White paper
Cisco public

Generative AI Inferencing Use Cases with Cisco UCS X-Series M7 Blade Servers Using 5th Gen Intel Xeon Scalable Processors

March 26, 2024

Contents

Executive summary 3

Challenges 3

Solution overview 3

Technology overview 5

Cisco UCS X-Series Modular System 5

Cisco UCS X9508 Chassis 6

Cisco UCS X210c M7 Compute Node 7

Cisco UCS Virtual Interface Cards (VICs) 8

Cisco UCS 6500 Series Fabric Interconnect 9

Cisco Intersight 10

Cisco Nexus Switching Fabric 11

Intel Xeon Scalable Processor family 11

5th Gen Intel Xeon Scalable processors 12

Intel Advanced Matrix Extensions (Intel AMX) 13

Intel Extension for PyTorch (IPEX) 14

Optimizations for DeepSpeed 14

Large Language Models (LLMs) 15

Meta's Llama 2 models 15

Setup details 16

Hardware and software components 16

Cisco UCS X-Series configuration - Cisco Intersight Managed Mode 17

Intel IPEX with DeepSpeed tests 18

Performance results 22

Meta Llama 2 inference results on 5th Gen Intel Xeon Scalable processor 22

Conclusion 25

References 26

Executive summary

Generative AI is revolutionizing industries, enabling text-to-image generation, realistic voice synthesis, and even the creation of novel scientific materials. However, unleashing the full potential of these powerful models requires a robust and optimized infrastructure. Generative AI models typically require massive amounts of data and complex algorithms, leading to significant computational demands during inference. Challenges include:

- **High computational workloads:** Inference often involves processing large amounts of data through complex neural networks, requiring high-performance computing resources.
- **Memory bandwidth demands:** Large models often require substantial memory bandwidth to handle data transfer efficiently.
- **Latency requirements:** Many applications require low latency inference to ensure real-time responsiveness.

Challenges

Generative AI workloads are characterized by their massive data-processing requirements, complex algorithms, and distributed computing architectures. Deploying Generative AI at scale presents unique challenges:

- **Intensive compute needs:** Large-scale models are computationally expensive. Suboptimal compute resources can result in slower response in practical scenarios. Moreover, different stages of the inference pipeline, from pre-processing to post- processing, require diverse compute capabilities.
- **Model complexity and size:** Large Language Models (LLMs) can have billions of parameters, potentially exceeding the memory capacity of single devices. Distributed inferencing across multiple machines can introduce complexities in model partitioning and require model optimization techniques.
- **Massive network demands:** Real-time responsiveness is often crucial for AI applications, making low latency critical. Large-scale models, distributed across servers, can generate high volumes of traffic between servers. Any degradation in performance will affect the Job Completion Time (JCT).
- **Infrastructure complexity:** Managing and orchestrating large-scale AI deployment requires robust infrastructure and intelligent automation.

Solution overview

A solution based on Cisco UCS® with Intel® Xeon® Scalable processors and Cisco Nexus® offers a compelling and scalable foundation for deploying generative AI at scale. This architecture offers a combination of:

- **Optimal performance:** Cisco UCS with Intel Xeon Scalable processors with specialized AI accelerators and optimized software frameworks significantly improves inferencing performance and scalability. Cisco Nexus 9000 Switches provide high bandwidth, low latency, congestion management mechanisms, and telemetry to meet the demanding networking requirements of AI/ML applications.
- **Balanced architecture:** Cisco UCS excels in both deep-learning and non-deep-learning compute, critical for the entire inference pipeline. This balanced approach leads to better overall performance and resource utilization.
- **Scalability on demand:** Cisco UCS seamlessly scales with your generative AI inferencing needs. Add or remove servers, adjust memory capacities, and configure resources in an automated manner as your models evolve and workloads grow using Cisco Intersight®.

The Cisco UCS X-Series Modular System, and C240 and C220 rack servers, support 5th Gen Intel Xeon Scalable processors so that you have the option to run inferencing in the data center or at the edge, using either a modular or a rack form factor.

Note: In this paper we have used Cisco UCS X-Series Modular System to run generative AI LLM benchmarks with Intel Extensions for PyTorch (IPEX) and DeepSpeed.

5th Gen Intel Xeon processors are engineered to seamlessly handle demanding AI workloads, including inference and fine-tuning on models containing up to 20 billion parameters, without an immediate need for additional hardware. Furthermore, the compatibility of 5th Gen with 4th Gen Intel Xeon processors facilitates a smooth upgrade process for existing solutions, minimizing the need for extensive testing and validation.

Intel Xeon processors are equipped with:

- Intel Advanced Matrix Extensions (Intel AMX) accelerator, an AI accelerator, is built into each core to significantly speed up deep-learning applications when 8-bit integer (INT8) or 16-bit float (bfloat16) datatypes are used.
- Higher core frequency, larger last-level cache, and faster memory with DDR5 speed up compute processing and memory access.
- Intel Advanced Vector Extensions 512 (Intel AVX-512) help with non-deep-learning vector computations.
- Improved cost-effectiveness is provided by combining the latest-generation AI hardware with software optimizations, which potentially lowers TCO by enabling the use of built-in accelerators to scale-out inferencing performance rather than relying on discrete accelerators, making generative AI more accessible and affordable.
- DeepSpeed provides high-performance inference support for large transformer-based models with billions of parameters, through enablement of multi-CPU inferencing. It automatically partitions models across the specified number of CPUs and inserts necessary communications to run multi-CPU inferencing for the model.

Intel has integrated optimizations into both Intel Extensions for PyTorch (IPEX) and DeepSpeed, enabling users to seamlessly use the DeepSpeed trained models to:

- Run on 5th Gen Intel Xeon Scalable processors without any modification.
- Run across multiple cores.
- Fully utilize CPU cores and reduce memory footprint, because some transformer stages are fused to run together.
- Be bound to cores, thus reducing interference.

The integration of 5th Gen Intel Xeon processors with Cisco UCS X-Series ensures seamless pairing, providing a wide range of options and scalability in performance.

Cisco UCS, powered by Intel Xeon Scalable processors, delivers a compelling solution for overcoming these challenges and maximizing generative AI performance. Some of the benefits are:

- **Faster inference:** Cisco UCS with Intel Xeon processors delivers faster inference speeds, enabling real-time applications and lower latency response times.
- **Increased model throughput:** the platform efficiently handles large-scale generative AI workloads, allowing you to process more data and achieve higher throughput.
- **Reduced operational costs:** lower TCO and improved energy efficiency lead to significant cost savings, making generative AI more accessible and affordable.
- **Simplified infrastructure management:** Cisco Intersight streamlines infrastructure management, freeing up valuable resources to focus on innovation and development. Cisco UCS X-Series Modular System, and C240 and C220 rack servers support 5th Gen and 4th Gen Intel Xeon Scalable processors so that you have the option to run inferencing in the data center or at edge using either a modular or a rack form-factor.

