White Paper

NVIDIA DGX-1 With Tesla V100 System Architecture

The Fastest Platform for Deep Learning

TABLE OF CONTENTS

Abstract

The NVIDIA® DGX-1™ with Tesla V100 (Figure 1) is an integrated system for deep learning. DGX-1 features 8 NVIDIA® Tesla® V100 GPU accelerators connect through NVIDIA® NVLink™, the NVIDIA high-performance GPU interconnect, in a hybrid cube-mesh network. Together with dual-socket Intel Xeon CPUs and four 100 Gb InfiniBand network interface cards, DGX-1 provides unprecedented performance for deep learning training. Moreover, the DGX-1 system software, powerful libraries, and NVLink network are tuned for scaling up deep learning across all eight Tesla V100 GPUs to provide a flexible, maximum performance platform for the development and deployment of deep learning applications in both production and research settings.

DGX-1 with V100 GPUs achieves dramatically higher throughput than DGX-1 with previous-generation NVIDIA Tesla P100 GPUs, achieving up to 3.1x faster deep learning training for convolutional neural networks. High-performance NVLink GPU interconnect improves scalability of deep learning training, improving recurrent neural network training performance by up to 1.5x compared to slower PCIe interconnect. More productivity and performance benefits come from the fact that DGX-1 is an integrated system, with a complete optimized software platform aimed at deep learning that ensures DGX-1 outperforms similar off-the-shelf systems.

Figure 1 NVIDIA DGX-1

1 INTRODUCTION

Deep learning is quickly changing the field of computer science as well as many other disciplines, and it is having a large impact on the economics of large enterprises such as Google [Metz 2015], Facebook [Statt 2016], and Amazon [Finley 2016]. Even as this new discipline and technology becomes mainstream, it is evolving rapidly. To serve increasingly sophisticated applications and deliver higher quality results, neural networks are becoming exponentially deeper and more complex. At the same time, neural networks deployed in mainstream interactive applications are being driven to infer and predict results faster.

The demand for deep learning performance is rapidly growing. Facebook CTO Mike Schroepfer recently noted that:

- Facebook's deployed neural networks process more than 6 million predictions per second;
- 25% of Facebook engineers are now using AI and machine learning APIs and infrastructure;
- Facebook has deployed more than 40 PFLOP/s of GPU capability in house to support deep learning across their organization [Schroepfer 2016:6:54].

In an April 2016 interview, former Baidu Chief Scientist Andrew Ng stated that Baidu has adopted an HPC (High Performance Computing) point of view to machine learning:

> *The faster we train our networks, the more iteration we can make on our datasets and models, and the more iterations we make, the more we advance our machine learning*. [Ng 2016a].

According to Ng, training one of Baidu's speech models requires 10 exaflops of computation [Ng 2016b:6:50].

As neural networks get deeper and more complex, they provide a dramatic increase in accuracy (for example, Microsoft Deep Residual Networks [He et al. 2015]), but training these higher accuracy networks requires much higher computation time, and their complexity increases prediction latency.

Aimed at satisfying this insatiable need for performance, the NVIDIA DGX-1 (shown in Figure 1) is the latest artificial intelligence (AI) supercomputer for deep learning training. NVIDIA's goal with DGX-1 was to create the world's fastest platform for training deep neural networks that can be deployed quickly and simply for plug-and-play use by deep learning researchers and data scientists. The architecture of DGX-1 draws on NVIDIA's experience in the field of high-performance computing as well as knowledge gained from optimizing deep learning frameworks on NVIDIA GPUs with every major cloud service provider and multiple Fortune 1000 companies.

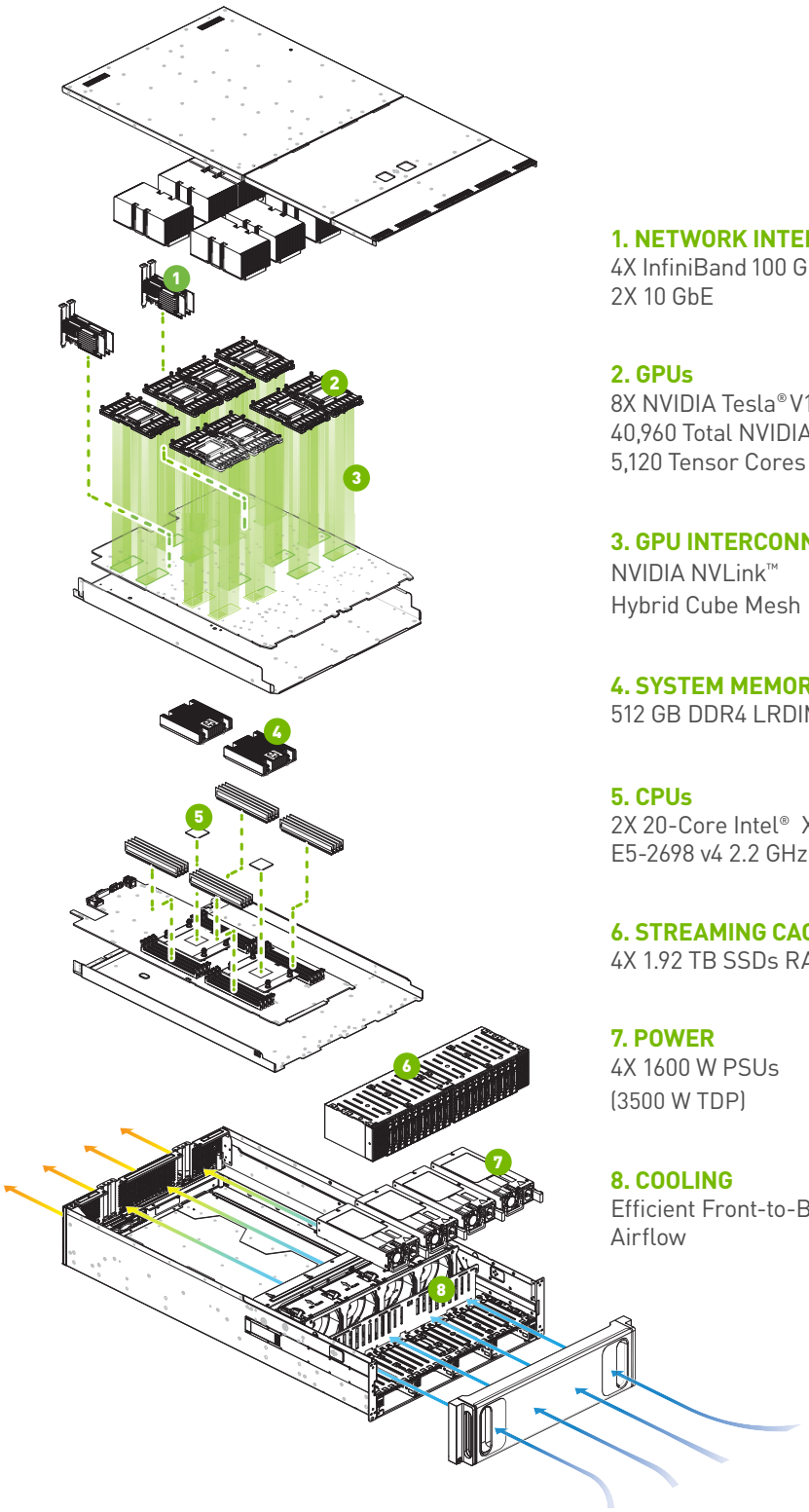
2 NVIDIA DGX-1 WITH V100 SYSTEM ARCHITECTURE

The NVIDIA® DGX-1™ is a deep learning system, architected for high throughput and high interconnect bandwidth to maximize neural network training performance. The core of the system is a complex of eight Tesla V100 GPUs connected in the hybrid cube-mesh NVLink network topology described in Section 3. In addition to the eight GPUs, DGX-1 includes two CPUs for boot, storage management, and deep learning framework coordination. DGX-1 is built into a three-rack-unit (3U) enclosure that provides power, cooling, network, multi-system interconnect, and SSD file system cache, balanced to optimize throughput and deep learning training time.

NVLink is an energy-efficient, high-bandwidth interconnect that enables NVIDIA GPUs to connect to peer GPUs or other devices within a node at an aggregate bidirectional bandwidth of up to 300 GB/s per GPU: over nine times that of current PCIe Gen3 x16 interconnections. The NVLink interconnect and the DGX-1 architecture's hybrid cube-mesh GPU network topology enable the highest achievable data-exchange bandwidth between a group of eight Tesla V100 GPUs.

Tesla V100's Page Migration Engine allows high bandwidth, low overhead sharing of data between GPUs and between each GPU and the system memory [NVIDIA Corporation 2016]. For multi-node, high performance clusters, DGX-1 provides high system-to-system bandwidth through InfiniBand (IB) networking with GPUDirect RDMA, and supports state-of-the art "scale-out" approaches such as those described in Section 3.3.

Figure 2 shows a diagram of DGX-1 system components.



1. NETWORK INTERCONNECT
4X InfiniBand 100 Gbps EDR
2X 10 GbE

2. GPUs
8X NVIDIA Tesla® V100 16 GB/GPU
40,960 Total NVIDIA CUDA® Cores
5,120 Tensor Cores

3. GPU INTERCONNECT
NVIDIA NVLink™
Hybrid Cube Mesh

4. SYSTEM MEMORY
512 GB DDR4 LRDIMM

5. CPUs
2X 20-Core Intel® Xeon®
E5-2698 v4 2.2 GHz

6. STREAMING CACHE
4X 1.92 TB SSDs RAID 0

7. POWER
4X 1600 W PSUs
(3500 W TDP)

8. COOLING
Efficient Front-to-Back
Airflow

Figure 2 DGX-1 with V100 system components

2.1 DGX-1 System Technologies

Tesla V100 (Figure 3) is the latest NVIDIA accelerator, designed for high performance computing and deep learning applications [NVIDIA Corporation 2017c]. The Tesla V100 accelerator features the GV100 GPU, which incorporates 80 streaming multiprocessors (SMs), each with:

- 8 Tensor Cores;
- 64 single-precision (FP32) cores;
- 64 integer (INT32) cores;
- 32 double-precision (FP64) cores;
- 256KB of register file (RF);
- Up to 96KB of shared memory (configurable).

Figure 3 The Tesla V100 Accelerator

Tesla V100 peak[1] computational throughput is:

- 125 Tensor TFLOP/s [NVIDIA Corporation 2017c]
- 7.8 TFLOP/s for FP64 computation
- 17.7 TFLOP/s for FP32]

To support this high computational throughput, Tesla V100 incorporates HBM2 (High Bandwidth Memory version 2). Tesla V100 includes 16 GB of HBM2 stacked memory with 900 GB/s of bandwidth; significantly higher than the bandwidth of GDDR5 RAM. Because HBM2 memory is stacked memory located on the same physical package as the GPU, it provides considerable space savings compared to traditional GDDR5, which enables high-density GPU servers like DGX-1.

Each Tesla V100 has 6 NVLink connections, each capable of 50 GB/s of bidirectional bandwidth, for an aggregate of up to 300 GB/s bidirectional bandwidth. NVLink and the DGX-1 interconnect topology and its implications are discussed in detail in Section 3.

The PCIe links between the GPUs and CPUs enable access to the system memory to enable working set and dataset streaming to and from the GPUs. The system memory capacity is four times the GPU memory capacity to enable simplified buffer management and balance for deep learning workloads. While twice the GPU memory footprint would normally be sufficient to manage background data moves and double buffering, four times gives greater flexibility for managing in-memory working sets and streaming data movement. In addition to the 512 GB of system memory, the eight Tesla V100 GPUs have a total of 128 GB HBM2 memory with net GPU memory bandwidth of 8 × 900 GB/s = 7.2 TB/s.

