Generative AI in the Enterprise – Model Training

A Scalable and Modular Production Infrastructure with NVIDIA for AI Large Language Model Training

April 2024

H20003

White Paper

Abstract

This white paper describes the Dell Reference Design for Generative AI Model Training with NVIDIA, a collaboration between Dell Technologies and NVIDIA to enable high performance, scalable, and modular full-stack generative AI model training solutions for large language models in the enterprise.

Dell Technologies AI Solutions

Dell
Reference Design

Copyright

Contents

Introduction

Overview

Generative AI, the branch of artificial intelligence (AI) that is designed to generate new data, images, code, or other types of content that humans do not explicitly program, is rapidly becoming pervasive across nearly all facets of business and technology.

In 2023, Dell Technologies and NVIDIA introduced a groundbreaking project for generative AI, with a joint initiative to bring generative AI to the world's enterprise data centers. This project delivered a set of validated designs for full-stack integrated hardware and software solutions that enable enterprises to create and run custom AI Large Language Models (LLMs) at scale using unique data that is relevant to their own organization.

An LLM is an advanced type of AI model that has been trained on an extensive dataset, typically using deep learning techniques, which is capable of understanding, processing, and generating natural language text. However, AI built on public or generic models is not well suited for an enterprise to use in their business. Enterprise use cases require domain-specific knowledge to train, customize, and operate their LLMs.

LLM training, often called to as pre-training, is the process of training a large-scale neural network model on a vast amount of text data before it is fine-tuned for specific tasks. This pre-training aims to equip the model with a broad understanding of language, including grammar, context, semantics, and common knowledge. The process is fundamental to the creation of foundation models like Llama 2 and GPT (Generative Pre-trained Transformer) and enables them to perform a wide range of language-related tasks, even those they were not explicitly trained for.

Dell Technologies and NVIDIA have designed a scalable, modular, and high-performance architecture that enables enterprises everywhere to create a range of generative AI solutions that apply to their businesses, reinvent their industries, and give them a competitive advantage.

This reference design for training is one in a series of designs for generative AI that focus on all facets of the generative AI life cycle, including inferencing, model customization, and model training. While these designs are focused on generative AI use cases, the architecture is more broadly applicable to more general AI use cases as well.

About this document

This technical white paper is a blueprint for generative AI model training. It describes a reference design for a modular platform for generative AI in the enterprise that focuses specifically on large model training, which is the process of training a model from scratch to be used for inferencing operations.

This document can be read alongside the associated white paper Generative AI in the Enterprise. The white paper provides an overview of generative AI, including its underlying principles, benefits, architectures, and techniques; the various types of generative AI models and how they are used in real-world applications; the challenges and limitations of generative AI; and descriptions of the various Dell and NVIDIA hardware and software components to be used in the architecture.

We also recommend reading the related design guides Generative AI in the Enterprise - Inferencing, which focuses on Inferencing and deployment of pre-trained models and Generative AI in the Enterprise – Model Customization, which focuses on retraining or fine tuning pre-trained models.

Audience

This document guide is intended for anyone that is interested in the implementation of solutions and infrastructure for generative AI, including professionals and stakeholders involved in the development, deployment, and management of generative AI systems. Key roles include IT executives and decision makers such as Chief Technology Officers (CTOs), Chief Information Officers (CIOs), and principal systems architects. Other audience members may include system administrators and IT operations personnel, AI engineers and developers, and data scientists and AI researchers.

Solution overview

This section contains or describes model training and when it would be employed, introduces some of the concepts and considerations that drive the design, and provides a high level view of the solution architecture.

Note: Although some aspects of this document may be considered pre-training, we will generally use the term training.

Model training

Model training in AI is a fundamental process in which a machine learning model learns to recognize patterns and make predictions by analyzing input data. It involves exposing the model to labeled training data, which consists of input-output pairs, and iteratively adjusting its internal parameters to minimize errors in predictions. Through this process, which is known as optimization or learning, the model gradually improves its ability to generalize and make accurate predictions on new, unseen data.

Training from scratch is the process of training a model without leveraging pre-existing knowledge or representations. In other words, the model starts with random or uninitialized parameters, and the training process initializes and updates these parameters solely based on the provided training data. When training from scratch, the model does not benefit from pre-trained weights or representations learned from previous tasks or datasets.

The success of model training hinges on several factors, including the quality and quantity of training data, the choice of algorithm or model architecture, and the optimization techniques employed during the training process. Ultimately, the trained model serves as a powerful tool for making predictions, classifying data, or generating insights in various domains, ranging from natural language processing and image recognition; to textual, audio, and visual content creation; to recommendation systems and numerous other use cases.

Training rationale

Training a model may be preferable to using a foundation model in certain scenarios that require:

- **Data availability:** There is an abundance of high-quality, task-specific data available for training, making it feasible to achieve better performance by

training from scratch. It also allows for complete control over the data used, enabling the removal of biases, inaccuracies, or undesirable content that might be present in the datasets used for existing pre-trained models.

- **Privacy or security concerns:** Pre-trained models may not be suitable due to concerns about sharing sensitive data or intellectual property, particularly in shared environments such as public clouds.
- **Innovation and research:** Training new LLMs provides opportunities for innovation and research, enabling exploration of novel architectures, training methodologies, and applications in natural language processing.
- **Contractual usage terms:** The terms and conditions under which the base LLM can be used may include constraints that do not align with organizational requirements.

In these cases, training a model from scratch allows for greater control over the model's architecture, training process, and performance optimization.

While pre-training a model is resource-intensive, requiring significant computational power, time, and data, there are specific scenarios where the benefits outweigh the costs, such as those described above.