Technology overview

Cisco UCS X-Series Modular System

The Cisco UCS X-Series (Cisco UCSX) is a modular, next-generation data center platform that builds on the unique architecture and advantages of the previous Cisco UCS 5108 Server Chassis system. The X-Series is a standards-based open system designed to be deployed and automated quickly in a hybrid-cloud environment. The following key enhancements in Cisco UCS X-Series simplify IT operations:

- **Cloud-managed infrastructure:** with the Cisco UCS X-Series, the management of the network infrastructure is moved to the cloud, making it easier and simpler for IT teams to respond quickly and at scale to meet the needs of your business. The Cisco Intersight cloud-operations platform allows you to adapt the resources of the Cisco UCS X-Series Modular System to meet the specific requirements of a workload. Additionally, you can seamlessly integrate third-party devices such as Pure Storage and VMware vCenter. This integration also enables global visibility, monitoring, optimization, and orchestration for all your applications and infrastructure.
- **Adaptable system designed for modern applications:** today's cloud-native and hybrid applications are dynamic and unpredictable. Application and DevOps teams frequently deploy and redeploy resources to meet evolving requirements. To address this, the Cisco UCS X-Series provides an adaptable system that does not lock you into a fixed set of resources. It combines the density, manageability, and efficiency of blade servers with the expandability of rack servers, allowing you to consolidate multiple workloads onto a single platform. This consolidation results in improved performance, automation, and efficiency for both hybrid and traditional data center applications.
- **Platform engineered for the future:** the Cisco UCS X-Series is designed to adapt to emerging technologies with minimal risk. It is a modular system that can support future generations of processors, storage, nonvolatile memory, accelerators, and interconnects. This eliminates the need to purchase, configure, maintain, power, and cool separate management modules and servers. Cloud-based management through Cisco Intersight ensures automatic updates and access to new capabilities delivered through a software-as-a-service model.
- **Broad support for diverse workloads:** the Cisco UCS X-Series supports a broad range of workloads, reducing the need for different products; this lowers both support and training costs and provides greater flexibility in your data center environment.

Cisco UCS X9508 Chassis

The Cisco UCS X9508 Chassis is engineered to be adaptable and flexible. With a midplane-free design, I/O connectivity for the X9508 chassis is accomplished with front-loading vertically oriented computing nodes that intersect with horizontally oriented I/O connectivity modules in the rear of the chassis. A unified Ethernet fabric is supplied with the Cisco UCS 9108 25G Intelligent Fabric Modules (IFMs). Cisco UCS X9508 Chassis' superior packaging enables larger compute nodes, thereby providing more space for actual compute components, such as memory, GPU, drives, and accelerators. Improved airflow through the chassis enables support for higher power components, and more space allows for future thermal solutions (such as liquid cooling) without limitations.

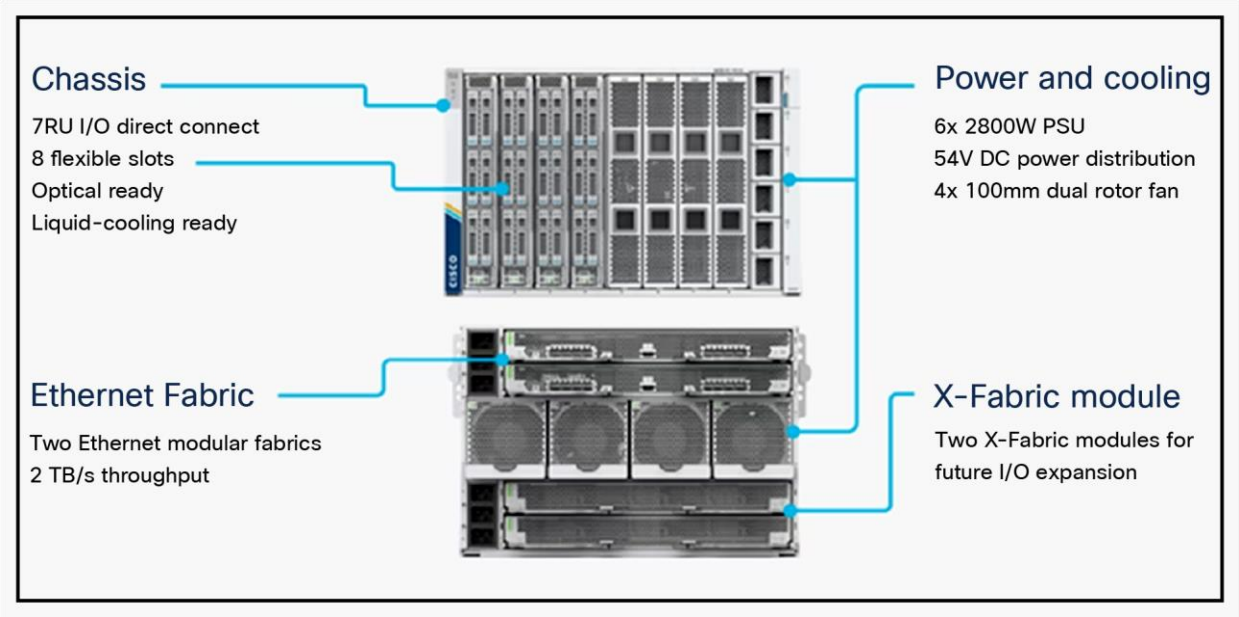


Figure 1.
Cisco UCS X9508 Chassis

The Cisco UCS X9508 Chassis (Figure 1) provides the following features and benefits:

- The 7RU chassis has 8 front-facing flexible slots. These slots can house a combination of computing nodes and a pool of future I/O resources, which may include Graphics Processing Unit (GPU) accelerators, disk storage, and nonvolatile memory.
- Two Cisco UCS 9108 IFMs at the top of the chassis connect the chassis to upstream Cisco UCS 6400 Series Fabric Interconnects (FIs). Each IFM offers these features:
  - The module provides up to 100 Gbps of unified fabric connectivity per computing node.
  - The module provides eight 25-Gbps Small Form-Factor Pluggable 28 (SFP28) uplink ports.
  - The unified fabric carries management traffic to the Cisco Intersight cloud-operations platform, Fibre Channel over Ethernet (FCoE) traffic, and production Ethernet traffic to the fabric interconnects.
- At the bottom of the chassis are slots used to house Cisco UCS X9416 X-Fabric Modules, which enables GPU connectivity to Cisco UCS X210c Compute Nodes.

- Six 2800-watt (W) Power Supply Units (PSUs) provide 54 Volts (V) of power to the chassis with N, N+1, and N+N redundancy. A higher voltage allows efficient power delivery with less copper wiring needed and reduced power loss.
- Efficient, 4x 100-mm, dual counter-rotating fans deliver industry-leading airflow and power efficiency. Optimized thermal algorithms enable different cooling modes to best support the network environment. Cooling is modular, so future enhancements can potentially handle open- or closed-loop liquid cooling to support even higher-power processors.

Cisco UCS X210c M7 Compute Node

The Cisco UCS X210 M7 Compute Node is high-performance, highly scalable, and designed for data centers and enterprise environments. Some of the key benefits of this server are:

- **Performance:** The Cisco UCS X210 M7 Compute Node is built to deliver exceptional performance. It features the latest Intel Xeon Scalable processors, providing high processing power for demanding workloads such as virtualization, database management, and analytics. The server's architecture is designed to optimize performance across a wide range of applications.
- **Scalability:** The Cisco UCS X210 M7 Compute Nodes offers excellent scalability options, allowing organizations to easily scale their computing resources as their needs grow. With support for up to eight CPUs and up to 112 DIMM slots per server, the server can accommodate large memory configurations and high core counts, enabling it to handle resource-intensive applications and virtualization environments.
- **Memory capacity:** The server supports a large memory footprint, making it suitable for memory-intensive workloads. It can accommodate a vast amount of DDR4 DIMMs, providing high memory capacity for applications that require significant data processing and analysis.
- **Enhanced virtualization capabilities:** The Cisco UCS X210 M7 Compute Node is designed to optimize virtualization performance. It includes features such as Intel Virtualization Technology (Intel VT-x) and Virtual Machine Device Queues (Intel VMDq), which improve virtual machine density and network performance in virtualized environments. These capabilities enable organizations to consolidate their workloads and achieve efficient resource utilization.
- **Simplified management:** The Cisco Unified Computing System™ (Cisco UCS) management software provides a unified and streamlined approach to server management. The Cisco UCS Manager software allows administrators to manage multiple servers from a single interface, simplifying operations and reducing management complexity. Additionally, the server integrates with Cisco's ecosystem of management tools, providing enhanced visibility, automation, and control.
- **High availability and reliability:** The Cisco UCS X210 M7 Compute Node is built with redundancy and fault tolerance in mind. It includes features such as hot-swappable components, redundant power supplies, and redundant fans, ensuring high availability and minimizing downtime. The server's architecture is designed to support mission-critical applications that require continuous operation.
- **Energy efficiency:** Cisco UCS servers are designed to be energy efficient. The Cisco UCS X210 M7 Compute Node incorporates power management features that optimize power usage and reduce energy consumption. This not only helps organizations reduce their carbon footprint but also lowers operating costs over time.

