Scaling Deep Learning Performance on the NVIDIA DGX-1 Server with WekaIO Matrix

WHITE PAPER

CONTENTS

List of Figures & Tables

Executive Summary

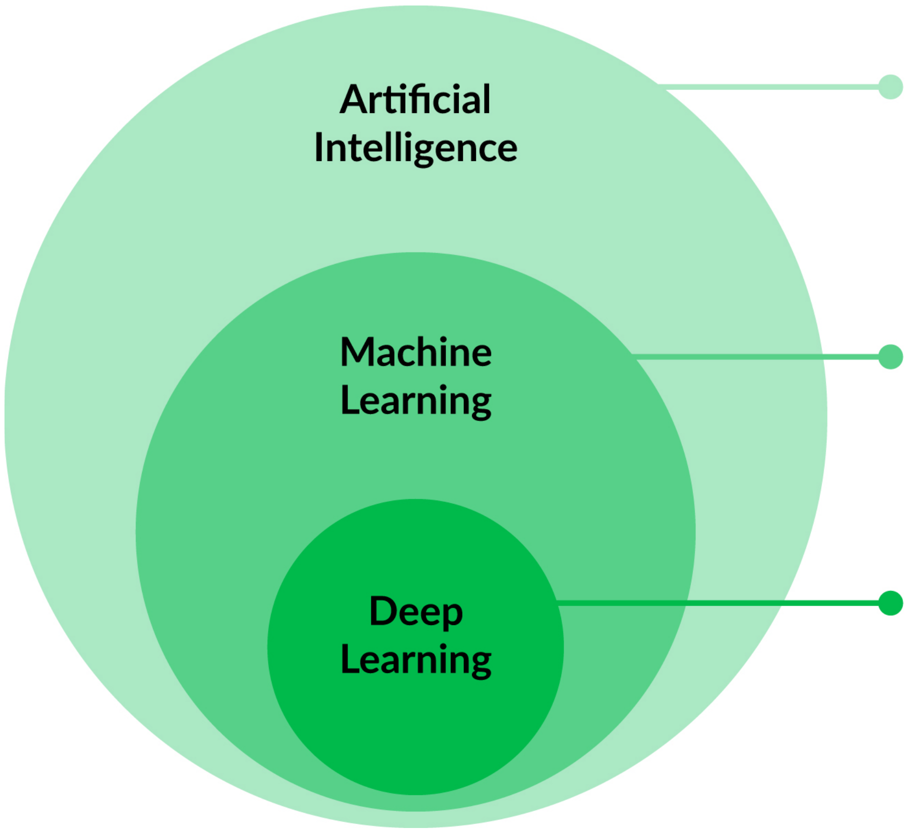
Deep Learning (DL) is an artificial intelligence (AI) breakthrough, using trained data models to solve problems beyond human levels of performance. Using DL to speed the likes of image recognition and fraud detection requires the processing of huge amounts of data at the lowest possible latencies. This paper addresses the relationship between training models and the high throughput, high bandwidth, low latency data access required to accelerate these new AI/DL insights at scale.

NVIDIA, Arista Networks, and WekaIO are innovators who have partnered their mutual technologies for AI/DL workloads. NVIDIA is the world leader in high-performance parallel computing on GPUs, accelerating DL workloads 10x beyond CPU architectures. Arista provides high bandwidth, low latency, dense datacenter switching as the common data fabric. WekaIO Matrix™ scales low latency storage performance for AI and high-performance computing (HPC) workloads. NVIDIA® DGX-1™ servers powered by eight NVIDIA Tesla® V100 Tensor Core GPUs are fully utilized by shared WekaIO Matrix data storage over Arista switches. Thus, the most demanding DL models can learn and infer better and faster, providing new insights in less time for greater business agility.

Intended Audience

IT managers, solutions architects and data scientists concerned with building a scalable infrastructure for modern AI and DL in the data center.

What is Deep Learning



Artificial Intelligence (AI)
Enabling machines to mimic human intelligence

Machine Learning (ML)
Subset of AI including statistical techniques enabling machines to improve at tasks with experience

Deep Learning (DL)
Subset of ML based on exposing multilayered neural networks to example data

Figure 1 – Artificial Intelligence Definitions

DL is a recent revolution in machine learning (ML) methods based on multiple layers of knowledge representation - usually artificial neural networks (NN) of various types — operating in a cascade. Figure 2 illustrates a high-level view of a typical deep neural network (DNN), showing the large number of levels and how they feed each other.

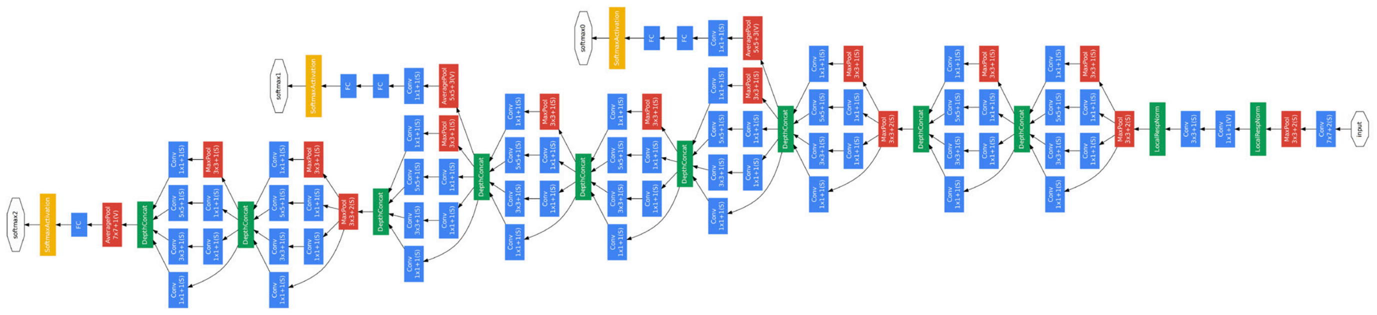


Figure 2 – GoogLeNet from Christian Szegedy et al., Going Deeper with Convolutions (2014)

It has been well represented in academia that DL model accuracy is highly correlated to the amount of data provided to the model and DL model accuracy improves as it is exposed to more training data (Figure 3). Only recently has technology permitted enough rapid data flow to get error rates low enough for practical use. While larger datasets increase accuracy, they also increase the demand for computation and storage performance. Multiplying the number of servers greatly increases performance but only if all have adequate access to data. A shared parallel file system is required to scale to these levels - sharing ensures data consistency between machines and parallelism ensures consistent performance across the machines.