1. Based on GPU Boost Clock

DGX-1 is provisioned with both Ethernet and InfiniBand (IB) network interfaces. Two 10-gigabit Ethernet interfaces provide remote user access into the system, and also access to remote storage. To connect multiple DGX-1 systems, each system has four high-bandwidth, low-latency EDR IB (Extended Data Rate InfiniBand) ports for a total of 800 Gb/s of bidirectional communication. With 8 V100 GPUs and 4 EDR IB ports, each DGX-1 system has one IB port for every two GPUs. In addition, the DGX-1 EDR IB is compatible with NVIDIA GPUDirect Remote Direct Memory Access (RDMA), providing the ability to transfer data directly from GPU memory in one system to GPU memory in another without involving either the CPU or the system memory.

Efficient, high-bandwidth streaming of training data is critical to the performance of DGX-1 as a deep learning system, as is reliable, low failure rate storage. Each system comes configured with a single 480 GB boot OS SSD, and four 1.92 TB SAS SSDs (7.6 TB total) configured as a RAID 0 striped volume for high-bandwidth performance. For working sets larger than 7 TB, data can be staged and cached through the SSDs in the background during a training run, with remote data being accessed via either the provided 10G Ethernet interfaces or the EDR IB interfaces.

In addition to high computational throughput and memory bandwidth, training deep neural networks requires high streaming data rates from disk storage. DGX-1 is designed to support large training datasets that are hundreds of gigabytes to terabytes in size. The four SAS SSDs together provide an aggregate streaming read bandwidth of 2 GB/s, which is sufficient to avoid disk streaming bottlenecks during training.

The thermal design power (TDP)[2] of DGX-1 is 3.5 kW, but actual power consumption is dynamic based on workload. For ease of deployment, DGX-1 is designed to be air cooled in most datacenter environments with inlet air from 5°C – 35°C.

2. TDP specifies the maximum system power used while running real-world applications, and is used to correctly size power and cooling requirements in data centers.

3 MULTI-GPU AND MULTI-SYSTEM SCALING WITH NVLINK AND INFINIBAND

Servers with two or more GPUs per CPU are becoming common as developers increasingly expose and leverage the available parallelism in their applications. While dense GPU systems provide a great vehicle for scaling single-node performance, multi-GPU application efficiency can be constrained by the performance of the PCIe (Peripheral Component Interconnect Express) bus connections between GPUs. Similarly, data center applications are growing outside the box, requiring efficient scaling across multiple interconnected systems. To address both of these needs, DGX-1 incorporates the NVLink high-speed GPU interconnect for multi-GPU scalability within a system, and multiple EDR InfiniBand ports to provide high bandwidth between many connected DGX-1 systems.

Given that communication is an expensive operation, developers must overlap data transfers with computation, or carefully orchestrate GPU accesses over PCIe interconnect to maximize performance. As GPUs get faster and GPU-to-CPU ratios climb, a higher-performance GPU interconnect provides users with more flexibility in communication scheduling, and is required to properly balance the higher throughput of the GPUs.

This challenge motivated the creation of the NVLink high-speed interconnect, which enables NVIDIA GPUs to connect to peer GPUs and/or to NVLink-enabled CPUs or other devices within a node. NVLink supports the GPU ISA, which means that programs running on NVLink-connected GPUs can execute directly on data in the memory of another GPU as well as on local memory. GPUs can also perform atomic memory operations on remote GPU memory addresses, enabling much tighter data sharing and improved application scaling.

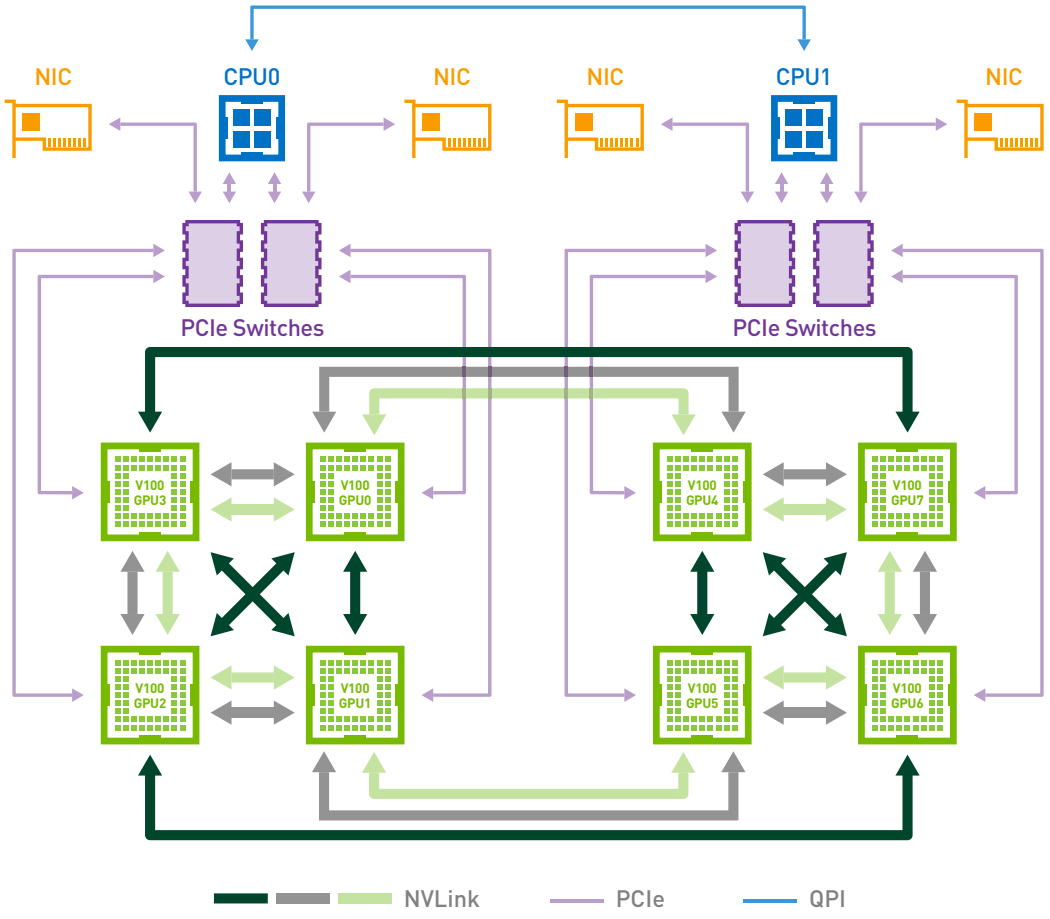
NVLink uses NVIDIA's new High-Speed Signaling interconnect (NVHS). NVHS transmits data over a differential pair running at up to 25 Gb/s. Eight of these differential connections form a "Sub-Link" that sends data in one direction, and two sub-links—one for each direction—form a "Link" that connects two processors (GPU-to-GPU or GPU-to-CPU). A single link supports up to 50 GB/s of bidirectional bandwidth between the endpoints. Multiple links can be ganged together for even higher bandwidth between processors. The NVLink implementation in Tesla V100 supports up to six Links, allowing for an aggregate maximum theoretical bidirectional bandwidth of up to 300 GB/s.

3.1 DGX-1 NVLink Network Topology for Efficient Application Scaling

DGX-1 includes eight NVIDIA Tesla V100 accelerators; providing the highest compute-density available in an air-cooled 3U chassis. Application scaling on this many highly parallel GPUs can be hampered by today's PCIe interconnect. NVLink provides the communications performance needed to achieve good scaling on deep learning and other applications. Each Tesla V100 GPU has six NVLink connection points, each providing a point-to-point connection to another GPU at a peak bandwidth of 25 GB/s in each direction. Multiple NVLink connections can be aggregated, multiplying the available interconnection bandwidth between a given pair of GPUs. The result is that NVLink provides a flexible interconnect that can be used to build a variety of network topologies among multiple GPUs. V100 also supports 16 lanes of PCIe 3.0. In DGX-1, these are used for connecting between the CPUs and GPUs and high-speed IB network interface cards.

The design of the NVLink network topology for DGX-1 aims to optimize a number of factors, including the bandwidth achievable for a variety of point-to-point and collective communications primitives, the ability to support a variety of common communication patterns, and the ability to maintain performance when only a subset of the GPUs is utilized. During the design, NVIDIA engineers modeled projected scaling of a variety of applications, such as deep learning, sorting, Fast Fourier Transforms (FFT), molecular dynamics, graph analytics, computational fluid dynamics, seismic imaging, ray tracing, and others. This paper focuses on the scaling of deep learning training.

The hybrid cube-mesh topology (Figure 4) can be thought of as a cube with GPUs at its corners and with all twelve edges connected through NVLink (some edges have two NVLink connections), and with two of the six faces having their diagonals connected as well. The topology can also be thought of as three interwoven rings of single NVLink connections.



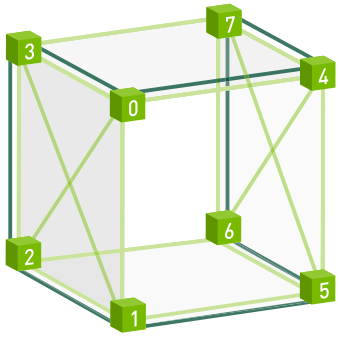


Figure 4 DGX-1 uses an 8-GPU hybrid cube-mesh interconnection network topology. The corners of the mesh-connected faces of the cube are connected to the PCIe tree network, which also connects to the CPUs and NICs.

The cube-mesh topology provides the highest bandwidth of any 8-GPU NVLink topology for multiple collective communications primitives, including broadcast, gather, all-reduce, and all-gather, which are important to deep learning. Using NVLink connections to span the gap between the two clusters of four GPUs relieves pressure on the PCIe bus and on the inter-CPU SMP link, and avoids staging transfers through system memory when transferring across the two clusters.

Furthermore, this 8-GPU cube-mesh topology is easily subdivided into fully connected halves, or into NVLink-connected GPU pairs. This flexibility is important when multiple applications share the server.

3.2 Scaling Deep Learning Training on NVLink

Deep neural networks learn many levels of abstraction, ranging from simple to complex concepts. The strength of deep models is that they are not only powerful but learnable. A deep neural network is trained by feeding it input and letting it compute layer-by-layer to generate output for comparison with a known correct answer. After computing the error at the output, this error flows backward through the network by back-propagation. At each step backward the model parameters are tuned in a direction that tries to reduce the error using a numerical optimization method such as stochastic gradient descent (SGD). This process sweeps over the data improving the model as it goes.

Training deep neural networks in parallel across multiple GPUs and/or multiple nodes requires distributing either the input data (“data parallel”), the model being trained (“model parallel”), or a hybrid of the two [Wu et al. 2015][Krizhevsky 2014]. Regardless of the approach, parallelizing across GPUs requires synchronization and communication of data (such as gradients) between GPUs. For example, in data-parallel approaches, separate parallel tasks must periodically resynchronize their gradients so that the model parameters are kept in sync across all parallel tasks. This amounts to an all-reduce operation.

Scaling is a measure of the improvement in time to solution when increasing the number of parallel processors applied to a problem. A common approach to scaling training deep neural networks is to increase the global batch size as the number of GPUs increases. Perhaps unsurprisingly, these so-called “weak” scaling approaches have high parallel efficiency, even with relatively slow interconnections among GPUs. Nevertheless, for many weak-scaling deep learning workloads, NVLink provides significant performance advantages.