Figure 2.
Cisco UCS X-Series X210c M7 Compute Node

Cisco UCS Virtual Interface Cards (VICs)

Cisco UCS X210c M7 Compute Nodes support multiple Cisco UCS VIC cards. This design uses the Cisco UCS VIC 15000 adapter.

Cisco UCS X210c M7 Compute Nodes support Cisco UCS VIC 15231, 15420, and 15422. For this white paper, we have used Cisco UCS VIC 15231.

Cisco UCS VIC 15231

Cisco UCS VIC 15231 fits the mLOM slot in the Cisco UCS X210c Compute Node and enables up to 100 Gbps of unified fabric connectivity to each of the chassis IFMs for a total of 200 Gbps of connectivity per server. Cisco UCS VIC 15231 connectivity to the IFM and up to the fabric interconnects is delivered through 100Gbps. Cisco UCS VIC 15231 supports 512 virtual interfaces (both FCoE and Ethernet) along with the latest networking innovations such as NVMeoF over FC or TCP, VxLAN/NVGRE offload, and so forth.

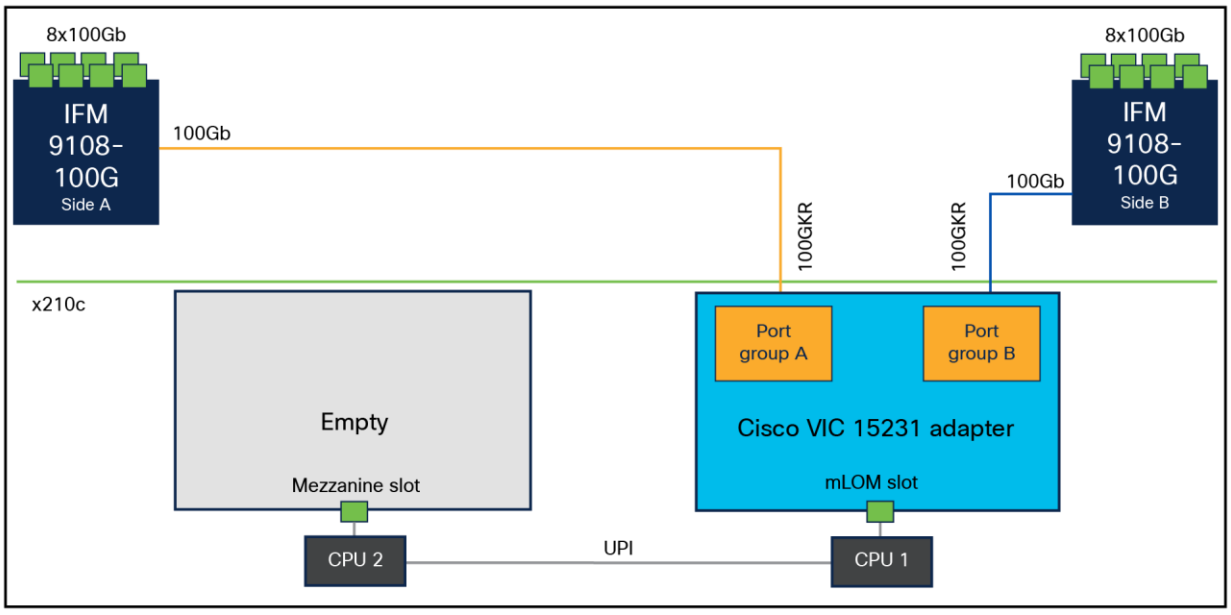


Figure 3.
Cisco UCS VIC 15231 in Cisco UCS X210c M7 Compute Node

Cisco UCS 6500 Series Fabric Interconnect

The Cisco UCS Fabric Interconnects (FIs) provide a single point of connectivity and management for the entire Cisco UCS system. Typically deployed as an active/active pair, the system's FIs integrate all components into a single, highly available management domain controlled by the Cisco UCS Manager or Cisco Intersight. Cisco UCS FIs provide a single unified fabric for the system, with low-latency, lossless, cut-through switching that supports LAN, SAN, and management traffic using a single set of cables.

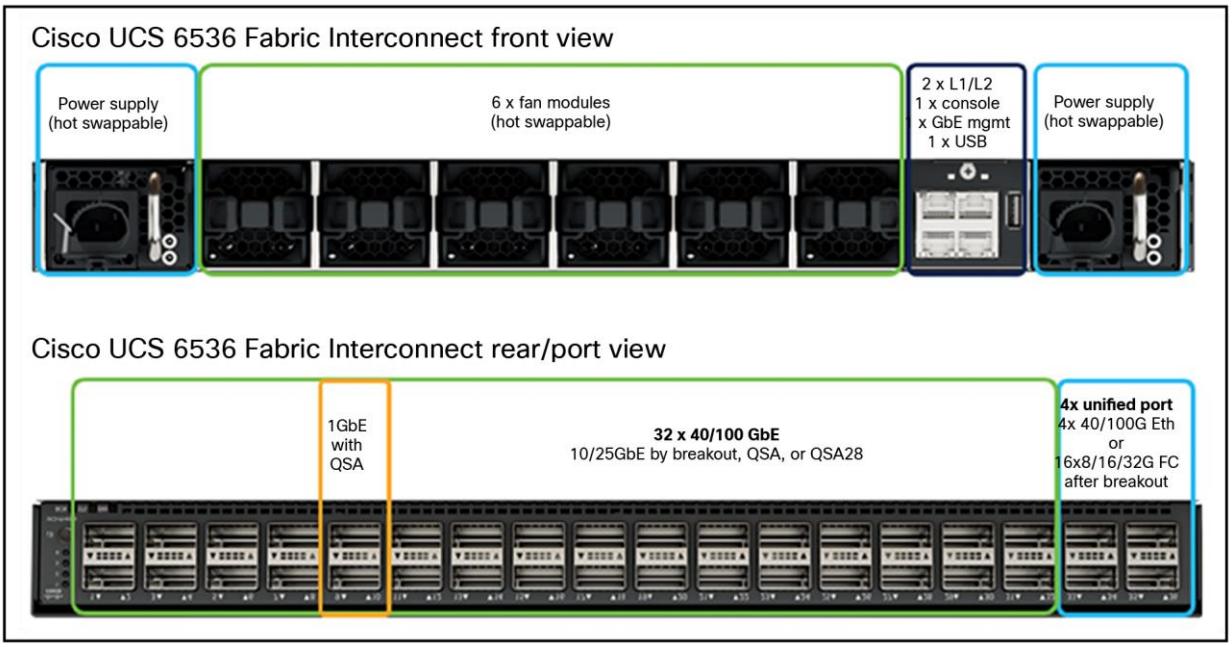


Figure 4.
Cisco UCS 6536 Fabric Interconnect – front and rear view

The Cisco UCS 6536 Fabric Interconnect utilized in the current design is a one-rack-unit (1RU) 1/10/25/40/100 Gigabit Ethernet, FCoE, and Fibre Channel switch offering up to 7.42 Tbps throughput and up to 36 ports. The switch has 32 40/100-Gbps Ethernet ports and 4 unified ports that can support 40/100-Gbps Ethernet ports or 16 Fiber Channel ports after breakout at 8/16/32-Gbps FC speeds. The 16 FC ports after breakout can operate as an FC uplink or FC storage port. The switch also supports two ports at 1-Gbps speed using QSA, and all 36 ports can breakout for 10- or 25-Gbps Ethernet connectivity. All Ethernet ports can support FCoE.