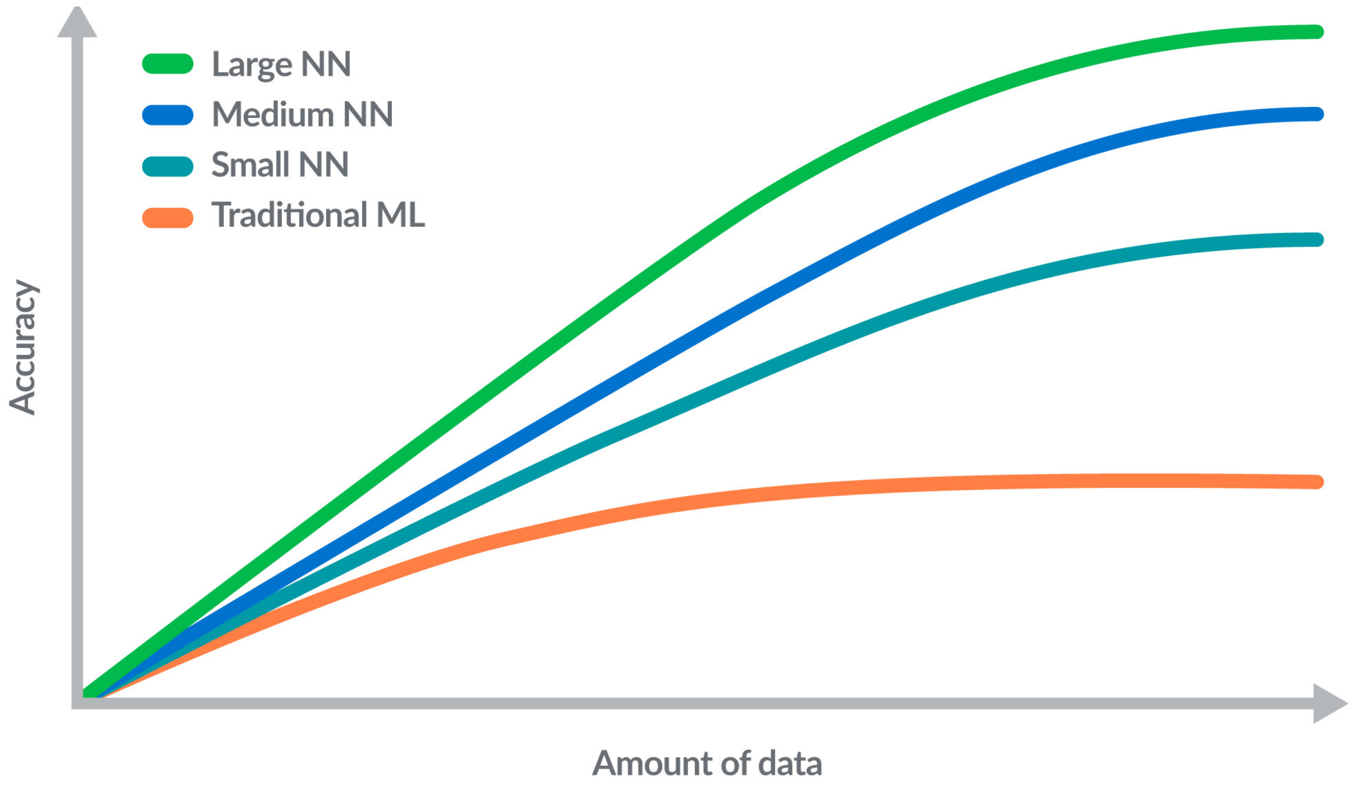


Figure 3 – Accuracy of Models Grows with the Amount of Training Data (Andrew Ng)

The success of DL is partially algorithmic, but the largest gains occurred with the introduction of NVIDIA GPUs to train models. Today's most advanced CPUs run at high speeds, but are capable of only a few threads of parallel execution on their cores. A GPU by comparison, has thousands of threads of parallel execution - multiple orders of magnitude greater than CPUs. Originally focused on visualization applications, NVIDIA has since transformed the GPU into powerful parallel floating-point processors capable of millions of floating-point operations per second, especially suitable to speed the training and processing of DL models.

Deep Learning Data Pipeline

As Figure 4 shows, DL pipelines consist of two overlapping paths, one for model training and the other for production inference. Training is the most computationally intensive path, involving many computations to train a model from specified input data. Once a model is considered trained, the production inference path uses ingested normalized input data to generate a conclusion, a much lighter computational requirement than the iterative back-propagation of training.

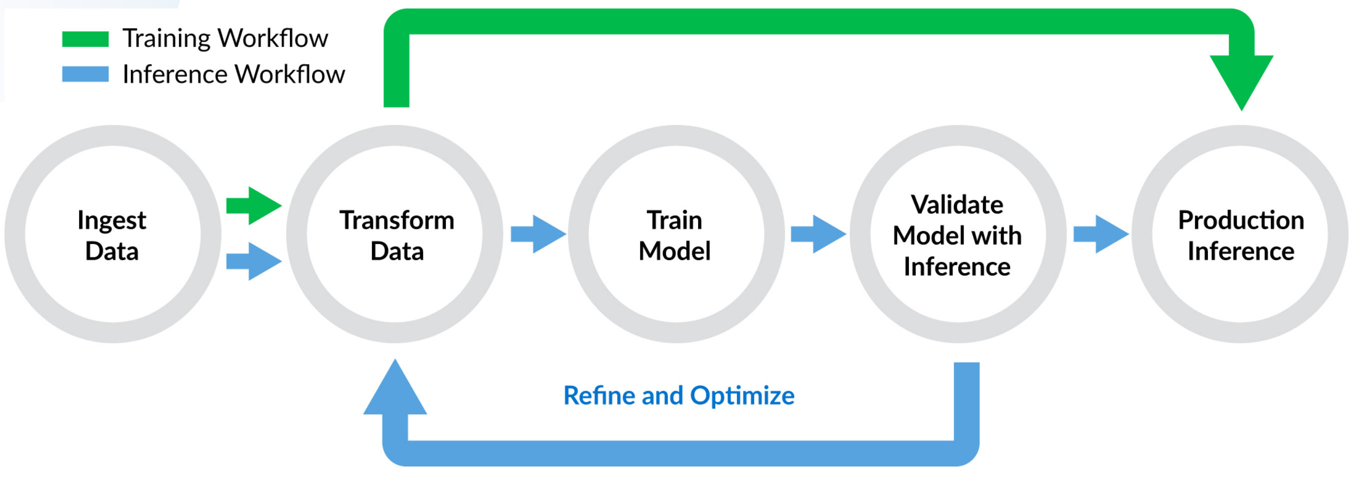


Figure 4 –Deep Learning Data Pipeline

Today's DL applications include complex, multi-stage pre-processing data pipelines with compute-intensive steps mainly carried out on the CPU.

1. **Ingest Data** into the DL pipeline. An example would be images and associated metadata. I/O usually consists of small file sequential reads often followed with large file sequential writes that are aggregated for increased throughput.

2. **Transform Data** to clean and store input data in a normalized form. For image data, this can consist of image decoding and various conversions in size and/or color, often performance is limited by server host CPUs. The NVIDIA Data Loading Library (NVIDIA DALI™)[1] architecture is a collection of highly optimized building blocks and an execution engine to accelerate input data pre-processing for DL applications. The NVIDIA DALI architecture can offload much of this work onto GPUs eliminating this bottleneck. To fully realize the benefit of this improvement, the underlying I/O storage must perform at a level to keep the pipeline saturated. Typically, I/O for this part of the pipeline is small file sequential writes.

