Dell EMC PowerScale Deep Learning Infrastructure with NVIDIA DGX A100 Systems for Autonomous Driving

December 2021

H18627.1

White Paper

Abstract

This document demonstrates the design and architecture of a large-scale deep learning (DL) infrastructure for the development of autonomous driving use cases. It also showcases how the Dell EMC PowerScale F800 All-Flash scale-out NAS powered by PowerScale OneFS and NVIDIA DGX A100 systems are used to accelerate and scale DL training workloads. Benchmark results based on object detection and semantic segmentation use cases are created by using prominent, automotive-focused and publicly available datasets.

Dell Technologies

Copyright

Published in the USA December 2021 H18627.1.

Contents

Executive summary

Overview

Computer vision and machine learning (ML) solutions are integrated into vehicles to improve safety, convenience, and the driver experience. Data coming from multiple sources such as camera, lidar, radar and ultrasonic sensors are processed and used to extract information and characteristics of the surrounding environment. Modern vehicles have moved from passive safety systems such as seat-belt pre-tensioning and airbags (designed to minimize injury resulting from collisions).to active safety systems, such as anti-lock braking (ABS) and autonomous emergency braking (AEB) to avoid collisions.

With the evolution of Artificial Intelligence (AI) and DL, the industry is developing embedded control units (ECUs) into vehicles with computer vision algorithms that can interpret real-time sensor data and point cloud data to make corresponding predictions and to derive actions.

The significant automotive safety improvement in the past was passive safety, mainly designed to minimize damage during an accident. Advanced driver assistance systems (ADAS) in many of today's vehicles can proactively help the driver to avoid accidents by utilizing innovative DL technologies. For example, blind spot detection can alert a driver as they try to move into an occupied lane, pedestrian detection notifies the driver that pedestrians are in front of or behind the car, AEB activates the brakes to avoid an accident or pedestrian injury. More ADAS features like path planning combine with sensor fusion, which brings us closer to the goal of an autonomous vehicle. As the level of accuracy and sophistication increases, autonomous driving (AD) can realize improved capabilities. Critical success factors include improved safety algorithms, increased and efficient computational power, and access to large, comprehensive verification datasets.

This paper focuses on the IT infrastructure challenges faced by automotive original equipment manufacturers (OEMs), and suppliers in developing DL algorithms for ADAS/AD and proposes a scale-out compute and storage solution. The solution is optimized for ADAS/AD workloads—delivering high performance, high concurrency, massive scalability, and flexibility.

Revisions

Table 1. Revisions

Date	Description
January 2021	Initial release
December 2021	Template update

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on this document. Contact the Dell Technologies team by email (subject line: Feedback for document: H18627.1).

Author: Frances Hu

Contributors: Florian Baumann

Note: For links to other documentation for this topic, see the PowerScale Info Hub.

DL in ADAS development

ADAS development cycle

The following figure illustrates the typical ADAS development lifecycle for automotive OEMs and suppliers leveraging the Dell EMC PowerScale scale-out NAS as the central data lake:

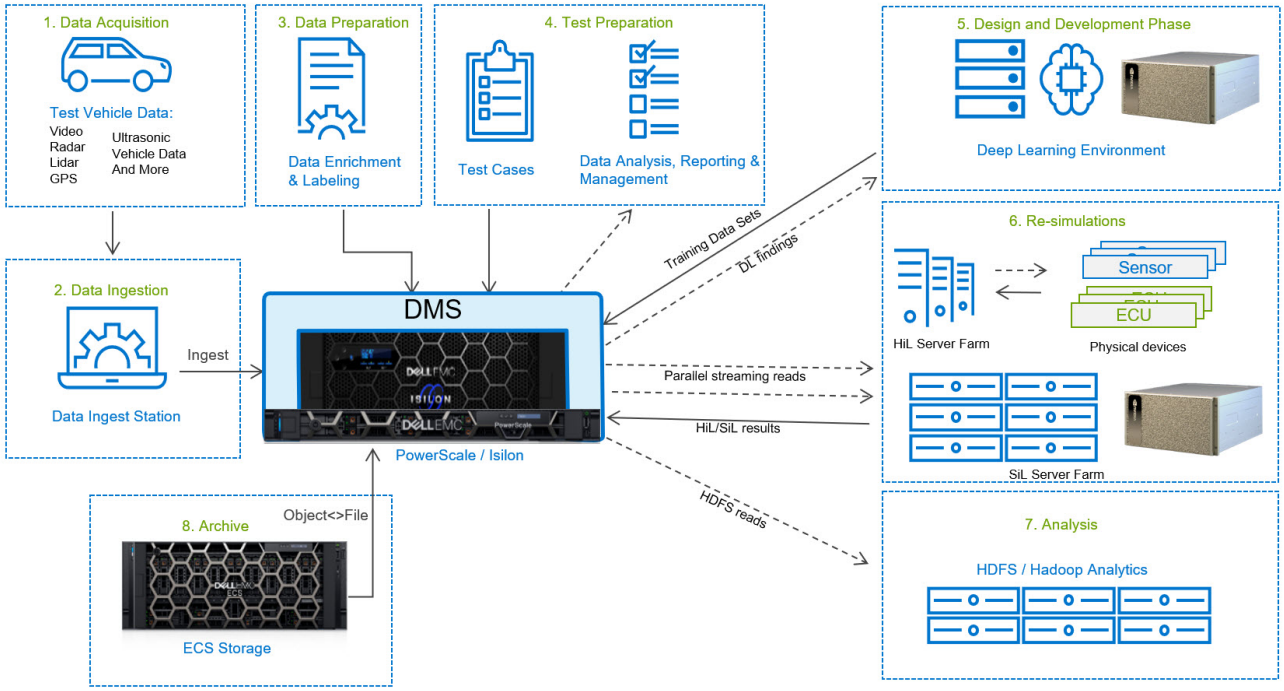

Figure 1. ADAS development lifecycle

1. **Data Acquisition**: Huge volumes of sensor data are captured by a fleet of test vehicles, including camera video sequences, ultrasonic proximity, radar, Light Detection and Ranging (LiDAR), Global Positioning System (GPS), and other sensor data. Sensors with very high-resolution such as 4K (or greater) cameras are being adopted. Some of these sensors will be beta samples of the actual sensors planned for the production vehicle, while other sensors will be capturing high-resolution reference data ("ground truth") around the test vehicle. Another important reference is the actions of the test driver – such as accelerator / brake / steering functions. Typically, we see ADAS developers generating real-world test data on the order of 2 TB per hour, or around 30–80 TB per car per day with some developers running test fleets with 50 or more vehicles. The data is stored with each vehicle in real-time using dedicated industrial data-logging hardware with removable solid-state storage disks. These drives are swapped out either daily or at the end of each shift—depending on the amount of data captured per shift. The drives are then either shipped directly to a centralized ingest server, transferred virtually through WAN lines, or transferred locally to tape, with the tapes then being shipped to a centralized ingest server for upload to the data lake (centralized storage).
2. **Data Ingestion**: During the data ingest process, which includes moving data from the vehicle to the data lake, custom copy stations apply data cleaning and lossless data compression algorithms to reduce the final amount of required storage and,

therefore, costs. Data cleansing can occur in-line to avoid multiple copies or move operations. Typically, 10%–15% of the recorded data is used to train the ML/DL algorithms and over 50% of the recorded data is used to run re-simulations as well.

3. **Data Preparation:** Once the data has been ingested, engineering teams will prepare the data, which may include trimming, decoding, data enrichment (labeling or ground truth generation), processing and adding metadata such as weather and traffic conditions. This requires a vast amount of compute and GPU resources in the high-performance computing (HPC) cluster to process the data, as well as fast storage—such as Dell EMC PowerScale storage, which meets both high capacity and high sequential read and write performance needs.
4. **Test Preparation**: Engineers can build test suites with various test use cases, as well as required closed-loop simulation and open-loop re-simulation (replay) validation jobs to verify ADAS models. With massive raw datasets, it is vital to be able to search through metadata quickly to find the corresponding sensor data for specific scenarios from within the vast data lake. Tests developed using captured sensor data tests against possible corner cases, with discrepancies between the ECU validation and test driver actions identified as potential bugs. The NVIDIA DRIVE Sim software and NVIDIA DRIVE Constellation AV simulator deliver a scalable, comprehensive, and diverse testing environment. DRIVE Sim is an open platform with plug-ins for third-party models from ecosystem partners, allowing users to customize it for their unique use cases.
5. **Design and Development Phase**: When the data is ready, the ADAS engineering teams can develop and build algorithms for smart cameras or ECU models through DL. ML models are created by training them on a huge amount of data (traffic signs, lanes, vehicles, and people). With huge amounts of data, it is strongly recommended to run multi-GPU training with data parallelism. Data parallelism allows a higher throughput of images during training, and therefore reduced training time.
6. **Re-simulations**: After algorithms are developed, it is crucial to run iterative tests using data fusion of all the sensors, GPS, weather, and road/environment data. On small projects, individual sensors and ECUs may be tested independently. Then all subsystems are tested together at the system level (sensor fusion). It is important to test corner-cases and complex interaction of various ADAS functions, sophisticated scenarios involving various road environments, pedestrians, other vehicles, and driver behavior to ensure the safety of the system. With various test cases, the engineering teams can schedule re-simulation jobs on the hardware-in-the-loop (HiL) and software-in-the-loop (SiL) computer clusters. Re-sim involves “replaying” the captured raw sensor data back through the test farm—usually with hundreds or even thousands of iterations running in parallel. For HiL this will be replayed in real-time, and for SiL it is often replayed faster than real-time. This workload requires the inherent high-concurrency benefit of the Dell EMC PowerScale scale-out NAS architecture. The NVIDIA DRIVE Sim software and NVIDIA DRIVE Constellation AV simulator deliver a scalable, comprehensive, and diverse testing environment. DRIVE Sim is an open platform with plug-ins for third-party models from ecosystem partners, allowing users to customize it for their unique use cases. For more information, refer to this link.
7. **Analysis:** Once testing is complete, engineers need to analyze the test results and determine whether additional validation is required. In-place analytics compare

ECU operation to original test driver actions to quickly identify potential bugs. Algorithms are refined to achieve the expected output results, and the revised ECU version can be uploaded to the test vehicles adopting a continuous improvement process. All the results are sent to data center storage to provide the engineering teams with on-demand access.

8. **Archive:** Following final validation, data used to develop the ECU algorithms can move to lower-cost archive storage. Archiving must meet regulatory and contractual commitments, which typically span multiple decades – the agreed "life of the vehicle." Many OEMs stipulate service-level agreements (SLAs) of 1-30 days for simulation data restoration time – for example, in the event of a safety recall – to allow quick turn-around of updates. Dell Technologies strongly recommends an active archive solution such as ECS cloud neutral object storage, implementing the S3 API.

Note: The preceding steps are not time sequential. Steps are concurrent and continuous to ensure high-quality solution outcomes and efficiency.

Challenge of DL training workload for ADAS

Training and validating new deep neural networks, such as those used in ADAS / AD development, require large datasets along with significant IT infrastructure that includes compute, networking and storage. The right infrastructure is crucial for safety-critical system development. These advanced algorithms must operate even within complex circumstances like varying weather conditions, visibility and road surface quality.

Key challenges of the DL training workload for ADAS are:

- **Explosive Data Growth**: A typical vehicle used for data collection in the ADAS system test use case includes multiple sensors such as LiDAR, RADAR, ultrasonic, GPS and cameras – all of which continuously generate data. Also, the vehicle controller area network (CAN) bus data and test driver captures the control information. This high level of visibility and redundancy builds a detailed picture to enable the vehicle to make reliable decisions in adverse weather conditions or in the event of an individual component failure. Due to the safety requirements for driving, development engineers need to ensure that the system used can detect objects sufficiently far away to operate safely at high speeds. This combination of vehicle speed and critical safety demands higher image resolutions than used in other industries, which in turn generates more data. Massive challenges occur in terms of the scale of the unstructured sensor data (videos, cloud point, images, text) that must be captured and replayed to test ADAS subsystems.