Nonetheless, for most enterprise use cases, fine-tuning an existing pre-trained foundation model is the best approach as it offers a practical balance between customization and resource efficiency. Pre-trained models like GPT or Llama 2 have already learned a vast amount of general knowledge, allowing businesses to tailor these models to specific tasks with minimal additional resources. This process significantly reduces the need for extensive computational power and data, making it a practical choice for achieving high-performance results in applications ranging from customer service automation to content generation.

Solution approach

Generative AI models have been growing in computation requirements. Training a model from scratch is typically a resource-intensive endeavor and can take considerable amounts of time. For example, according to OpenAI, for Chat GPT-3 with 175B parameters, the model size is approximately 350 GB and it would take 34 days to train on1024 NVIDIA A100 GPUs, or 355 years on a single V100 GPU.

As another example, recently Databricks released DBRX, an open general purpose LLM with 132B parameters that was trained using next-token prediction. It was trained on 3072 NVIDIA H100 GPUs and took three months to complete.

Clearly, training LLMs requires significant computational resources, including multiple GPUs and distributed training setups. Training times can vary depending on factors such as the size of the model, the complexity of the task, the size of the dataset, and the hardware infrastructure available. Therefore, the selection of these factors is an important consideration.

There are several aspects to training in order to create a pre-trained LLM and the choices you make during the planning phases have implications on the training time and the resulting performance of the model. The considerations include the acquisition and preparation of the data, the selection of the model architecture, the development, and

training of the model, including tokenization. These steps are explained below, followed by discussion of the design of the infrastructure on which the training will take place.

Training process

The process of training for a large language model typically involves the following steps:

1. Data collection and preprocessing
    - Collect a large corpus of text data from various sources, such as books, articles, websites, and other textual sources, or select an existing dataset. The amount of training data that is used has a direct impact on training time and the performance of the model.
    - Preprocess the text data by tokenizing it into words or subwords, removing punctuation, lowercasing, and other normalization techniques.
2. Architecture selection
    - Choose an appropriate architecture for training your model. Common model architectures used as starting points include Generative Pre-trained Transformers (GPT), Bidirectional Encoder Representations from Transformers (BERT), and LLMs designed for Natural Language Processing (NLP) tasks such as Llama 2.
3. Initialization and parameterization
    - Define the architecture of the model, including the number of layers, hidden units, attention heads, and other hyperparameters.
    - Initialize the model's parameters randomly.
4. Objective function and training task definition
    - Define the pre-training task, which typically involves predicting the next word in a sequence (language modeling) or predicting masked words in a sentence (masked language modeling)
    - Formulate the objective function or loss function based on the pre-training task, such as cross-entropy loss for language modeling or masked language modeling.
5. Training procedure
    - Train the language model on the pre-training task using a large dataset and parallel processing techniques to accelerate training.
    - Utilize techniques like mini-batch training, gradient descent optimization, and regularization methods to optimize the model's parameters and minimize the loss function.

While the next two steps are not within the scope of this document, it is important to understand the full context.

6. Evaluation and fine-tuning
    - Evaluate the performance of the pre-trained language model on held-out validation data to assess its generalization capability.

- Fine-tune the pre-trained model on downstream tasks or specific domains using task-specific labeled data, if preferred.

7. Model deployment
    - Deploy the now pre-trained language model for inference on new text data or integrate it into applications and systems for various natural language processing tasks, such as text generation, text classification, and sentiment analysis.

Overall, the process of pre-training for a large language model involves collecting and preprocessing data, selecting an appropriate architecture, defining the pre-training task, training the model, evaluating its performance, and deploying it for inference on new tasks.

More on model architectures

Transformers are the default choices for NLP applications. Depending on your preferences, some of the key elements to consider for model selection include the number of layers in transformer blocks, the number of attention heads, and loss function and hyperparameters. The size and configuration of the model directly influences the compute time required to train the model. During the training, the model is presented with a sequence of tokens and trained to predict the next token in the sequence. The model adjusts its weights based on the difference between its predicted token and the actual token in the sequence. This process is repeated millions of times, until the model reaches a certain level of performance, or the model has learned enough, and additional training is not changing the model's accuracy.

To decrease training time, different parallelism techniques can be used, such as data parallelism and model parallelism. Model parallelism can be implemented as sequence parallelism, pipeline parallelism, and tensor parallelism. Each of these techniques has unique workload characteristics and infrastructure requirements.

Other architecture considerations

When selecting an architecture for your LLM, several factors should be considered:

- The current state-of-the-art for LLM model is the transformer architecture based on the 2017 paper Attention is All You Need.
- Model architectures with larger parameter counts will have larger training dataset requirements based on the Chinchilla Scaling Laws.
- Decoder only, decoder/encoder and encoder only are different transformer architecture options. Many commercial LLMs used for text summarization, chat, and code generation are based on decoder-only architectures.
- Hyperparameter selection – this includes hyperparameters such as number of layers, number of attention heads, learning rate, and more. Many well-known open source LLMs publish this information so this can be a good starting point for architectures based fully or partly on those models.
- Industry leaderboards such as Hugging Face LLM Leaderboard best models. This leaderboard includes best models based on parameter count, performance on domain-specific datasets, and more. This information may be useful in

deciding on an architecture similar to a high-performing model based on a specific parameter count and dataset may be appropriate.

Infrastructure considerations

There are multiple considerations regarding the various hardware infrastructure components for a generative AI training system, including high performance compute and memory, high-speed networking, and scalable, high-capacity, and low-latency storage to name a few.

Compute

The time to train a large model depends so many things beyond even the number of server nodes and the number and types of GPUs per node, such as parameters, precision, model architecture and different algorithms and techniques used to train the model. On top of various model dependencies, the underlying AI frameworks and libraries may make the training faster and more efficient with each new software release. We can be sure of one thing; model sizes are increasing and even a set of eight XE9680 servers each with 8-way H100 NVIDIA GPU can take up to 4 days to train a smaller 5B parameter LLM.