3. **Train Model** by passing each dataset member through the model. Each data has the effect of adjusting parameters at each layer, contributing to a future conclusion. Supervised training is the most common method, with each input accompanied by a desired output value. Unsupervised training examples lack a provided output value, requiring the model to organize the data based on common characteristics e.g. grouping similar items into a cluster.

4. **Validate Model** using input data that the model hasn't yet seen. Inference-based on this data helps to ensure accuracy and avoid false positives. High error rates indicate the need for further training. I/O in this stage can be characterized as large file sequential reads.

5. **Production Inference** occurs when a trained and validated model is used in a production environment. Input data is often aggregated into larger files containing many items reduced to short tensor values to maximize throughput.

Storage Challenges for Deep Learning at Scale

Application workflows commonly scale across a server cluster rather than scaling up within a single large system. This is especially true for AI and HPC workloads, allowing ever more CPU and GPU resources to be leveraged. In large scale DL environments, a dataset can scale to tens of petabytes of data that needs to be shared across a GPU cluster.

[1]https://docs.nvidia.com/deeplearning/sdk/dali-developer-guide/docs/index.html

Server local storage is limited in available capacity and makes cluster-wide consistency a real problem. If a complete copy of the data will not fit in available local storage, smaller chunks will need to be copied to each server as needed.

The solution is a shared file system, maintaining all stored data in a single, large and expandable namespace. Attached GPU servers can simply use a single shared version of data at high speed without the complexity and overhead of local storage management. All nodes in the cluster see the same directories and files in the same way.

For a shared storage system to be an effective data repository for AI/ML workflows, it must be able to support all I/O demands – sequential and random access patterns, large and small files, and metadata. Existing shared storage systems - often referred to as Network Attached Storage (NAS) - use the traditional Network File System (NFS) or legacy parallel file systems (e.g. Lustre or GPFS) designed for limited specialized applications. Figure 5 depicts the difference between traditional NAS systems and parallel filesystems, especially WekaIO Matrix.

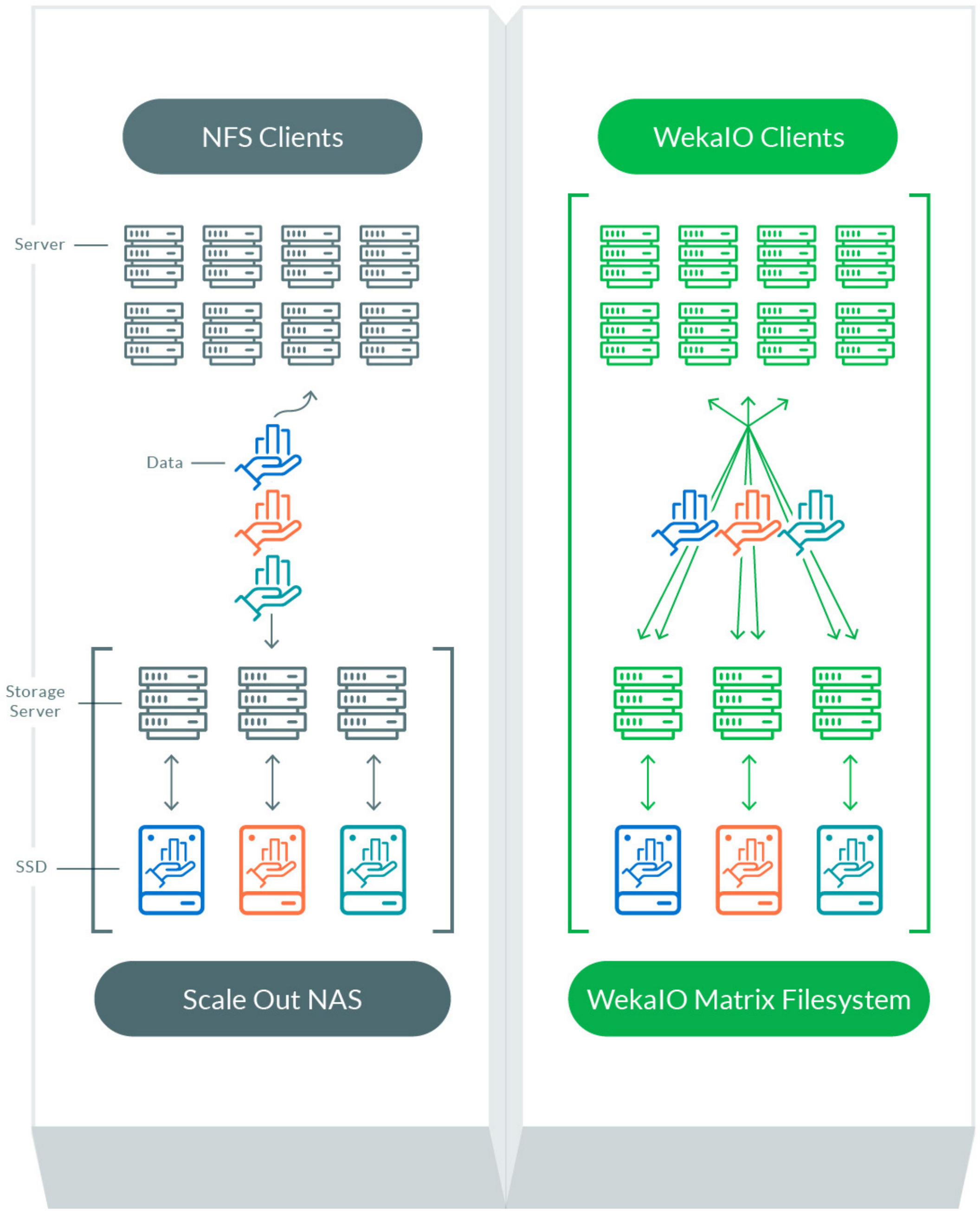


Figure 5 – Comparison of WekaIO Matrix Parallelism to Scale Out NAS

While NFS is ubiquitous and has been used widely for decades, its performance is limited by its serial nature. NFS requires that both data and request information reside in the same message, adding processing overhead and latency. It also limits the amount of data that can be transferred in a given message packet. When NFS was designed in the 1980s for user group file sharing over 10 Mbit Ethernet, it provided acceptable performance for that use case.

Legacy parallel file systems were designed to support a small number of high bandwidth applications doing large sequential I/O with very little metadata. These systems run data services separate from metadata services, often with dedicated infrastructure for each. They are difficult to manage in modern environments and often perform poorly because metadata can account for up to 65% of I/O activity, in most cases more than what the system was architected to handle.

To overcome these issues, Matrix breaks up files and dynamically distributes both data and metadata across all NVMe devices in a storage cluster and presents a fully POSIX-compliant parallel file system to the GPUs. File data and metadata can now be accessed in parallel. The resulting performance is vastly superior to NFS storage, and metadata operations are far faster than legacy parallel filesystems. As illustrated in Figure 5, a single Matrix client connects to any or all Matrix storage servers in parallel, limited only by available network connectivity. Each Matrix server is extremely fast on its own using NVMe storage, but their summed performance is available to any client without individual connection or server limits. In contrast, clients access NAS systems through a single connection, creating an unavoidable bottleneck and greatly limiting data access.

Deep Learning Infrastructure

Once a DL workflow is understood, implementation choices must be made. How much GPU horsepower is required to attain the required performance? Can this reside in a single server or are multiple servers required for performance and/or required application redundancy? The computational load determines the required storage capacity, bandwidth, and latency. Finally, the interconnect between compute, storage and model inputs/users must ensure ample bandwidth for all. If data flow is impeded by incorrect choices, the power of the GPUs and its computational efficiency will be wasted.

Compute: NVIDIA DGX-1 Server

NVIDIA DGX servers are fully-integrated supercomputer building blocks purpose-built for AI and HPC applications. DGX-1 delivers the equivalent computing performance of hundreds of traditional CPU servers. A DGX-1 server (Figure 6) contains eight Tesla V100 GPUs, connected in a hybrid mesh topology using NVIDIA NVLink™ technology. NVLink technology enables direct high-bandwidth, low latency communications between GPUs within the server, avoiding bottlenecks caused by the server PCIe bus. The DGX-1 server also includes four 100 Gbps network interconnects facilitating outstanding connectivity to storage.

Figure 6 – NVIDIA DGX-1 Server

NVIDIA DGX servers are configured with the DGX software stack including the server operating system and NVIDIA-optimized DL containers for maximum GPU-accelerated performance. These containers made available through the NVIDIA GPU Cloud (NGC) offers pre-optimized versions of popular DL frameworks including TensorFlow™, PyTorch and Caffe2. Each container has everything necessary to run supported applications in a highly performant yet portable fashion without requiring users to build their own low-level development environments.

Networking: Arista 7368X Data Center Switch

The Arista 7368X is a modular high performance 40/100/400 GbE data center switch, specifically intended for next generation applications such as AI/ML. Most importantly for AI/ML is the ability of the switch to extend the high bandwidth of intra-GPU communications between GPU servers and storage with a high radix non-blocking architecture. High-radix interconnection networks offer significantly better cost/performance and lower latency than conventional (low-radix). Not only does the Arista 7368X switch offer 12.8 Tbps of throughput, but the consistent low 700ns of latency across all ports further empowers multi-server environments and shared storage.

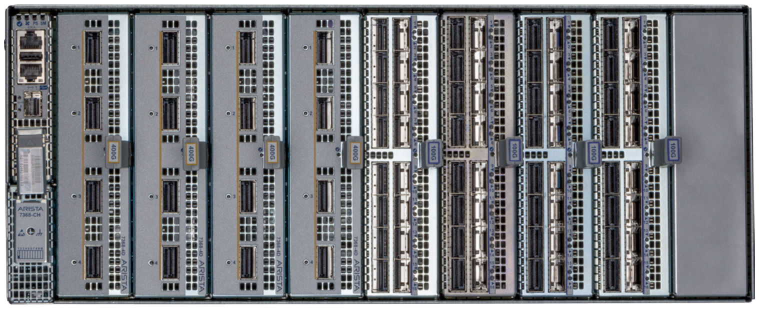


Figure 7 – Arista 7368X Data Center Switch

In addition to both high performance and port density, the Arista 7368X switch provides configurable classes of forwarding with an integrated packet buffer and advanced traffic management and congestion control along with network monitoring and telemetry features. This flexibility allows full customization for AI/DL as well as HPC environments. For this benchmark, the Arista 7368X switch has been configured with 128 100GbE ports, ample for servers and storage to leverage multiple high bandwidth connections.

Storage: WekaIO Matrix Deployed on Standard x86 Servers

WekaIO Matrix, the world's fastest and most scalable filesystem[2], is designed to transcend the limitations of local storage, traditional NFS storage, and block storage using a legacy parallel file system, making it ideal for data-intensive AI and HPC workloads. Matrix is a clean-sheet design integrating NVMe-based flash storage, standard X86 compute servers, object storage and low latency interconnect fabrics such as InfiniBand or 100 GbE into an NVMe over-Fabrics architecture, creating an extremely high-performance scale-out storage system. Matrix performance scales linearly as more servers are added to the cluster.

In addition to parallel access, Matrix also supports all the common file access protocols including NFS, SMB and REST for maximum compatibility and interoperability. Hadoop and Spark environments also benefit from the performance of a shared file system through a fully integrated connector that allows Matrix to replace HDFS and function as a single, easy to manage data lake for all forms of analytics (Figure 8).

Data protection is achieved using a resilient, space efficient distributed data protection schema whereby reliability increases as the number of storage nodes grows. Matrix also ensures data integrity through an end-to-end checksum. Integrated tiering to multiple S3 targets enables the cost-effective data lifecycle management for older or less used training data in addition to providing cloud bursting capabilities for organizations that require on-demand GPU resources.

[2]WekaIO Matrix has scored first place in the IO-500 ten node challenge [https://www.vi4io.org/io500/list/19-01/10node], a high-performance computing benchmark that measures bandwidth and metadata. It has also scored first place in the suite of SPEC 2014 [http://spec.org/sfs2014/results/] file system benchmarks that measure file performance and latency.

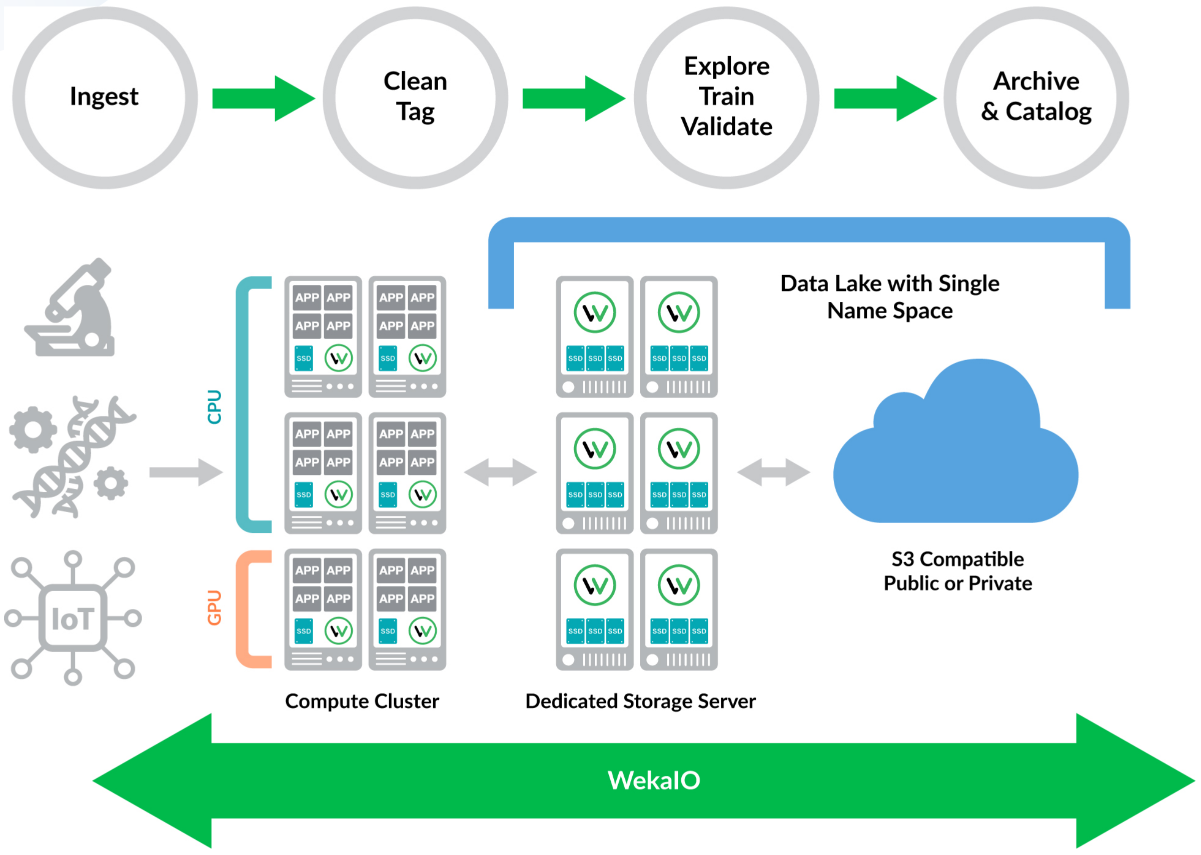


Figure 8 – WekaIO Matrix as a Data Lake for AI/DL Datasets

WekaIO can be deployed on any standard x86 based server from commercial server vendors. The following table provides a recent list of servers that WekaIO has tested and qualified.

Vendor	Platform
Dell	R640
HPE	Apollo 2000, DL360
Penguin Computing	Relion
Supermicro	BigTwin, 1029U

Table 1 – List of x86 Platform Options

For this benchmark, Matrix is running on the WekaIO Supermicro reference architecture [https://www.weka.io/wp-content/uploads/2018/05/Supermicro-Reference-Architecture.pdf]. This is a storage cluster based on two server chassis, with each chassis containing four nodes. Each node contains a CPU, NVMe storage and high-bandwidth networking. The exact configuration is detailed in the Benchmark Architecture section.

Figure 9 – WekaIO Matrix on Eight Node Server System – in 4U Form Factor

Evaluating Deep Learning Performance

Benchmarking DL requires the construction of an environment that closely corresponds to those used in DL research and production. It is necessary to pick a base DL software framework, a standardized (or at least commonly available) dataset, well-known models, and typical training and production workflows.

TensorFlow is an open-source DL software framework that originated at Google and is widely used for both research and production. TensorFlow is currently the most popular DL framework and has full NGC container support. Many of the DL benchmarks accepted as standard are based on TensorFlow (see https://www.tensorflow.org/guide/performance/benchmarks).

ImageNet is a large collection of hand-annotated images designed for use in visual object recognition software research. ImageNet contains more than 20,000 categories with an annotation database that is freely available from ImageNet. An annual ImageNet challenge has encouraged the development of new efficient models that can correctly classify and detect objects and scenes in standardized data.

Why are batch sizes important in AI for Training?

Batch size is an important adjustable hyper-parameter of the model training that sets the number of training examples utilized in one iteration. The effects of batch size are a bit of an open-ended question in data science and selecting the optimal batch size is not a simple process. It is generally best to start with standard default benchmark batch sizes like 128, 256, and 1024 and tune appropriately.

Tuning batch size is impacted by both hardware (e.g. available memory) and workload implementation (e.g. precision, memory management). The amount of GPU memory available is also an important factor, making it easier to measure per iteration the impact the hardware will have on the overall performance.

A larger batch size can improve the effectiveness of the optimization steps resulting in more rapid convergence of the model parameters. Larger batch sizes may also improve performance by reducing communication overhead caused by moving the training data to the GPU. This causes more compute cycles to run on the GPU with each iteration. All of this depends on other hyper-parameters and the characteristics of the dataset (large or small) being used as well as the network model and computational framework.

Larger batches generally enable more efficient use of GPU resources. For our tests, we achieved optimal results using the following batch sizes:

Benchmark Model: Training	Batch Size
AlexNet	1024
GoogLeNet (aka Inception)	1024
Inception v2	416
Inception v3	256
Inception v4	320
ResNet-50	256
ResNet-152	256

Benchmark Model: Inference	Batch Size
AlexNet	1024
GoogLeNet (aka Inception)	256
Inception v2	1024
Inception v3	256
Inception v4	256
ResNet-50	256
ResNet-152	256

Table 2 – List of Benchmark Batch Sizes

Benchmark Architecture

The system configuration used in this paper for the benchmarks consisted of the following:

- A nine-node cluster of NVIDIA DGX-1 servers, connected via four 100GbE ports
- One Arista 7368X Data Center Switch with 128 100 GbE ports
- One WekaIO Matrix reference architecture consisting of
    - WekaIO Matrix version 3.2.2
    - Two Supermicro BigTwin chassis, each containing four nodes
        - Each node equipped with six Micron 9200 NVMe drives
        - Each node connected via two 100 GbE ports

The software configuration consists of the tensorflow:19.01-py3 container from NGC.

Benchmarking across multiple servers was coordinated with Horovod, an open-source Python package originally developed by Uber[3]. Distributed TensorFlow processes use Horovod to manage communications between them.

[3]Sergeev, A., Del Balso, M. (2018) Horovod: fast and easy distributed deep learning in TensorFlow. arXiv:1802.05799

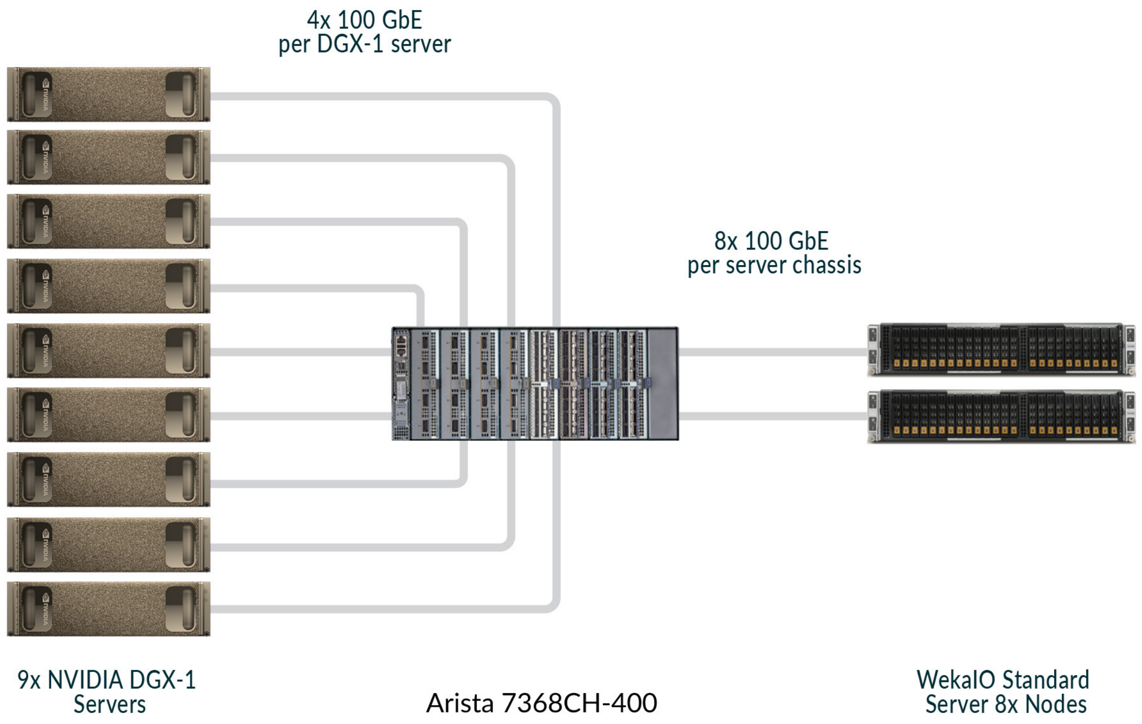


Figure 10 – Benchmark Hardware Architecture

Benchmark Model	Reference Paper
AlexNet	https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
GoogLeNet (aka Inception)	https://arxiv.org/pdf/1409.4842v1.pdf
Inception v2	https://arxiv.org/abs/1512.00567
Inception v3	https://arxiv.org/abs/1512.00567
Inception v4	https://arxiv.org/pdf/1602.07261.pdf
ResNet-50	https://arxiv.org/abs/1512.03385
ResNet-152	https://arxiv.org/abs/1512.03385

Table 3 – List of Benchmarks Conducted

An initial test was conducted on a single DGX-1 server to demonstrate scaling from one to eight GPUs in a single server environment.

The second phase of the testing measured scaling across multiple DGX-1 servers, scaling from one to nine servers (from eight to 72 GPUs).

Benchmark Results – Training

Performance Scaling across GPUs – Single DGX-1 Server

Six common DL models were used for the single server training benchmark (Figure 11). Each of these were run in an NVIDIA NGC TensorFlow container on a single DGX-1 server connected as shown in Figure 10. As a single DGX-1 server, however powerful, is only a small load for this WekaIO Matrix storage configuration, all training occurs at full GPU performance levels. These results scale linearly, showing that the server and its connected storage operates at full utilization.

Benchmark Training Results – ResNet-50, ResNet-152, Inception-V3, Inception-V4

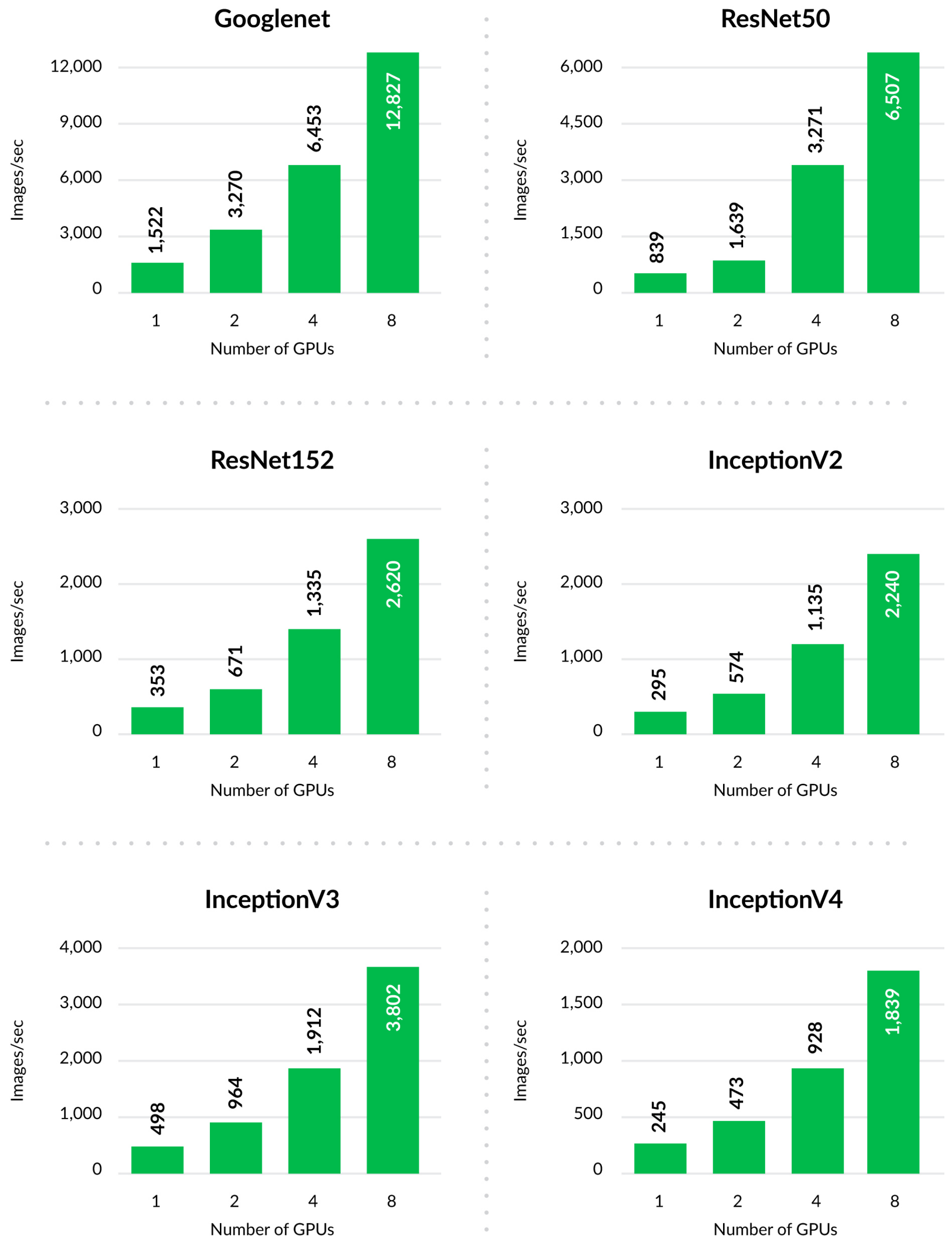


Figure 11 – DL Training Results - Single DGX-1 Server

Performance Scaling across GPUs – Nine DGX-1 Servers

Having established linear scaling across GPUs in a single DGX-1 server, the second phase of testing used a similar set of DL models to investigate performance scaling across a nine DGX-1 servers - 72 GPUs in total (Figure 12). As per the single server tests, each server ran these benchmarks from an NVIDIA NGC TensorFlow container. Linear scaling is often difficult to produce due to the complexities and interactions of data flows between GPUs, CPUs and servers. The multiple server benchmarks were performed without the DALI architecture, thus magnifying the impact of preprocessing the input data. Even so, the observed performance is near linear and increases as the number of servers grow due to the effect of parallelism, demonstrating that even the small WekaIO Matrix configuration used can keep nine DGX-1 servers saturated with data for maximum server and GPU utilization.

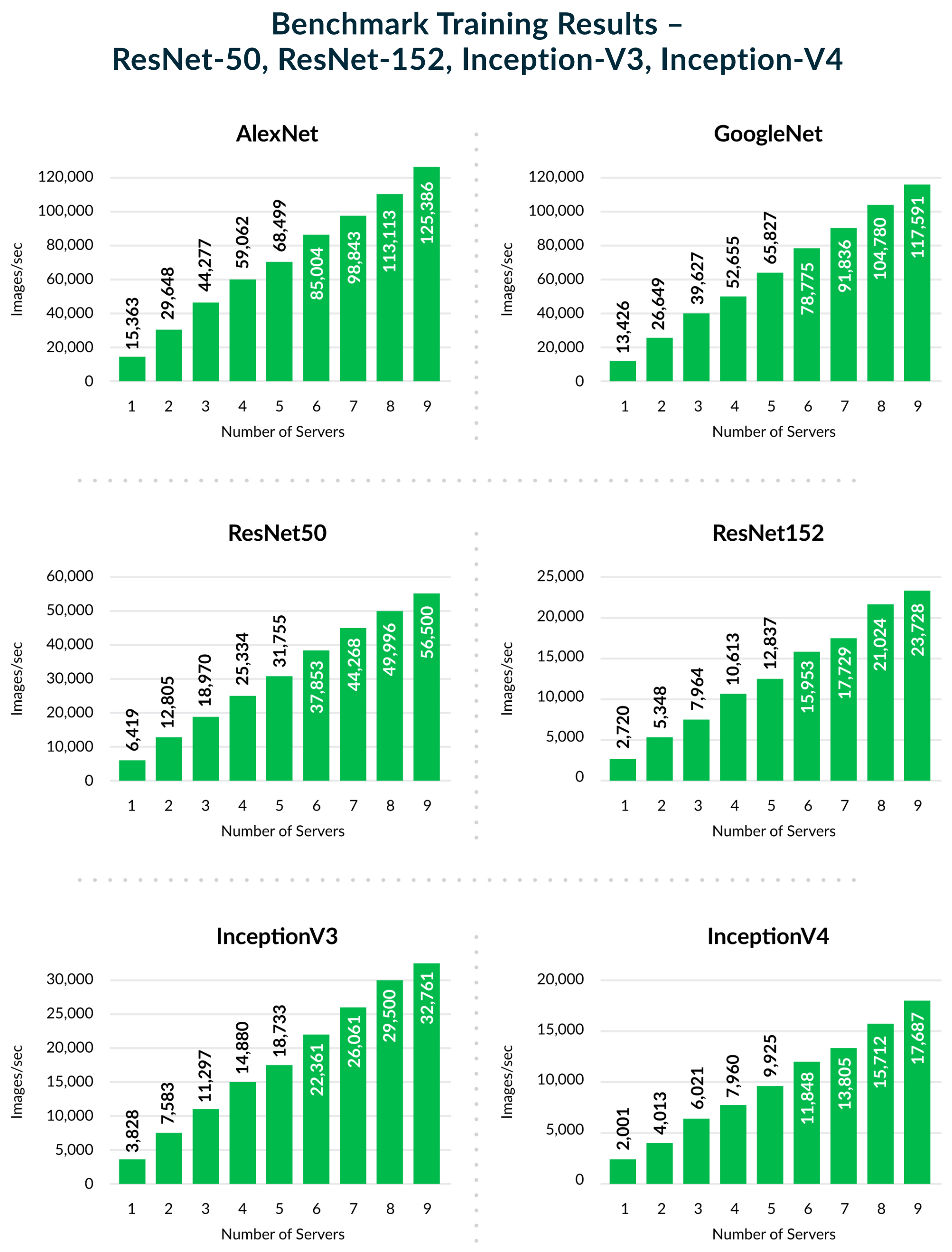


Figure 12 – DL Training Results - Multiple DGX-1 Servers

Benchmark Results - Inference

A configuration identical to that used for multi-server training was used to assess the impact of scaling production inference. Performing only inference requires significantly less computation from each server and far more IO, yet WekaIO Matrix continues to deliver linear scaling (Figure 13).

The conclusion from all benchmarks is that a small WekaIO configuration provides the storage throughput enabling nine DGX-1 servers to use their full capabilities. Whether training or inference across all servers and GPUs, performance scales linearly. As the DL cluster grows, WekaIO Matrix can easily grow to support it in size and bandwidth, simply by adding more Matrix nodes.

Benchmark Training Results – ResNet-50, ResNet-152, Inception-V3, Inception-V4

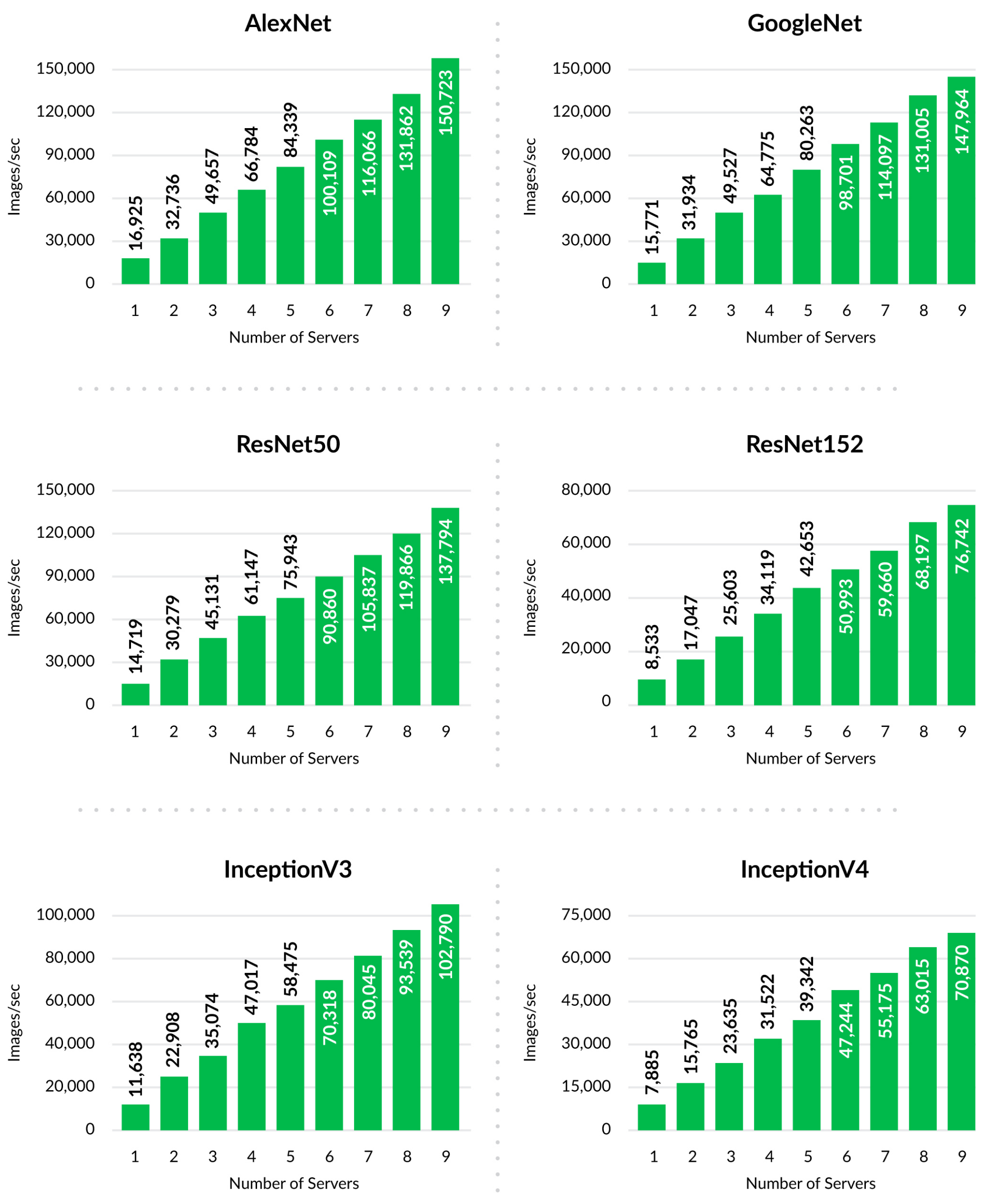


Figure 13 – DL Inference Results - Multiple DGX-1 Servers

High Performance Computing with WekaIO Matrix

High-Performance Computing (HPC) has traditionally been used to solve the largest numerically intensive problems in science and engineering. Current supercomputers use myriad compute nodes, harnessed by a low latency interconnect to solve problems at much larger scale than any single node can handle. GPUs now provide critically important performance boosts for this class of system due to their great parallelism. Infrastructures for AI/DL and HPC increasingly overlap, enabling new converged models of computation - large numerically computed datasets refined, selected and optimized with DL. Convergence brings the need for infrastructure that provides increased flexibility along with increased operational efficiency and simplicity because most IT organizations have limited experience architecting, managing and maintaining HPC systems.

Modern HPC systems must accommodate not only large sequential I/O, but random I/O workloads as well. These systems support a variety of applications and users, even more so with the addition of AI/DL. A modern HPC storage system must be a master of all trades so Matrix has been designed for this and similar high performance, high flexibility workflows.

Transcending previously discussed issues with legacy storage, WekaIO Matrix combines:

- Software-based configuration simplicity and variety of deployment choices
- High bandwidth, low latency data and metadata
- Efficient parallel access, allowing performance to scale with more WekaIO servers
- Full interoperability via traditional NFS and SMB
- Integrated automatic tiering to any S3 compatible object storage for improved efficiency and ROI
- Support for hybrid and cloud-based DL implementations for peak-needs bursting and disaster recovery
- A single namespace that servers as a common data lake for all analytic applications

The Standard Performance Evaluation Corporation (SPEC) created SPEC SFS® 2014 as a standard benchmark for comparing storage performance across different vendor platforms. This benchmark consists of several workload types including large software builds, transactional database, VDI and electronic design automation (EDA). Published results from February 2019 (https://www.spec.org/sfs2014/results/sfs2014.html) show Matrix is the clear front runner on performance and on latency across all benchmarks.

IO-500 is a new benchmark created by the Virtual Institute for I/O (VI4IO) as a representative, scalable, portable and trustworthy measure of bandwidth and latency for a variety of HPC workloads. See https://www.vi4io.org/io500/about/start for further details on IO-500.

By limiting IO-500 to 10 client nodes, this benchmark challenges single client performance from a storage system. WekaIO Matrix - ½ cabinet with eight chassis totaling 32 storage nodes - demonstrated performance for 10 clients 31% faster than 40 cabinets, 154 storage nodes from the world's fastest supercomputer. Even with 12x more RAM and 3.2x more NVMe, the supercomputer could only achieve second place to Matrix. Figure 14 shows the January 2019 revised results https://www.vi4io.org/io500/list/19-01/10node.

Conclusions

Powerful GPUs, storage and networking together enable the effective use of DL at scale and speed. Medical image interpretation, fraud detection, autonomous vehicles and natural language processing are only the beginning of the value that DL can provide.

This paper has examined the performance of a DL infrastructure based on NVIDIA DGX-1 servers, Arista networking, and WekaIO Matrix reference implementation storage.

Our results show that performance scales linearly from one to eight Tesla V100 GPUs within a single DGX-1 server; performance also scales linearly from one to nine DGX-1 servers, all sharing the same Matrix storage.

In many machine learning deployments, each DGX-1 server is dependent on its own local storage to serve the training models, requiring elaborate data staging in order to scale. Using WekaIO Matrix shared storage allows multiple DGX-1 servers to scale transparently without storage bottlenecks up to the limits of Matrix itself. The unique value of WekaIO Matrix is to provide the convenience of centrally managed shared storage in the form of a data lake, the scalability of a supercomputer, and performance in excess of even NVMe local storage.

Using several image classification models to benchmark this architecture, it is evident that training is more compute than I/O bound, allowing many DGX-1 servers to be served by a small WekaIO Matrix cluster. Inference, being much less computationally intensive, can utilize much more storage bandwidth depending on the required response time. Matrix scales independently of compute resources for greater flexibility, allowing storage nodes to be tailored for the exact environment, whether dedicated training, production or a mix, including HPC.

Areas for further testing of WekaIO Matrix include multi-node implementation of DALI architecture using the NVIDIA TensorRT™ engine. Briefly, the TensorRT engine rewrites parts of the execution graph to allow for faster prediction times. The TensorRT engine and TensorFlow are tightly integrated to combine the flexibility of TensorFlow with the powerful optimizations of the TensorRT engine.

WekaIO Matrix has shown its ability to deliver high storage performance to a GPU cluster consisting of high-performance servers. As DL training datasets increase in size and models increase in complexity, each training session will require more resources. Matrix has the power and flexibility to keep current GPU clusters free of I/O bottlenecks and to easily scale to accommodate future needs. While beyond the scope of this paper, Matrix also supports cloud GPUs for organizations that run in the public cloud or wish to utilize on-demand resources for peak computation periods.

Additional Resources

NVIDIA DALI - GPU Technology Conference 2018 presentation about DALI, T. Gale, S. Layton and P. Tredak [slides: http://on-demand.gputechconf.com/gtc/2018/presentation/s8906-fast-data-pipelines-for-deep-learning-training.pdf; recording: http://on-demand.gputechconf.com/gtc/2018/video/S8906/]

NVIDIA TensorRT - GPU Technology Conference 2018 presentation about accelerating TensorFlow Inference with New TensorRT Integration, J. Bernauer [slides: http://on-demand.gputechconf.com/gtc/2018/presentation/s81009-accelerate-tensorflow-inference-with-new-tensorrt-integration.pdf; recording: http://on-demand.gputechconf.com/gtc/2018/video/S81009/]

WEKA.IO

2001 Gateway Place, Suite 400W, San Jose, CA 95110 USA T 408.335.0085 E info@weka.io www.weka.io