A common deep learning workload is the training of convolutional neural networks for computer vision tasks such as image recognition and understanding. Implementations commonly use weak data-parallel scaling for training. Figure shows training performance and scaling for the Inception-V3 deep neural

network architecture using the Caffe2 framework. NVLink provides about a 15% overall performance benefit compared to PCIe when training on eight GPUs.

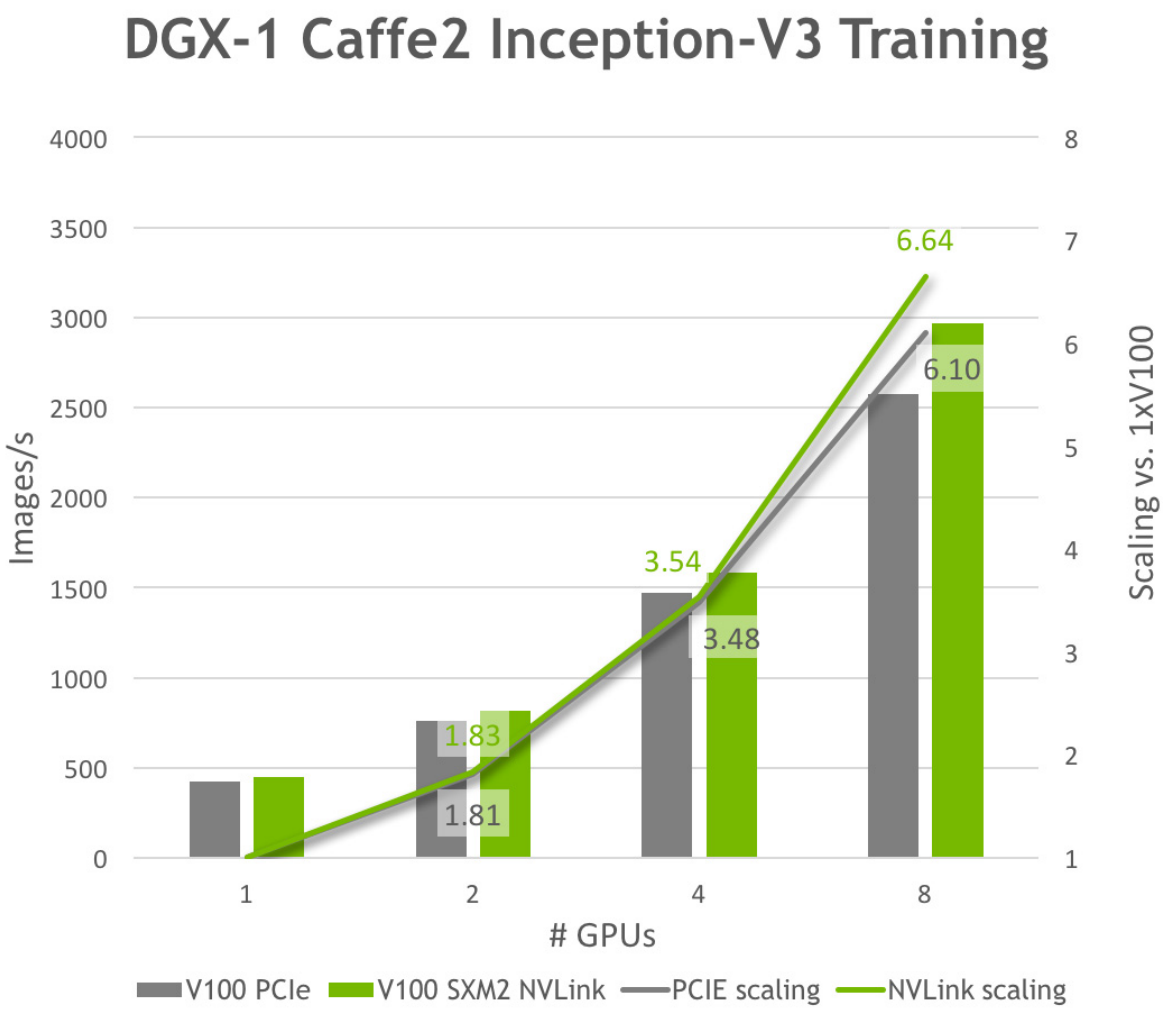


Figure 5 DGX-1 with V100 NVLink performance and scaling for mixed-precision training of the Inception-v3 neural network architecture using Caffe2 and the ImageNet dataset with a batch size of 128 per GPU. The bars show performance on one, two, four, and eight GPUs, comparing an off-the-shelf system of eight Tesla V100 GPUs using PCIe for communication (gray) with eight Tesla V100 GPUs in a DGX-1 using NVLink communication (green). The lines show the speedup compared to a single GPU. Tests used NVIDIA DGX containers version 17.11, processing real data with cuDNN 7.0.4, NCCL 2.1.2.

The training of recurrent neural networks (RNNs) are another common deep learning workload. Due to high inter-GPU communication requirements, RNNs—in particular long short-term memory (LSTM) networks—often benefit more from faster NVLink interconnect. RNNs are commonly used for speech recognition and natural language processing. An example is the “Sequence-to-Sequence” (Seq2Seq)

neural machine translation (NMT) technique [Sutskever et al., 2014], with several implementations and improvements including Open Neural Machine Translation (OpenNMT) [Klein et. al, 2017] and Sockeye [Hieber et. al, 2017]. As Figure 6 shows, DGX-1 with V100 GPUs and NVLink provides significant multi-GPU performance and scaling advantages for an NMT implementation of Seq2Seq relative to a V100 PCIe configuration.

Seq2Seq NMT models are used for language translation as well as image and video captioning [Venugopalan et. al, 2015]. Seq2Seq is an RNN made up of an encoder, which ingests an input sequence, and a decoder, which generates the output sequence (e.g., a translation). The encoder and decoder are parameterized by their embedding size, known as the RNN's learning capacity. Increasing the embedding size can improve the network's accuracy at the cost of lower training throughput (as with any other network, care must be taken to not overfit). With larger embeddings, the performance of NVLink becomes more valuable.

Figures 6 and 7 show training performance for a Sockeye NMT language translation model with a multilayer perceptron (MLP) attention model. Research has shown that adding attention to the decoder can improve the performance of a Seq2Seq network [Luong, et. al, 2015 and Bahdanau, et. al, 2016]. Figure 6 shows training performance and scaling of the NMT model on DGX-1 with an encoder and decoder embedding size of 512, comparing communication with NVLink to PCIe. NVLink provides about 30% overall performance benefit compared to PCIe when training on eight GPUs.

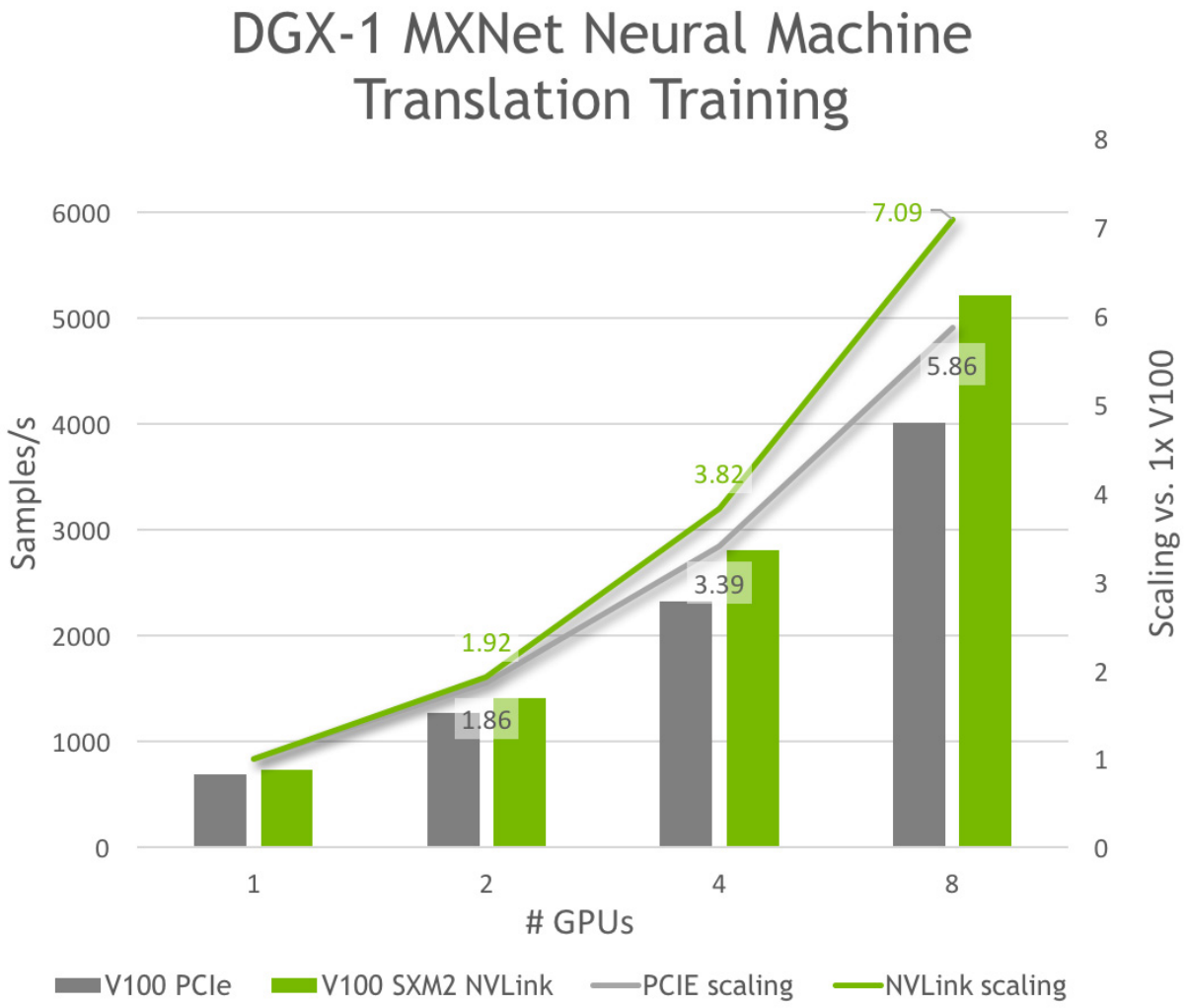


Figure 6 DGX-1 and V100 PCIe performance and scaling for single-precision training of a neural machine translation model with MLP attention and encoder/decoder embedding size of 512 and a batch size of 256 per GPU. The bars show performance on one, two, four, and eight GPUs, comparing an off-the-shelf system of eight Tesla V100 GPUs using PCIe for communication (gray) with eight Tesla V100 GPUs in a DGX-1 using NVLink for communication (green). The lines show the speedup compared to a single GPU. Tests used NVIDIA DGX containers version 17.11, processing real data with cuDNN 7.0.4, NCCL 2.1.2.

As with CNNs, NVLink benefits vary with differences in RNN configuration that affect the balance of computation and communication. Figure 7 shows that NVLink benefits the performance of Seq2seq more as the network embedding size increases, with a 2.54x improvement vs. PCIe for an embedding size of 2048. We ran three eight-GPU tests on a DGX-1 with and without NVLink. We varied the encoder and decoder embedding sizes from 512 to 2048, decreasing the per-GPU batch size with increasing embedding size due to GPU memory constraints.