  To illustrate, a typical SAE Level 2 ADAS project, capturing 200,000 km of driving at an average speed of 65 km/h, would generate over 3,076 hours of test data, requiring approximately 6.2 petabytes (PB) of storage. Note that even within SAE L2 solutions, the total number of ADAS sensors required varies with functionality (lane departure warning, self-parking, and more). Multiple sensors are typically required. A typical SAE Level 3 ADAS project, which typically requires 1,000,000 km of driving, could generate 30 PB of raw sensor test data per test vehicle. As most ADAS developers have multiple cars, typical total storage for a single production vehicle development averages between 50 – 200 PBs of data.

- **Fast training cycle**: To assure safety and reliability, the neural networks designed must utilize millions of parameters which generate more compute-intensive

requirements for the underlying systems and hardware architecture. To accelerate time-to-market, neural network training must be as fast and efficient as possible. First, the deeper the network, the higher the number of parameters and operations needed to store many intermediate results in GPU memory. Second, training usually proceeds in the method of mini-batches, I/O throughput is thus the primary performance metric of concern in DL training.

- **Test and validation**: Validation is a key stage of the ADAS development cycle. Since most ADAS systems are intended to improve safety, the robustness and reliability of the trained model is paramount. This demands exhaustive testing and verification on the trained algorithm to represent diverse traffic scenarios and dimensions, which might include road geometry, driver and pedestrian behaviors, traffic conditions, weather conditions, vehicle characteristics and variants, spontaneous component faults, security, and more.
- **High quality labeled data**: The availability of labeled data is critical for ADAS DL training. High quality labeled data yields better model performance. Labels are added either manually (often via crowd sourcing) or automatically by image analysis, depending on the complexity of the problem. Labeling massive collections of training data is a tedious task and requires significant effort.

Key components

Introduction

This section describes the key components recommended for distributed DL targeted at ADAS/AD development.

PowerScale storage for DL

Dell EMC PowerScale all-flash storage platforms, powered by the PowerScale OneFS operating system, provide a powerful yet simple scale-out storage architecture to speed access to massive amounts of unstructured data, while dramatically reducing cost and complexity. With a highly dense design that contains four nodes within a single 4U chassis, PowerScale all-flash delivers extreme performance and efficiency for your most demanding unstructured data applications and workloads – including ADAS/AD. The Dell EMC PowerScale family includes four all-flash nodes recommended for DL workloads:

- **PowerScale F200:** Provides the performance of flash storage in a cost-effective form factor to address the needs of a wide variety of workloads. Each node allows you to scale raw storage capacity from 3.84 TB to 15.36 TB per node and up to 3.8 PB of raw capacity per cluster. The F200 includes in-line compression and deduplication. The minimum number of PowerScale nodes per cluster is three while the maximum cluster size is 252 nodes.
- **PowerScale F600:** With NVMe flash drives, the F600 provides larger capacity with massive performance in a cost-effective compact form factor to power the most demanding workloads. Each node allows you to scale raw storage capacity from 15.36 TB to 61.4 TB per node and up to 15.48 PB of raw storage per cluster. The F600 includes inline software data compression and deduplication. The minimum number of nodes per cluster is three while the maximum cluster size is 252 nodes.
- **PowerScale F800**: Provides massive performance and capacity and delivers up to 250,000 IOPS and 15 GB/s aggregate throughput in a single chassis configuration and up to 15.75M IOPS and 945 GB/s of aggregate throughput in configurations of

up to a 252-nodes cluster. Each chassis houses 60 SSDs with a capacity choice of 1.6 TB, 3.2 TB, 3.84 TB, 7.68 TB, or 15.36 TB per drive. This allows you to scale raw storage capacity from 96 TB to 924 TB in a single 4U chassis and up to 58 PB in a single cluster.

- **PowerScale F810**: Provides massive performance and capacity along with inline data compression and deduplication capabilities to deliver extreme efficiency. The F810 delivers up to 250,000 IOPS and 15 GB/s aggregate throughput in a single chassis configuration and up to 15.75M IOPS and 945 GB/s of aggregate throughput in a 252-node cluster. Each F810 chassis houses 60 SSDs with a capacity choice of 3.84 TB, 7.68 TB, or 15.36 TB per drive. This allows you to scale raw storage capacity from 230 TB to 924 TB in a 4U chassis and up to 58 PB of raw storage in a single cluster. Depending on your specific dataset and workload, F810 inline data compression and deduplication delivers up to a 3:1 reduction in storage requirements, this increasing the effective capacity up to 138 PB per cluster. For more information, see the document PowerScale All-Flash Scale-Out NAS Specification Sheet.

Figure 2. Dell EMC Isilon F800/F810

Dell EMC PowerScale families have the following features to benefit DL:

- Low latency, high throughput, and massively parallel I/O for AI. This shortens time for training and testing analytical models on data sets from tens of TB to hundreds of PB on AI platforms such as TensorFlow, SparkML, Caffe, or proprietary AI platforms. Isilon F810/F800 performance characters are:
  - Up to 250,000 file IOPS per chassis, up to 15.75M IOPS per cluster
  - Up to 15 GB/s throughput per chassis, up to 945 GB/s per cluster
  - 230 TB to 924 TB raw flash capacity per chassis; up to 58 PB per cluster (All-Flash)
- The ability to run AI in-place on data using multi-protocol access. Most data used for DL training is also used for other workloads, like HiL and SiL validation. These workloads, which use captured sensor data, typically require lower cost hybrid storage, such as Isilon H5600 but in one PowerScale cluster. This eliminates the need to migrate/copy data and results over to a separate AI stack. Organizations can perform DL and run other IT apps on same data already on PowerScale by adding PowerScale all-flash nodes to the existing cluster.
  - Multi-protocol support such as SMB, NFS, HTTP, and native HDFS to maximize operational flexibility

- Enterprise grade features out-of-box. This enables organizations to manage AI data throughout the lifecycle of the ADAS project with minimal cost and risk, while protecting data and meeting regulatory requirements.
  - Enterprise data protection and resiliency
  - Robust security options
  - Economical, long-term archival with fast recovery
  - Data Management System (DMS) using a single pane of glass
  - Container Storage Interface (CSI) driver for provisioning of persistent storage
  - Ansible Module to automate and orchestrate configuration and management (link)
- Extreme scale support. Organizations can achieve AI at scale in a cost-effective manner by leveraging PowerScale for DL as well as other ADAS workflows. Enabling them to handle multi-petabyte data sets with high resolution content and high performance without the need to re-architect their data center, and/or performance degradation.
  - Seamlessly tier between all flash, hybrid, and archive nodes via SmartPools
  - Grow-as-you-go scalability with up to 58 PB capacity per cluster
  - Up to 252 nodes may be connected to form a single cluster with a single namespace and a single coherent cache
  - Data Management solutions spanning multiple clusters for Exabyte scale
  - Depending on your specific dataset and workload, F810 inline data compression and deduplication delivers up to a 3:1 reduction in storage requirements, this increasing the effective capacity up to 138 PB per cluster

NVIDIA DGX A100 system

The NVIDIA DGX A100 system is a universal system for all AI workloads, offering unprecedented compute density, performance, and flexibility in the world's first 5-petaflop AI system. Each DGX A100 system features eight of the world's most advanced accelerators, the NVIDIA A100 Tensor Core GPU, enabling enterprises to consolidate training, inference and analytics into a unified, easy-to-deploy AI infrastructure that includes direct access to NVIDIA AI experts.

It allows organizations to standardize on a single system that can speed through any type of AI task at any time and dynamically adjust to changing compute needs over time. This unmatched flexibility reduces costs, increases scalability, and makes the DGX A100 system the foundational building block of the modern AI data center.

With the DGX SuperPOD program, customers can significantly reduce time to DC deployment and ultimately, time-to-market. Customers of the SuperPOD program have seen 6+ months reduction in DC deployment time. AI infrastructure requires extremely high-speed storage to handle a variety of data types in parallel, such as text, tabular data, audio, and video.

Figure 3. NVIDIA DGX A100 system

NVIDIA A100 GPU

The NVIDIA A100, built on the latest Ampere architecture, supports both training and inference workloads. In the case of AV development, DNN training and validation (RePlay) can be supported by one, unified infrastructure.

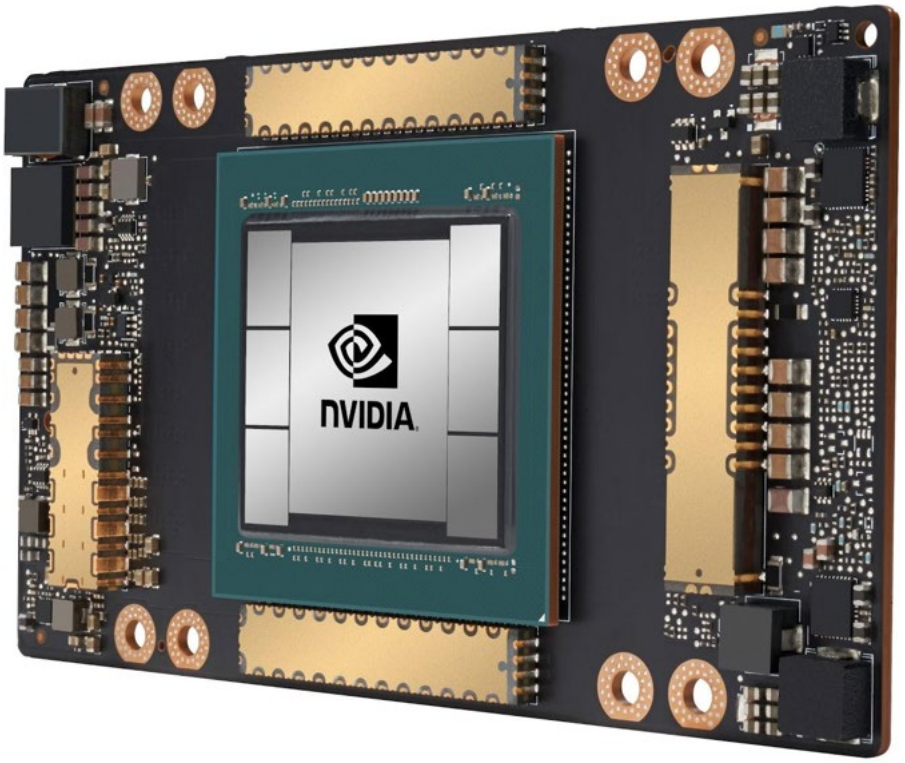

Figure 4. NVIDIA A100 GPU

DL models are very complex and large, and a DL framework is an interface, library or a tool which allows developers to tackle DL tasks easily and quickly, without requiring in-depth understanding of all the details of the underlying algorithms. These frameworks provide a clear and concise way for defining models using a collection of pre-built and pre-optimized components. Popular DL frameworks include TensorFlow, Keras, PyTorch, and Caffe.

Key characteristics of a well-designed DL framework include:

- Optimized for GPU performance
- Easy to understand and code
- Extensive community support
- Process parallelization to reduce computation cycles
- Automatically computed gradients
- Cloud-native app development capability (Docker, Kubernetes etc.)

Training with large datasets and DL networks can be accelerated by using multiple GPUs and/or more servers, but only if the underlying infrastructure is architected correctly.

In the market, there are some popular platforms and toolkits to allow developers to test distributed execution of different DL platforms on GPU clusters including MPI-based Uber Horovod and the Microsoft Distributed Machine Learning Toolkit (DMTK), available on the Microsoft website. Horovod is a distributed training framework for TensorFlow, Keras, PyTorch, and MXNet. The goal of Horovod is to make distributed DL fast and easy to use. These platforms are designed to make large-scale parallel distributed DL jobs easy and better.

