A Scalable AI Data Pipeline for Storing and Processing Ingested Data

Sanhita Sarkar, Ph.D
Global Director,
Analytics Software Development

August 8, 2019
## Topics Covered

<table>
<thead>
<tr>
<th></th>
<th>An AI Data Pipeline - Stages and Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Disaggregated Architectures for AI Implementations</td>
</tr>
<tr>
<td>3</td>
<td>Training and Inference Performance on a Disaggregated Architecture</td>
</tr>
<tr>
<td>4</td>
<td>Data Ingestion Performance with Apache Kafka© to an NVMe™ All-Flash Array</td>
</tr>
<tr>
<td>5</td>
<td>A Real-World Use Case</td>
</tr>
<tr>
<td>6</td>
<td>Summary and Best Practices</td>
</tr>
</tbody>
</table>
AI Pipeline is Data Intensive

Data Sources
- IoT and Sensors
- Business Processes
- Legacy

Ingest
- Transient
- Centralized
- High Performance

Data Preparation
- ETL - Data Cleansing, Pre-processing

Model Training
- Deep Learning Frameworks
- Trained Models

Model Serving
- Deploy Trained Models
- Inference

Data
- Speed and Scale
- Hours and Days
- Days and Weeks
- Seconds to Results
AI Pipeline has Varying Characteristics and Performance Requirements

Data Sources
- IoT and Sensors
- Business Processes
- Legacy

Ingest
- Transient
- Centralized
- High Performance

Data Preparation
- ETL - Data cleaning, pre-processing

Model Training
- Deep Learning Frameworks
- Trained Models

Model Serving
- Deploy Trained Models
- Inference

Data Variety
- Sensors

Data Volume
- Business Processes

Data Velocity
- High Performance

Data Disparity
- Legacy

Data Hours and Days
- Speed and Scale
- Hours and Days
- Days and Weeks
- Seconds to Results

Data Quality
- Throughput
- Response Time
- Data Access Latency

Model Quality
- Model Quality

Flash Memory Summit 2019, Santa Clara, CA
© 2019 Western Digital Corporation or its affiliates. All rights reserved.
AI Pipeline has Varying Infrastructure Requirements

**Data Sources**
- IoT and Sensors
- Business Processes
- Legacy

**Ingest**
- Transient
  - Throughput-oriented, Random I/O Performance and Capacity Tier (NVMe™/Hybrid)
- Centralized
  - Throughput-oriented, Globally Accessible Capacity Tier (Object Storage)
- High Performance
  - Throughput-oriented, High Performance Tier (NVMe™)

**Data Preparation**
- ETL - Data Cleansing, Pre-processing
  - Throughput-oriented, Random I/O Performance and Capacity Tier (NVMe™/Hybrid)
  - Throughput-oriented, Globally Accessible Capacity Tier (Object Storage)
  - Throughput-oriented, High Performance Tier (NVMe™)

**Model Training**
- Deep Learning Frameworks
  - Low Latency, Throughput-oriented, Scalable, Random I/O Performance Tier (NVMe™)

**Model Serving**
- Deploy Trained Models
  - Scalable, Large/Sequential I/O, Capacity Archival Tier (Object Storage)

**Speed and Scale**
- Throughput-oriented, Random I/O Performance and Capacity Tier (NVMe™/Hybrid)
- Throughput-oriented, Globally Accessible Capacity Tier (Object Storage)
- Throughput-oriented, High Performance Tier (NVMe™)

**Hours and Days**
- Days and Weeks
- Seconds to Results

Flash Memory Summit 2019, Santa Clara, CA
© 2019 Western Digital Corporation or its affiliates. All rights reserved.
Topics Covered

1. An AI Data Pipeline - Stages and Requirements
2. Disaggregated Architectures for AI Implementations
3. Training and Inference Performance on a Disaggregated Architecture
4. Data Ingestion Performance with Apache Kafka© to an NVMe™ All-Flash Array
5. A Real-World Use Case
6. Summary and Best Practices
Aggregated vs. Disaggregated Architecture for AI

Model Training

- Ingest
- Object storage system
- Data Preparation
- Training data
- GPU server(s) collocated with Flash
- Training data
- Trained models

Data Network

Model Serving

- Clients
- Inference
- Clients

Centralized Data Repository

- Ingest
- Object storage system
- Data Preparation
- Training data
- GPU server(s) collocated with Flash
- Training data
- Trained models

Model Training

- Ingest
- Object storage system
- Data Preparation
- Training data
- Shared pool of NVMe™ Flash Storage
- Trained models

Data Network

Model Serving

- Clients
- Inference
- Clients

- Model training is limited to the flash storage capacity integrated in the GPU servers and this incurs multiple data transfers from the object storage, once data grows over capacity on the servers.
- Incurs delays in model serving and inference.