The Cisco UCS 6536 Fabric Interconnect (FI) is a core part of the Cisco Unified Computing System, providing both network connectivity and management capabilities for the system. The Cisco UCS 6536 Fabric Interconnect offers line-rate, low-latency, lossless 10/25/40/100 Gigabit Ethernet, Fibre Channel, NVMe over Fabric, and Fibre Channel over Ethernet (FCoE) functions.

The Cisco UCS 6536 Fabric Interconnect provides the communication backbone and management connectivity for the Cisco UCS X-Series compute nodes, Cisco UCS X9508 chassis, Cisco UCS B-Series blade servers, Cisco UCS 5108 B-Series server chassis, and Cisco UCS C-Series rack servers. All servers attached to a Cisco UCS 6536 Fabric Interconnect become part of a single, highly available management domain. In addition, by supporting a unified fabric, the Cisco UCS 6536 Fabric Interconnect provides both LAN and SAN connectivity for all servers within its domain.

From a networking perspective, the Cisco UCS 6536 FI uses a cut-through architecture, supporting deterministic, low-latency, line-rate 10/25/40/100 Gigabit Ethernet ports, a switching capacity of 7.42 Tbps per FI and 14.84 Tbps per unified fabric domain, independent of packet size and enabled services. It enables 1600Gbps bandwidth per X9508 chassis with X9108-IFM-100G in addition to enabling end-to-end 100G Ethernet and 200G aggregate bandwidth per X210c compute node. With the X9108-IFM-25G and the IOM 2408, it enables 400Gbps bandwidth per chassis per FI domain. The product family supports low-latency, lossless 10/25/40/100 Gigabit Ethernet unified network fabric capabilities, which increases the reliability, efficiency, and scalability of Ethernet networks. The 6536 fabric interconnect supports multiple traffic classes over a lossless Ethernet fabric from the server through the fabric interconnect. Significant TCO savings come from the unified-fabric-optimized server design in which Network Interface Cards (NICs), Host Bus Adapters (HBAs), cables, and switches can be consolidated.

Cisco Intersight

As applications and data become more distributed from core data center and edge locations to public clouds, a centralized management platform is essential. Achieving IT agility is difficult without a consolidated view of the infrastructure resources and centralized operations. Cisco Intersight provides a cloud-hosted management and analytics platform for all Cisco UCS and other supported third-party infrastructure across the globe. It provides an efficient way of deploying, managing, and upgrading infrastructure in the data center, ROBO, edge, and co-location environments.

Cisco Intersight API can help users to programmatically:

- Simplify the way they manage their infrastructure.
- Automate configurations and provision for their data center.
- Save long provisioning time.

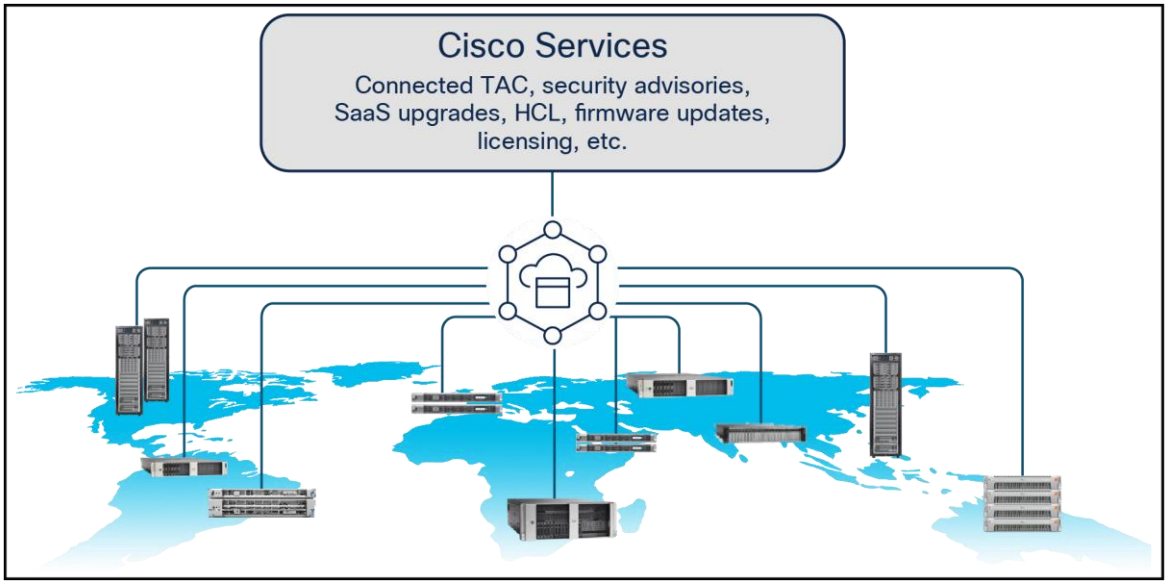


Figure 5.
Cisco Intersight overview

The main benefits of Cisco Intersight infrastructure services follow:

- Simplify daily operations by automating many daily manual tasks.
- Combine the convenience of a SaaS platform with the capability to connect from anywhere and manage infrastructure through a browser or mobile app.
- Stay ahead of problems and accelerate trouble resolution through advanced support capabilities.
- Gain global visibility of infrastructure health and status along with advanced management and support capabilities.
- Upgrade to add workload optimization when needed.

Cisco Nexus Switching Fabric

Cisco Nexus 9000 Series Switches offer both modular and fixed 1/10/25/40/100 Gigabit Ethernet switch configurations with scalability up to 60 Tbps of nonblocking performance with less than five-microsecond latency, wire speed VXLAN gateway, bridging, and routing support.

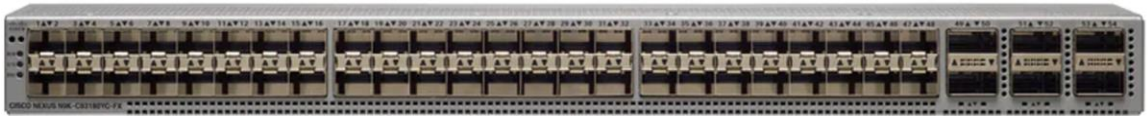


Figure 6.
Cisco Nexus 93180YC-FX3 NX-OS Mode Switch

The Cisco Nexus 9000 series switch featured in this design is the Cisco Nexus 93180YC-FX3 NX-OS Mode Switch configured in Cisco NX-OS standalone mode. NX-OS is a purpose-built data center operating system designed for performance, resiliency, scalability, manageability, and programmability at its foundation. It provides a robust and comprehensive feature set that meets the demanding requirements of virtualization and automation.

The Cisco Nexus 93180YC-FX3 switch is a 1RU switch that supports 3.6 Tbps of bandwidth and 1.2 bpps. The 48 downlink ports on the 93180YC-FX3 can support 1-, 10-, or 25-Gbps Ethernet, offering deployment flexibility and investment protection. The six uplink ports can be configured as 40- or 100-Gbps Ethernet, offering flexible migration options.

Intel Xeon Scalable Processor family

The latest generation of Intel Xeon Scalable processors comes with built-in accelerators and featured technologies that help optimize workload-specific performance, accelerate AI capabilities, reduce data center latency, reduce data bottlenecks, and balance resource consumption. Intel Accelerator Engines are purpose-built integrated accelerators on Intel Xeon Scalable processors that deliver performance and power efficiency advantages across today’s fastest-growing workloads.