NVIDIA NGC

The NVIDIA NGC container registry provides researchers, data scientists and developers with simple access to a comprehensive catalog of GPU-accelerated software for AI, DL, and HPC that take full advantage of NVIDIA DGX A100 systems. NGC provides containers for today's most popular AI frameworks such as RAPIDS, Caffe2, TensorFlow, PyTorch, MXNet and TensorRT, which are optimized for NVIDIA GPUs. The containers integrate the framework or application, necessary drivers, libraries and communications primitives and they are optimized across the stack by NVIDIA for maximum GPU-accelerated performance. NGC containers incorporate the NVIDIA CUDA Toolkit, which provides the NVIDIA CUDA Basic Linear Algebra Subroutines Library (cuBLAS), the NVIDIA CUDA Deep Neural Network Library (cuDNN), and much more. The NGC containers also include the NVIDIA Collective Communications Library (NCCL) for multi-GPU and multi-node collective communication primitives, enabling topology awareness for DL training. NCCL enables communication between GPUs inside a single DGX A100 system and across multiple DGX A100 systems.

Reference architecture

Introduction

The following figure illustrates the reference architecture showing the key components of the solution as it was tested and benchmarked. For an in-depth review of the reference

architecture, see Dell EMC PowerScale and NVIDIA DGX A100 Systems for Deep Learning.

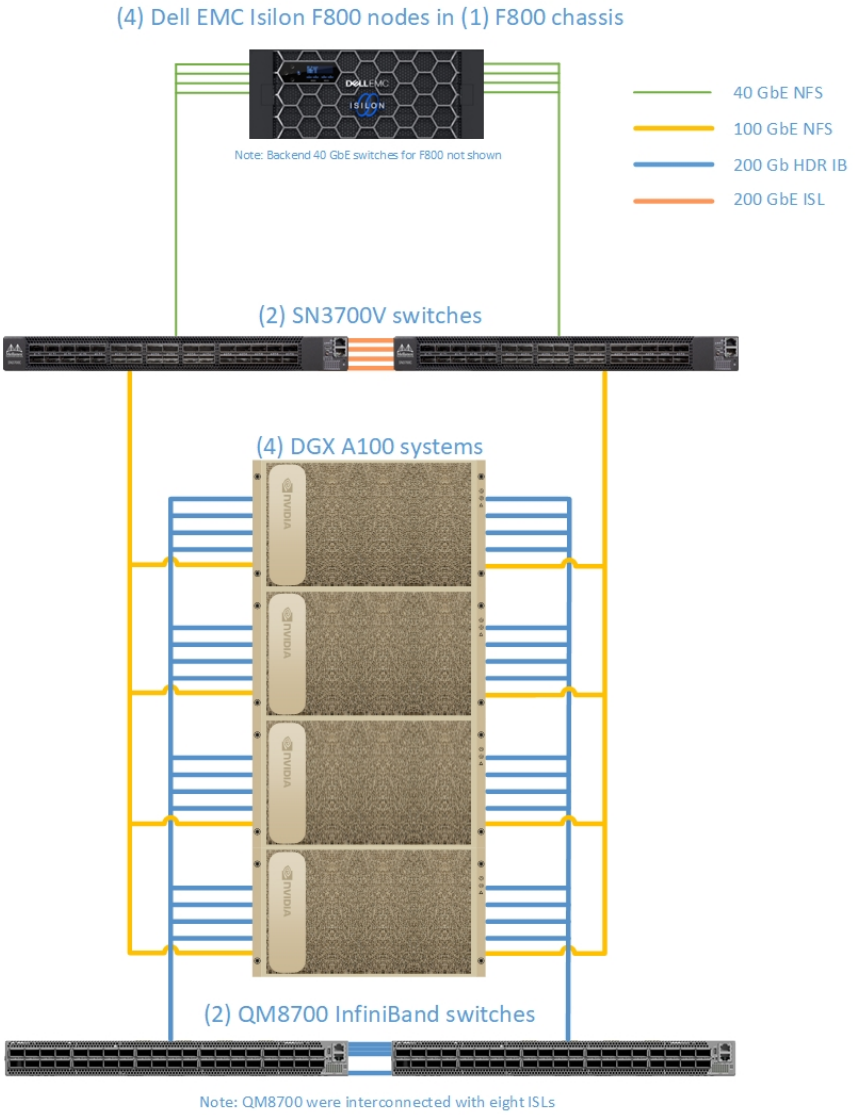

Figure 5. Dell EMC ADAS/AD DL reference architecture

ADAS deep learning reference architecture

Note: In a customer deployment, the number of DGX A100 systems and PowerScale storage nodes will vary and can be scaled independently to meet the requirements of the specific DL workloads.

The hardware architectures include these key components:

- **Compute**: Four NVIDIA DGX A100 systems.

  The DGX A100 system is a fully integrated, turnkey hardware and software system that is purpose-built for DL workflows. Each DGX A100 system is powered by eight NVIDIA A100 Tensor Core GPUs that are interconnected using NVIDIA NVSwitch technology, which provides an ultra-high bandwidth low-latency fabric for inter-GPU communication. This topology is essential for multi-GPU training, eliminating the

bottleneck that is associated with PCIe-based interconnects that cannot deliver linearity of performance as GPU count increases. The DGX A100 system is also equipped with eight single-port NVIDIA Mellanox ConnectX-6 VPI HDR InfiniBand adapters for clustering and two dual-port ConnectX-6 VPI Ethernet adapters for storage and management networking, all capable of 200Gb/s.

- **Storage**: A critical component of DL solutions is high-performance storage. One Dell EMC PowerScale F800 with 4 nodes was used in this solution with a mounted NFS share on the NVIDIA DGX A100 systems. It is uniquely suited for modern DL applications – delivering the flexibility to deal with any data type; the scalability for data sets ranging in the PBs; and the concurrency to support the massive concurrent I/O requests from the GPUs.
- **Networking**: The solution consists of two network fabrics:
  - The NVIDIA Mellanox SN3700V Ethernet switches provide the high speed "front-end" Ethernet connectivity between the Isilon F800 cluster nodes and NVIDIA DGX A100 systems. The F800 nodes connect with 40GbE connections, the DGX A100 systems connect with 100GbE connections, and the SN3700 switches automatically forward traffic across the different speed connections with minimal latency. Based on the NVIDIA Spectrum-2 switch ASIC and purpose built for the modern datacenter, the SN3700V switch combines high-performance packet processing, rich datacenter features, cloud network scale and visibility. A flexible unified buffer to ensure fair and predictable performance across any combination of ports and speeds from 10Gb/s to 200Gb/s, and an Open Ethernet design supports multiple network OS choices including NVIDIA Cumulus Linux, NVIDIA Onyx, and Software for Open Networking in the Cloud(SONiC).
  - The NVIDIA Mellanox QM8700 InfiniBand switches provide high-throughput, low-latency networking between the DGX A100 systems. Designed for both EDR 100Gb/s and HDR 200 Gb/s InfiniBand links, they minimize latency and maximize throughput for all GPU-to-GPU communication between systems. The QM8700 switches support Remote Direct Memory Access (RDMA) and in-network computing offloads for AI and data analytics to enable faster and more efficient data transfers. They support NVIDIA GPUDirect, Mellanox SHARP for network-based AI and analytics offloads (such as MPI AllReduce), and Mellanox SHIELD for maximum resiliency in a self-healing network. Learn more about the NVIDIA Mellanox Quantum QM8700 InfiniBand switches.

The software architectures include these key components:

- **Docker containers or virtual machines**: There has been a dramatic rise in the use of software containers for simplifying deployment of applications at scale. You can use either virtual machines or containers encapsulated by all the application's dependencies to provide reliable execution of DL training jobs. A docker container is more widely used now for bunding an application with all its libraries, configuration files and environment variables so that the execution environment is always the same. To enable portability in Docker images that leverage GPUs, NVIDIA developed the Docker Engine Utility for NVIDIA GPUs which is also known as the NVIDIA Container Toolkit, an open-source project that provides a command-line tool. The publicly available CSI driver for the scale-out NAS PowerScale provides support for provisioning of persistent storage.

Apart from the DL architecture, the development environment needs a rich ecosystem of open-source tools and partner-led solutions specific to the unique requirements of each customer, as shown in the following figure. Dell Technologies has put together the following map of workflow and possible ecosystem solutions. More information can be collected from a blog “What DevOps for AD/ADAS looks like” here: Link.

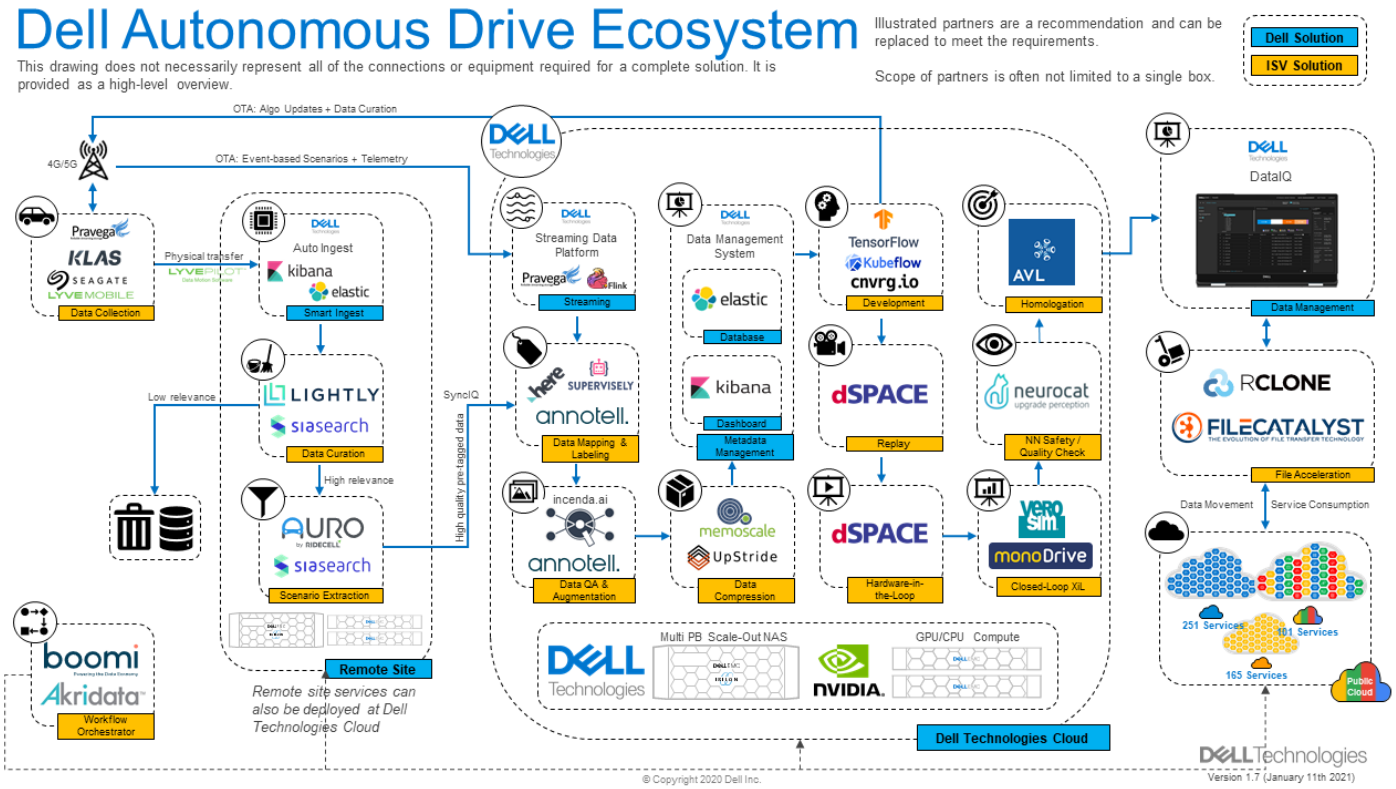

Figure 6. Dell autonomous drive ecosystem

Design considerations for distributed ADAS/AD DL training solution

Introduction

This section describes the key considerations and best practices of designing distributed ADAS DL solution.