GPU Memory

Large models also have large memory requirements. For example, a 7B parameter model would need 7x4 = 28GB of GPU memory just to store those parameters in memory at FP32 precision. This is based on the fact that a 32bit floating point value needs 4 bytes of memory, so 7B parameters of FP32 values needs 28GB of memory. Or it would need 14GB of GPU memory at FP16 precision. Since FP16 only requires 2 bytes, it only needs 14GB of memory.

Larger models like ChatGPT3 with 175B parameter would require 350GB of GPU memory just to load the model. The NVIDIA H100 GPU has only 80GB of GPU memory and that means that a 175B model would have to be split across multiple GPUs.

GPU memory is one of the most significant hurdles for training LLMs. The state-of-the-art optimizers like Adam converge much faster than traditional stochastic gradient descent due to tracking the first and second order momentum, however, to track the momentum, Adam must keep two additional values for each parameter in the model, thus adding a 2X memory overhead. For deep learning models like transformers, activations of all layers need to be in memory for backpropagation and this causes memory cost of such models to increase proportionally to number of layers.

GPU Connectivity

Parallelism techniques, like model parallelism, where a language model is split over multiple GPUs with each GPU storing one or more different layers, provides huge improvement in the time to train the model. However it brings the challenge of required data communication between GPUs. Typically, those communications go over the PCIe bus and sometimes may require going over the even slower CPU-to-CPU NUMA interconnect which may not meet the requirements of the performance and throughput needed for this kind of workload. That is why optimized GPU interconnects like NVIDIA NVLink are good solutions for linking GPU pairs in a machine and provide efficient communication when the model is split between 2 GPUs. When the models are split among four or six or eight GPUs then NVIDIA NVSwitch comes into play, where a system

like XE9680 with 8 NVIDIA H100 SXM GPUs delivers all-to-all GPU communication in the box between the GPUs.

Networking

Once we scale the model across sufficient number of GPUs to satisfy the memory capacity requirements using model parallelism, we use data parallelism to further scale the performance and reduce training time This is typically accomplished in 3 steps. First, copies of the model are dispatched to each node (or groups of nodes if using model parallelism along with data parallelism). Then, we shard the data and distribute to n nodes or node-groups. Finally, all results are aggregated for the backpropagation step with an all-reduce operation, requiring low latency and high throughput connection between nodes. Data parallel uses a collective communication library (such as NVIDIA's NCCL) to synchronize the transfer of the gradients.

For distributed training, high-speed networking like InfiniBand, with its low latency and high bandwidth capabilities, significantly reduces the communication overhead between processors. This acceleration is essential for synchronizing large volumes of data and parameters efficiently, ensuring that the distributed components of the model can be updated at the lowest possible latency. NVIDIA's GPUDirect RDMA further enhances data transfer efficiency by allowing direct, high-speed, low-latency transfers between GPUs across multiple nodes, bypassing the traditional multistep data copy process.

Storage

As mentioned, training large models takes a significant amount of time on a large number of GPUs. Thus, there is a requirement to do application level checkpointing. In case the training is disrupted for any reason, one can restart the training by restoring the last known good checkpoint. In case of LLM training, checkpointing is essentially a snapshot of the model parameters (weights) after some defined time period or defined number of steps.

This is where high performance storage comes into play as the size of the checkpoint file depends on the size of the model and the number of checkpoint files depends on how many GPUs hold that model in memory during the training. For example, 175B parameters trained with tensor parallelism = 8 and pipeline parallelism = 8 means, 8 GPUS in 8 nodes working together on training that model – thus 64 GPUs hold a unique copy of the model, and each GPU will create one 40GB checkpoint file and the model will create approximately 2.4TB of data from 64 files. Checkpoint writes are single threaded from each GPU and checkpoint reads are simultaneously read by all 64 GPUs for restore if needed.

Architecture overview

This reference design for generative AI model training is designed to address the challenges of training LLMs for enterprise use cases. LLMs have shown tremendous potential in natural language processing tasks but require specialized infrastructure for efficient development and deployment.

This design serves as a starting point, offering organizations guidelines and best practices to implement scalable, efficient, and reliable infrastructure specifically tailored for generative AI model training. While its primary focus is LLM training, the architecture can be adapted for AI model customization and inferencing, as explained in the associated papers.