Intel Xeon Scalable processors are designed to meet your organization's computing needs, whether for empowering solid foundations for AI innovation and HPC, supporting critical workloads at the edge, or building a secure cloud. They offer optimized performance, scale, and efficiency across a broad range of data center, edge, and workstation workloads.

5th Gen Intel Xeon Scalable processors

5th Gen Intel Xeon Scalable processors are designed to help boost performance, reduce costs, and improve power efficiency for today's demanding workloads, enabling you to achieve improved business outcomes.

These processors deliver impressive performance-per-watt gains across all workloads, with higher performance and lower Total Cost of Ownership (TCO) for AI, databases, networking, storage, and High-Performance Computing (HPC). They offer more compute, larger shared last-level cache, and faster memory at the same power envelope as the previous generation. They are also software- and platform compatible with the 4th Gen Intel Xeon processors, so you can minimize testing and validation when deploying new systems for AI and other workloads.

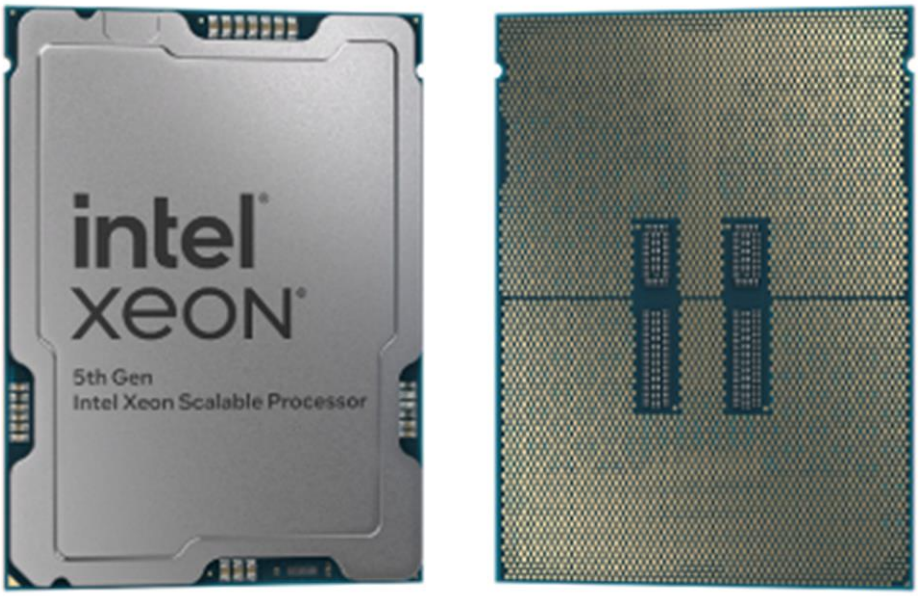


Figure 7.
5th Gen Intel Xeon Scalable Processor for AI

Some of the key features of 5th Gen Intel Xeon Scalable processors include:

- Built-in AI accelerators on every core; Intel Advanced Matrix Extensions (Intel AMX) for a big leap in deep-learning inference and training performance.
- Intel AI software suite of optimized open-source frameworks and tools.
- Out-of-the-box AI performance and E2E productivity with 300+ validated deep-learning models.
- Higher core count for faster compute processing with better scalability for training and inferencing parallel tasks.
- Boosted performance for memory-bound and latency-sensitive workloads with faster memory (DDR5 offers up to 1.7x bandwidth improvement over DDR4).
- Up to 320 MB last-level cache shared across all cores — an up-to 2.7x increase in last-level cache.

Intel Advanced Matrix Extensions (Intel AMX)

Intel Advanced Matrix Extensions (Intel AMX) enables Intel Xeon Scalable processors to boost the performance of deep-learning training and inferencing workloads by balancing inference, which is the most prominent use case for a CPU in AI applications, with more capabilities for training.

Many Intel customers are taking advantage of Intel AMX to enable better outcomes for their organizations. With 5th Gen Intel Xeon processors, customers can experience up to 14x better training and inference vs. 3rd Gen Intel Xeon processors.

Primary benefits of Intel AMX include:

- **Improved performance:**
  - CPU-based acceleration can improve power and resource utilization efficiencies, giving you better performance for the same price.
  - For example, 5th Gen Intel Xeon Platinum 8592+ with Intel AMX BF16 has shown up to 10.7x higher real-time speech Recognition Inference Performance (RNN-T) and 7.9x higher performance/watt vs. 3rd Gen Intel Xeon processors with FP32.4.
- **Reduced Total Cost of Ownership (TCO):**
  - Intel Xeon Scalable processors with Intel AMX enable a range of efficiency improvements that help with decreasing costs, lowering TCO, and advancing sustainability goals.
  - As an integrated accelerator on Intel Xeon Scalable processors that you may already own, Intel AMX enables you to maximize the investments you have already made and get more from your CPU, removing the cost and complexity typically associated with the addition of a discrete accelerator.
  - Intel Xeon Scalable processors with Intel AMX can also provide a more cost-efficient server architecture compared to other available options, delivering both power and emission reduction benefits.
- **Reduced development time:**
  - To simplify the process of developing deep-learning applications, Intel works closely with the open-source community, including the TensorFlow and PyTorch projects, to optimize frameworks for Intel hardware, upstreaming Intel's newest optimizations and features so they are immediately available to developers. This enables you to take advantage of the performance benefits of Intel AMX with the addition of a few lines of code, reducing overall development time.

For more information on Intel AMX, see: https://www.intel.com/content/www/us/en/products/docs/accelerator-engines/advanced-matrix-extensions/overview.html.

Intel Extension for PyTorch (IPEX)

Intel collaborated with the PyTorch open-source community to improve Deep-Learning (DL) training and inference performance. Intel Extension for PyTorch (IPEX) is an open-source extension that optimizes DL performance on Intel processors. Many of the optimizations will eventually be included in future PyTorch mainline releases, but the extension allows PyTorch users to get up-to-date features and optimizations more quickly. In addition to CPUs, Intel Extension for PyTorch will also include support for Intel GPUs in the near future.

Intel Extension for PyTorch provides additional performance on Intel processors. Intel would upstream most of the optimizations to the mainline PyTorch while continuously experimenting with new features and optimizations for the latest Intel hardware. Intel's open-source project is available in GitHub repository. Intel Extension for PyTorch provides a lot of specific optimizations for these LLMs. On the operator level, the extension provides a highly efficient GEMM kernel to speed up the linear layer, and customized operators to reduce the memory footprint.

Intel Extension for PyTorch (IPEX) is optimized for Intel processors. IPEX provides performance optimizations for CP- training with both float32 and bfloat16. Low precision datatype bfloat16 has been natively supported from 3rd Generation Xeon Scalable processors onward, with AVX512 instruction sets. This support is continued in newer generations of Intel Xeon Scalable processors with Intel AMX instruction sets. At the same time, the support of auto mixed precision with bfloat16 for CPU and bfloat16 optimization of operators has been massively enabled in Intel Extension for PyTorch, and partially upstreamed to the PyTorch master branch. Users can get better performance and user experience with IPEX auto mixed precision.

Optimizations for DeepSpeed

DeepSpeed is a deep-learning optimization software for scaling and speeding up deep-learning training and inference. DeepSpeed inference refers to the feature set in DeepSpeed that is implemented to speed up inference of transformer models. It initially supported only CUDA GPUs. We recently added support for CPUs, specifically 4th Gen Intel Xeon Scalable processors. Features currently implemented include automatic tensor parallelism (AutoTP), bfloat16 and int8 datatype support, and binding cores to rank.