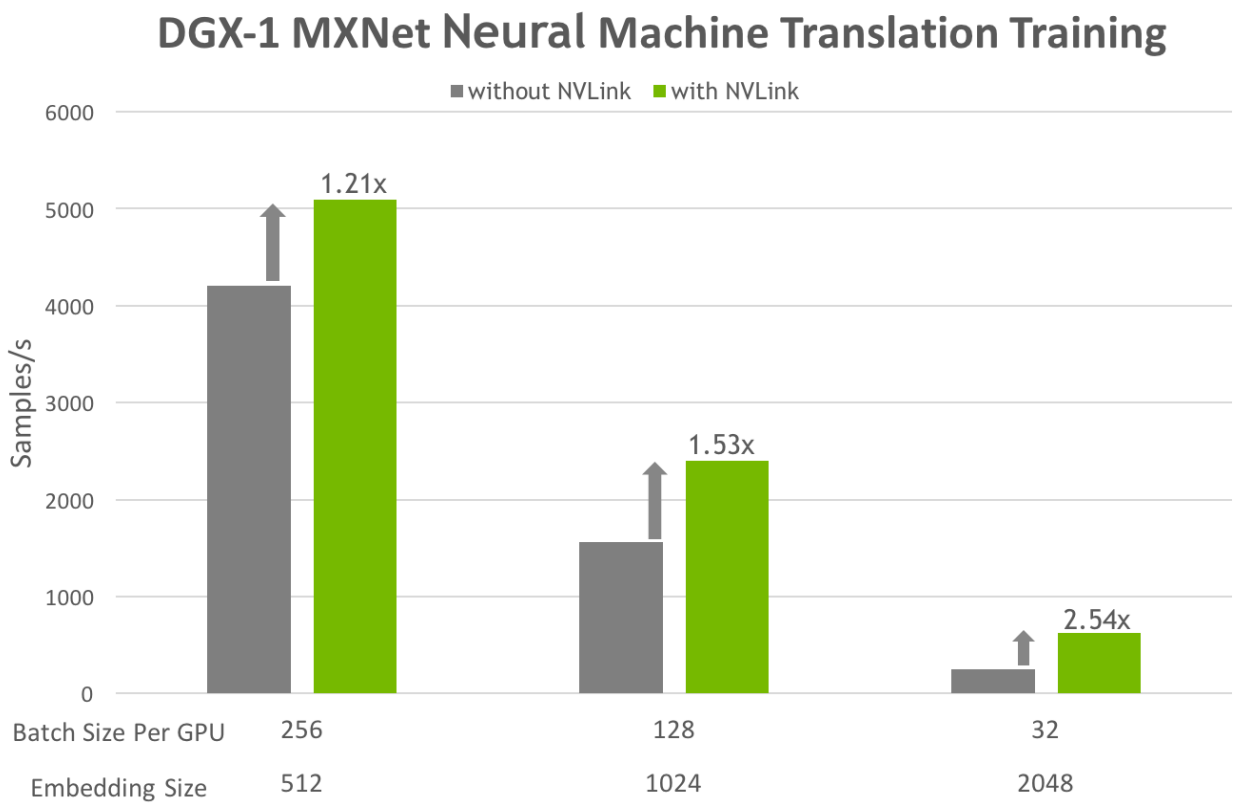


Figure 7 Sockeye neural machine translation single-precision training with MXNet using MLP attention on DGX-1, demonstrating significant NVLink performance benefits. The bars present performance on eight Tesla V100 GPUs in a DGX-1 when using NVLink for communication (green), and when using PCIe for communication (gray). Performance benefits increase with the encoder/ decoder embedding size. Results are the average number of samples per second processed during a single epoch of training with the German to English dataset. Tests used NVIDIA DGX MXNet container version 17.11, processing real data with cuDNN 7.0.4, NCCL 2.1.2.

In general, the largest NVLink advantages occur when scaling to all eight GPUs, where multiple NVLink connections provide bandwidth far exceeding what PCIe provides. Communication-heavy applications such as recurrent neural networks show even larger performance benefits from NVLink, as the 2.54x speedup in Figure 7 demonstrates.

3.3 InfiniBand for Scaling to Multiple DGX-1 Systems

Multi-system scaling of the latest computational workloads, especially deep learning, requires strong communications between GPUs, both inside the system and between systems, to match the significant GPU performance of each system and to improve performance of scaling workloads. In addition to NVLink for high speed communication internally between GPUs, DGX-1 also uses Mellanox ConnectX-4 EDR 100Gb InfiniBand ports to provide extremely low-latency and high-bandwidth communication between systems to reduce bottlenecks. The latest InfiniBand standard, EDR IB, configured in DGX-1 provides:

- Four EDR IB ports providing simultaneous 400 Gb/s in and 400 Gb/s out of each DGX-1 system
- Low-latency communication and built-in primitives and collectives to accelerate large computations across multiple systems;
- High performance computing network topology support to enable data transfer between multiple systems simultaneously with minimal contention;
- NVIDIA GPUDirect RDMA[3] across InfiniBand for direct transfers between GPUs in multiple systems.

DGX-1 comes configured with four EDR IB ports providing 800 Gb/s total bidirectional bandwidth that can be used to build a high-speed cluster of DGX-1 systems. Four EDR IB ports balance intra- and inter-node bandwidth, and in certain use cases can be fully consumed by inter-node communication. When compared to typical networking technologies such as Ethernet, InfiniBand provides twenty times the bandwidth and four times lower latency even across a large multi-system cluster (see Table 1).

3. NVIDIA GPUDirect Remote Direct Memory Access (RDMA) protocol provides the ability to transfer data directly between GPUs in two different systems across the InfiniBand network without involving the CPU or system memory. This reduces latency, and significantly increases performance of multi-systems.

Table 1 Multi-system DGX-1 cluster with InfiniBand provides 20x network performance.

	Typical multi-system cluster	DGX-1 multi-system cluster
Number of systems	124	124
Technology	Dual 10 Gb Ethernet	Quad EDR IB
Single system peak network Bandwidth	40 Gb/s	800 Gb/s
Full Cluster peak Bisection Bandwidth	310 GB/s	6,012 GB/s[a]
System wide latency	5 us	1.28 us[b]

a. InfiniBand full cluster peak bandwidth of 6,012 GB/s computed based on 124 systems with 4 EDR IB ports per system at 200 Gb/s (100 Gb/s in and 100 Gb/s out simultaneously) * 64/66 bit encoding divided by 8 bits/byte to convert to bytes times 0.5 to calculate bisection bandwidth.
b. InfiniBand latency of 1.28 us based on system to system latency of 1.01 us plus 3 switch hops at 0.09 us each.

The latest DGX-1 multi-system clusters use a network based on a fat-tree topology providing easily-routed, predictable, contention-free communication between systems (see Figure 8). A fat tree is a tree-structured network topology with systems at the leaves that connect up through multiple switch levels to a central top-level switch. Each level in a fat tree has the same number of links providing equal bandwidth. The fat-tree topology ensures the highest communication bisection[4] bandwidth and lowest latency for all-to-all and all-gather type collectives that are common in computational and deep learning applications.

In addition, the internal NVLink cube-mesh network is connected to the external InfiniBand network in a way that provides optimized performance. Placement of the EDR IB ports in the DGX-1 system at the corners of the NVLink hybrid cube-mesh network provide balanced access for GPUs to IB and allow direct GPU-to-GPU RDMA communication across IB (see Figure 4). Since each EDR IB link has similar bandwidth to a x16 PCIe Gen 3 interface, system-to-system interconnect is well-balanced. From an application point of view, GPUDirect RDMA and the unique NVLink and IB network design provide the ability for any GPU kernel to directly access any other GPU's memory in the network with minimal overhead, latency, and contention.

4. Bisection bandwidth is the total bandwidth available between two halves of a networked cluster. It is determined by splitting the system network down the center and adding the bandwidth of all the links that were split.

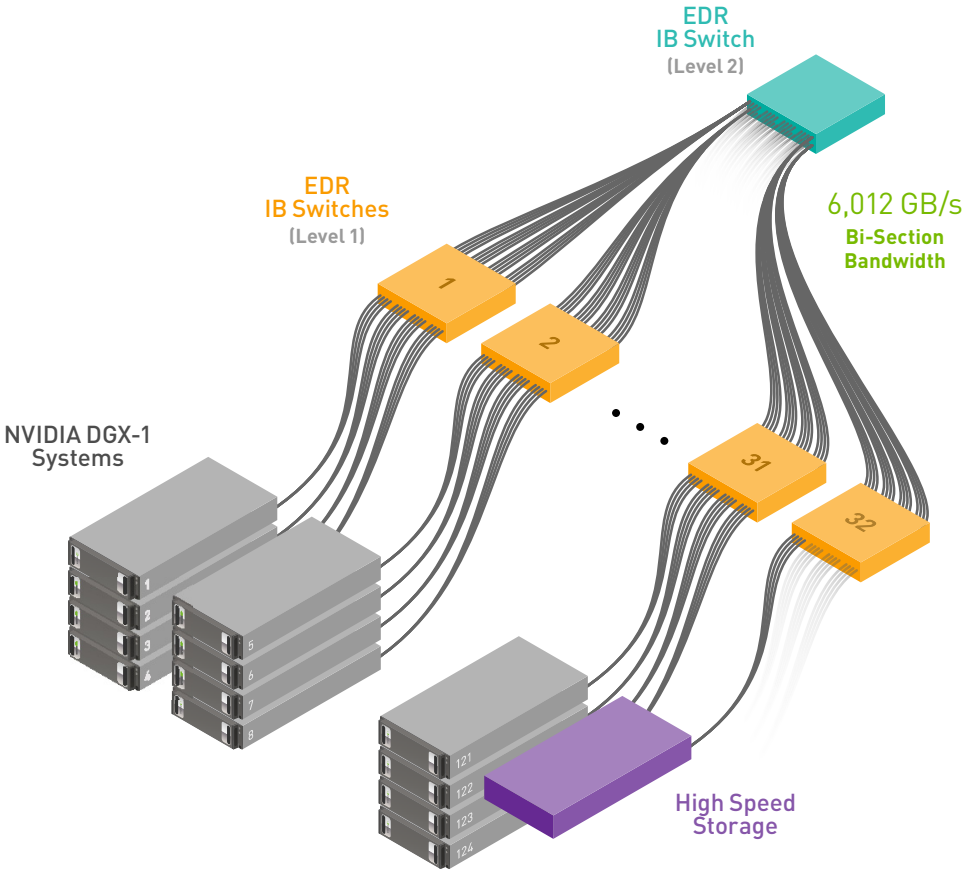


Figure 8 Example Multi-System Cluster of 124 DGX-1 Systems Tuned for Deep Learning

While the reference system architecture shown in Figure 8 is based on 124 systems, the fat-tree topology used is extensible to much larger configurations with more switches while still maintaining the high performance characteristics of InfiniBand. With the design in Figure 8, each first-level switch supports four DGX-1 systems, and up to 32 first-level and 16 second-level switches can be configured to support up to a maximum of 128 systems. To grow larger, a third switch level would be added or a Director-class IB switch can be used for level 2.

4 DGX-1 SOFTWARE

The DGX-1 software has been built to run deep learning at scale. A key goal is to enable practitioners to deploy deep learning frameworks and applications on DGX-1 with minimal setup effort. The design of the platform software is centered around a minimal OS and driver install on the server, and provisioning of all application and SDK software in Docker (see Section 4.2) containers through the DGX Container Registry[5], maintained by NVIDIA. Containers available for DGX-1 include multiple optimized deep learning frameworks, the NVIDIA DIGITS deep learning training application, third-party accelerated solutions, and the NVIDIA CUDA Toolkit. Figure 9 shows the DGX-1 deep learning software stack.