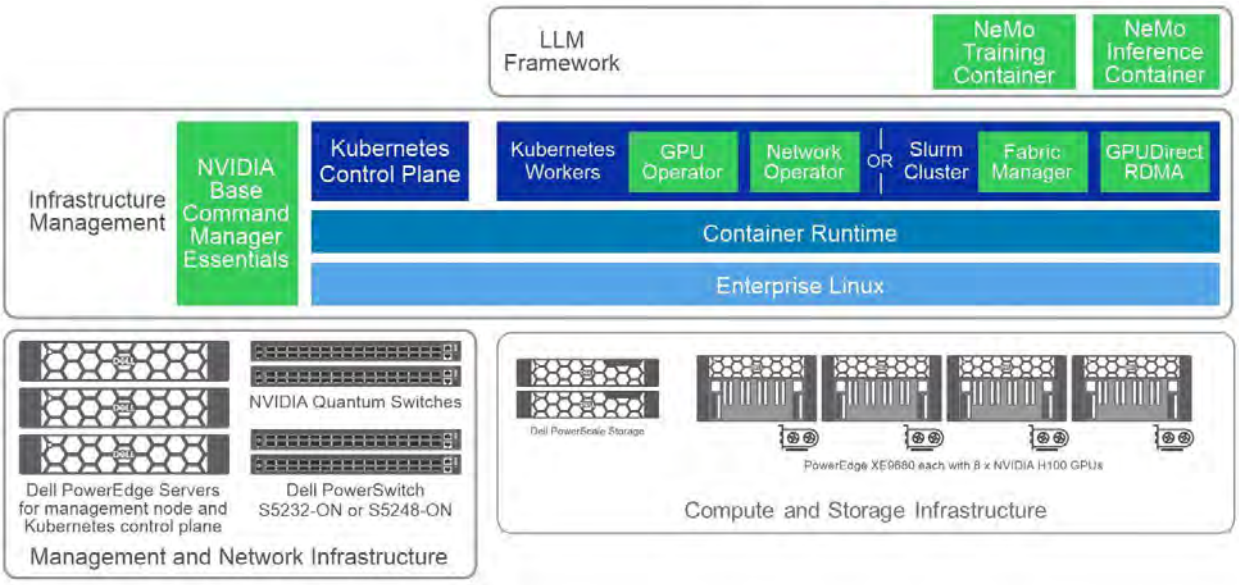

Figure 1. High level solution architecture

The solution design presented here is modular and each of the components can be independently scaled depending on the customer's workflow and application requirements.

Servers and GPUs

Dell Technologies offers a range of acceleration-optimized servers with an extensive portfolio of NVIDIA GPUs. The Dell PowerEdge XE9680 server is featured in this design for generative AI training.

The PowerEdge adaptive compute approach enables servers engineered to optimize the latest technology advances for predictable profitable outcomes. The improvements in the PowerEdge portfolio include:

- Focus on acceleration: Support for the most complete portfolio of GPUs, delivering maximum performance for AI, machine learning, and deep learning

training and inferencing, high performance computing (HPC) modeling and simulation, and advanced analytics.

- Optimized thermal design: New thermal solutions and designs to address dense heat-producing components, and in some cases, front-to-back, air-cooled designs.
- Dell multivector cooling: Streamlined, advanced thermal design for airflow pathways within the server

The primary hardware components used in this solution are described below.

PowerEdge XE9680 server

The PowerEdge XE9680 server is a high-performance server made for demanding AI, machine learning, and deep learning workloads that enable you to rapidly develop, train, and deploy large machine learning models.

The PowerEdge XE9680 server is the industry's first server to ship with eight NVIDIA H100 GPUs and NVIDIA AI software. It is designed to maximize AI throughput, providing enterprises with a highly refined, systemized, and scalable platform to help them achieve breakthroughs in NLP, recommender systems, data analytics, and more. Its 6U air-cooled design chassis supports the highest wattage next-generation technologies up to 35C ambient. And it features high-speed networking with NVIDIA ConnectX-7 smart network interface cards (SmartNICs).

NVIDIA H100 Tensor Core GPU

The NVIDIA H100 Tensor Core GPU delivers unprecedented performance, scalability, and security for every workload. With NVIDIA fourth-generation NVLink Switch System, the NVIDIA H100 GPU accelerates AI workloads with a dedicated Transformer Engine for trillion parameter language models. The NVIDIA H100 GPU uses breakthrough innovations in the NVIDIA Hopper architecture to deliver industry-leading conversational AI, speeding up large language models by 30 times over the previous generation.

Dell PowerScale storage

Dell PowerScale supports the most demanding AI workloads with all-flash NVMe file storage solutions that deliver massive performance and efficiency in a compact form factor. There are several models used in the generative AI solution architecture, all powered by the PowerScale OneFS operating system and supporting inline data compression and deduplication. PowerScale clusters can scale up to 252 nodes, 186PB of capacity, and over 2.5TB read/write throughput within a single namespace.

PowerScale was originally designed and optimized for industries with workloads that require extreme read and write concurrency. This long heritage and deep in-market experience ideally positioned PowerScale to meet the AI infrastructure market's needs when it emerged.

PowerScale F710

PowerScale's continuous innovation extends into the AI era with the introduction of the next generation of PowerEdge-based nodes, including the PowerScale F710. The new PowerScale all-flash nodes leverage Dell PowerEdge 16G servers, unlocking the next generation of performance. On the software front, the F710 takes advantage of significant performance improvements in PowerScale OneFS 9.7. Combining the latest hardware and software innovations, the F710 can tackle the most demanding workloads with ease.

Networking components

Future-ready networking technology helps you improve network performance that lowers overall costs and network management complexity and experience flexibility to adopt new innovations.

Dell PowerSwitch Z9432F-ON

The Dell PowerSwitch Z9432F-ON 100/400GbE fixed switch consists of Dell's latest disaggregated hardware and software data center networking solutions, providing state-of-the-art, high-density 100/400 GbE ports and a broad range of functionality to meet the growing demands of today's data center environment. This innovative, next-generation open networking high-density aggregation switch offers optimum flexibility and cost-effectiveness for the Web 2.0, enterprise, mid-market, and cloud service providers with demanding compute and storage traffic environments.

NVIDIA NVLink and NVSwitch

NVIDIA NVLink is a fast, scalable interconnect that enables multi-GPU systems with seamless, high-speed communication between every GPU. The fourth generation of NVIDIA NVLink technology provides 1.5 times higher bandwidth and improved scalability for multi-GPU system configurations. A single NVIDIA H100 Tensor Core GPU supports up to 18 NVLink connections for a total bandwidth of 900 gigabytes per second (GB/s), over seven times the bandwidth of PCIe Gen5.

NVIDIA QM9790

The NVIDIA Quantum-2-based QM9700 and QM9790 switch systems deliver 64 ports of NDR 400Gb/s InfiniBand per port in a 1U standard chassis design. A single switch carries an aggregated bidirectional throughput of 51.2 terabits per second (Tb/s), with a landmark of more than 66.5 billion packets per second (BPPS) capacity.

Network architecture

Figure 2 shows the network architecture. It shows the network connectivity for the PowerEdge training nodes, PowerScale storage, and the three control plane nodes that incorporate NVIDIA Base Command Manager Essentials and other software components.