- Model training can scale independently on a disaggregated pool of GPUs, shared flash and object storage, with no subsequent data transfers.
- Inference by the model serving client is faster due to immediate access to trained models on the shared flash storage.
Option 1: Disaggregated Architecture of an AI Data Pipeline

Centralized, High Capacity Landing Zone

- Raw S3 Object
- Raw S3 Object
- Raw S3 Object

Object Storage System

- Model v₁
- Model v₂
- Model vn

Data Preparation

- Batched

GPU Cluster

- Produce & Publish CNN/RNN Models
- Train and Validate Model

AI framework (TensorFlow, PaddlePaddle, PyTorch, Caffe, etc)

High Performance Rack-scale Flash Storage

- Training Data
- Validation Data

Model Training

- Model vn+1
- Model vn+2
- Model v2n

Model Serving

AI Model Serving

- Deploy
- Inference

Ingest

Clients

Clients

Clients
Implemented Workflow: Option 1

ActiveScale™ (Object Storage System)

- ActiveScale bucket
  - Gets notifications of newly ingested S3 objects

Data Pipeline Service / Kafka Producer

- Kafka Topic
  - Publishes the system metadata of the new S3 objects to a Kafka topic

AI Model Serving

- TensorFlow/PaddlePaddle/PyTorch
- GPU Cluster
- Models
- AI Model Training
- Gets the S3 objects from ActiveScale
- Subscribes to the Kafka topic to get the S3 locations of the newly ingested objects
- Uses AI/Deep Learning algorithms for analyzing the objects, and generating labels, and custom metadata
- Stores the labels, paths and custom metadata of the S3 objects in Elasticsearch

Data Preparation

- Elasticsearch©
- Gets notifications of newly ingested S3 objects
- Publishes the system metadata of the new S3 objects to a Kafka topic
- Kafka Topic

Key-Value Store

- Elasticsearch©
- Inference Score
  - Score < threshold
  - Notifies S3 locations of failed objects to be used for re-training the model
  - Data Preparation
  - AI Model Serving
  - AI Model Training

AI Model Training

- TensorFlow/PaddlePaddle/PyTorch
- GPU Cluster
- Training Data
- Gets AI model from N5800

IntelliFlash™ N5800

- Models
- Training Data

Search and Visualize

- Custom metadata of image/text/video S3 objects
- Kibana™

Dashboard

- Kibana™
- Elasticsearch©
Option 2: Disaggregated Architecture of an AI Data Pipeline

*Model Training*
- AI framework (TensorFlow, PaddlePaddle, PyTorch, Caffe, etc)

*Model Serving*
- AI Model Serving

**High Performance Rack-scale Flash Storage**
- Ingest
- Raw Data
  - Training Data
  - Validation Data

**GPU Cluster**
- Produce & Publish CNN/RNN Models
- Train and Validate Model

**AI Model Serving**
- Deploy
  - Inference

**High Capacity Data Archive**
- Object Storage System
  - Archived Training Data
  - Archived Raw Data
Implemented Workflow: Option 2

**Data Preparation**

- Raw Data
- Apache Kafka©

**IntelliFlash N5800**

- Training Data
- Models
- Key-Value Storage

**AI Model Training**

- TensorFlow/ PaddlePaddle/ PyTorch
- GPU Cluster

**S3™ Cloud Connector**

- Compresses and moves old files and models

**AI Model Serving**

- TensorFlow/ PaddlePaddle/ PyTorch

**ActiveScale (Object Storage System)**

- ActiveScale bucket
- ActiveScale bucket

**Dashboard**

- Kibana™

**Key-Value Store**

- Elasticsearch©

**Search and Visualize**

- Custom metadata of image/text/video files

**Inference**

- Score < threshold
- Score > threshold

**Notifies locations of failed data files to be used for re-training the model**

**Gets the input data and models from IntelliFlash N5800**

**Uses AI/Deep Learning algorithms for analyzing the input data files, and generating labels, and custom metadata**

**Stores the labels, paths and custom metadata of the files in Elasticsearch**

**Compresses and moves old files and models**

**Flash Memory Summit 2019, Santa Clara, CA © 2019 Western Digital Corporation or its affiliates. All rights reserved.**
### Topics Covered

1. An AI Data Pipeline - Stages and Requirements
2. Disaggregated Architectures for AI Implementations
3. Training and Inference Performance on a Disaggregated Architecture
4. Data Ingestion Performance with Apache Kafka© to an NVMe™ All-Flash Array
5. A Real-World Use Case
6. Summary and Best Practices
On a disaggregated architecture comprising an NVMe all-flash array, and

- a single 8-GPU server, the training performance with most AI models scales almost linearly up to 8 GPUs, except for AlexNet and LeNet, where training performance scales linearly up to 2 GPUs.