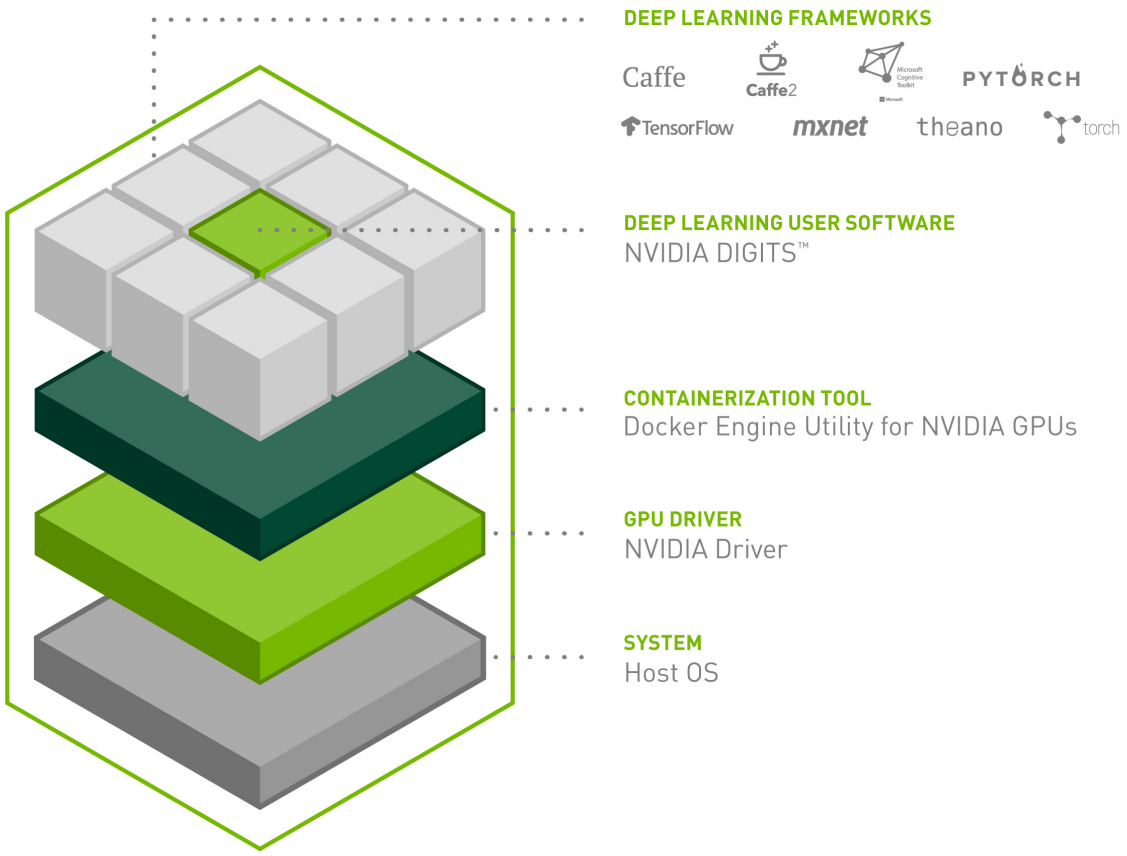


Figure 9 The DGX-1 Deep Learning Software Stack.

5. NVIDIA's Docker container registry service. See http://docs.nvidia.com/dgx/dgx-registry-guide/

This software architecture has many advantages:

- Since each deep learning framework is in a separate container, each framework can use different versions of libraries like libc, cuDNN, and others, and not interfere with each other.
- As deep learning frameworks are improved for performance or bug fixes, new versions of the containers are made available in the DGX Container Registry.
- The system is easy to maintain, and the OS image stays clean, since applications are not installed directly on the OS.
- Security updates, driver updates, and OS patches can be delivered seamlessly.

The deep learning frameworks and the CUDA Toolkit include libraries that have been custom-tuned to provide high multi-GPU performance on DGX-1.

The remainder of this section covers the key components of the DGX-1 software stack (above the GPU Compute Software Driver) in detail. Section 5 provides details of optimizations to deep learning frameworks for DGX-1.

4.1 NVIDIA CUDA Toolkit

CUDA is a parallel-computing platform and programming model created by NVIDIA to give application developers access to the massive parallel processing capability of GPUs. CUDA is the foundation for GPU acceleration of deep learning as well as a wide range of other computation- and memory-intensive applications ranging from astronomy, to molecular dynamics simulation, to computational finance. Today there are over 400 GPU-accelerated applications that leverage the CUDA parallel computing platform [NVIDIA 2017b]. DGX-1 is not only the fastest platform for deep learning, but the most advanced CUDA platform for a wide variety of GPU-accelerated applications.

The NVIDIA CUDA Toolkit provides a comprehensive environment for C and C++ developers building GPU-accelerated applications. The CUDA Toolkit includes NVCC, the CUDA C++ compiler for NVIDIA GPUs, a suite of libraries of GPU-accelerated algorithms, debugging and profiling tools, examples, and comprehensive programming guides and documentation. While the CUDA Toolkit can be directly installed on DGX-1, it is also provided as a Docker container image which can be used as the base layer for any containerized CUDA application (as Figure 9 shows). In addition, the full CUDA Toolkit is embedded in every deep learning framework container image.

4.2 Docker Engine Utility for NVIDIA GPUs

Over the last few years there has been a dramatic rise in the use of software containers for simplifying deployment of data center applications at scale. Containers encapsulate an application's dependencies to provide reproducible and reliable execution of applications and services without the overhead of a full virtual machine.

A Docker container is a mechanism for bundling a Linux application with all of its libraries, configuration files, and environment variables so that the execution environment is always the same, on whatever Linux system it runs (see Figure 10). DGX-1 uses Docker containers as the mechanism for deploying deep learning frameworks.

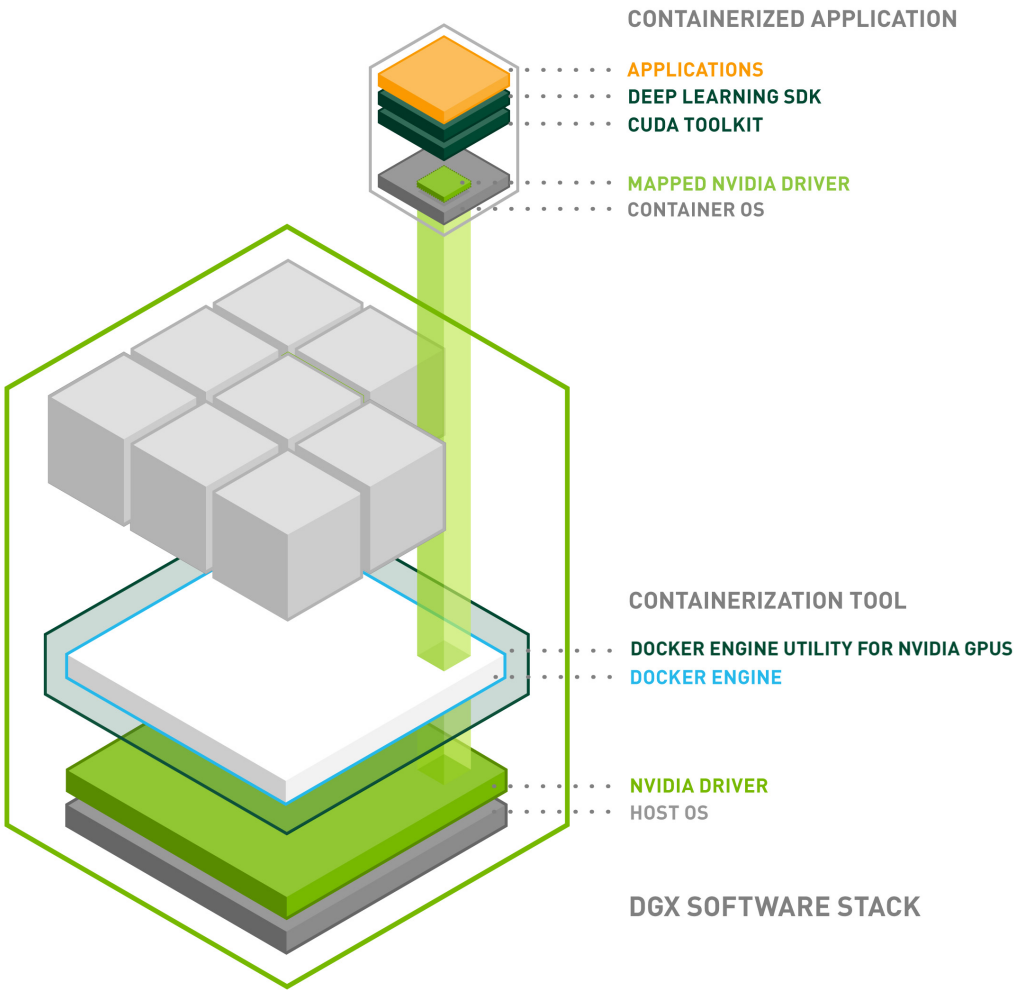


Figure 10 Docker containers encapsulate application dependencies to provide reproducible and reliable execution. The Docker Engine Utility for NVIDIA GPUs maps the user-mode components of the NVIDIA driver and the GPUs into the Docker container at launch.

Docker containers generally strive to be platform- and hardware-agnostic. They achieve this by separating user-mode code (in the container) from kernel-mode code. This separation presents a problem when using specialized hardware such as NVIDIA GPUs, since GPU drivers consist of a matched set of user-mode and kernel-mode modules. An early workaround to this problem was to fully install the NVIDIA drivers inside the container and map in the devices corresponding to the NVIDIA GPUs on launch. This solution is brittle because the version of the host driver must exactly match the version of the driver installed in the container. This requirement drastically reduced the portability of these early containers, undermining one of Docker's more important features.

To enable portability in Docker images that leverage GPUs, NVIDIA developed the Docker Engine Utility for NVIDIA GPUs [NVIDIA Corporation 2015], also known as the nvidia-docker utility, an open-source project that provides a command-line tool to mount the user-mode components of the NVIDIA driver and the GPUs into the Docker container at launch, as Figure 10 shows.

4.3 NVIDIA Deep Learning SDK

NVIDIA provides a complete suite of GPU-accelerated libraries built on top of the CUDA parallel computing platform. The following two libraries provide GPU-accelerated primitives for deep neural networks:

- the CUDA Basic Linear Algebra Subroutines library (cuBLAS): cuBLAS is a GPU-accelerated version of the complete standard BLAS library that delivers significant speedup running on GPUs. The cuBLAS generalized matrix-matrix multiplication (GEMM) routine implements a key computation used in deep neural networks; for example, in computing fully connected layers.
- the CUDA Deep Neural Network library (cuDNN): cuDNN provides highly tuned implementations for standard routines such as forward and backward convolution, pooling, normalization, and activation layers.

When deployed using Docker containers for DGX-1, deep learning frameworks are automatically configured to use parallel routines optimized for the Tesla V100 GPU architecture in DGX-1.

4.4 NCCL

The NVIDIA Collective Communication Library (NCCL, pronounced “Nickel”) is a library of topology-aware multi-GPU collective communication primitives that can be easily integrated into applications. Initially developed as an open-source research project[6], NCCL is designed to be light-weight, and is dependent only on common C++ and CUDA libraries. NCCL can be deployed in single-process or multi-process applications, handling required inter-process communication transparently. The NCCL API is designed to be familiar to anyone with experience using MPI collectives such as broadcast, reduce, gather, scatter, all-gather, all-reduce, or all-to-all.