Datasets preparation

Here are some key design considerations for preparing ADAS DL datasets:

- **Collect real-world raw datasets**: An end-to-end DL system requires large amounts of training data for the development of Deep Neural Networks (DNN) that can successfully interpret virtually all driving scenarios likely to be encountered in the real world. OEMs and Tier-1 suppliers typically deploy fleets of test cars outfitted with sensors to capture this data. Sensor data rates of around 2 TB/hour per car are common.
- **Data and scenario diversity**: The recorded sensor data is used to train DNNs in the data center, with over 1 million images per DNN viewed as a best practice for

relatively simple DNNs (complex DNNs typically will require up to 3 million images). A RAND report highlights that 11 billion miles from realistic road scenarios are needed to assure the safety of ADAS/AD DNN models. The data also must be diverse enough to cover different types of road scenarios to verify the model accuracy.

- **Public ADAS/AD dataset utilization**: Existing public AD datasets can be leveraged heavily to verify the algorithm accuracy as well. As shown in the following table, many multimodal datasets from different countries are now available, where each is typically comprised of images, range sensor (lidar, radar) data and GPS/IMU data. These are available to researchers and developers to accelerate the development and validation of ADAS systems (subject to individual license restrictions).

Table 2. Major public ADAS/AD datasets

Dataset	Year	Volumes	Diversity	Annotations
ApolloScape	2018	• 140,000 annotated images • 20,000 Lidar 3D point cloud annotation data • No radar	• Include GPS/IMU data/timestamps • Different times in the day • Mainly in Beijing, China	• Semantic • Instance-wise • Dense pixel annotations • 2D/3D boxes
BDD100K	2017	• 120,000,000 with 100,000 annotated images (1280 * 720) • No lidar/radar	• Multiple cities • Multiple scene type • Different times in the day • Include GPS/IMU data/timestamps • Multiple weather	• Semantic • Instance-wise • Dense pixel annotations • 2D boxes
Cityscapes	2016	• 25,000 annotated images (1280 * 720) • No lidar/radar	• 50 cities • Several months • Daytime • Good weather conditions • Include GPS/timestamp metadata	• Semantic • Instance-wise • Dense pixel annotations • 2D boxes
KITTI	2012	• 15,000 annotated images (1248 * 384) • 15,000 Lidar 3D point cloud annotation data • No radar	• Include GPS/IMU data/timestamps • Daytime • Mainly in Karlsruhe	• Semantic • Instance-wise • Dense pixel annotations • 2D/3D boxes
Lyft Dataset	2019	• 55,000 3D annotated images • HD Mapping data	• 1,000 driving scenes in multiple cities • Different times in the day	• Semantic • Instance-wise • Dense pixel annotations • 2D/3D boxes

Dataset	Year	Volumes	Diversity	Annotations
nuScenes	2019	• 1,400,000 with 40,000 annotated images • 390,000 Lidar 3D point cloud data • 1,400,000 radar sweeps	• 1,000 driving scenes in multiple cities • Different times in the day • Multiple weather	• Semantic • Instance-wise • Dense pixel annotations • 2D/3D boxes
Waymo Open Dataset	2019	• 200,000 annotated images (1920 * 1280 & 1920 * 886) • Lidar 3D point cloud data • 12 million 3D labels and 1.2 million 2D labels	• 1,000 driving scenes in multiple cities • Different times in the day • Multiple weather (day and night, dawn and dusk, sun and rain)	• Semantic • Instance-wise • Dense pixel annotations • 2D/3D boxes

Data ingestion and data management

Here are some key design considerations for data ingestion and data management for ADAS DL workflow:

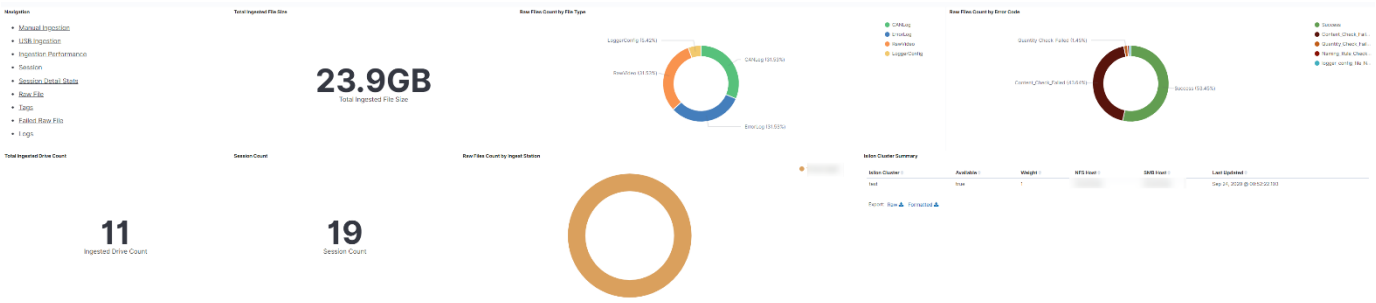

Figure 7. Dell EMC Data Management System dashboard

DMS is integrated into the managed service offering for ingesting data from test vehicles in an automated fashion. Specific metadata types and tags are configurable and can be customized for each customer. The service provider can manage logistics to receive the storage media from the vehicles and insert them into ingestion stations. This will automatically initiate processes that dynamically decide which cluster/location to upload the raw data to—in such a fashion that all ingestion processes are well balanced across the storage environment. DMS then initiates the data transfer automatically while enriching the data with appropriate metadata tagging, resulting in all data and tags being logged in a global index that tracks the data sets across the namespace and clusters. At the completion of the data transfer and indexing, DMS can launch post-processing jobs within the DL and HPC server grid as preconfigured through the scheduling platform. Functions available can include accurately splitting and merging data files as required, decoding the data into human readable form, automatically updating the index upon completion and other functions as defined by the customer. These processes are

wrapped within a series of real-time monitoring, status dashboards, and logs to alert of any errors or anomalies.

Dell EMC DataIQ product, as shown in the following figure, solves these challenges with its ability to break down storage silos and unify the unstructured data environment. It provides a single UI to view sensor data across heterogenous storage environments including the public cloud.

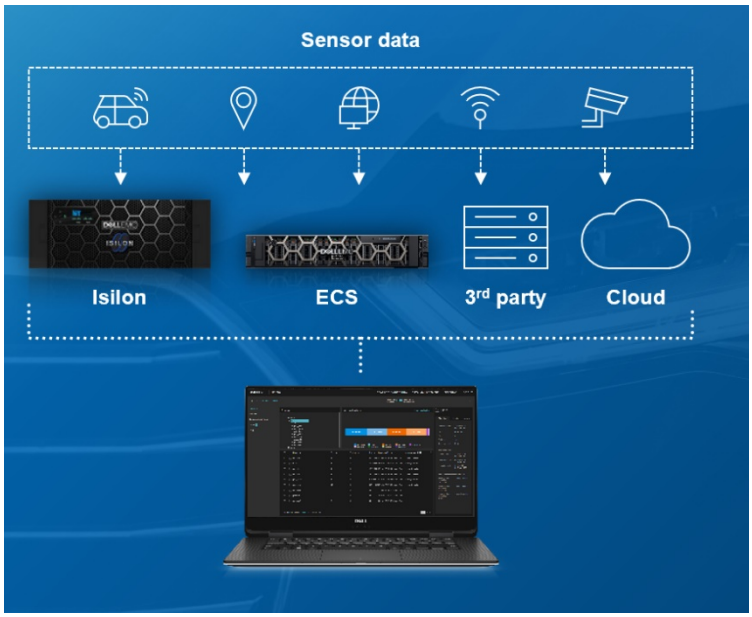

Figure 8. New hybrid cloud solution for ADAS development with Dell EMC DataIQ

DataIQ also equips organizations with the ability to rapidly search across billions of files and locate data, which can then be precisely moved or copied to on-prem or public cloud storage as needed, ensuring the right projects have access to pertinent data. This can help free up space on higher performance tiers and enables organizations to archive data as long as is required. The cost savings can be immense, and data can be retrieved easily from a well-curated, searchable archive for re-simulation and DL development in the future.

Data transform

Training neural networks with images requires developers to first normalize those images. Moreover, images are often compressed to save storage. Developers have therefore built multi-stage data processing pipelines that include loading, decoding, cropping, resizing and many other augmentation operators. Here are some key design considerations for data ingestion and data management for an ADAS DL workflow:

- **Consider Data augmentation strategy**: Dataset augmentation applies transformations to training data. Transformations can be as simple as flipping an

image, or as complicated as applying neural style transfer. By performing data augmentation, it is possible to increase training dataset size and diversity. This can help to prevent a neural network from learning irrelevant patterns (common with small datasets) that would degrade overall training accuracy. Here are two common methods for data augmentation:

- **Offline augmentation** is to create new augmented data which is stored on the filesystem. This can help effectively increase the training sample size many times over with variety of different augmentation technologies.

- **Use NVIDIA Data Loading Library (DALI):** The DALI is a portable, open source library for decoding and augmenting images, videos, and speech to accelerate DL applications. As shown in the following figure, DALI is a set of highly optimized building blocks plus an execution engine used to accelerate input data pre-processing for DL applications.

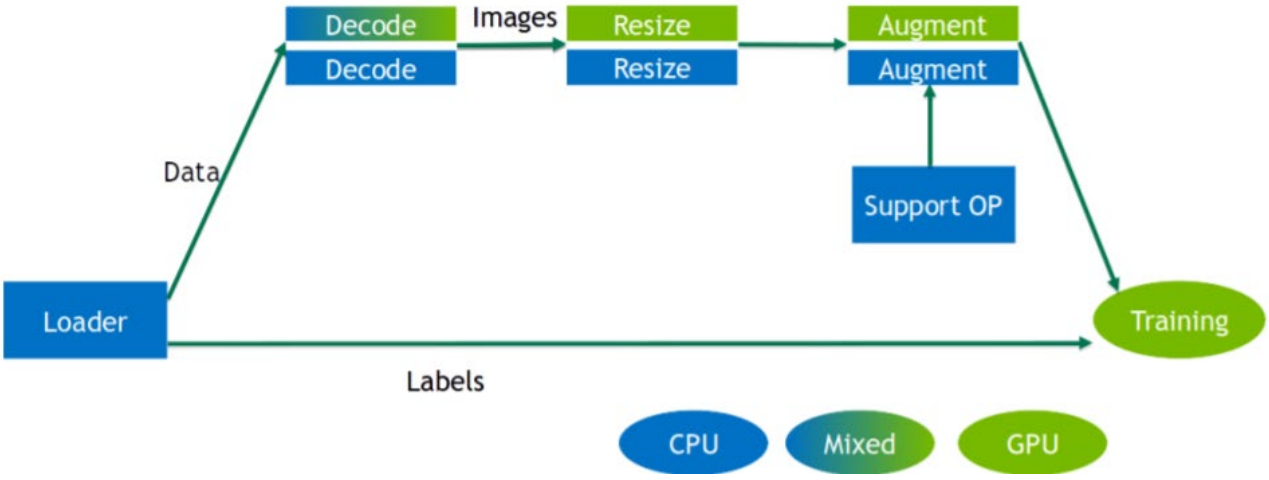

Figure 9. DALI inside architecture (©NVIDIA)

DALI provides performance and flexibility for accelerating different data pipelines. DALI reduces latency and training time, mitigating bottlenecks, by overlapping training and pre-processing. It provides a drop-in replacement for built in data loaders and data iterators in popular DL frameworks for easy integration and retargeting to different frameworks. Here are some key features:

- Easy-to-use Python API
- Transparent scaling across multiple GPUs
- Accelerated image classification (ResNet-50), object detection (SSD) workloads and speech recognition models such as Jasper and RNN-T
- Flexible graphs that let developers create custom pipelines
- Support of multiple data formats—LMDB, RecordIO, TFRecord, COCO, JPEG, wav, Free Lossless Audio Codec (FLAC), Ogg, H.264, and HEVC

- Ability for developers to add custom audio, image, and video processing

For more information, refer to NVIDIA blog.