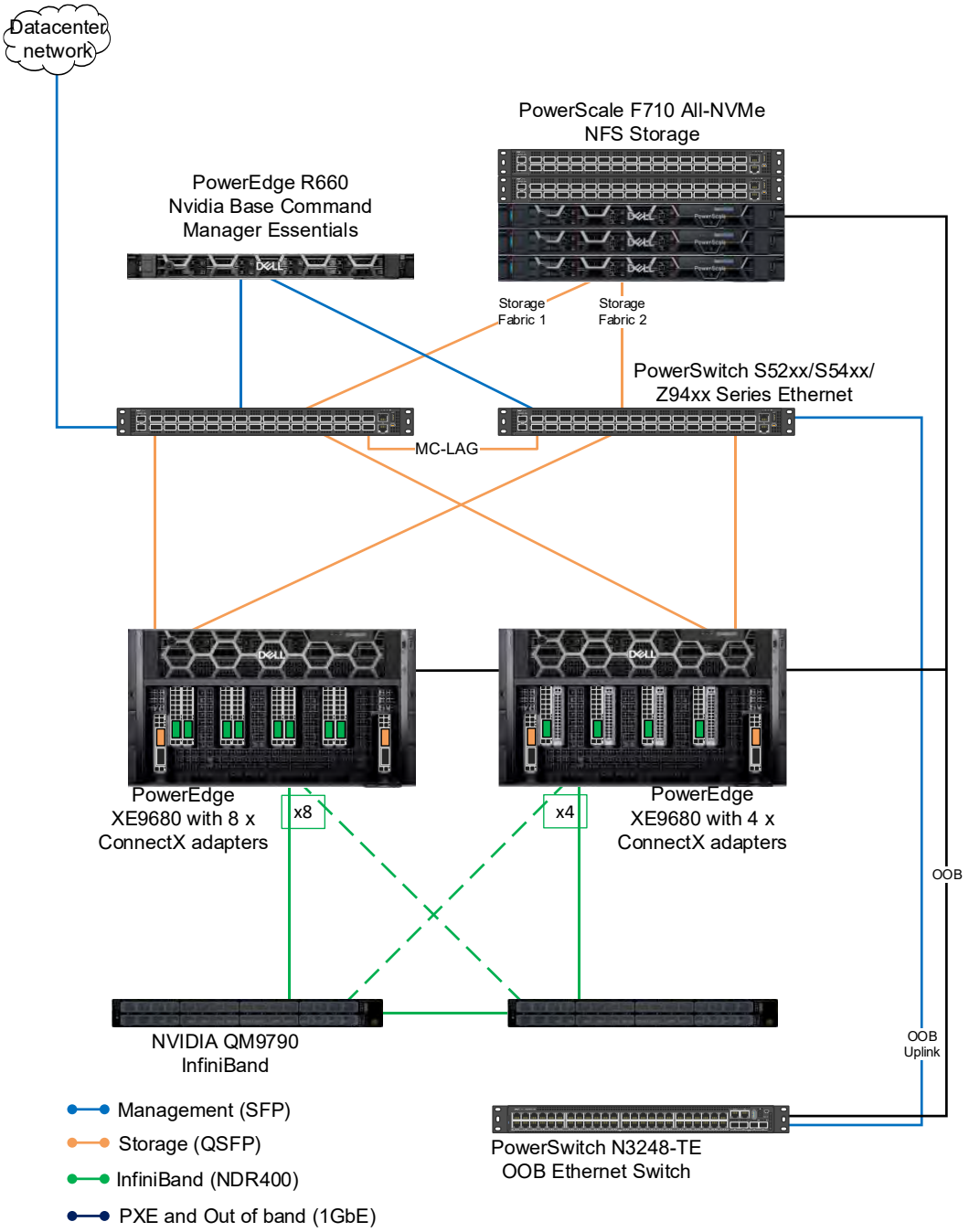

Figure 2. Networking design showing two options for XE9680 connectivity

In this reference design, we evaluated two network configurations. The first setup equipped each XE9680 with 8 NVIDIA ConnectX-7 SmartNIC network adapters, ensuring a dedicated network port for each GPU. In contrast, the second setup featured 4 NVIDIA ConnectX-7 SmartNIC network adapters per XE9680. Our objective was to validate both configurations by measuring their training times. This comparison aims to provide readers with the data necessary to choose the most suitable configuration for their needs.

Our network design accommodates a minimum of eight PowerEdge XE9680 servers, each equipped with 8 NVIDIA Connect-X 7 adapters. When deploying eight servers, we establish 32 inter-switch links to guarantee unblocked communication on two 64 port QM970.

Rail-optimized network architecture

LLM training often necessitates clusters significantly larger than 8 nodes. When scaling beyond eight servers, the network architecture must be carefully considered. In the PowerEdge XE9680 model, each NVIDIA GPU can be equipped with a dedicated InfiniBand network adapter, enabling communication with GPUs in other servers. This communication, facilitated by GPUDirect RDMA, doesn't interrupt the CPU, thereby reducing latency. Collective communications are crucial for the performance of modern distributed LLM training. The NVIDIA Collective Communication Library (NCCL), part of the Magnum IO Library, implements GPU-accelerated collective operations like all-gather and all-reduce. NCCL is aware of the network topology and is optimized to achieve high bandwidth and low latency over various interconnects, including PCIe, NVLink, and InfiniBand.

A new feature, known as PXN (PCI × NVLink), was introduced in NCCL 2.12. This feature allows a GPU to communicate with a NIC on the node via NVLink and then PCI, bypassing the CPU and avoiding the use of QPI or other inter-CPU protocols, which can't deliver full bandwidth. As a result, each GPU can access other NICs as needed, even though it primarily uses its local NIC.

Considering this, a rail optimized network architecture is recommended for large cluster designed for LLM training. This architecture consists of leaf-spine network switches as shown in the figure.

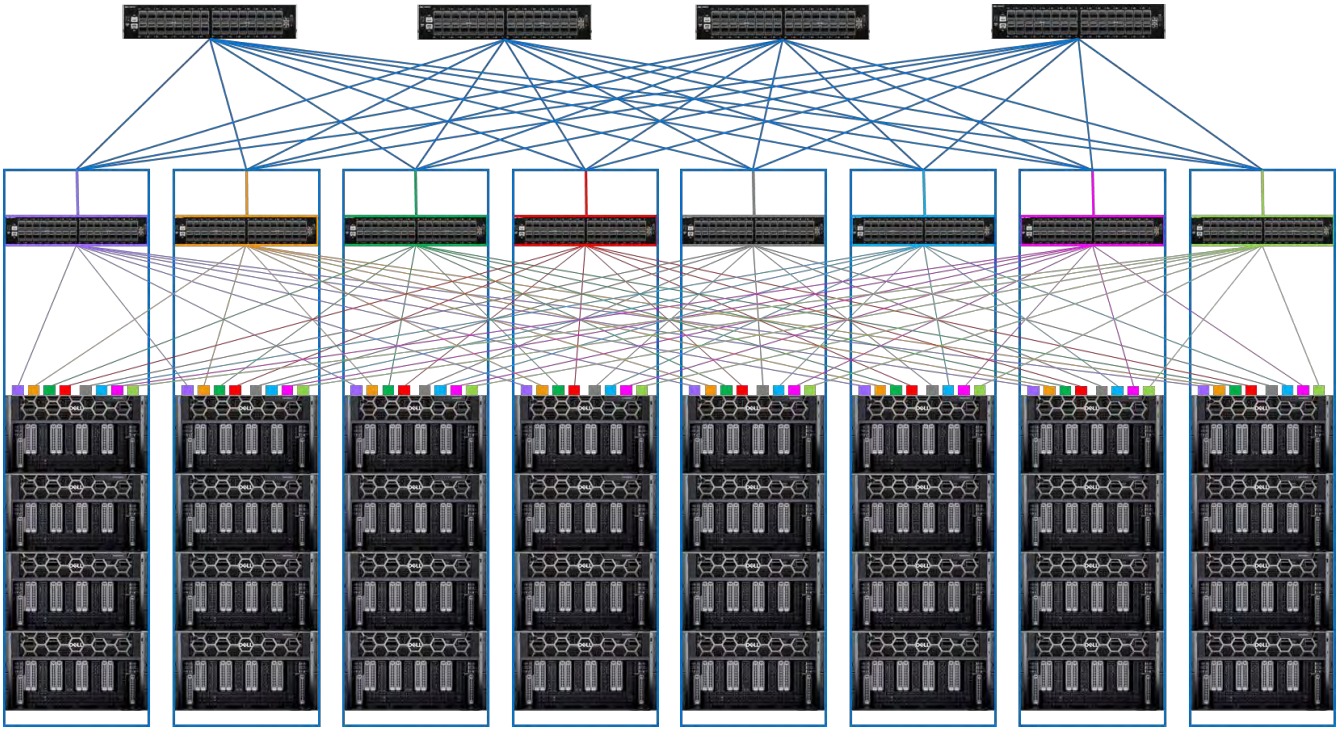

Figure 3. Rail optimized network architecture for distributed LLM training

The rail-optimized network architecture consists of the following design elements:

- Each of the eight pods houses 4 PowerEdge XE9680 servers and one PowerSwitch Z9664F-ON, which functions as a spine switch.
- Each of PowerEdge XE9680 server is equipped with 8 InfiniBand network adapters (NVIDIA Connect-X 7) that connect to the PowerSwitch Z9664F-ON spine switch.
- Each PowerSwitch Z9664F-ON spine switch has 64 ports. Half of these ports (32) connect to the PowerEdge XE9680 servers, while the remaining half connect to 4 x PowerSwitch Z9664F-ON leaf switches.
- Each leaf switch is connected to each spine switch using 8 links. This setup results in a 1:1 subscription ratio between the server to the leaf switches and leaf to spine switches.

Software design

The NVIDIA AI software stack is the primary software used in this design. NVIDIA enterprise software solutions are designed to give IT admins, data scientists, architects, and designers access to the tools they need to easily manage and optimize their accelerated systems.

NVIDIA AI Enterprise

NVIDIA AI Enterprise, the software layer of the NVIDIA AI platform, accelerates the data science pipeline and streamlines development and deployment of production AI including generative AI, computer vision, speech AI and more. This secure, stable, cloud-native platform of AI software includes over 100 frameworks, pretrained models, and tools that accelerate data processing, simplify model training and optimization, and streamline deployment. Some of the capabilities include:

- **Data preparation**: Improve data processing time by up to 5 times while reducing operational costs by 4 times with the NVIDIA RAPIDS Accelerator for Apache Spark.
- **AI Training**: Create custom, accurate models in hours, instead of months, using NVIDIA TAO Toolkit and pretrained models.
- **Optimization for inference**: Accelerate application performance up to 40 times over CPU-only platforms during inference with NVIDIA TensorRT.
- **Deployment at scale**: Simplify and optimize the deployment of AI models at scale and in production with NVIDIA Triton Inference Server.

Since the full stack is maintained by NVIDIA, organizations can count on regular security reviews and patching, API stability, and access to NVIDIA AI experts and support teams to ensure business continuity and AI projects stay on track.

NVIDIA NeMo

NVIDIA AI Enterprise includes NVIDIA NeMo, a framework to build, customize, and deploy generative AI models with billions of parameters. The NVIDIA NeMo framework provides an accelerated workflow for training with 3D parallelism techniques. It offers a choice of several customization techniques and is optimized for at-scale inference of large-scale models for language and image applications, with multi-GPU and multinode

configurations. NVIDIA NeMo makes generative AI model development easy, cost-effective, and fast for enterprises.

NeMo framework's AutoConfigurator searches for the Hyper-Parameters (HPs) that achieve the highest throughput for training and inference for LLMs using NeMo-Framework. AutoConfigurator is intended to quickly iterate over different model configurations, such as parallelism, to find the best configuration with minimal time and money spending.

NVIDIA Base Command Manager Essentials

NVIDIA AI Enterprise includes NVIDIA Base Command Manager Essentials. a NVIDIA's cluster manager for AI infrastructure. It facilitates seamless operationalization of AI development at scale by providing features like operating system provisioning, firmware upgrades, network and storage configuration, multi-GPU and multinode job scheduling, and system monitoring, thereby maximizing the utilization and performance of the underlying hardware architecture.

NVIDIA BCM supports automatic provisioning and management of changes in nodes throughout the cluster's lifetime. With an extensible and customizable framework, it has seamless integrations with the multiple HPC workload managers, including Slurm IBM Spectrum LSF, OpenPBS, Univa Grid Engine, and others. It offers extensive support for container technologies including Docker, Harbor, Kubernetes, and operators. It also has a robust health management framework covering metrics, health checks, and actions. NVAIE suite of products and licenses can be obtained when buying your Dell system.

Validation findings

This Dell Reference Design for Generative AI Model Training aims to simplify and accelerate the deployment of complex infrastructure for generative AI by providing proven reference architectures. It helps customers by reducing the guesswork and potential risks associated with designing and implementing initial custom solutions.

We validated our training with Llama 2 model architectures with our reference design to ensure the Dell and NVIDIA hardware and software are optimized and integrated to deliver reliable and high-performance solutions.

Validation system configurations

We validated our reference design with up to six PowerEdge XE9680 nodes. In most cases, pretraining models with billions of parameters necessitate a significantly larger cluster, often comprising hundreds of GPUs. Our validation aims to demonstrate the feasibility of pretraining using Dell infrastructure equipped with NVIDIA GPUs and software. Furthermore, the cluster size we employed for validation should adequately support pretraining models containing billions of parameters.

The following tables list the hardware configurations and software components used for generative AI model training in this design.

Table 1. System configuration

Component	Configuration 1	Configuration 2
Compute server for model customization	6 x PowerEdge XE9680 servers	6 x PowerEdge XE9680 servers
GPUs per server	8 x NVIDIA H100 SXM GPUs	8 x NVIDIA H100 SXM GPUs
Ethernet Network adapters	2 x NVIDIA ConnectX-6 DX Dual Port 100 GbE	2 x NVIDIA ConnectX-6 DX Dual Port 100 GbE
Ethernet Network switch	2 x PowerSwitch S5232F-ON	2 x PowerSwitch S5232F-ON
InfiniBand Network adapter	8 x NVIDIA ConnectX-7, Single Port NDR OSFP PCIe, No Crypto, Full Height	4 x NVIDIA ConnectX-7, Single Port NDR OSFP PCIe, No Crypto, Full Height
InfiniBand Network switch	QM9790	QM9790

We tested three different size deployments, consisting of 2, 4, and 6 node PowerEdge XE9680 servers. We evaluated two network configurations:

- Configuration 1: Each PowerEdge XE9680 is equipped with 8 x NVIDIA ConnectX-7 InfiniBand adapters.
- Configuration 2: Each PowerEdge XE9680 is equipped with 4 x NVIDIA ConnectX-7 InfiniBand adapters.

Table 2. Software components and versions

Component	Details
Operating system	Ubuntu 22.04.1 LTS
Cluster management	NVIDIA Base Command Manager Essentials 10.23.12
Slurm cluster	Slurm 23.02.4
AI framework	NVIDIA NeMo Framework v23.11

A Slurm cluster, powered by the "Simple Linux Utility for Resource Management" software, is a high-performance computing environment that efficiently manages and schedules computing tasks across multiple nodes. This open-source system is efficient at job scheduling, tracking resource availability, and prioritizing tasks based on user-defined requirements. It uses job queues and provides fairness mechanisms, ensuring that higher-priority jobs are accommodated without neglecting lower-priority ones.

Slurm offers access control features, facilitating user management and access policies, and is designed to handle node failures gracefully, redistributing jobs to maintain efficiency. We used Slurm for LLM training as it offers seamless scheduling components like batch scheduling, preemption, and multiple queues, making it efficient for orchestrating long-running tasks such as LLM training.

Validation results

Model training or pretraining yields a foundational LLM by training it on a large corpus of data. We validated our design to ensure the functionality of model training technique available in the NeMo framework. Our goal in this validation was not to train a model to convergence and generate a complete foundational model, but rather to train for a defined number of steps in order to achieve the goals described here.

The following list provides the details of our validation setup:

- **Model architectures**: We trained primarily with 7B and 70B Llama 2 model architectures. We also trained with GPT model architectures.
- **Foundation model pre-training using NeMo Framework**: See the NeMo documentation for available playbooks.
- **Cluster configuration:** We used a Slurm for cluster management and job scheduling.
- **Dataset**: We used Pile datasets for this validation. The Pile is an 825 GiB diverse, open source language modelling data set that consists of 22 smaller, high-quality datasets combined together, derived primarily from academic or professional sources.
- **Time for training**: Usually, data scientists train a model until it reaches convergence, a point influenced by factors like the dataset, model complexity, and chosen hyperparameters. Our aim was not to achieve convergence for every scenario, as it is specific to our chosen dataset and parameters, offering limited insight into a customers' needs. To maintain a consistent metric across all scenarios, we conducted training jobs for a minimum of 500 steps.

Model architecture selection

Among the various LLMs available, we selected the 7B and 70B parameters of Llama 2 architectures to use for training, based on several key factors:

- Resource use: Using these two models sizes helped us better understand the infrastructure resource usage and requirements for various training workloads for a range of model sizes.
- Ease of use: The models have been readily available for consumption, along with recipes and cookbook implementations, making modification to the codebase easier for customers' use cases.

Parallelism

Below are the tensor and pipeline parallelism values we used for the models. Tensor parallelism and pipeline parallelism are generated by NeMo Megatron launcher based on model parameter size and number of GPUs.

Table 3. Parallelism for Llama 2 architecture for training

Model	Configuration
Llama 2 7B	Tensor Parallelism = 2 Pipeline Parallelism = 1 Micro batch size = 1 Global batch size = 144 Sequence length = 4096

Model	Configuration
Llama 2 70B	Tensor Parallelism = 4 Pipeline Parallelism = 4 Micro batch size = 1 Global batch size = 144 Sequence length = 4096

Training times

The following table shows the time that we measured for model pretraining. It includes the time to train the model for 500 steps. It does not include time to initialize the model, load the dataset, checkpointing and validation.

Table 4. Time in minutes for model training for various models[1]

Models	Number of nodes	Training time for 500 steps in Configuration 1	Training time for 500 steps in Configuration 2
Llama 2 7B	2	61	63
Llama 2 7B	4	32	35
Llama 2 7B	6	22	25
Llama 2 70B	6	230	244

With regard to performance results, please note the following:

- **Training time for LLMs:** Typically, pretraining a large language model (LLM) requires around 100,000 steps or more. However, in our case, we've conducted validation using only 500 steps. The purpose of this validation is to demonstrate the functionality of both the hardware and software stack. For more detailed information on estimated training times for various model types, please refer to the NeMo documentation.
- **Comparison of configurations:** We evaluated two configurations:
    - Configuration 1: Each PowerEdge XE9680 is equipped with 8 x ConnectX InfiniBand adapters.
    - Configuration 2: Each PowerEdge XE9680 is equipped with 4 x ConnectX InfiniBand adapters.

  While Configuration 1 performs slightly better, the improvement is not significant. For clusters with fewer than 8 nodes, Configuration 2 may offer a better cost-benefit ratio. However, in large clusters, network communication becomes crucial, and we recommend equipping each PowerEdge XE9680 server with 8 NVIDIA ConnectX-7 adapters.

[1] All performance data contained in this report was obtained in a rigorously controlled environment. Results obtained in other operating environments may vary significantly. Dell Technologies does not warrant or represent that a user can or will achieve similar performance results.

Note that training results are highly dependent upon workload, specific application requirements, and system design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, this workload should not be used as a substitute for a specific customer application benchmark when critical capacity planning and/or product evaluation decisions are contemplated. For benchmarking on PowerEdge server, refer to MLPerf benchmarking page.

Conclusion

The Dell Reference Design for Generative AI Model Training, developed in collaboration with NVIDIA, provides a comprehensive, scalable, and high-performance architecture for training large language models (LLMs). This design addresses the challenges of LLM training, offering a modular solution that can be tailored to various enterprise use cases.

The design leverages the power of NVIDIA's AI software stack, including NVIDIA AI Enterprise and NVIDIA NeMo, to streamline the development and training of generative AI models. It also provides a robust Dell infrastructure for efficient model training, with considerations for network architecture, software design, and parallelism techniques to optimize training times.

The validation of this design using Llama 2 model architectures demonstrates its effectiveness in delivering reliable and high-performance solutions for generative AI model training. The design offers flexibility in terms of network configurations and model architectures, allowing organizations to choose the most suitable setup for their needs.

In conclusion, the Dell Reference Design for Generative AI Model Training serves as a valuable guide for organizations looking to harness the power of generative AI. It simplifies the deployment of complex infrastructure for generative AI, reducing potential risks associated with designing and implementing custom solutions. This design, therefore, plays a crucial role in enabling enterprises to leverage generative AI to reinvent their industries and gain a competitive advantage.

While this design focuses on model training, also known as pre-training, it is the third in a series of validated designs for generative AI that focus on all facets of the generative AI life cycle, including training, model customization, and inferencing. While these designs focus on generative AI use cases, the architecture is more broadly applicable to more general AI use cases as well.

With this project, Dell Technologies and NVIDIA enable organizations to deliver full-stack generative AI solutions built on the best of Dell infrastructure and software, combined with the latest NVIDIA accelerators, AI software, and AI expertise. This combination of components enables enterprises to use purpose-built generative AI on-premises to solve their business challenges. Together, we are leading the way in driving the next wave of innovation in the enterprise AI landscape.

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on this document and the information that it contains. Please contact the Dell Technologies Solutions team by email.

References

The following links provide additional information about the solution design and components in this paper.

Dell Technologies documentation

The following Dell Technologies web sites and documentation provide additional and relevant information.

- Dell Technologies AI Solutions
- Dell Technologies Info Hub for AI
- White Paper – Generative AI in the Enterprise
- Design Guide – Generative AI in the Enterprise - Inferencing
- Dell PowerEdge XE Servers
- Dell PowerEdge Accelerated Servers and Accelerators (GPUs)
- Dell PowerScale Storage
- Dell ECS Enterprise Object Storage
- Dell ObjectScale Storage
- Dell PowerSwitch Z-series Switches
- Dell OpenManage Systems Management
- Dell Professional Services for Generative AI

NVIDIA documentation

The following NVIDIA web sites and documentation provide additional and relevant information.

- NVIDIA AI Enterprise NVIDIA NeMo
- Nvidia Base Command Manager Essential
- NVIDIA Data Center GPUs
- NVIDIA Networking