DeepSpeed Accelerator Abstraction allows users to run large language models seamlessly on various deep-learning acceleration hardware with DeepSpeed. It offers a set of accelerator runtime and accelerator op builder interfaces that can be implemented for different hardware. This means users can write large language model code without hardware-specific code. With DeepSpeed Accelerator Abstraction, the same large language model can run on different hardware platforms without the need to rewrite the model code.

DeepSpeed supports using CPUs as accelerators. The DeepSpeed model using the DeepSpeed Accelerator Abstraction interface can run on CPUs without changing the model code. DeepSpeed checks if Intel Extension for PyTorch is installed in the environment. If this package is installed, DeepSpeed will use the CPU as an accelerator; otherwise, the CUDA device will be used as an accelerator.

DeepSpeed builds on top of PyTorch. It has been highly optimized for CPU inference and training. Intel Extension for PyTorch adds state-of-the-art optimizations for popular LLM architectures, including highly efficient matrix multiplication kernels to speed up linear layers, and customized operators to reduce the memory footprint. The runtime software components for DeepSpeed inference on CPU are shown in Figure 8. Intel oneAPI Deep Neural Network Library (oneDNN) uses Intel AVX-512 VNNI and Intel AMX optimizations. Intel oneAPI Collective Communications Library (oneCCL) is a library that implements the communication patterns in deep learning. Intel Neural Compressor was used to convert the LLMs from FP32 datatype to bfloat16 or int8 datatype.

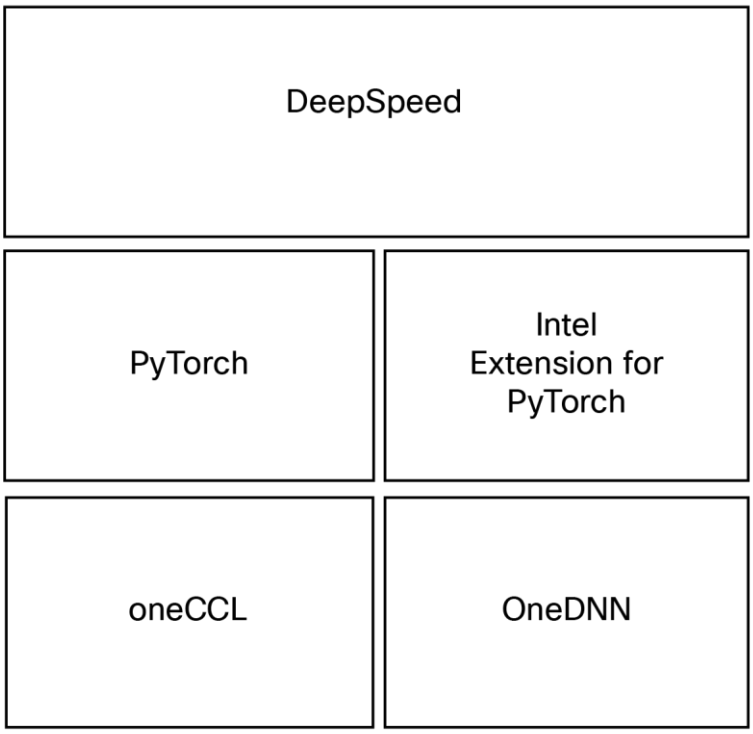


Figure 8.
Software components for DeepSpeed inference on CPU

For more information on Llama 2 inference on Intel Xeon Scalable processors with DeepSpeed, see: https://www.intel.com/content/www/us/en/developer/articles/technical/llama-2-on-xeon-scalable-processor-with-deepspeed.html.

Large Language Models (LLMs)

Generative AI (GenAI) workloads and models have gained widespread attention and popularity. LLMs have emerged as the dominant models driving these Generative AI applications. They are, essentially, complex algorithms trained on massive amounts of text data, allowing them to learn the patterns and nuances of language. This enables them to perform various tasks such as text generation, answering questions, language translation, writing different kinds of creative content, and more.

Inferencing refers to the process of using a trained LLM model to generate outputs based on new, unseen input data. This involves feeding the input data to the model and obtaining the model's predicted output, such as a text continuation, a translation, or an answer to a question.

Meta's Llama 2 models

Llama 2 is a family of transformer-based autoregressive causal language models. Autoregressive language models take a sequence of words as input and recursively predict—that is, output—the next word(s).

Llama 2 is a collection of pretrained and fine-tuned generative text models ranging in scale from 7 billion to 70 billion parameters. These fine-tuned LLMs, called Llama-2-Chat, are optimized for dialogue use cases. Llama-2-Chat models outperform open-source chat models on most benchmark tests.

Llama 2's training corpus includes a mix of data from publicly available sources. It constitutes more than two trillion tokens of training data.

During self-supervised pretraining, LLMs are provided the beginning of sample sentences drawn from a massive corpus of unlabeled data and tasked with predicting the next word. In training the model to minimize divergence between ground truth (the actual next word) and its own predictions, the model learns to replicate linguistic and logical patterns in the training data. Though there are no details on specific data sources, it is said that Meta's Llama 2 was trained with two trillion tokens — numerically represented words, word parts, phrases, and other semantic fragments that transformer-based neural networks use for language processing from publicly available sources.

These LLM models find usage in latency-sensitive applications such as chatbots, where benchmarking with hyperparameters for token latency is a critical measure of their successful deployments.

Setup details

Hardware and software components

The table below provides details on the hardware and software components used in this white paper.

Table 1. Details of the hardware components used in this white paper

Hardware components	Details	Versions	Quantity
Compute model	Cisco UCS X-Series blade chassis can host a combination of Cisco UCS X210c M7 Compute Nodes and a pool of IO resources that include GPU accelerators, disk storage, and non-volatile memory. The chassis has a Cisco UCS 9108 100G IFM providing 100G connectivity to the compute nodes on each side of the fabric.		1
Cisco UCS X210c M7 Compute node	Each node is equipped with 2x Intel 5th Gen Xeon Scalable 8568Y+ processors each with 48 cores running at 2.3GHz base frequency 350W. Each node has 16x 32G memory (total of 512GB) running at 5600 MT/s. Each compute node has one Cisco UCS VIC 15231 providing 100Gbps connectivity on each side of the fabric.	Firmware version-5.2(99.230063)	1
Cisco UCS 6536 Fabric Interconnect	Cisco UCS 6536 Fabric Interconnect provides both network connectivity and management capabilities for the system.	Bundle version - 4.3(2.230117) Cisco Nexus OS version - 9.3(5)I43(2a)	2
Cisco Nexus switch	Cisco Nexus 93360YC-FX2 Switch provides management and storage traffic.	Cisco Nexus OS version - 10.2(5)	2

Cisco UCS X-Series configuration - Cisco Intersight Managed Mode

Cisco Intersight Managed Mode standardizes policy and operation management for the Cisco UCS X-Series. The compute nodes in the Cisco UCS X-Series are configured using server profiles defined in Cisco Intersight. These server profiles derive all the server characteristics from various policies and templates. At a high level, configuring Cisco UCS servers using Intersight Managed Mode consists of the following steps:

1. Set up Cisco UCS Fabric Interconnect for Cisco Intersight Managed Mode

   Initial configuration wizard enables customers to choose whether to manage the fabric interconnect through Cisco UCS Manager or the Cisco Intersight platform. Customers can switch the management mode for the Fabric Interconnects (FIs) between Cisco Intersight and Cisco UCS Manager at any time; however, Cisco UCS FIs must be set up in Intersight Managed Mode (IMM) for configuring the Cisco UCS X-Series system.

2. Claim a Cisco UCS fabric interconnect in the Cisco Intersight platform

   After setting up the Cisco UCS 6536 Fabric Interconnects for Cisco Intersight Managed Mode, the FIs can be claimed for either a new or an existing Cisco Intersight account. When a Cisco UCS fabric interconnect is successfully added to Cisco Intersight, all future configuration steps are completed in the Cisco Intersight portal.