Data model training

The ability to train neural network data models with many hidden layers, as well as the ability to train them with large datasets in a short amount of time, at scale, is critical to ADAS/AD development. To assure safety and reliability, the neural networks designed for driving operations will utilize many permutations of parameters which will generate more compute-intensive requirements for the underlying systems and hardware architecture. In distributed DL platforms, the model needs to be synchronized across all nodes. It also requires the careful management and distributed coordination of computation and communication across all nodes.

- **Data parallelism vs model parallelism**: Data parallelism, as shown in the following figure, is generally easier to implement. Each device works with a different part of the overall dataset and the devices collectively update a shared model. These devices can be located on a single machine or across multiple machines.

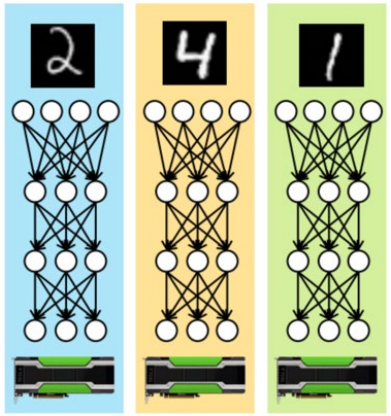

Figure 10. Data parallelism flow

Most DL frameworks use data parallelism to partition the workload over multiple devices. The following figure shows the details of the process of data parallelism to distribute training processes across multiple GPU systems and devices.

Most DL frameworks use data parallelism to partition the workload over multiple devices. The following figure shows the details of the process of data parallelism to distribute training processes across multiple GPU systems and devices.

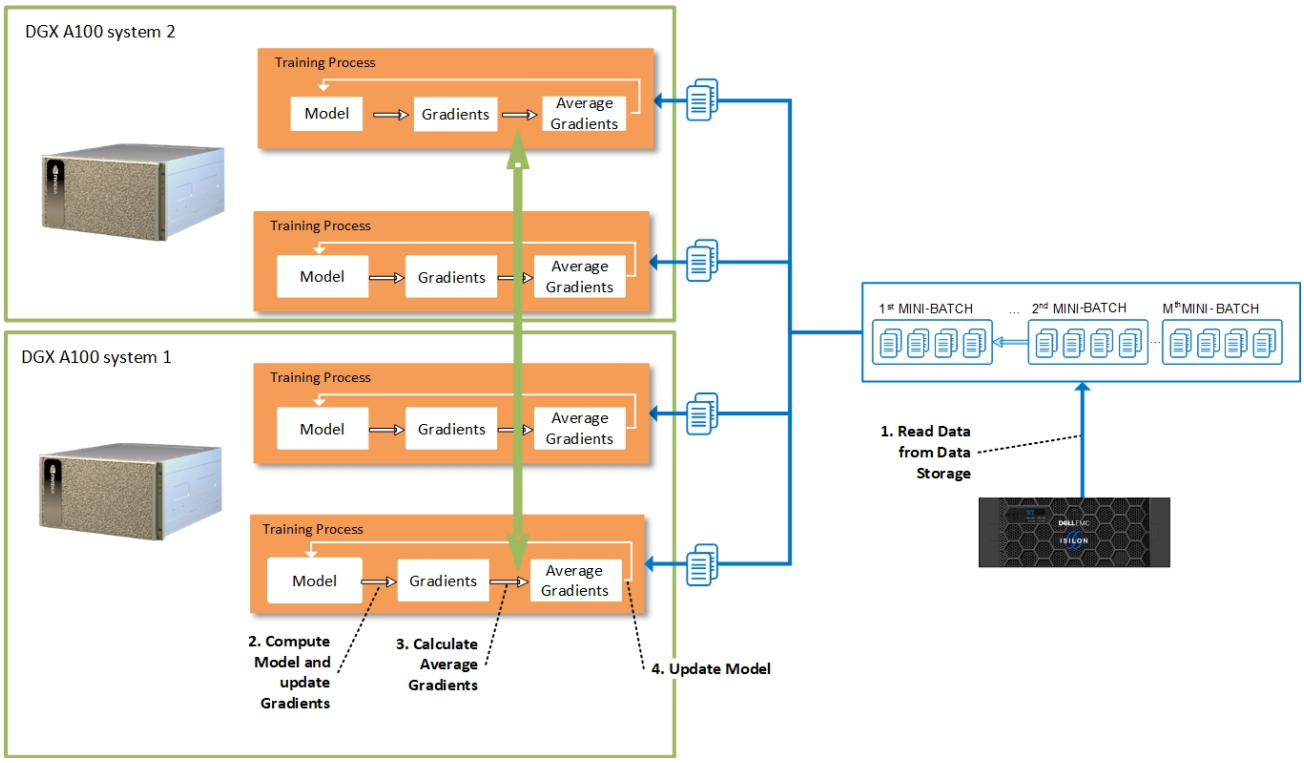

Figure 11. Data parallelism approach to distribute training processes

Data parallelism also requires less communication between nodes as it benefits from high amount of computations per weight. Assume for example, there are n devices, where each device receives a copy of the complete model and trains it with 1/nth of the data. The results such as gradients and the updated model itself are communicated across these devices.

To ensure efficient training, the network bandwidth across the nodes cannot become a bottleneck. Also note it is inefficient—and bad practice—to store training data on the local disks of every worker node, which forces the copying of terabytes of data to each worker node before the actual training can be started.

When models are so large that they don't fit into device memory, then an alternative method, called model parallelism, is employed. With model parallelism, as illustrated in the following figure, different devices are assigned the task of learning different parts of the model.

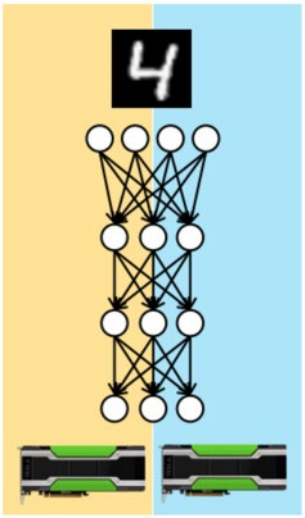

Figure 12. Model parallelism flow

Model parallelism requires more careful consideration of dependences between the model parameters. Model parallelism may work well for GPUs in a single server that shares a high-speed bus. It can be used with larger models as hardware constraints per node are no longer a limitation.

- **Leverage open source DL toolkits like Horovod:** Horovod is a distributed training framework for TensorFlow, Keras, PyTorch, and MXNet. The goal of Horovod is to make distributed DL fast and easy to use. Horovod use Message Passing Interface (MPI) model to be much more straightforward and require far less code changes than the Distributed TensorFlow with parameter servers. Horovod currently supports models that fit into one server but may span multiple GPUs.
- **Use the latest DL framework**: Leverage the latest DL framework and toolkit, to avoid potential performance issue or bugs, as well as compatible cuDNN and NVIDIA Collective Communications Library (NCCL). The newer version always delivers great performance improvements and bug fixes.

Infrastructure

Here are some key design considerations on designing infrastructure for distributed DL:

- **Build high network bandwidth between nodes:** Distributed DL is a very compute-intensive task. To accelerate computation, training can be distributed across multiple machines connected by a network. During the training, constant synchronization between GPUs within and across the servers is required. Limited network bandwidth is one of the key bottlenecks towards the scalability of distributed DNN training. Most important metrics for the interconnection network are low latency and high bandwidth. It is recommended to use InfiniBand or 100Gbps Ethernet Network to maximum the network bandwidth between nodes. With enough network bandwidth, GPU utilization across nodes performs similar to that of single GPU configurations.
- **Choose high throughput, scale-out centralized storage**: With scalable distributed DL, the distribution of the training data (batches) to the worker nodes is crucial. It is inefficient to store training data in the local disks or Non-Volatile Random-Access Memory (NVRAM) on every worker node and copy terabytes of data across each worker node before the actual training can be started. Using a

high throughput, scale-out storage for centralized storage for the training data, offers a more convenient, higher performance and cost-effective solution for ADAS/AD DL training.

GPU servers can have 20-30k cores and each one request its own read thread. Most file systems max out at between 2,000–6,0000 open read threads, but the PowerScale OneFS operating system doesn't have a performance limit on open threads. DL workloads also require high throughput random reads during the training and data processing, as well as high throughput sequential writes during data ingest. With distributed DL on multiple GPUs, to full utilize GPUs a steady flow of data from centralized storage into multiple GPUs jobs is critical for obtaining optimal and timely results. Data preprocessing (CPU) and model execution of a training step (GPU) run in parallel during the training and require high throughput data reads from storage.

An ADAS high performance DL system requires equally high-performance storage with scalability to multiple petabytes within a single file system. For a typical SAE level 3 project, this requirement ranges between 50 to 100 PB of data. The high-performance storage scalability is crucial to meet different business requirements of ADAS DL projects.

Dell EMC PowerScale all-flash storage platforms, powered by the PowerScale OneFS operating system, provide a powerful yet simple scale-out storage architecture that scales up to 33 petabytes per cluster. It allows DL workloads to access massive amounts of unstructured data with high performance, while dramatically reducing cost and complexity.

- **Leverage NVIDIA Container Toolkits on NVIDIA GPUs:** NVIDIA offers ready-to-run GPU-accelerated containers with the top DL frameworks. NVIDIA also offers a variety of pre-built containers which allow users to build and run GPU accelerated Docker containers quickly. To deploy DL applications at scale, it is very easy to leverage containers encapsulating an application's dependencies to provide reliable execution of application and services even without the overhead of a full virtual machine. For more detail information, refer to the NVIDIA Container Toolkit page.
- **Use Kubernetes to manage containers for distributed DL**: Kubernetes is an increasingly popular option for training deep neural networks at scale, as it offers flexibility to use different ML frameworks via containers as well as the agility to scale on demand. It allows researchers to automate and accelerate DL training with their own Kubernetes GPU cluster. For more information, see How to automate DL training with Kubernetes GPU-cluster.
- **Use NVIDIA DL GPU Training System (DIGITS) for user interface**: DIGITS is a wrapper for NVCaffe and TensorFlow that provides a graphical web interface to those frameworks in the following figure, rather than dealing with them directly on the command-line.