Docker containers for DGX-1 include a version of NCCL that optimizes these collectives for the DGX-1 architecture’s 8-GPU hybrid cube-mesh NVLink network. When deployed using these containers, deep learning frameworks such as NVCaffe, Torch, Microsoft Cognitive Toolkit, and TensorFlow automatically use this version of NCCL when run on multiple GPUs.

6. http://github.com/NVIDIA/nccl

There are numerous approaches to implementing collectives efficiently. However, it is critical that our implementation takes the topology of interconnects between processors into account. To illustrate this, consider a broadcast of data from GPU0 to all other GPUs in the PCIe tree topology pictured in Figure 11.

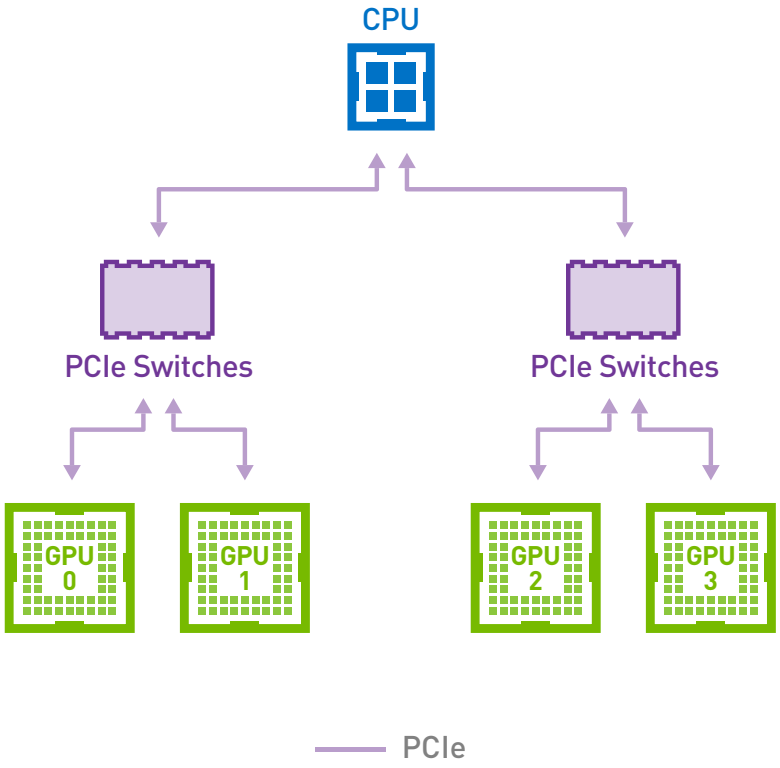


Figure 11 A common PCIe topology for 4 GPUs attached to a single CPU. Purple arrows represent PCIe x16 connections

A two-step tree algorithm is one approach: in the first step the data is sent from GPU0 to a second GPU, and in the second step both of these send data to the remaining processors. However, there is a choice. Either send data from GPU0 to GPU1 in the first step and then GPU0 to GPU2 and GPU1 to GPU3 in the second, or perform the initial copy from GPU0 to GPU2 and then GPU0 to GPU1 and GPU2 to GPU3 in the second step. Examining the topology, it is clear that the second option is preferred, since sending data simultaneously from GPU0 to GPU2 and GPU1 to GPU3 would cause contention on the upper PCIe

links, halving the effective bandwidth for this step. In general, achieving good performance for collectives requires careful attention to the interconnect topology.

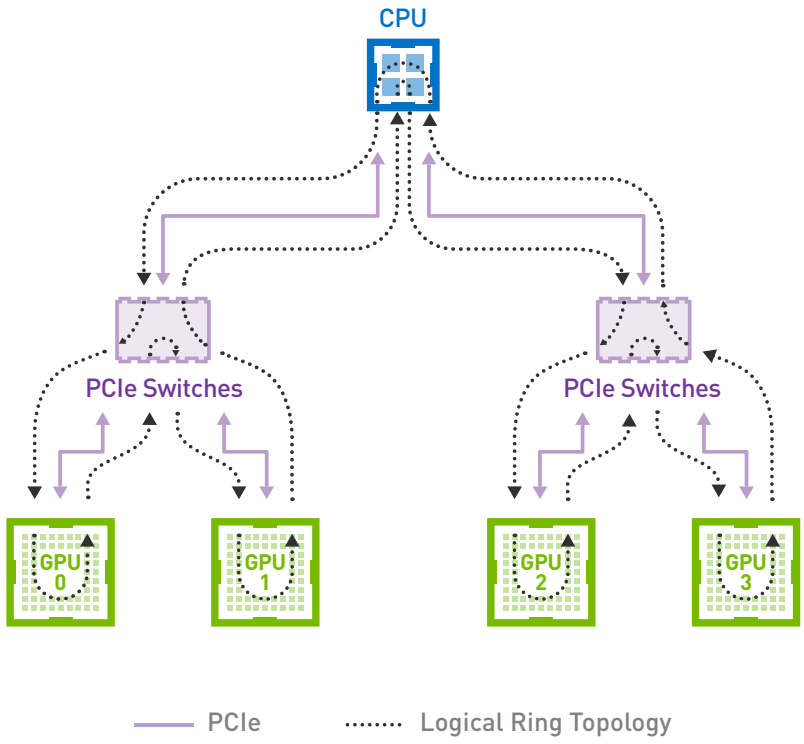


Figure 12 Ring order of GPUs in PCIe tree

To optimize Broadcast bandwidth, an even better approach is to treat the PCIe tree topology as a ring, as Figure 12 shows. The broadcast is then performed by relaying small chunks of the input around the ring from GPU0 to GPU3. Interestingly, ring algorithms provide near optimal bandwidth for nearly all of the standard collective operations, even when applied to tree-like PCIe topologies, provided that the correct ring order is selected.

In order to provide maximum bandwidth, NCCL implements ring-style collectives, and implicitly indexes the GPUs into the optimal ring order under the hood. This provides great performance for applications while freeing developers from having to worry about specific hardware configurations.

The 8-GPU hybrid cube-mesh network can be thought of as three interwoven bidirectional rings of single NVLink connections, as Figure 13 shows. Treating the topology this way ensures that the performance of collectives other than all-to-all is largely equivalent.

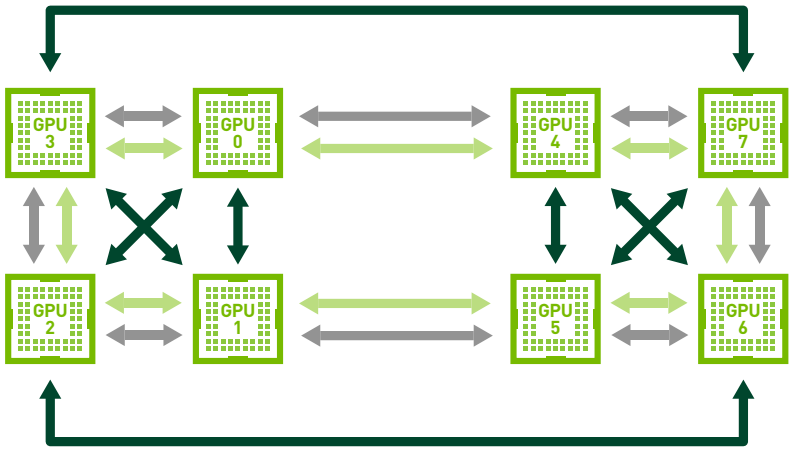


Figure 13 The DGX-1 NVLink hybrid cube-mesh topology can be treated as three interwoven bidirectional rings of single NVLink connections, shown here in gray, light green, and dark green.

Inter-GPU transfers for deep learning are performed using these three distinct bidirectional rings. Each ring connects all eight GPUs and together they use all six links of each Volta GPU in both directions. With this approach, reduction and broadcast operations can be performed at a speed of more than 130 GB/s, compared to 10 GB/s using PCIe on previous hardware generations. This performance is essential to achieving high scaling for deep learning training.

5 DEEP LEARNING FRAMEWORKS AND TOOLS FOR DGX-1

The NVIDIA Deep Learning SDK accelerates widely-used deep learning frameworks such as NVCaffe, Caffe2, Microsoft Cognitive Toolkit, MXNet, TensorFlow, Theano, PyTorch, Torch, and TensorRT. The following sections describe the deep learning frameworks and tools NVIDIA has optimized for DGX-1.

The DGX-1 software stack provides containerized versions of these frameworks optimized for the system. These frameworks, including all necessary dependencies, are pre-built, tested, and ready to run. For users who need more flexibility to build custom deep learning solutions, each framework container image also includes the framework source code to enable custom modifications and enhancements, along with the complete software development stack described in Section 4.

Most deep learning frameworks have begun to merge support for half-precision training techniques that exploit Tensor Core calculations in Volta. Some frameworks include support for FP16 storage and Tensor Core math. To achieve optimum performance, you can train a model using Tensor Core math and FP16 mode on some frameworks. For information about which frameworks are optimized for Volta, see http://docs.nvidia.com/deeplearning/sdk/mixed-precision-training/index.html.

For the latest list of NVIDIA's optimizations and changes for DGX-1, select your deep learning framework listed in the Deep Learning DGX Documentation Release Notes.

5.1 NVCaffe

Caffe[7] is a deep learning framework made with flexibility, speed, and modularity in mind. It was originally developed by the Berkeley Vision and Learning Center (BVLC) and by community contributors.

NVIDIA Caffe [NVIDIA Corporation, 2017a], also known as NVCaffe, is an NVIDIA-maintained fork of BVLC Caffe tuned for NVIDIA GPUs, particularly in multi-GPU configurations. It includes multi-precision support as well as other NVIDIA-enhanced features and offers performance specially tuned for the NVIDIA DGX-1.

NVCaffe supports single and multi-GPU execution.

The following list summarizes NVIDIA's NVCaffe optimizations and changes for DGX-1.

- Integration with cuDNN v7.
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.

7. http://caffe.berkeleyvision.org/

- Automatic selection of the best cuDNN convolution algorithm.
- NVCaffe includes support for FP16 storage and Tensor Core math. To achieve optimum performance, you can train a model using Tensor Core math and FP16 mode on NVCaffe.
- 16-bit (half) floating point train and inference support.
- Mixed-precision support for storing and/or computing data in either 64-, 32- or 16-bit formats. Precision can be defined for every layer (forward and backward passes may be different), or it can be set globally.
- A parallelized parser and image transformer for improved I/O performance.
- Optimized GPU memory management for data and parameters storage, I/O buffers and workspace for convolutional layers.
- Parallel data parser and transformer for improved I/O performance.
- Parallel back-propagation and gradient reduction on multi-GPU systems.
- Fast solvers implementation with fused CUDA kernels for weights and history update.
- Multi-GPU test phase for even memory load across multiple GPUs.
- Backward compatibility with BVLC Caffe and NVCaffe 0.15.
- Extended set of optimized models (including 16-bit floating point examples).

5.2 Caffe2

Caffe2[8] is a deep-learning framework designed to easily express all model types, for example, convolutional neural networks (CNNs), recurrent neural networks (RNNs), and more, in a friendly Python-based API, and execute them using a highly efficient C++ and CUDA backend.

Caffe2's flexible API lets users define models for inference or training using expressive, high-level operations. The Python interface allows easy control and visualization of the inference or training process.

Caffe2 supports single and multi-GPU execution, along with multi-node execution.

The following list summarizes NVIDIA's Caffe2 optimizations and changes for DGX-1.

- Integration with cuDNN v7.
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.

8. https://research.fb.com/downloads/caffe2/

- Caffe2 includes support for FP16 storage and Tensor Core math. To achieve optimum performance, you can train a model using Tensor Core math and FP16 mode on Caffe2.