3. Configure a Cisco UCS domain profile

   A Cisco UCS domain profile configures a fabric-interconnect pair through reusable policies, allows configuration of the ports and port channels, and configures the VLANs and VSANs to be used in the network. It defines the characteristics of and configures the ports on the fabric interconnects. The Cisco UCS domain profile can be assigned to one fabric-interconnect domain.

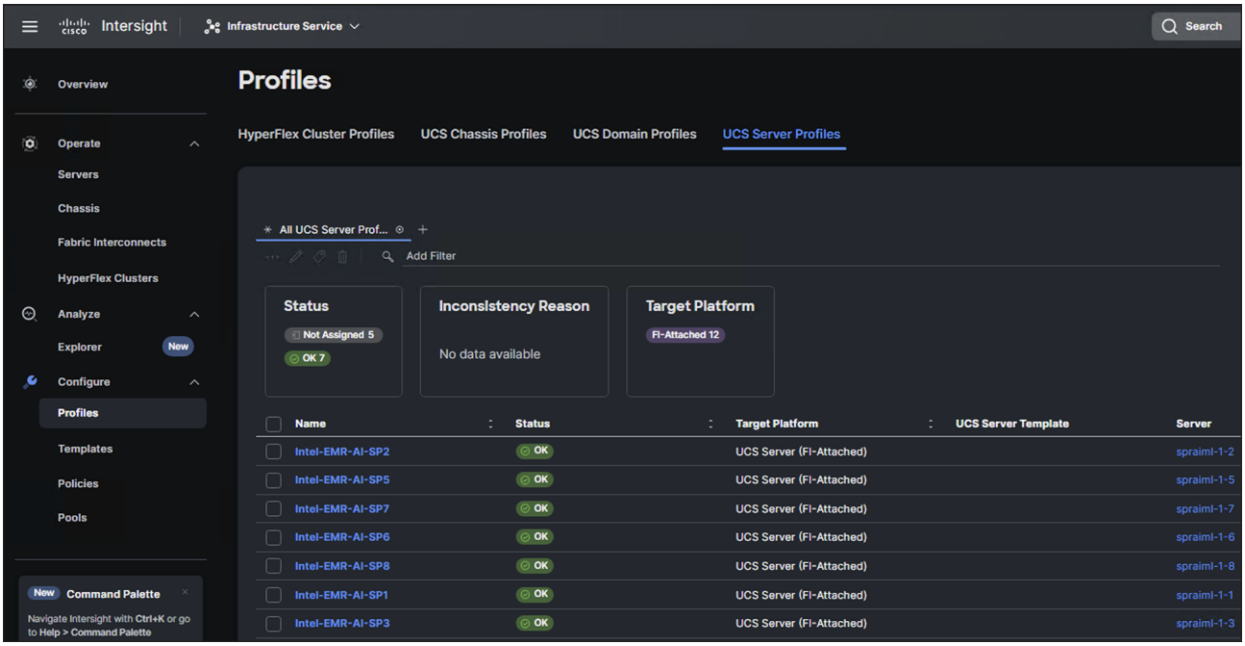
4. Configure a server profile template

   A server profile template enables resource management by simplifying policy alignment and server configuration. A server profile template is created using the server profile template wizard. The server profile template wizard groups the server policies into the following four categories: compute policies, network policies, storage policies, and management policies.

5. Deploy the server profiles

   Server profiles are derived from server-profile templates and deployed on bare-metal UCS servers that are claimed in Cisco Intersight.

When the server profiles are deployed successfully, you can see the profile status as shown in the screenshot below:



Once the server profile is deployed, an automated OS install can be triggered from Cisco Intersight.

Note: We have installed Red Hat Enterprise Linux 8.9 on the UCS X-Series node with 5th Gen Intel Xeon Platinum 8568Y+ processors for running IPEX with DeepSpeed benchmark scripts.

Intel IPEX with DeepSpeed tests

The Intel Extensions for PyTorch (IPEX) repository contains Dockerfile and benchmarking command lines to test both single-node single processing and scaling to multiprocessors using DeepSpeed. The benchmark was run on a single Cisco UCS X210c Compute Node. This compute node has two CPUs; the benchmark used the two sockets to establish communication between them to query the models and demonstrate Intel's DeepSpeed auto capability. DeepSpeed can be achieved with two sockets in a single node or with multiple nodes with two sockets in each node.

In this paper, we have limited our scope to testing DeepSpeed performance on a single node with two processors. Multinode multiprocessor performance is out of the scope of this paper.

Using a script that varies hyperparameters for LLMs, we tested bfloat16 and weight-only-quantized int8 precisions based on Meta's Llama 2 models:

- `meta-llama/Llama-2-7b-hf`
- `meta-llama/Llama-2-13b-hf`

We have captured the performance on 5th Gen Intel Xeon Platinum 8568Y+ processors by varying some of the parameters while running the benchmark scripts. When benchmarking using LLM-based models, it is important to understand the parameters associated with model training/inferencing performance. The following are some of the most important parameters (limited to those used in the DeepSpeed benchmark script):

- **Tokens:** In LLMs, tokens can be regarded as units of text that the models process and generate. They can represent an individual character, word, sub-word or even larger linguistic unit, depending on the specific tokenization approach used. Tokens act as a bridge between the raw text data and the numerical representations that LLMs can work with.
- **Tokenization:** Tokenization is the process of breaking a chunk of text into tokens. It involves segmenting the text into meaningful units to capture its semantic and syntactic structure. Various tokenization techniques are employed, such as word-level tokenization, subword-level tokenization, or character-level tokenization. Each technique has its advantages and tradeoffs, depending on the specific language and task requirements.
- **Input tokens:** For running the benchmark script, JSON file (prompts.json) with basic input prompts based on the varying input token sizes (input token size can be provided by the user) were used. The IPEX benchmark code reads the value that is provided and retrieves the correct prompt from the JSON file.
- **Output tokens:** This is the number of tokens it will end up sending to the screen in real time (the output token size can also be provided by the user).
- **Batch sizes:** Batching the dataset means dividing the datasets into smaller parts to be fed into the algorithm. The concept of batching is used in the scenario where multiple requests need to be processed and completed at once. Batch sizes can be loosely tied to the concept of number of concurrent user requests.
- **First (1st) token latency:** The latency of the 1st token signifies the time duration the LLM model takes to generate the initial token following receipt of a user prompt.
- **Second (2nd) token latency:** Once the first token is generated, the time taken to provide the 2nd token is termed the 2nd token latency. The evaluation of token latencies plays a crucial role in establishing a positive user experience, maintaining conversation flow, enabling interactivity, and enhancing the effectiveness and engagement of language models across a wide array of applications.