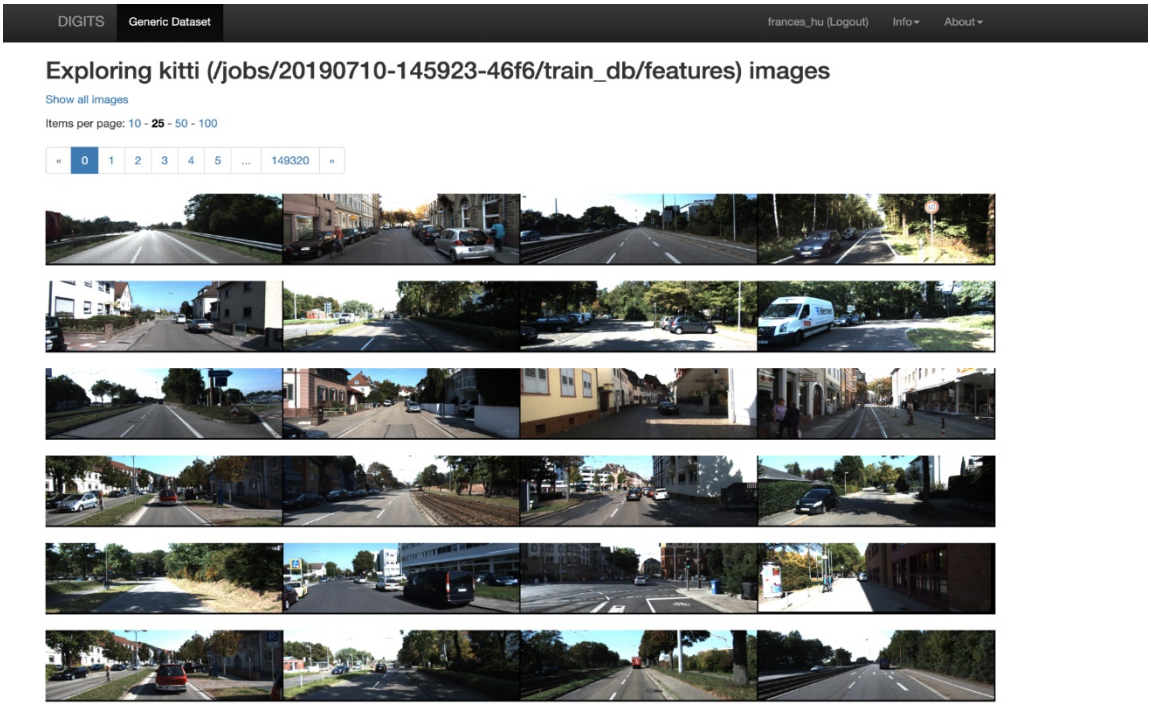

Figure 13. NVIDIA DIGITS for DL user interface platform

DIGITS simplify common DL tasks such as managing data, designing, and training neural networks on multi-GPU systems, monitoring performance in real time with advanced visualizations, and selecting the best performing model from the results browser for deployment. DIGITS is completely interactive so that data scientists can focus on designing and training networks rather than programming and debugging.

ADAS DL training performance and analysis

Introduction

In this section, the performance of DL training is tested for Cityscapes object detection and semantic segmentation, as shown in Table 2. The well-known dataset used was the Cityscapes Benchmark which consists of 2,975 training images with high quality dense pixel annotation, 500 validation images and 1,525 test images. This dataset is commonly used by ADAS DL researchers for benchmarking and comparison studies. Cityscapes has labeled 30 different classes.

ADAS dataset and CNN models

Since the entire Cityscapes raw dataset, including label data, is only 11 GB and can easily fit into system memory. The size of the dataset was increased 557 times to exercise the storage system (Isilon F800) more realistically. We did this by applying random rotation data augmentation techniques to each JPEG image in the dataset. This is standard practice in data analytics and DL to increase size of data sets. In total, this “557x” dataset contained 2,025,975 images and annotations with 5.4 TB in total. The average PNG image pixel is 1024 x 2048 and average size is 2,193KB.

The performance test utilized in this document uses this data to train two different convolutional neural network (CNN) models, shown in the following table, that are used for semantic image segmentation and object detection.

Table 3. Convolutional neural network models used in the tests

CNN model	Purpose	Dataset	DL framework	Distributed framework
SSD: Single Shot MultiBox Detector	Real-Time Object Detection	Cityscapes Benchmark data with 972,825 training images (2.8 TB in total)	PyTorch 1.4	PyTorch Distributed Data Parallel
Semantic-Segmentation	Hierarchical Multi-Scale Attention for Semantic Segmentation.	Cityscapes Benchmark data with 2,025,975 training images (5.4 TB in total)	PyTorch 1.3	PyTorch Distributed Data Parallel

Hardware and software configuration

In this test, the hardware comprises four DGX A100 systems with four Isilon F800 nodes, and network switches. The following tables show the detailed hardware and software configurations.

Table 4. Hardware configuration

Component	Number	Configuration
Storage	1	Dell EMC Isilon F800 4-nodes cluster • Front-end Network: 2* 40Gb Ethernet • Drives: 15.4 TB SSD x 60 with 924 TB capacity
Compute Node	4	NVIDIA DGX A100 system: • 8 NVIDIA A100 Tensor Core GPUs with 40GB • Two 64-Core AMD EPYC 7742 @3.3 GHz • 1 TB RAM • 2x Dual-Port NVIDIA Mellanox ConnectX-6 VPI 200 Gb/s Ethernet • 8x Single-Port NVIDIA Mellanox ConnectX-6 VPI 200Gb/s HDR InfiniBand
Storage Fabric Switch	2	NVIDIA Mellanox SN3700V Ethernet switches
Compute Fabric Switch	2	NVIDIA Mellanox QM8700 InfiniBand HDR Switch

Table 5. Software configurations

Component	Version
Dell EMC Isilon – OneFS	• 8.2.1.0 • Patches: 8.2.1_KGA-RUP_2020-04_268538, 8.2.1_UGA-PATCH-INFRA_2019-11_263088, 8.2.1_UGA-RUP_2020-04_268536
NVIDIA Mellanox SN3700V – NCLU Version	1.0-cl4.2.1u1
NVIDIA Mellanox SN3700V – Distribution Release	4.2.1
NVIDIA Mellanox QM8700 Product Release	3.9.0606
DGX A100 system – Base OS	4.99.11
DGX A100 system – Linux kernel	5.3.0-59-generic

Component	Version
DGX A100 system – NVIDIA Data Center GPU Driver	450.51.06
DGX A100 system – Ubuntu	18.04.5 LTS
NVIDIA NGC PyTorch Image	nvcr.io/nvidia/mxnet:20.06-py3

Test methodology

In order to measure the training time and bandwidth requirements for the distributed DL platform, different training procedures were carefully executed by using different configuration setups.

Real-time object detection model training test methodology

Here are some key test methodologies that we used to during the training benchmark:

- In order to measure the distributed training performance of the solution, we used SSD (Single Shot MultiBox Detector) model from the NVIDIA DL Examples GitHub repository. This suite of benchmarks performs training of augmented Cityscapes labeled images. This dataset contains 972,825 training images totaling 2.8 TB. This dataset is commonly used by automotive DL researchers for benchmarking and comparison studies. The solution used CityscapesScripts to convert annotations in standard PNG format to COCO format for SSD training.
- Training of SSD requires computationally costly augmentations. To fully utilize GPUs during the training, we are using the NVIDIA DALI library to accelerate data preparation pipelines.
- Scale-out trainings were performed on HW configurations ranging from one DGX A100 system with eight A100 GPUs to four DGX A100 systems with 32 A100 GPUs. This enabled the measurement of the training time, throughput performance, and provided a basic understanding of the training performance.
- In order to measure the training time across multi-GPUs and determine the corresponding bandwidth requirement, we trained SSD model for multi-epochs with the following setup:
  - SGD with momentum: 0.9
  - Learning rate: 2.6e-3 * number of GPUs * (batch_size / 32)
  - batch size: 16 per GPU
  - number of worker threads: 20
  - no warmup
  - ResNet-50 is used as backbone
- Prior to each execution of the benchmark, the L1 and L2 caches on Isilon F800 were flushed with the command isi_for_array isi_flush. It is worth noting, however, that the training process will read the same files repeatedly and after just several minutes, much of the data will be served from one of these caches.
- Multi-GPU training with Distributed Data Parallel – the NVIDIA model uses APEX's DDP to implement efficient multi-GPU training with NCCL.
- Excepting the training model, we also used the following command to evaluate the training benchmark:

```
python -m torch.distributed.launch --nproc_per_node=8 --
nnodes=3 --node_rank=0 \
     main.py --batch-size 32\
               --mode training \
               --num-workers 20\
               --epochs 20\
               --data /mnt/cityscapes/ssd3T
```

Semantic segmentation model training test methodology

Here are some key test methodologies that we used during the training benchmark:

- In order to measure the distributed training performance of the solution, we used NVIDIA semantic segmentation from the GitHub repository to train the model. This training was performed on augmented cityscapes labeled images. This dataset contains 2,025,975 training images in 5.4 TB.
- Scale-out trainings were performed to measure the training time and throughput performance in order to provide a basic understanding of the training performance. Hardware configurations used for this testing ranged from one NVIDIA DGX A100 compute nodes with 8 A100 GPUs to 4 NVIDIA DGX A100 compute nodes with 32 A100 GPUs.
- In order to measure the training time across multi-GPUs and evaluate bandwidth requirement, we trained semantic segmentation model in multiple epochs with the following setup:
  - batch size:8 per GPU
  - number of worker threads: 10
  - no warmup
  - deepv3Plus
- Prior to each execution of the benchmark, the L1 and L2 caches on Isilon F800 were flushed with the command `isi_for_array isi_flush`. In addition, the Linux buffer cache was flushed on all compute nodes by running sync; `echo 3 > /proc/sys/vm/drop_caches`. However, note that the training process will read the same files repeatedly and after just several minutes, much of the data will be served from one of these caches.
- Multi-GPU training with Distributed Data Parallel – the NVIDIA model uses Apex's DDP to implement efficient multi-GPU training with NCCL. For example, we used the following command to evaluate the training benchmark:

```
python -m torch.distributed.launch --nproc_per_node=8 --
nnodes=4 --node_rank=0 \
     train.py  --bs_trn 8\
               --apex \
               --fp16 \
               --crop_size "800,800"\
               --num_workers 10\
               --max_epoch 20\
               --arch deepv3.DeepV3PlusW38
```

Test results and performance analysis

Training throughput

The following figure shows the Single Shot MultiBox Detector model training benchmark throughput results. It is obvious from the results that image throughput scales linearly from 8 to 32 A100 GPUs.

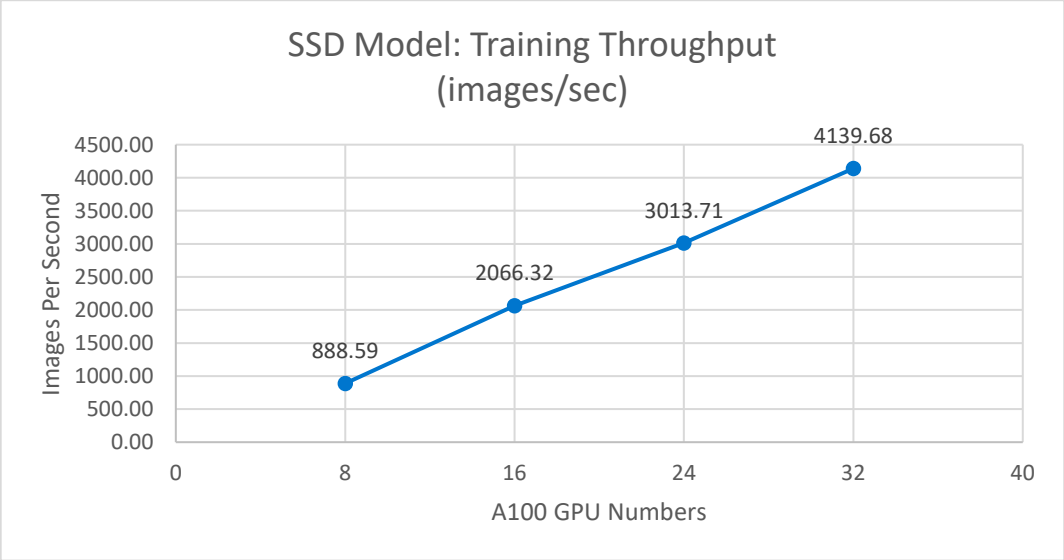

Figure 14. SSD model training benchmark from 8 GPUs to 32 GPUs

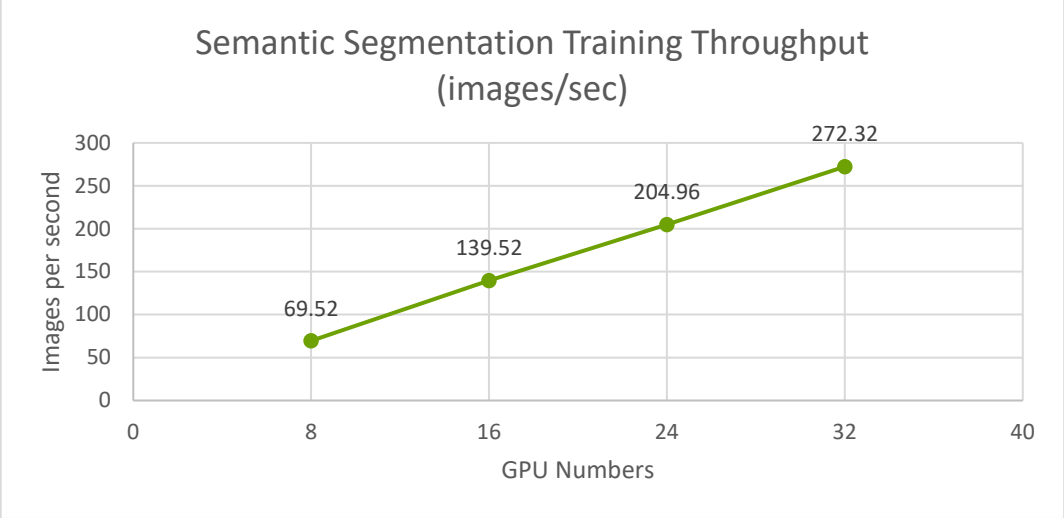

Figure 15. Semantic segmentation training benchmark from 8 GPUs to 32 GPUs

Training time and system metrics: SSD model

The following table shows the test results for the SSD model training time and bandwidth for each epoch. As shown in Figure 16, the average storage throughput grew almost linearly from 8 to 32 GPUs and reduced training time as well.