5.3 Microsoft Cognitive Toolkit

The Microsoft Cognitive Toolkit[9] (also known as CNTK), is a unified deep-learning toolkit that allows users to easily realize and combine popular model types such as feed-forward deep neural networks (DNNs), convolutional neural networks (CNNs), and recurrent neural networks (RNNs). Microsoft Cognitive Toolkit implements stochastic gradient descent (SGD) learning with automatic differentiation and parallelization across multiple GPUs and servers[10]. Microsoft Cognitive Toolkit can be called as a library from Python or C++ applications, or executed as a standalone tool using the BrainScript model description language.

NVIDIA and Microsoft worked closely to accelerate the Microsoft Cognitive Toolkit on GPU-based systems such as DGX-1 and Azure N-Series virtual machines. This combination offers startups and major enterprises alike tremendous ease of use and scalability since a single framework can be used to first train models on premises with the DGX-1 and later deploy those models at scale in the Microsoft Azure cloud[11].

Microsoft Cognitive Toolkit supports single and multi-GPU execution.

The following list summarizes NVIDIA's Microsoft Cognitive Toolkit optimizations and changes for DGX-1.

- Integration with cuDNN v7.
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.
- Integration of Volta hardware support.
- Image reader pipeline improvements allow AlexNet [Krizhevsky et al. 2012] to train at over 12,000 images/second.
- Reduced GPU memory overhead for multi-GPU training by up to 2 GB per GPU.
- Dilated convolution support.

9. https://www.microsoft.com/en-us/research/product/cognitive-toolkit/
10. Source: https://github.com/Microsoft/CNTK#what-is-the-microsoft-cognitive-toolkit
11. For information on using Microsoft Cognitive Toolkit in Azure, see https://github.com/Microsoft/CNTK/wiki/CNTK-on-Azure

5.4 MXNet

MXNet[12] is a deep learning framework designed for both efficiency and flexibility, which allows you to mix symbolic and imperative programming to maximize efficiency and productivity. At the core of MXNet is a dynamic dependency scheduler that automatically parallelizes both symbolic and imperative operations on the fly. A graph optimization layer on top of the scheduler makes symbolic execution fast and memory efficient. MXNet is portable and lightweight, and scales to multiple GPUs and multiple machines.

MXNet supports single and multi-GPU execution.

The following list summarizes NVIDIA's MXNet optimizations and changes for DGX-1.

- Integration with cuDNN v7.
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.
- MXNet includes support for FP16 storage and Tensor Core math. To achieve optimum performance, you need to train a model using Tensor Core math and FP16 mode on MXNet.
- Improved input pipeline for image processing.
- Optimized embedding layer CUDA kernels.
- Optimized tensor broadcast and reduction CUDA kernels.
- Optimized input pipeline for image processing

5.5 TensorFlow

TensorFlow[13] is an open-source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) that flow between them. This flexible architecture lets you deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device without rewriting code.

TensorFlow was originally developed by researchers and engineers working on the Google Brain team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research. The system is general enough to be applicable in a wide variety of other domains, as well.

12. http://mxnet.io
13. https://www.tensorflow.org

For visualizing TensorFlow results, the TensorFlow Docker image also contains TensorBoard (https://www.tensorflow.org/get_started/summaries_and_tensorboard). TensorBoard is a suite of visualization tools. For example, you can view the training histories as well as an image of the network model.

TensorFlow supports single and multi-GPU execution.

The following list summarizes NVIDIA's TensorFlow optimizations and changes for DGX-1.

- Integration with cuDNN v7.
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.
- TensorFlow supports FP16 storage and Tensor Core math. Models that contain convolutions or matrix multiplications using the `tf.float16` data type will automatically take advantage of Tensor Core hardware whenever possible.
- Replacement of libjpeg with libjpeg-turbo.
- Support for the ImageNet preprocessing script.

5.6 Theano

Theano[14] is a Python library that allows you to efficiently define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays. Theano has been powering large-scale computationally intensive scientific investigations since 2007.

Theano supports single and limited multi-GPU execution.

The following list summarizes NVIDIA's Theano optimizations and changes for DGX-1.

- Integration with cuDNN v7.
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.
- Theano includes support for FP16 storage and Tensor Core math. To make use of Tensor Core math, set the `dnn.conv.algo_xxx` configuration parameter to `time_once` or `time_on_shape_change`.
- Runtime code generation: evaluate expressions faster.
- Extensive unit-testing and self-verification: detect and diagnose many types of errors.

14. http://deeplearning.net/software/theano/

5.7 PyTorch

PyTorch[15] is a Python package that provides two high-level features:

- Tensor computation (like NumPy) with GPU acceleration;
- Deep neural networks (DNNs) built on a tape-based autograd system.

You can reuse your favorite Python packages such as NumPy, Scipy and Cython to extend PyTorch when needed.

PyTorch supports single and multi-GPU execution.

The following list summarizes NVIDIA's PyTorch optimizations and changes for DGX-1.

- Integration with cuDNN v7 with support for Tensor Core math when available.
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.
- PyTorch includes support for FP16 storage and Tensor Core math. To achieve optimum performance, you can train a model using Tensor Core math and FP16 mode on PyTorch.
- Supports Tensor Core operations for convolutions and GEMMs on Volta hardware.
- The examples directory contains examples of ImageNet and LSTM training scripts that uses FP16 data, as well as how to do training with FP16.
- Matrix multiplication on FP16 inputs uses Tensor Core math when available.
- A custom batch normalization layer is implemented to use cuDNN for batch normalization with FP16 inputs.

15. http://pytorch.org/

5.8 Torch

Torch[16] is a scientific computing framework with wide support for deep learning algorithms. Torch is easy to use and efficient, thanks to an easy and fast scripting language, Lua, and an underlying C/CUDA implementation. Torch offers popular neural network and optimization libraries that are easy to use yet provide maximum flexibility to build complex neural network topologies.

Torch supports single and multi-GPU execution.

The following list summarizes NVIDIA's Torch optimizations and changes for DGX-1.

- Integration with cuDNN v7.
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.
- Buffering of parameters to be communicated by NCCL to reduce latency overhead.
- cuDNN bindings for recurrents networks (RNN, GRU, LSTM), including persistent versions which greatly improve the performance of small batch training.
- Dilated convolution support.
- Support for 16- and 32-bit floating point (FP16 and FP32) data input to cuDNN routines.
- Support for operations on FP16 tensors (using FP32 arithmetic).

5.9 DIGITS

The NVIDIA Deep Learning GPU Training System (DIGITS)[17] puts the power of deep learning into the hands of engineers and data scientists.

DIGITS can be used to rapidly train highly accurate deep neural network (DNNs) for image classification, segmentation and object detection tasks. DIGITS simplifies common deep learning 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.

The following list summarizes NVIDIA's DIGITS optimizations and changes for DGX-1.

16. http://torch.ch
17. https://developer.nvidia.com/digits

- Integration with cuDNN v7.
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.
- DIGITS runs on top of NVCaffe, Torch, and TensorFlow frameworks which are optimized for DGX.

5.10 TensorRT

NVIDIA TensorRT™ is a high-performance deep learning inference optimizer and runtime that delivers low latency, high-throughput inference for deep learning applications on NVIDIA GPUs. TensorRT takes a network definition and optimizes it by merging tensors and layers, transforming weights, choosing efficient intermediate data formats, and selecting from a large kernel catalog based on layer parameters and measured performance.

After you have trained a neural network, you can optimize and deploy the model for GPU inferencing with TensorRT. The TensorRT container provides an easy to use container for TensorRT development. The container allows for the TensorRT samples to be built, modified and executed. For more information about optimizing and deploying using TensorRT, see the Deep Learning SDK Documentation.

The TensorRT API includes importers for trained deep learning models in a variety of standard formats. Once imported, TensorRT can optimize a network and generate a run-time engine for deployment. TensorRT also includes an infrastructure that allows you to perform inference using reduced-precision arithmetic to take advantage of the high performance and efficiency reduced-precision capabilities of Pascal and Volta GPUs.

TensorRT has both C++ and Python APIs. The C++ API allows developers to import, calibrate, generate and deploy networks using C++. Networks can be imported directly from NVCaffe, or from other frameworks via the UFF format. They may also be created programmatically by instantiating individual layers and setting parameters and weights directly.

TensorRT 3.0 introduces a Python API to allow developers to easily parse models (for example from NVCaffe, TensorFlow, NumPy compatible frameworks) and generate and run PLAN files within Python-based development environments. Currently, all TensorRT functionality except for INT8 calibrators and RNN support is exposed via the Python API. The Python API also introduces compatibility with Numpy arrays for layer weights and GPU-resident input and output data (via PyCUDA). The Python API provides a set of utility functions to address common tasks including parsing NVCaffe and UFF models and writing PLAN files.

The following list summarizes NVIDIA's TensorRT optimizations and changes for DGX-1.

- Integration with cuDNN v7.

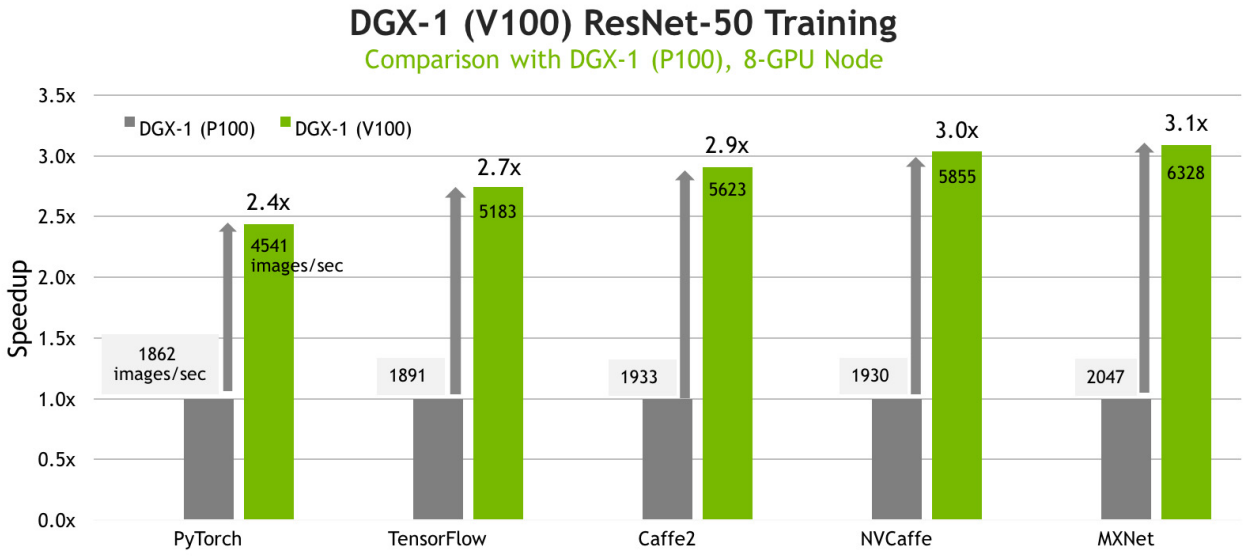
- Integration with CUDA 9.
- Integration with cuBLAS.
- Integration with latest version of NCCL with NVLink for improved multi-GPU scaling.

6 RESULTS: DGX-1 FOR HIGHEST DEEP LEARNING PERFORMANCE

The performance of DGX-1 for training popular deep neural networks speaks volumes about the value of an integrated system for deep learning.