- multiple GPU servers, the training performance scales linearly with the number of servers, irrespective of the choice of AI models.
On a disaggregated architecture comprising a single 8-GPU server and an NVMe all-flash array –

- the average I/O throughput during training using the ResNet-50 model (compute intensive) is ~800 MB/s, the GPU utilization being 97-100% (size of image data is 164 GB, each image being ~100 KB)

- the average I/O throughput during training using the LeNet model (I/O intensive) is ~2.5 GB/s, the GPU utilization being 17-20%. So the LeNet model yields ~3x the I/O throughput, compared to ResNet-50.
The inference throughput is measured as the aggregated images/sec inference results using ImageNet datasets across multiple GPU containers.

On a disaggregated architecture comprising an NVMe all-flash array, and

- a single 8-GPU server, results show that the inference image processing rates are between ~3x to ~3.5x the training rates of the corresponding TensorFlow models.

- multiple GPU servers, users have the flexibility to run mixed AI workloads for training and inference, by dedicating one or two GPUs to inference for every 8 GPUs, rest being allocated to training.
Example Configurations: GPU servers and an IntelliFlash N5800 Array

- An example allocation strategy of IntelliFlash N-series arrays is considered for executing AI workloads –
  - 30% of the I/O bandwidth for model training, and the remaining 70% for various phases like data preparation, inference, and other activities

- **Considering the above allocation strategy**, example configurations are derived using the I/O throughput achieved on a disaggregated architecture comprising a single IntelliFlash N5800 array and an 8-GPU server, while using ResNet-50 and LeNet models for training -
  - A single IntelliFlash N5800 array can scale up to nine 8-GPU servers running ResNet-50 model for the training phase, with 100% utilization of GPUs.
  - With LeNet model, a single IntelliFlash N5800 array can optimally scale up to three 8-GPU servers for the training phase.

<table>
<thead>
<tr>
<th></th>
<th>IntelliFlash N5800</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Arrays</td>
<td>1</td>
</tr>
<tr>
<td>Mixed I/O Throughput GB/s (80% reads, 20% writes)</td>
<td>23.5</td>
</tr>
<tr>
<td>Number of 8-GPU servers (using compute-intensive ResNet-50 model for training)</td>
<td>9</td>
</tr>
<tr>
<td>Number of 8-GPU servers (using I/O-intensive LeNet model for training)</td>
<td>3</td>
</tr>
</tbody>
</table>
## Topics Covered

<table>
<thead>
<tr>
<th></th>
<th>An AI Data Pipeline - Stages and Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Disaggregated Architectures for AI Implementations</td>
</tr>
<tr>
<td>3</td>
<td>Training and Inference Performance on a Disaggregated Architecture</td>
</tr>
<tr>
<td>4</td>
<td>Data Ingestion Performance with Apache Kafka© to an NVMe™ All-Flash Array</td>
</tr>
<tr>
<td>5</td>
<td>A Real-World Use Case</td>
</tr>
<tr>
<td>6</td>
<td>Summary and Best Practices</td>
</tr>
</tbody>
</table>
For each benchmark run on 4 clients:
- Record size: 500 bytes/message
- Sends 400 M messages at 4 GB/s
- Publishes to a Kafka topic, 8 M messages at a time.

For each benchmark run, Kafka cluster can process:
- 8 M messages/sec, for the message size of 500 bytes.
- Average latency of acknowledgment to the clients is ~40 ms/message.

Apache Kafka® Connect Cluster
Worker 1
Connector 1
Connector 32
Worker 2
Connector 33
Connector 64
Worker 3
Connector 65
Connector 96
Worker 4
Connector 97
Connector 128

Each connector in the Kafka Connect Cluster subscribes to a Kafka topic in the Kafka Cluster, and sinks the data to the N5800 array.
A total of 128 connectors is writing to a single N5800 array.

- Achieved average write throughput to a single N5800 array is ~ 3.82 GB/s.
- Projected throughput will increase to 7.9 GB/s for higher ingestion rates, while using 8 worker nodes in the Kafka Connect cluster.
Each connector (known as sink connector) is assigned to a partition of the respective Kafka topic, i.e., the number of connectors is equal to the number of partitions/files.

Write throughput increases linearly with the number of connectors.

With a single IntelliFlash N5800 array, 128 sink connectors and 128 Kafka partitions, a 4-node Kafka Connect cluster provides a write throughput of 3.82 GB/s, for an ingestion rate of 4 GB/s.

A maximum of 7.9 GB/s write throughput with a single IntelliFlash N5800 array can be achieved with 8 (projected) Kafka Connect worker nodes, for ingestion rates higher than 4 GB/s.

This test helps to determine the number of connectors to configure in the Kafka Connect cluster, based on the number of N5800 arrays, the input ingestion rates, and the available I/O throughput from the flash arrays.

The CPU usage is 80% per worker node (having 32 connectors) in the 4-node Kafka Connect cluster to achieve a max throughput with 128 connectors.

4 JVMs are used for each worker node in the Kafka Connect cluster, with a JVM heap size of 64 GB.
Topics Covered

1. An AI Data Pipeline - Stages and Requirements
2. Disaggregated Architectures for AI Implementations
3. Training and Inference Performance on a Disaggregated Architecture
4. Data Ingestion Performance with Apache Kafka© to an NVMe™ All-Flash Array
5. A Real-World Use Case
6. Summary and Best Practices
**Implemented Real-World Use Case**

**ActiveScale™ System**

- **Data Pipeline Service**
  - Publishes the metadata of image S3 objects to a Kafka topic

- **Kafka Topic**
  - Subscribes to the Kafka topic to get the S3 locations of the images

- **AI Model Serving**
  - Uses AI/Deep Learning algorithms for analyzing the images, and generating custom metadata
  - Stores the S3 object paths and custom metadata in Elasticsearch©

- **Dashboard**
  - Kibana™

- **Key-Value Store**
  - Elasticsearch©

- **AI Model Training**
  - TensorFlow
  - GPU Cluster
  - Training Data

- **IntelliFlash™ N5800**
  - Gets AI model from N5800
  - Score < 80%
  - Score > 80%

**Car Image S3™ Objects**

- Search and Visualize
  - Custom metadata of car image S3 objects

**Flower Image S3™ Objects**

- Search and Visualize
  - Custom metadata of flower image S3 objects

**Notifies S3 locations of failed flower image objects to be used for re-training the model**

Flash Memory Summit 2019, Santa Clara, CA
© 2019 Western Digital Corporation or its affiliates. All rights reserved.
# Topics Covered

<table>
<thead>
<tr>
<th></th>
<th>An AI Data Pipeline - Stages and Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Disaggregated Architectures for AI Implementations</td>
</tr>
<tr>
<td>3</td>
<td>Training and Inference Performance on a Disaggregated Architecture</td>
</tr>
<tr>
<td>4</td>
<td>Data Ingestion Performance with Apache Kafka© to an NVMe™ All-Flash Array</td>
</tr>
<tr>
<td>5</td>
<td>A Real-World Use Case</td>
</tr>
<tr>
<td>6</td>
<td>Summary and Best Practices</td>
</tr>
</tbody>
</table>
Summary and Best Practices

• Implementing a disaggregated architecture of GPU compute, a shared pool of IntelliFlash N-series arrays and ActiveScale system(s) has multiple benefits while executing AI workloads –
  ➢ Subsequent data transfers in and out of local SSDs of GPU servers can be avoided, as the data grows over capacity.
  ➢ Inference is faster due to immediate access to trained models on the shared flash storage.
  ➢ Businesses have the ability to scale GPU servers and shared flash arrays independently to meet the changing needs of their AI workloads.
  ➢ Users have the flexibility to run mixed AI workloads for training and inference.
  ➢ With a preferred allocation strategy of the I/O bandwidth, various teams can share and scale the IntelliFlash N-series arrays to serve multiple GPU servers in a cost-effective manner.
  ➢ A high capacity object storage system like ActiveScale, as a component of the disaggregated architecture, may be used as a landing zone for the ingested data as well as an archival solution.

• As a best practice to attain an optimal ingestion performance with Kafka to IntelliFlash N-series arrays, tuning the following parameters is recommended -
  ➢ Number of connectors and worker nodes in the Kafka Connect cluster, based on the number of N-series arrays, the input ingestion rates, and the available I/O throughput from the arrays;
  ➢ Based on the I/O throughput requirement, high-speed network interfaces and topology need to be configured for the Kafka cluster, the worker nodes of the Kafka Connect cluster, and the IntelliFlash N-series array(s) to eliminate network bottlenecks.