Table 6. SSD training time and storage bandwidth for each epoch with 2.8 TB dataset

GPUs	Average training time for each epoch	Average storage bandwidth per epoch	Peak storage bandwidth per epoch
8	18 minutes 15 seconds	2,080 MB/s	2,267 MB/s
16	7 minutes 51 seconds	2,350 MB/s	4,177 MB/s
24	5 minutes 23 seconds	2,860 MB/s	4,680 MB/s
32	3 minutes 55 seconds	3,420 MB/s	5,363 MB/s

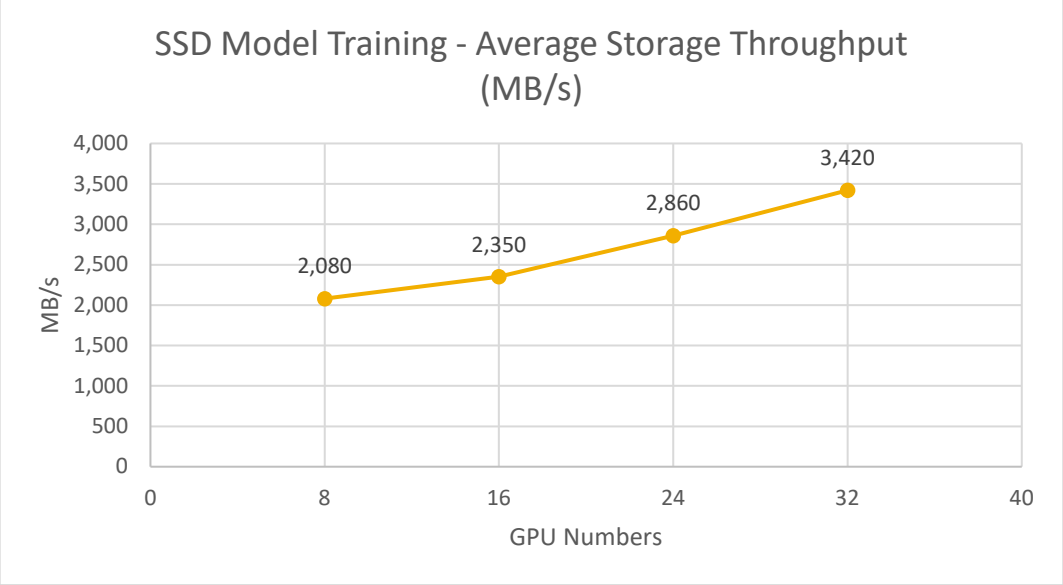

Figure 16. SSD model training average storage bandwidth from 8 GPUs to 32 GPUs

The following table lists the system metrics during the training. This indicates that the GPUs were fully utilized by leveraging the NVIDIA DALI library.

Table 7. SSD model training system metrics

System metrics	Percentage
Average GPU utilization	94%
Average GPU memory utilization	50%
Average compute node CPU utilization	35%
Average compute node memory utilization	99%

As shown in the following figure, during the training time, the peak storage throughput reached to 42.9Gb/s (5.36GB/s) with three DGX systems with 24 GPUs. We observed that the training is a heavily read intensive workload as it needed to read entire dataset from storage for training.

Figure 17. SSD model training storage throughput with three DGX A100 systems

We also observed that the storage throughput dropped after one epoch training. This was because the total dataset is only 2.8 TB. While running more than one epoch with three DGX systems (total 3 TB RAM), the system cache exceeded the total dataset. During the training, some of the data became cached on the system side, which reduced the storage throughput required during the training.

For storage configuration considerations, it is crucial to plan sizing for peak throughput on Isilon F800 storage during the training in order to minimize training times as required for business development needs.

During the training time, GPUs are fully utilized at 94%, as shown in the following figure.

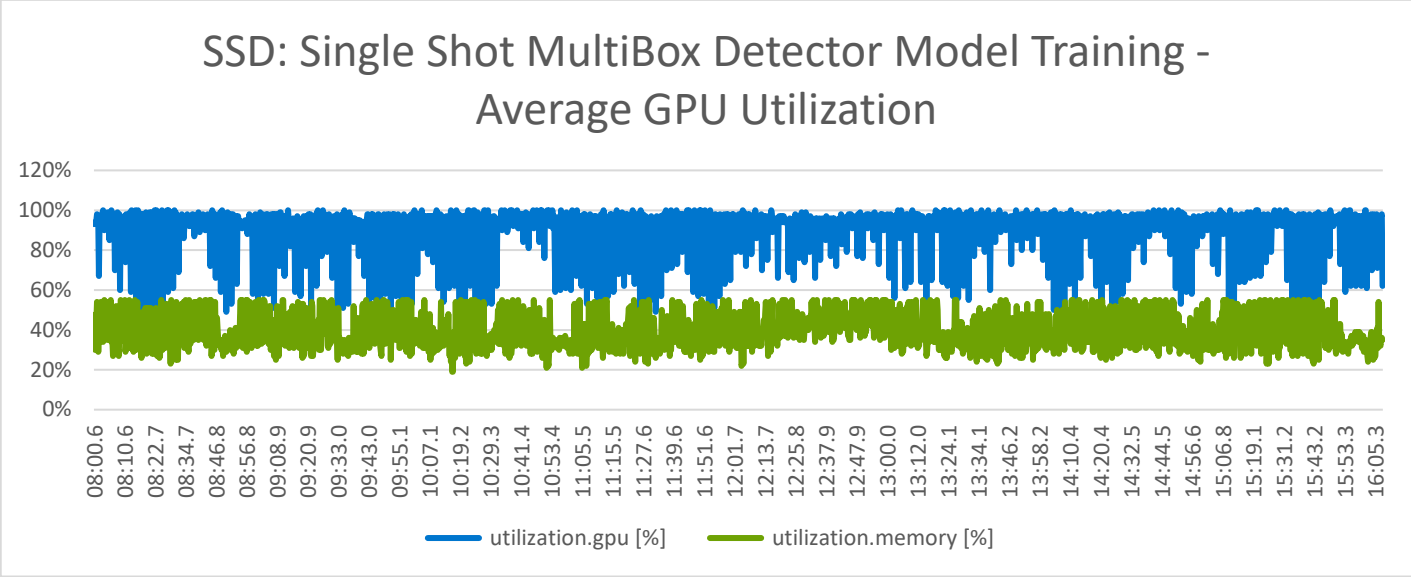

Figure 18. SSD model training with average 32 A100 GPU utilization

Training time and system metrics: Semantic segmentation model

The following table shows the test results for the semantic segmentation model training time and bandwidth. As shown in Figure 19, the average throughput scaled linearly from 8 to 32 GPUs and reduced training time as well.

Table 8. Semantic segmentation training time and storage bandwidth for each epoch with 5 TB dataset

GPUs	Training Time for each epoch	Average storage bandwidth for each epoch	Peak storage bandwidth for each epoch
8	8 hours 5 minutes	145.64 MB/s	138.43 MB/s
16	4 hours 2 minutes	303.75 MB/s	385.69 MB/s
24	2 hours 44 minutes	451.25 MB/s	570.32 MB/s
32	2 hours 4 minutes	550.34 MB/s	698.85 MB/s

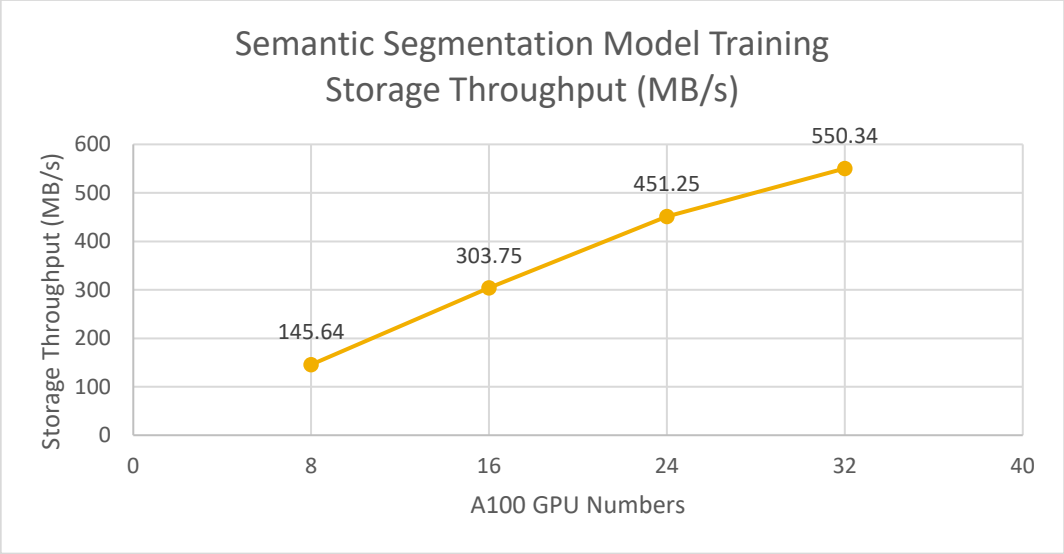

Figure 19. Semantic segmentation model training throughput from 8 GPUs to 32 GPUs

The following table lists the system metrics during the training. This indicates that the GPUs were fully utilized by leverage NVIDIA DALI library.

Table 9. DeepLabv3+ model training system metrics

System metrics	Percentage
Average GPU utilization	96%
Average GPU memory utilization	36.7%
Average compute node CPU utilization	9.86%
Average compute node memory utilization	98%

As shown in the next figure, during the training time, the average storage throughput reached to 4.4Gb/s (0.55 GB/s) and was read intensive. The storage throughput is

consistent during the training because the total dataset exceeded the total compute server cache.

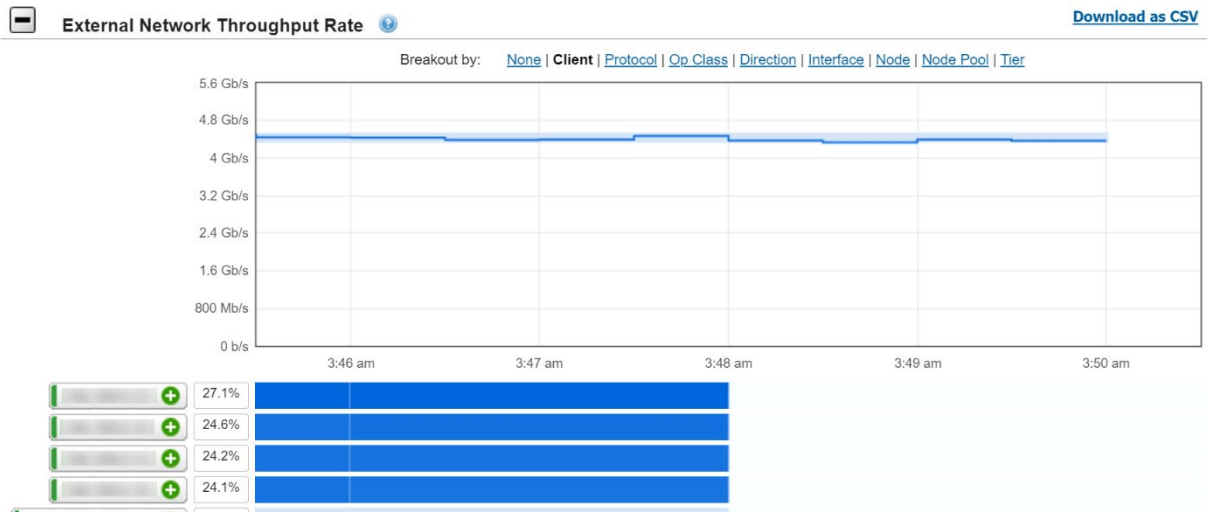

Figure 20. Semantic segmentation model training with 32 GPU storage throughput

Test results comparison

From our test results shown in the following figure, DGX A100 systems delivered over twice the training performance of an eight V100 GPU system, such as the NVIDIA DGX-1 system. The combination of the groundbreaking A100 GPUs with massive computing power and high-bandwidth access to large DRAM, and fast interconnect technologies, makes the DGX A100 system optimal for dramatically accelerating complex networks.

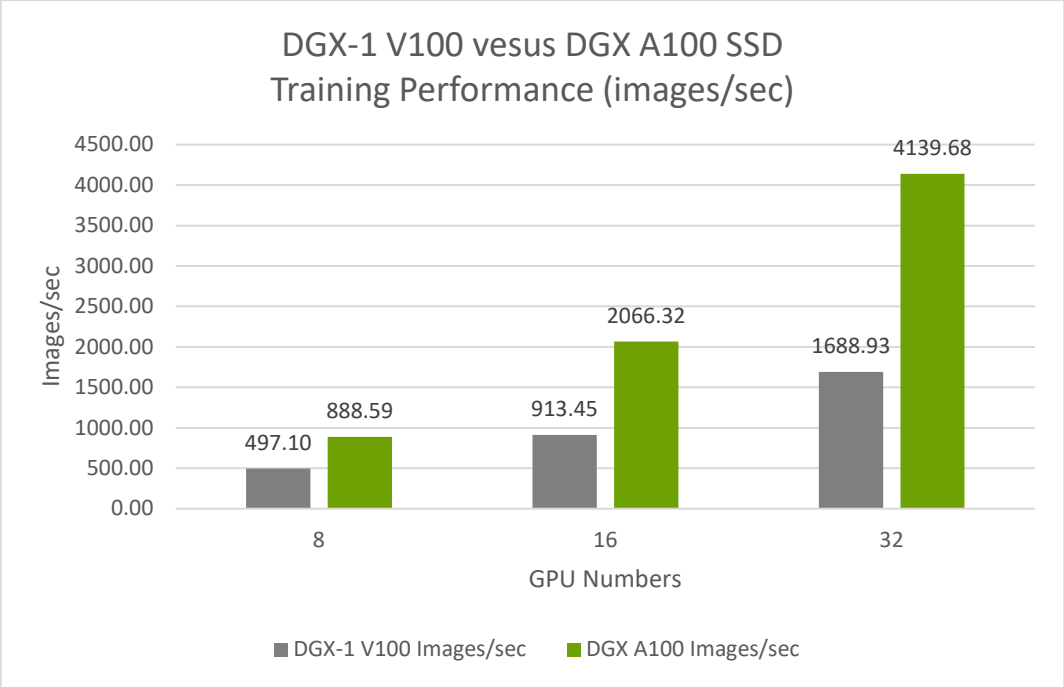

Figure 21. SSD model training performance comparisons

The DGX A100 system required higher storage throughput over V100 GPU system, as shown in the next figure. In our test results, we observed the average storage throughput dropped after testing with four DGX A100 systems. This was because the total dataset is only 2.8 TB. While running more with three DGX systems (total 3 TB RAM), the system cache exceeded the total dataset.

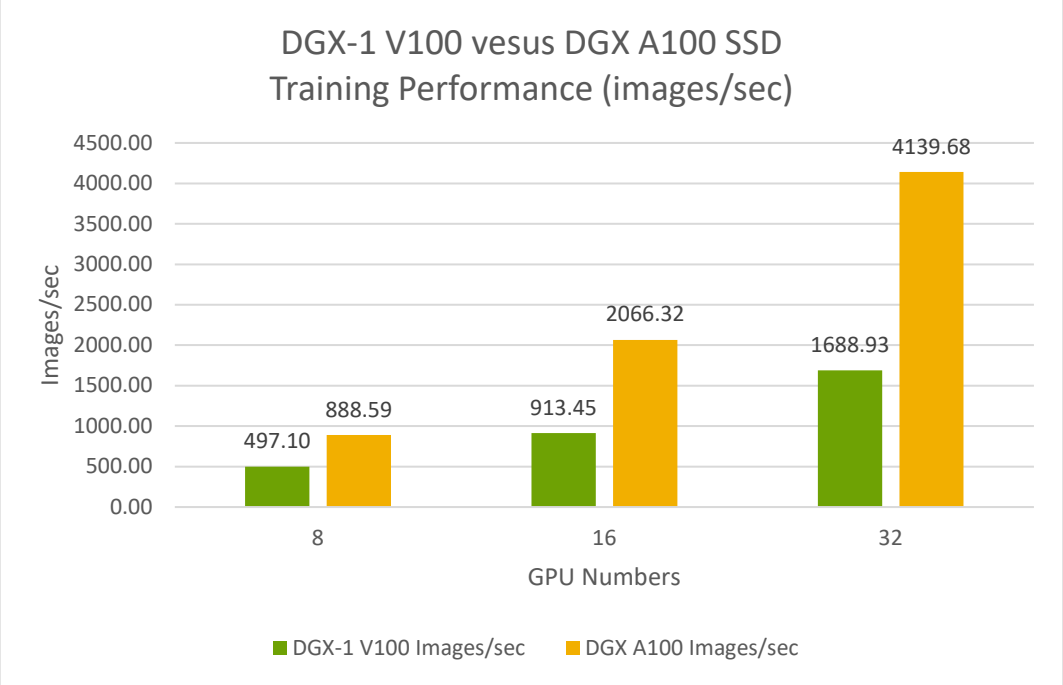

Figure 22. SSD model average storage throughput comparisons

Conclusion

DL training key considerations

Here are some key considerations for DL training that we observed during the tests:

- **Different CNN models generate varying throughput requirements**: Comparing the test results, different CNN models can greatly vary the performance requirements for storage bandwidth. Complex CNN models (like DeepLabv3+) may generate less storage throughput than simple CNN models (like SSD). A high-resolution dataset can generate higher storage bandwidth needs even for the same CNN models. Dell Technologies published a whitepaper Dell EMC PowerScale and NVIDIA DGX A100 Systems for Deep Learning which provides a sizing guideline for different CNN models.
- **Multiple GPUs and systems are particularly beneficial to large dataset training**: From our tests, we were able to observe the training time for average one epoch reduced from 18 minutes to 3 minutes by using 32 GPUs with 2.8 TB datasets. Multiple DGX A100 systems reduced the training time by 6x for large dataset DL model development.
- **Storage throughput requirement grows linearly with the increase of total GPUs during training:** From our test results, storage throughput grows linearly with the increase of GPU numbers during the model training. During the sizing for DL infrastructure, it is important to plan storage for future GPU growth as well.

- **Larger dataset can generate higher bandwidth requirements during training**: It is observed that a large dataset which exceeds the compute system cache will consistently generate higher bandwidth requirements. During the sizing, it is crucial for ADAS development to size your infrastructure with consideration of future data growth.
- **NVIDIA DALI allows the training to run at full speed:** GPU utilization is generally much higher than CPU utilization. In most of the training cases, the average GPU utilization can reach up to 97%. Low GPU usage could be caused by CPU bottlenecks. When the GPU utilization is low, the CPU is busy with data fetching from storage to memory or from working on small operations. DALI reduces latency and training time, mitigating bottlenecks, by overlapping training and pre-processing which maximizes the training speed. DALI is primarily designed to do preprocessing on a GPU, but most operations also have a fast CPU implementation.
- **Batch size should be large enough to reduce training time:** It is observed that larger batch size will decrease the training time with higher GPU utilization. The larger batch size will also increase the throughput, so eliminating storage bottlenecks is also required to accelerate the training time. Also, some research by Facebook has shown that in distributed learning, the learning rate scaling rule is surprisingly effective for a broad range of batch sizes. It is recommended to increase learning rate following batch size. For more information, refer to Facebook article Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour.

Summary

This document discussed key features of the scale-out NAS Dell EMC PowerScale as a powerful persistent storage solution for ADAS DL solutions. We presented a powerful and robust hardware architecture for DL by combining NVIDIA DGX A100 systems with embedded NVIDIA A100 GPUs and all-flash PowerScale storage. We ran several object detections and reported system performance based on the rate of images processed and throughput profile of I/O to disk. We also monitored and reported the CPU, GPU utilization and memory statistics that demonstrated that the system, GPU, and memory resources were fully utilized while the Dell EMC PowerScale was still capable of providing more I/O.

DL algorithms have a diverse set of requirements with various compute, memory, I/O, and disk capacity profiles. That said, the architecture and the performance data points presented in this whitepaper can be utilized as the starting point for building DL solutions tailored to varied set of resource requirements. More importantly, all the components of this architecture are linearly scalable and can be expanded to provide DL solutions that can manage tens to thousands of petabytes of data.

While the solution presented here provides several performance data points and speaks to the effectiveness of PowerScale in handling large scale DL workloads, there are several other operational benefits of persistent data for DL on PowerScale:

- The ability to run AI in-place on data using multi-protocol access.
- Enterprise grade features out-of-box.
- The ability to share data via tiering with other ADAS workloads (ex: HiL/SiL) without copying data to lower-performance, hybrid storage tiers (typical for HiL/SiL)
- Seamlessly tier to higher density nodes for cost-effective archiving while maintaining SLAs

In summary, PowerScale based DL solutions deliver the capacity, flexible performance, and high concurrency to eliminate the I/O storage bottlenecks for AI. The DGX A100 system delivers over twice the training performance of an eight V100 GPU system, such as the DGX-1 system using the same cityscapes dataset. This provides a solid foundation for large scale, enterprise-grade DL solutions with a future proof scale-out architecture that meets your AI needs of today and scales for the future.

Technical support and resources

Dell.com/support is focused on meeting customer needs with proven services and support.

Dell Technologies Automotive Landing page provide extended material to build-up an AD stack

Storage technical documents and videos provide expertise that helps to ensure customer success on Dell EMC storage platforms.

Additional resources include:

- Dell EMC PowerScale and NVIDIA DGX A100 Systems for Deep Learning
- Dell EMC PowerScale: Storage Solution for Autonomous Driving
- Dell EMC PowerScale for ADAS and Autonomous Driving
- Top 5 reasons to choose Dell EMC PowerScale for AD/ADAS
- Solving the storage conundrum in ADAS development and validation
- Cityscapes Public Dataset
- Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour
- NVIDIA SSD300 v1.1 For PyTorch
- NVIDIA Semantic-Segmentation
- NVIDIA DALI