6.1 ResNet-50 Training Results

The graph in Figure 14 demonstrates that DGX-1 with V100 GPUs achieves much higher throughput than DGX-1 with previous-generation NVIDIA Tesla P100 GPUs for training the ResNet-50 convolutional neural network across a variety of different frameworks. The Tesla V100 GPU provides much higher performance compared to the Tesla P100 GPU. Not only does V100 have 40% more FP32 CUDA Cores, it also adds new Tensor Cores, which can provide up to 8x higher throughput for mixed-precision matrix multiply-and-accumulate, a core computation in deep neural networks (see Section 2.1). Tesla V100 also has higher peak memory bandwidth and faster second-generation NVLink interconnect. Figure 14 shows that the high performance of Tesla V100 translates to up to 3.1x higher training performance for the ResNet-50 network on all eight GPUs of DGX-1, and similar speedups can be measured on other convolutional neural networks.



Best throughput achievable on each platform.
DGX-1 (P100) using FP32, DGX-1 (V100) using mixed precision (FP16 and FP32) using deep learning framework containers version 17.11

Figure 14 DGX-1 deep learning training speedup comparing eight Tesla V100 GPUs in a DGX-1 with eight Tesla P100s in a DGX-1 for the ResNet-50 deep neural network architecture on the popular PyTorch (0.2.0+), TensorFlow (1.3.0+), Caffe2 (0.8.1+), NVCaffe (0.16.4) and MXNet (0.12.0+) deep learning frameworks.

- *Other software used: NCCL 2.1.2, CUDA 9.0.176, cuDNN 7.0.4, Ubuntu 16,04, NVIDIA Linux DIsplay Driver 384.81*
- *NVIDIA deep learning container version: 17.11*
- *Floating point precision: DGX-1 (P100) using FP32, DGX-1 (V100) using mixed precision (FP16 and FP32)*
- *Batch sizes:*

Framework	***DGX-1 (P100) Batch Size***	***DGX-1 (V100) Batch Size***
PyTorch	*128*	*256*
TensorFlow	*128*	*256*
Caffe2	*64*	*128*
NVCaffe	*64*	*128*
MXNet	*96*	*128*

6.2 Sequence-to-Sequence Training Results

Figure 15 demonstrates the benefits of NVLink for training sequence-to-sequence recurrent neural networks (RNNs), which are commonly used for machine translation tasks. Compared to convolutional neural networks, which tend to be compute-bound, scaling RNNs to multiple GPUs can result in a communication bottleneck, which the higher inter-GPU bandwidth of NVLink can help reduce. Figure 15 shows that DGX-1 with Tesla V100 and NVLink can train the Seq2Seq RNNs up to 1.5x faster on all eight GPUs than an off-the-shelf system with eight Tesla V100 GPUs interconnected via PCIe.

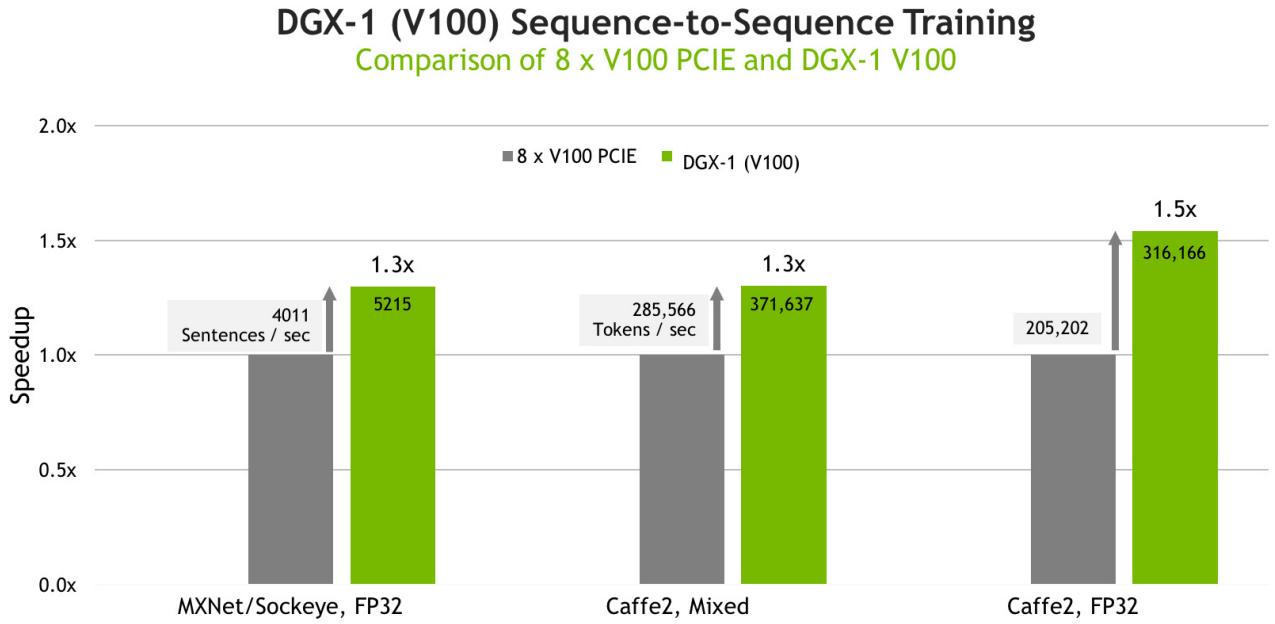


Figure 15 DGX-1 sequence-to-sequence recurrent neural network training speedup comparing eight NVLink-connected Tesla V100 GPUs in a DGX-1 with eight PCIe-connected Tesla V100 GPUs in an off-the-shelf system. PCIe results run on a SMC 4028GR-TRT with dual Intel Xeon E5-2698v4 CPUs and 256GB DDR4-2133 RAM, with eight PCIe Tesla V100 accelerators. MXNet (0.12.0+) and Caffe2 (0.8.1+) deep learning frameworks used in all tests. See Figure 14 for other software used. Seq2Seq networks implemented in each framework were trained with an embedding sizes of 512 and a batch size of 256 per GPU. MXNet results used MLP attention and Caffe2 results used regular attention. MXNet reports results in sequences per second, while Caffe2 reports results in tokens per second, hence the larger values.

6.3 Conclusion

Powerful Volta GPUs and high-performance NVLink interconnect are just part of the DGX-1 story. More productivity and performance benefits come from the fact that DGX-1 is an integrated system, with a complete software platform aimed at deep learning, as described in Sections 4 and 5. This includes the deep learning framework optimizations such as those in NVCaffe, cuBLAS, cuDNN, and other GPU-accelerated libraries, and NVLink-tuned collective communications through NCCL. This integrated software platform, combined with Tesla V100 and NVLink, ensures that DGX-1 outperforms similar off-the-shelf systems.

To learn more about NVIDIA DGX-1, visit http://www.nvidia.com/dgx1.

References

Bahdanau, D., KyungHyun Cho, K., and Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. 2016. https://arxiv.org/pdf/1409.0473.pdf

Finley, K. 2016. Amazon's Giving Away the AI Behind Its Product Recommendations http://www.wired.com/2016/05/amazons-giving-away-ai-behind-product-recommendations/

He, K., Zhang, X., Ren, S., and Sun, J. 2015. Deep Residual Learning for Image Recognition. *arXiv [cs.CV]*. http://arxiv.org/abs/1512.03385

Hieber, F., Domhan, T., Denkowski, M., Vilar, D., Sokolov, A., Clifton, A., Post, M. Sockeye: A Toolkit for Neural Machine Translation. 2017 https://arxiv.org/abs/1712.05690

Klein, G. and Kim, Y. and Deng, Y. and Senellart, J. and Rush. A.~M. OpenNMT: Open-Source Toolkit for Neural Machine Translation. 2017 https://arxiv.org/abs/1701.02810

Krizhevsky, A. 2014. One weird trick for parallelizing convolutional neural networks. *arXiv [cs.NE]*. http://arxiv.org/abs/1404.5997.

Krizhevsky, A., Sutskever, I., and Hinton, G.E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In: F. Pereira, C.J.C. Burges, L. Bottou and K.Q. Weinberger, eds., *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 1097–1105. http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Luong, M., Hieu Pham, H. and Manning. C., Effective Approaches to Attention-based Neural Machine Translation, 2015 https://arxiv.org/pdf/1508.04025.pdf

Metz, C. 2015. TensorFlow, Google's Open Source AI, Signals Big Changes in Hardware Too. http://www.wired.com/2015/11/googles-open-source-ai-tensorflow-signals-fast-changing-hardware-world/

Ng, A. 2016a. Baidu's Chief Scientist on Intersection of Supercomputing, Machine Learning. http://www.nextplatform.com/2016/04/01/baidus-chief-scientist-intersection-supercomputing-machine-learning/.

Ng, A. 2016b. AI: The New Electricity. https://www.youtube.com/watch?v=4eJhcxfYR4I.

NVIDIA Corporation. 2015. Docker Engine Utility for NVIDIA GPUs. *Github*. https://github.com/NVIDIA/nvidia-docker.

NVIDIA Corporation. 2016. NVIDIA® Tesla® P100 — The Most Advanced Data Center Accelerator Ever Built. Featuring Pascal P100, the World's Fastest GPU. http://www.nvidia.com/object/pascal-architecture-whitepaper.html

NVIDIA Corporation. 2017a. *NVCaffe branch.* https://github.com/NVIDIA/caffe

NVIDIA Corporation. 2017b. *GPU-Accelerated Applications*. http://www.nvidia.com/content/gpu-applications/PDF/gpu-applications-catalog.pdf

NVIDIA Corporation. 2017c. NVIDIA® Tesla® V100 GPU Architecture — The World's Most Advanced Data Center Accelerator. http://www.nvidia.com/object/volta-architecture-whitepaper.html

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015. http://arxiv.org/abs/1501.02876

Schroepfer, M. 2016. F8 2016 Day 1 Keynote. https://developers.facebook.com/videos/f8-2016/keynote/.

Statt, N. 2016. Exploring Facebook's massive, picture-painting AI brain. http://www.theverge.com/2016/7/13/12172904/facebook-ai-big-sur-machine-learning-prineville-data-center

Ilya Sutskever and Oriol Vinyals and Quoc V. Sequence to Sequence Learning with Neural Networks. NIPS 2014.

Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., and Saenko, K. Sequence to Sequence—Video to Text. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015.

Wu, R., Yan, S., Shan, Y., Dang, Q., and Sun, G. 2015. Deep Image: Scaling up Image Recognition. *arXiv [cs.CV]*. http://arxiv.org/abs/1501.02876.

Notice

ALL INFORMATION PROVIDED IN THIS WHITE PAPER, INCLUDING COMMENTARY, OPINION, NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and other changes to this specification, at any time and/or to discontinue any product or service without notice. Customer should obtain the latest relevant specification before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer. NVIDIA hereby expressly objects to applying any customer general terms and conditions with regard to the purchase of the NVIDIA product referenced in this specification.

NVIDIA products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

NVIDIA makes no representation or warranty that products based on these specifications will be suitable for any specified use without further testing or modification. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to ensure the product is suitable and fit for the application planned by customer and to do the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this specification. NVIDIA does not accept any liability related to any default, damage, costs or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this specification, or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this specification. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA. Reproduction of information in this specification is permissible only if reproduction is approved by NVIDIA in writing, is reproduced without alteration, and is accompanied by all associated conditions, limitations, and notices.

ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the NVIDIA terms and conditions of sale for the product

Trademarks

NVIDIA, the NVIDIA logo, CUDA, Pascal, Tesla, NVLink, and DGX-1 are trademarks or registered trademarks of NVIDIA Corporation in the United States and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright