge). This data is used to train the AI algorithms necessary for autonomous operation (the core). Because the datasets are growing exponentially and need to be stored for reuse, they need to be stored in a scalable, low-cost platform (the cloud). Today, automotive companies are quite literally driving IT technology to its limits. Global retailers face similar challenges when creating inference models based on data gathered from point-of-sale devices across hundreds of retail locations around the world.

Some people would have you believe that the AI data challenge is only about delivering performance. Performance is essential in the core of an AI pipeline. However, you need a data pipeline that encompasses the entire data flow, from ingest to archive, ensuring your operational success while delivering optimal performance, efficiency, and cost at every phase.

This white paper discusses AI infrastructure challenges and describes how NetApp can help you build a data pipeline that enables deep learning (DL). Because deep learning is the most demanding AI workflow in terms of computation and I/O, a data pipeline designed for deep learning also accommodates other AI and big data workflows. (See Figure 1.)

Figure 1) A data pipeline designed for deep learning can also accommodate other AI and big data workflows.

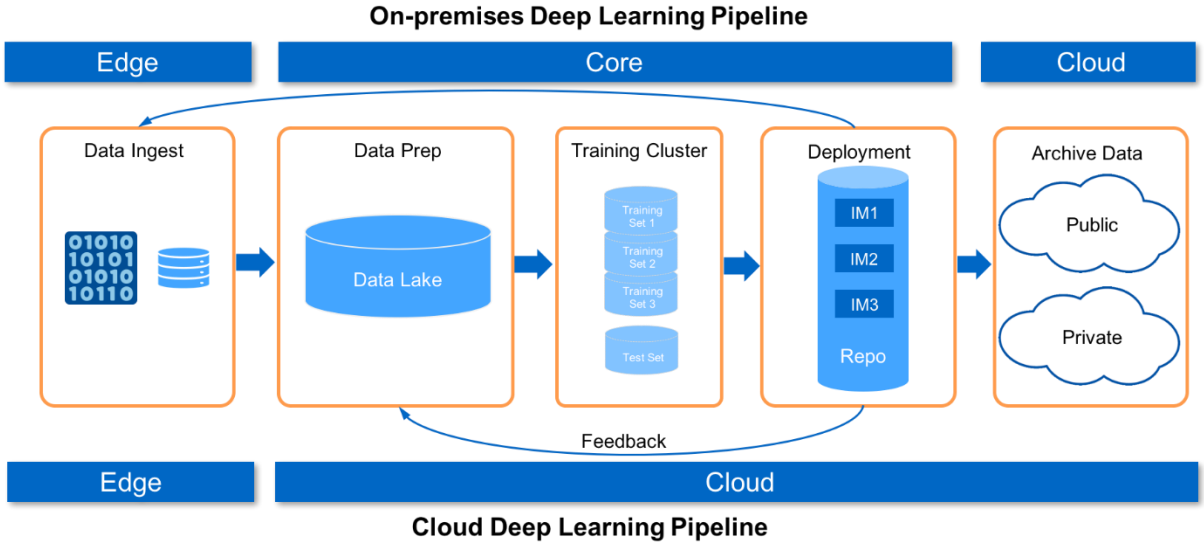


2 Data Flow in a Deep Learning Pipeline

When designing a data pipeline for AI or deep learning, you should begin by considering these steps, shown in Figure 2:

1. **Data ingest.** Ingestion usually occurs at the edge: for example, capturing data streaming from cars or point-of-sale devices. Depending on the use case, IT infrastructure might be needed at or near the ingestion point. For instance, a retailer might need a small footprint in each store, consolidating data from multiple devices.
2. **Data prep.** Preprocessing is necessary to normalize data before training. Preprocessing takes place in a data lake, possibly in the cloud in the form of an S3 tier or on the premises as a file store or object store.
3. **Training.** For the critical training phase of deep learning, data is typically copied from the data lake into the training cluster at regular intervals. Servers used in this phase often use graphics processing units (GPUs) or custom silicon to parallelize operations, creating a tremendous appetite for data. Raw I/O bandwidth is crucial.
4. **Deployment.** The resulting model is pushed out for testing and then moved to production. Depending on the use case, the model might be deployed back to edge operations. Real-world results of the model are monitored, and feedback in the form of new data flows back into the data lake, along with new data to iterate on the process.
5. **Archive.** Cold data from past iterations may be saved indefinitely. Many AI teams archive cold data to object storage in either a private or public cloud.

Figure 2) A deep learning pipeline can exist either on the premises or in the cloud.



Many companies have attempted to build out this type of data pipeline, either in the cloud or on the premises. This approach usually involves commodity hardware and a brute-force approach to data management. The cloud can become cost prohibitive. Moving large amounts of data out of the cloud quickly becomes expensive; after data is committed there, the rest of your pipeline will likely end up running in the cloud. In either case, bottlenecks inevitably arise as the projects transition to production use and the amount of data increases.

The biggest bottlenecks occur during the training phase, where massive I/O bandwidth with extreme I/O parallelism is needed to feed data to the deep learning training cluster for processing. Following the training phase, the resulting inference models are often stored in a DevOps-style repository, where they benefit from ultralow-latency access.

However, if data doesn't flow smoothly through the entire pipeline, beginning with ingest, your AI data pipeline will never achieve full productivity. You'll have to commit increasing amounts of staff time to manage the pipeline.

3 Accelerating I/O in a Deep Learning Pipeline

Whether you execute your AI workflow on the premises or in the cloud, operational bottlenecks can extend the time needed to complete each training cycle. This extra time reduces the productivity of your pipeline and uses valuable staff time.

This section describes solutions to address I/O bottlenecks from edge to core to cloud, including:

- Bottlenecks at the edge that slow down data ingest
- Bottlenecks on the premises
- Bottlenecks in the cloud

The three phases at the core of the pipeline in particular—data prep, training, and deployment—create unique I/O requirements that must be specifically addressed.

3.1 Eliminate Bottlenecks at the Edge

The amount of data generated by smart edge devices and a large number of ingestion points can overwhelm compute, storage, and networks at the edge. This data can create bottlenecks as it moves into your data center or the cloud.

By applying edge-level analytics, you can process and selectively pass on data during ingest. This approach requires infrastructure at the edge with high-performance, ultralow-latency storage. Many NetApp customers are taking a hierarchical approach, with infrastructure at the last mile and sensors at the edge acting as endpoints.

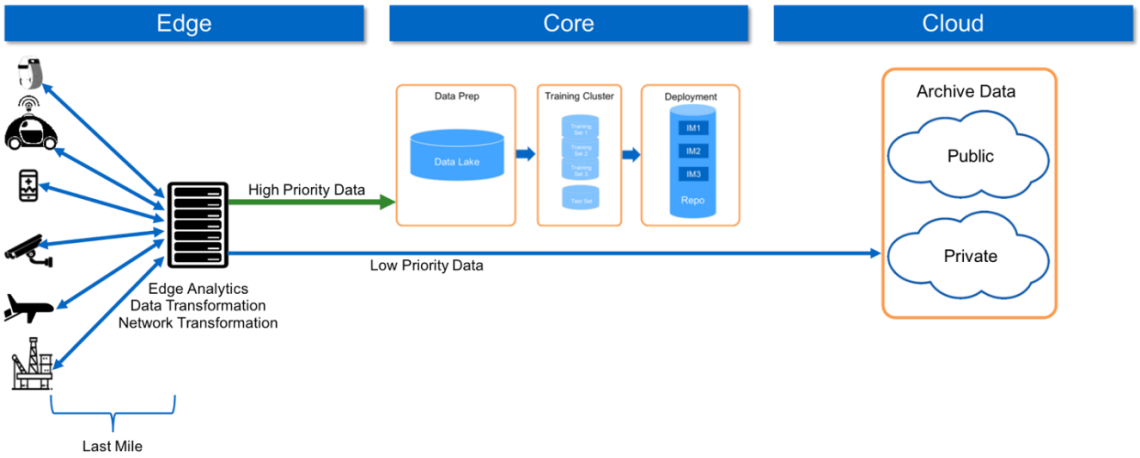
Sensors from manufacturing equipment feed data into infrastructure deployed in each plant to aggregate and analyze data, so it can be passed selectively up the chain. This approach also makes sense for autonomous vehicles (where each endpoint can generate up to 7TB of data per day), retail, and many other fields.

Tiered Data Management at the Edge

Using edge-level analytics, you can create different tiers of data service. With this approach, some data is prioritized—using either simple filtering or advanced analytics and AI—and passed efficiently into the AI/machine learning (ML)/DL pipeline. Other data is deprioritized and may be either discarded or managed with a different class of service.

Depending on requirements, each tier of data can be processed with different transformations to achieve the necessary levels of storage efficiency and security. For example, low-priority data might be compressed, deduplicated, encrypted, and stored in a cloud repository for compliance or in case it’s needed for later processing. (See Figure 3.)

Figure 3) Using edge analytics with data tiering, data from the edge can be separated into high-priority data destined for the core and low-priority data to be archived.



The ability to process analytics at the edge is a function of the compute power available. We're seeing compute and cloud vendors competing for the edge footprint with various strategies. For example, NVIDIA is bringing GPU power to the edge to enable AI for applications such as self-driving cars. One thing that all these solutions have in common from a data perspective is incorporation of commodity DAS lacking intelligent data management. There's an obvious need here for intelligent data storage.

Smart Data Movers

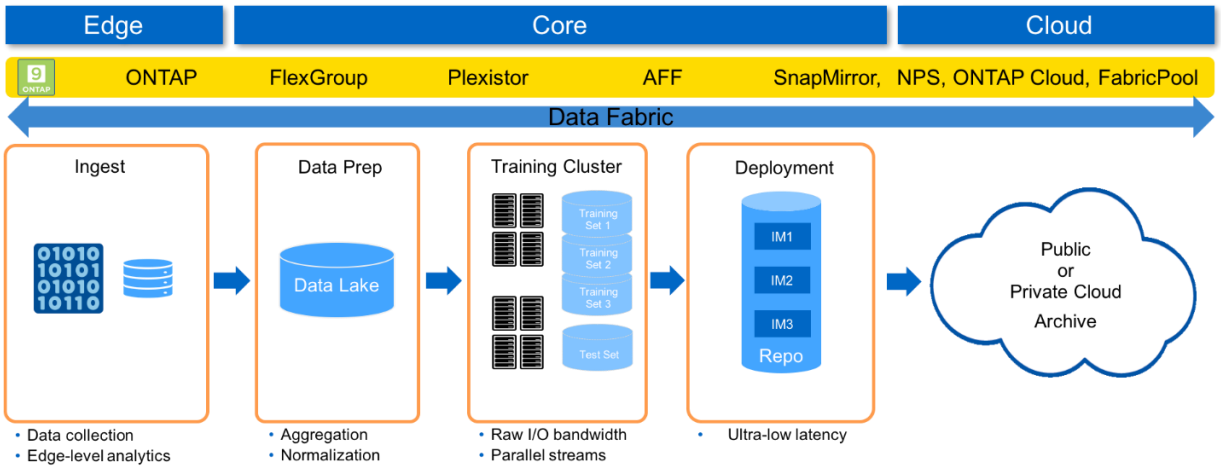
To deliver data at high bandwidth from the edge, one of the key elements is having a smart data mover. In the most common architecture today, data is moved using full data moves in the form of S3 puts. This approach has the disadvantage of moving data wholesale without applying any data transformation.

This crude method can be replaced with smart data movers that coalesce data, apply data transformations to reduce the data footprint, and apply network transformations to move only changed blocks. This approach can dramatically accelerate data movement and reduce bandwidth requirements.

3.2 Eliminate Bottlenecks on the Premises

If the core of your deep learning pipeline is on the premises, as in Figure 4, then you have direct control over data lake, training cluster, and inference model deployment.

Figure 4) A deep learning pipeline with the core of the pipeline on the premises.



Data Lake

As data flows in from the edge, it gets collected in a data lake. An improperly implemented data lake becomes a bottleneck as the amount of data grows. A data lake can take the form of a Hadoop deployment with the Hadoop file system (HDFS) or can be implemented using either an object store or a file store. HDFS is not optimized for performance and typically maintains three copies of each data object, slowing write performance and increasing cost.

Object stores were originally intended for cloud archiving, not performance, but in many cases they have become the de facto datastore for big data projects. As you saw earlier, for deep learning, object stores leave a lot to be desired where performance is concerned.

Turning to file stores, scale-out file systems such as Lustre and GPFS are designed for high-performance computing (HPC) batch processing; they don't deal well with small file workloads. Data flowing into the data lake from smart edge devices tends to be in the form of many small files, for which these systems are not optimized, so performance suffers.

NetApp® AFF, especially when used with NFS and ONTAP® FlexGroup volumes, overcomes the limitations of other data lake approaches. FlexGroup groups can deliver high performance for both bandwidth-oriented batch workloads and small file workloads. The other data lake solutions mentioned—HDFS, object storage, Lustre, GPFS, and other scale-out file storage—might do one or the other. However, they can't deliver good performance for both sequential and random I/O.

Training Cluster

The current state of the art for deep learning training clusters is a scale-out cluster with 32 to 64 servers and 4 to 8 GPUs per server. From an I/O standpoint, you have to keep all those GPUs 100% busy. That means delivering a parallel I/O stream to each CPU core. In turn, each CPU core has an affinity to a GPU. The CPU processes its stream, coalesces the I/O, and feeds the data to the GPU.

This process introduces I/O bottlenecks in the following ways:

- Data has to be streamed quickly and efficiently from the data lake into the training cluster.
- Up to 256 parallel I/O streams (32 to 64 servers, each with 4 to 8 GPUs) need to be kept fully loaded and staged to feed GPUs so they never have to wait for data.

The NetApp ONTAP software architecture uniquely satisfies both of these requirements. The data lake can be designed using hybrid flash nodes, which can stream data into the training cluster with extremely high bandwidth. All-flash storage nodes supporting the training cluster can deliver bandwidth up to 18GBps per two-controller HA pair and sub-500-microsecond latencies, providing the bandwidth to support many I/O streams in parallel. NetApp also offers a technology roadmap that allows you to continue to grow the I/O performance of your AI/ML/DL pipeline as your needs grow.

Deployment

After training completes, the resulting inference models are put into a DevOps-style repository and subjected to inference testing and hypothesis validation. This stage is where it is important to deploy storage systems that support extreme low latency.

With NetApp, a single storage architecture addresses all the performance needs for the core of your deep learning pipeline. Although this approach has immediate advantages, the current state of the art for most customers is to operate separate clusters for each stage of the pipeline. Big data pipelines might be in place with a data lake already deployed. You might want to implement just the new elements needed for deep learning as a separate project and copy data from phase to phase. As data continues to grow, however, you'll need to further unify the pipeline. AFF also makes this unification possible.

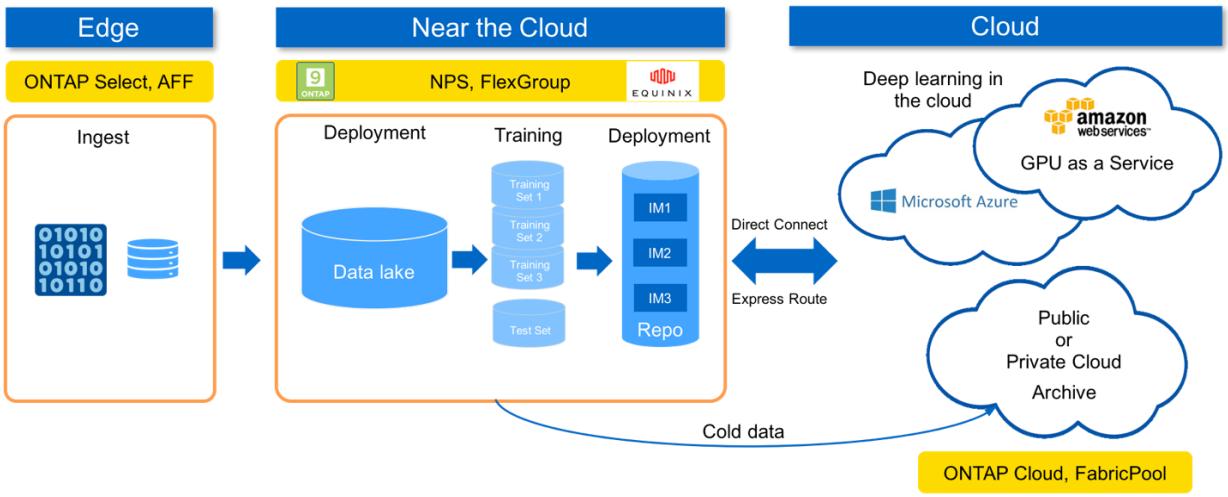
3.3 Eliminate Bottlenecks in the Cloud

You might decide to deploy deep learning in the cloud for agility and ease of consumption. However, the same potential bottlenecks apply when you run your deep learning pipeline in the cloud:

- Can your data lake deliver the necessary performance for data ingest? Can it stream data into the training cluster?
- Can your cloud provider deliver the I/O parallelism required for the training cluster?
- How can you deliver the ultralow latency required for finished inference models?
- What if you need to make sure of data sovereignty for sensitive data?

NetApp Private Storage (NPS) lets you store your data near the cloud so that you can use public cloud compute capabilities and other services, while maintaining full control over your data. (See Figure 5.) NPS brings the same architecture and the same performance described in the previous section to the public cloud. Data sovereignty issues are eliminated, and your data never gets locked into the cloud.

Figure 5) By positioning data near the cloud, you can take advantage of cloud compute while delivering greater data acceleration and maintaining greater control.



If your data lake absolutely must live in the cloud, the NetApp Data Fabric enables you to store and seamlessly manage NFS data in either the Azure or AWS cloud service.

4 File System and Data Architecture for a Deep Learning Pipeline

In an AI pipeline, there are different I/O characteristics for data flowing in from the edge compared to data flowing from the data lake into the training cluster. This section discusses the specific set of choices that you must consider to smooth the flow of data through the data pipeline and into the training cluster.

Think of the GPUs in your training cluster as a high-performance car. A good data pipeline is like the difference between taking that car out on a racetrack and taking it out on the freeway at rush hour. To obtain maximal results from your deployment for AI, including ML and DL, the data pipeline is perhaps the single most important consideration, yet it is often overlooked. The optimal data architecture takes into account I/O needs across the edge, data lake, and training cluster.

Object storage is not designed to deliver the level of performance that your data pipeline requires. Object stores were originally intended for cloud archiving rather than performance, but in many cases, they have become the de facto datastore for big data projects. For deep learning in particular, object stores leave a lot to be desired where performance is concerned.

The file system and data architecture you choose should account for all the factors that are important to your AI environment. File-based storage remains a superior choice, but there are many factors to consider, as noted in Table 1.

Table 1) Key questions and considerations.

Key Questions	Key Considerations
Which file systems should be considered?	• A scale-out file system such as Lustre or GPFS • HDFS, a commonly used big data file system • NFS, the most widely deployed shared file system for technical applications for the last 30 years
Can the file system accommodate and federate both structured and unstructured data from a variety of data sources without sacrificing performance?	• Log and sensor data • Databases, including RDBMS and NoSQL • Random I/O for many types of databases: table scans, document and collection reads in NoSQL, columnar reads in columnar databases, and key-value random reads in key-value databases • Sequential I/O for in-memory databases and in-memory engines such as Spark • Email logs • Home directories • Other sources
Does it provide performance for small, random I/O compared to sequential I/O?	• Some data sources generate random I/O, while others are sequential • File system must be able to balance performance between both types of I/O
What are the performance and capabilities of the data movers?	• Greatest performance • Most efficient data movement
Can it help you automate the data lifecycle?	• Intelligent filtering to determine what data goes to the core compared to archive tiers • Real-time performance for filtering decisions
Does it support the latest storage and memory media, enabling nondisruptive advances in performance and latency?	• Storage tiers that can meet the price-performance for your datastore, including storage-class memory (SCM), Nonvolatile Memory Express (NVMe), flash, hybrid flash, disk, and cloud • Nondisruptive data movement across tiers • Scale-out designs for adding incremental performance

4.1 Data Flow into the Training Cluster

In addition to the considerations listed in Table 1, there are some nuances in the way that data flows into the training cluster that are important to understand. These factors affect:

- Where I/O is coalesced
- Requirements for a single namespace
- Metadata scaling

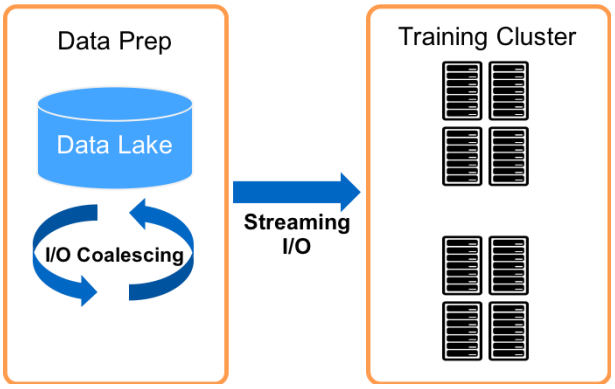
I/O Coalescing

Data curation is a function of the data source. I/O coalescing can happen in two different locations:

- In the data lake as a part of data curation and transformation, resulting in streaming I/O into the training cluster
- In the training cluster itself, which results in random I/O from the data lake

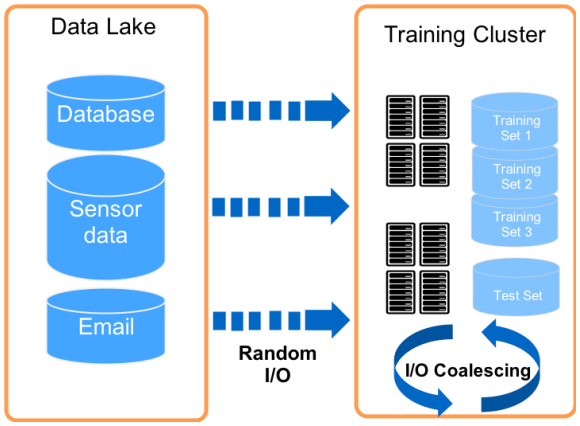
When you're dealing with an unstructured data lake, that's a file system almost by definition. It has the ability to curate the data and lay it out as a set of coalesced file streams. These file streams can be nicely aligned with the training cluster, allowing data to stream directly into cluster CPUs to preload and feed GPUs. (See Figure 6.)

Figure 6) Unstructured data allows data to be coalesced in the data lake and streamed into the training cluster.



In contrast, with data sources such as databases, sensor logs, file logs, emails, and so on, it might be impossible to have nice curated reads that let you stream data into the cluster. In these cases, data is accessed using random reads, and I/O coalescing happens in the training cluster itself. (See Figure 7.)

Figure 7) Structured data is read using small random I/Os and coalesced in the training cluster.



Depending on the types of data sources you have, your data architecture might need to be able to deliver both large sequential reads and small random reads into the training cluster.

Single Namespace

AI datasets have the potential to grow to massive size, leading to tremendous data sprawl. Accommodating this growth requires a scale-out file system with a single namespace that can scale performance linearly to a single client node or multiple client nodes accessing the same data in parallel. An architecture that can continue to scale as you add compute and capacity is critical.

There can be different types of client access to this single namespace, each with implications for performance. Certain training models are considered “async”: the dataset is partitioned statically across training cluster nodes with single-node access to regions of the namespace, resulting in a “single client active” scenario.

Other training models run synchronously. The training model and its dataset have tight coupling, and the dataset is shared across all cluster nodes with simultaneous access. This “multiclient active” scenario is the most demanding case from a performance standpoint.

There are other uses cases where a multilayered neural network trains the layers of the network on different nodes. The nodes serve as a model pipeline where the model progresses from one node to the next. This approach results in the entire dataset being read repeatedly, one node at a time, in a “sweeping hand” style of access.

As you evaluate file systems capable of addressing these usage patterns, you’ll find that NFS has been applied to a diverse range of workloads. These workloads range from its roots of HPC and home directories to databases such as Oracle and SQL running on NAS storage to SAP and more recently virtualization and big data. This long history of using NFS across a variety of workloads enables it to handle both the random and sequential I/O. This I/O can be generated by diverse access patterns to the namespace, especially when combined with the benefits of all-flash storage in a linear scale-out cluster.

As a relatively new file system, HDFS has had limited exposure to diverse data workloads and performance characteristics. Big data vendors have been undertaking significant (and proprietary) rewrites to deal with the performance needs in the transition from MapReduce to Spark. AI introduces another wrinkle in the HDFS story.

Relying on a big data–specific file system such as HDFS can mean more data copies and silos as you find yourself doing yet another data copy from HDFS into a high-performance scale-out file system for AI.

Metadata Performance

The access patterns discussed earlier also have implications for metadata performance. Each node in the training cluster might query metadata independently, so metadata access performance must scale linearly with the growth of the file system. Metadata access with file systems such as Lustre and GPFS can become a bottleneck due to reliance on separate metadata servers and storage.

4.2 Other Performance Factors

There are a variety of other factors that you’ll want to take into account when selecting a file system for your AI data pipeline that affect both performance and usability. These include:

- Ease of management
- Quality of service (QoS)
- Cloning capabilities
- Ecosystem of client-side caching solutions
- Ability to perform in-place AI/DL with a unified file system across the data lake and AI/DL tiers
- Best-in-class media support
- Future-proofing

Ease of Management

As you evaluate file systems, it’s important to ask management-related questions. Can the file system scale autonomously and automatically without management intervention? How much time and technical expertise does the file system take to manage? How easy is it to find people with the necessary expertise?

Scale-out file systems such as Lustre and GPFS can be challenging to configure, maintain, monitor, and manage. By comparison, NFS is easy to manage, and NFS expertise is widespread.

Quality of Service

QoS can also be an important element of your data architecture. You might be building multitenant training clusters with price tags running into the millions of dollars. QoS plays a key role in your ability to deliver multitenancy, enabling multiple activities to share the same resources.

- Does the file system offer QoS?
- Is QoS integrated end to end?
- Can you apply limits and maximums on performance consumption across storage, networks, and compute to partition service levels for different training models?

Cloning Capabilities

Part of the multitenancy requirement is to satisfy different job functions in your organization. You might have a set of training models in various stages of development resulting in different use cases:

- Early training
- Model validation
- Predeployment
- Production deployment

The ability to clone datasets and assign different QoS settings to each clone allows you to provide different performance service-level agreements for different use cases. Space-efficient cloning is therefore a must-have for a multitenant cluster.

Client-Side Caching

Use of a client-side cache helps further accelerate performance by providing a data buffer that enables uninterrupted data flow as the training dataset is accessed from training cluster nodes. A file system that supports an ecosystem of client caching products (whether open source or commercial) can provide substantial advantages.

A variety of open-source and commercial options exist for NFS-based storage. Few if any client-side caching products currently exist for Lustre, GPFS, or HDFS. Almost none are open source and widely available.

In-Place AI/DL with a Unified File System

There will be situations in which you want to use the same data to serve both big data analytics workloads and AI/ML/DL workloads. For AI that is applied postprocess—such as for surveillance, fraud detection, and so on—the right file system makes it possible to accomplish both workloads without the need for data copies. The dataset resides in a single location. Both in-place analytics and in-place AI/ML/DL compute processing are applied (possibly with client-side caching, as just discussed), without copying data into dedicated file systems for your data lake and training cluster.

However, if real-time performance is a key requirement or a key competitive differentiator, you will likely continue to need a dedicated data copy for the training cluster.

Support for State-of-the-Art Media and Memory

Finally, pick a file system that can support the latest advancements in media and memory so that your data pipeline's performance can continue to evolve in lockstep with the technology roadmap. Is the file system optimized for flash today? Is it seamlessly extensible to support new technologies, and are vendors actively innovating in areas such as, NVMe over Fabrics (NVMe-oF), NVDIMM, and 3D XPoint?

Flash today is capable of latencies around 500 microseconds. NVMe-oF takes that down to 200 microseconds. NVDIMM, 3D XPoint, and persistent memory are poised to take latencies to less than 100 microseconds, less than 10 microseconds, and eventually nanoseconds. Your data pipeline vendor needs to be making sustained investments to keep pace with this evolution across server-based and shared-storage solutions.

Future-Proofing Your Data Architecture and File System Choices

The whole AI field is evolving very quickly, but it can be impractical or impossible to rebuild from scratch every six months to a year. As a final consideration, you should try to make technology choices that are as future-proof as possible. It is important to seamlessly and nondisruptively evolve different layers of technology such as file system, interconnect, deployment location, media, and memory type in a chosen infrastructure. This capability provides a long-term return on investment and the ability to absorb technology evolutions as they occur.

Your choice of file system today will likely depend on your team's existing comfort levels, skillset, and prior expertise. You will likely want to factor in past deployment experience, existing deployments, and existing infrastructure.

As an example, if you're comfortable with and looking to deploy on FC or InfiniBand, you might go with a SAN architecture and Lustre or GPFS. Over time, you might decide the 100GbE or 400GbE roadmap with NFS is better for your needs. A well-planned data architecture is able to accommodate and future-proof the solution, allowing you to seamlessly switch your file system without replacing infrastructure.

Similarly, you might choose NFS today but decide you need a SAN-, NVMe-, or NVMe-oF-based file system or a persistent memory–based data layout in the future. A future-proofed architecture allows you to evolve datastore technologies without needing to replace your entire deployed infrastructure.

The criteria outlined in this chapter should give you a good foundation on which to select a file system and data architecture best suited to your AI/ML/DL needs. We believe that the combination of NFS running on NetApp AFF storage is the best choice based on our ability to address these needs and evolve in place to accommodate the latest technologies.

5 NetApp Technologies and the Deep Learning Pipeline

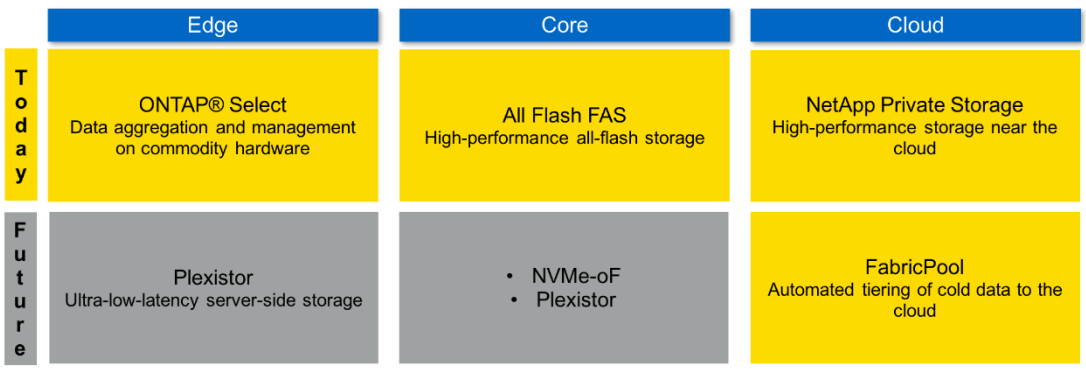
The NetApp Data Fabric includes data management technologies to satisfy the needs of the entire deep learning pipeline. (See Figure 8.) Cloud providers by themselves don't encompass the edge and might struggle with I/O performance. Other storage vendors attempt to solve the bandwidth problems during training, but can't deliver ultralow latency, and they lack the technology necessary to cover the entire workflow. This situation is where the NetApp Data Fabric offers distinct advantages.

At the edge, NetApp offers ONTAP Select, which runs on commodity hardware to enable data aggregation and advanced data management. Our upcoming Plexistor technology will facilitate ingest, especially in situations where the rate of ingest is extremely high.

To address storage needs for both the data lake and training cluster, NetApp AFF storage delivers both high performance and high capacity while reducing the need for time-consuming data copies. NetApp is working to deliver NVMe-oF and Plexistor to further extend AFF capabilities. NPS delivers many of the same benefits for deep learning pipelines in the cloud.

And, for archiving cold data, FabricPool migrates data to object storage automatically based on defined policies.

Figure 8) NetApp technologies for the Data Fabric.



6 Future-Proof Your Deep Learning Pipeline

The size of deep learning datasets and I/O requirements of your deep learning pipeline will almost certainly continue to grow as you increase the number of servers and CPUs, GPUs, and purpose-built AI silicon continue to grow in power. The NetApp roadmap incorporates a number of elements that will enable you to scale I/O to keep pace. These include:

- **NVMe-oF.** Incorporating NVMe-oF as part of the AFF architecture allows NetApp to drive latencies an order of magnitude lower.
- **Plexistor.** In June 2017, NetApp acquired Plexistor, giving us a server-side storage technology that drives down latencies even further, extending the NetApp Data Fabric into the server. Plexistor can be deployed at the edge, in the core, and in the cloud, accelerating data ingest, edge analytics, and training.

In addition, if the core of your AI pipeline is on the premises today, you might want to look at strategies that facilitate the evolution of core hardware. You should also plan for the possibility that you might need to transition from an edge-to-core-to-cloud strategy to an edge-to-cloud strategy in the future.

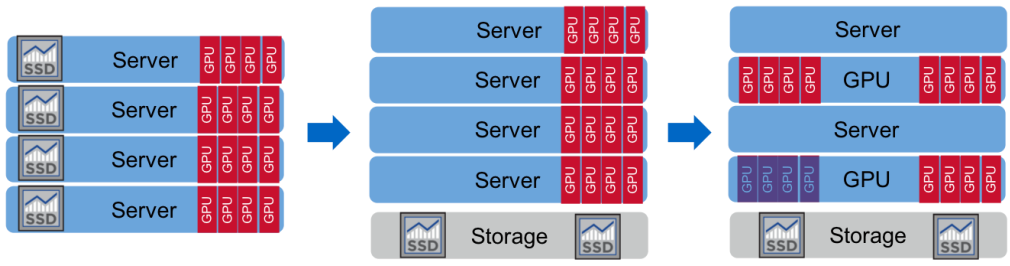
6.1 Plan for Hardware Evolution in the Core

One advantage of the cloud is that you can consume a deep learning service without having to understand the intricacies of the hardware stack. However, you pay a price for that convenience in terms of loss of control. When you consider the core of the on-premises deep learning pipeline, one of the key trends is the continued evolution of the hardware required.

There's an intense battle over who will be the hardware vendor of choice for deep learning. While NVIDIA has a clear lead at the moment, there are many emerging technologies. Each cloud vendor is building custom hardware. Google's tensor processing unit (TPU) is one example. Many startups are also building custom AI hardware.

One trend to watch for is the ability to separate the server infrastructure from the GPU infrastructure, allowing the two to evolve independently. A solution that abstracts GPU hardware from server and storage hardware (see Figure 9) can evolve more easily (and at lower cost) to take advantage of new developments.

Figure 9) The core of your AI/ML/DL pipeline will continue to evolve.



NetApp offers converged infrastructure with a variety of server partners, including Cisco (FlexPod®), Fujitsu (NFLEX™, a converged infrastructure from NetApp and Fujitsu), and other vendors. This fact means you can easily take advantage of NetApp storage in conjunction with a variety of server platforms when building out your deep learning cluster.

Conclusion: Take Control of Your Data Pipeline and Your AI Future

The guidelines in this white paper are intended to help you plan your data pipeline and include:

- Choose the best file system and data architecture to meet your needs today while keeping an eye on the future.
- Accelerate the flow of data through your pipeline, whether it's on the premises or in the cloud.
- Implement intelligent data management at the edge to better cope with data growth.
- Move data more intelligently and efficiently from the edge to the core or the cloud.
- Be prepared to transition to an edge-to-cloud model if that becomes necessary.
- Create a more agile core hardware architecture that can evolve quickly.

By taking these actions, you can eliminate bottlenecks and achieve greater throughput while also future-proofing the investments you make in your AI infrastructure.

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.