To run Intel IPEX with DeepSpeed benchmark script, we followed these steps:

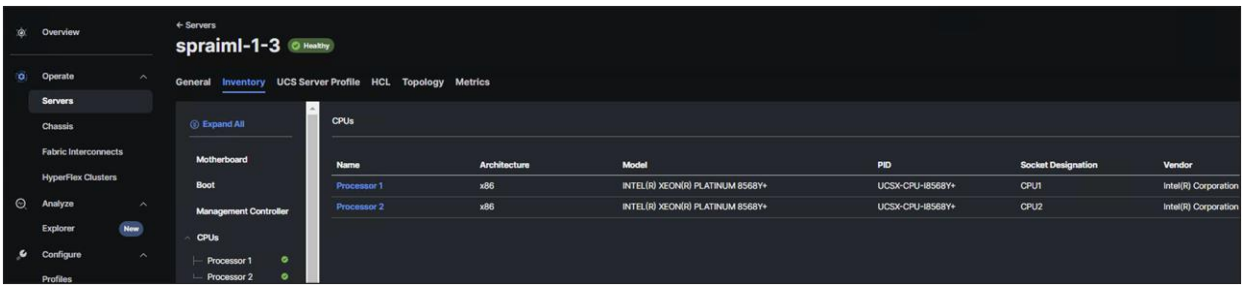
- Get access to Llama 2 models:

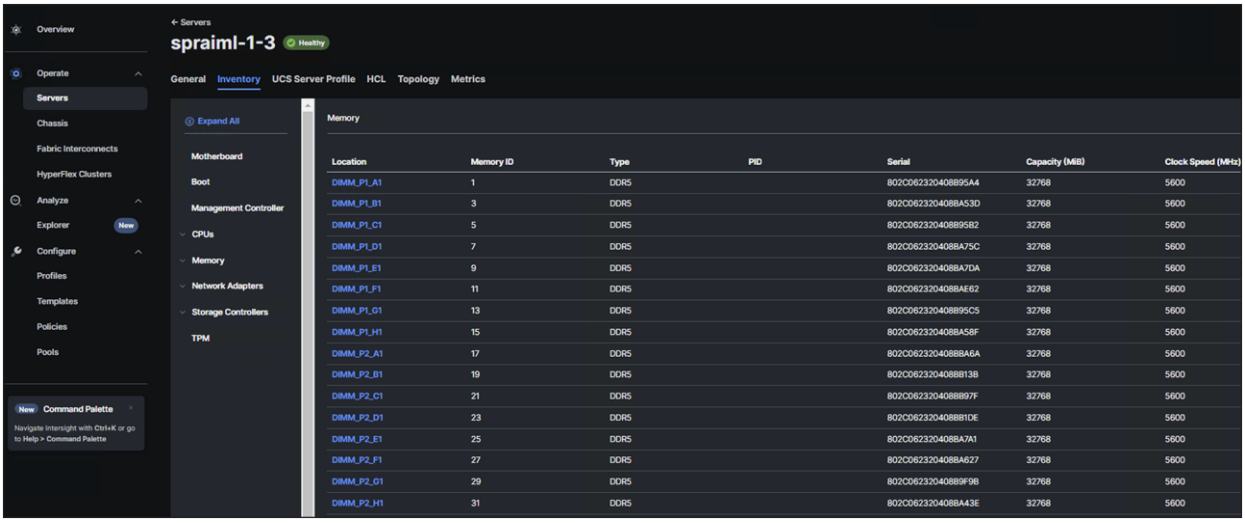
  Set up a Hugging Face account and accept the license from Meta website to get a Hugging Face token.

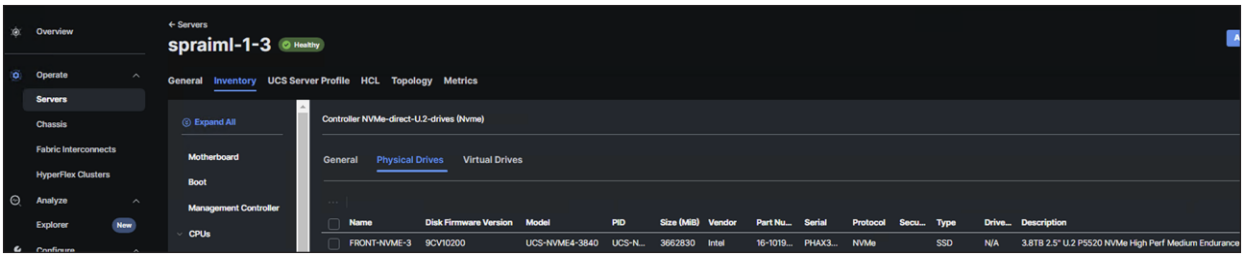
  Once the accounts are approved, you will find your token in settings in Hugging Face: https://huggingface.co/docs/hub/security-tokens.

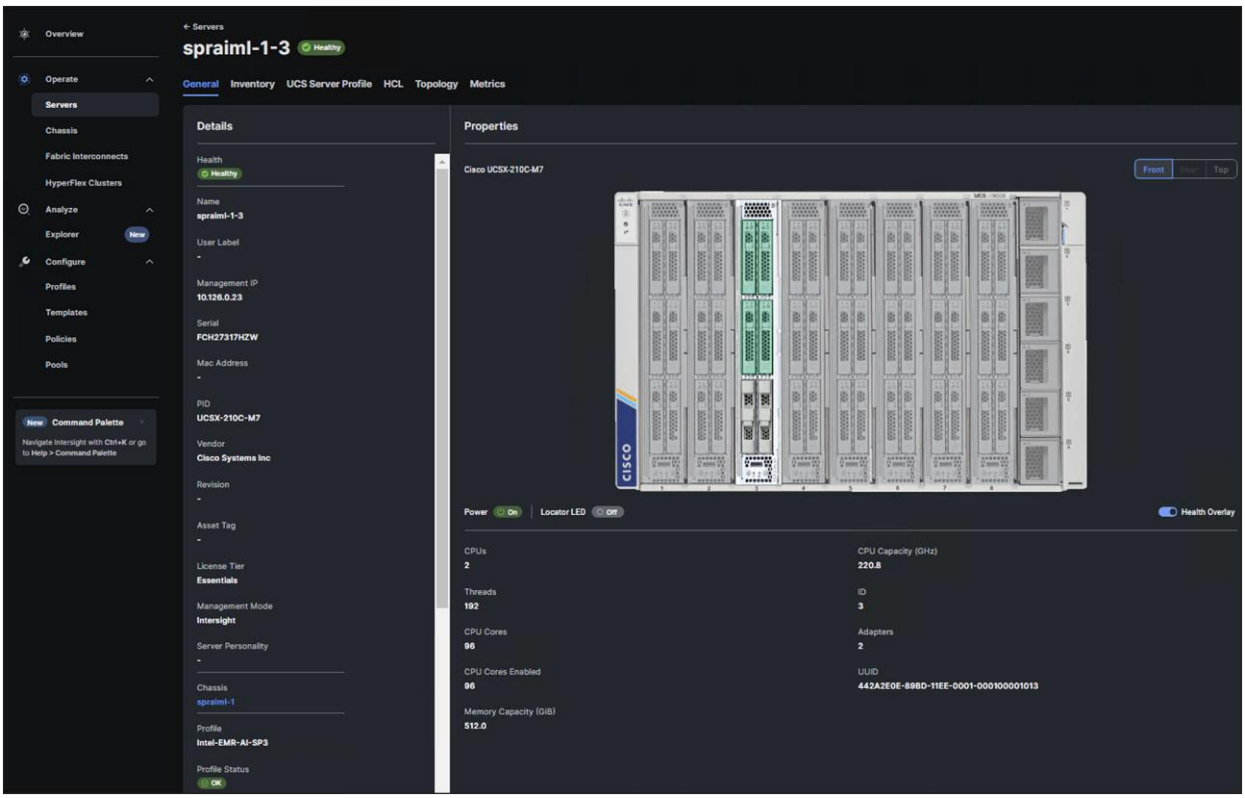
- Hardware required to run the tests: the Cisco UCS X210c Compute Nodes are equipped with the 5th Gen Intel Xeon Platinum processors and satisfy and meet the memory and NVMe drive requirements needed to run the benchmark. For more details on the hardware components used, refer to the “Setup details” section of this paper.

  Specific details regarding the Cisco UCS X210c Compute Node used to run the DeepSpeed benchmark script can be seen in Cisco Intersight. The following screen shots show details on the type of CPU, memory capacity and the NVMe drive used for running the benchmark tests.









Build a Dockerfile based container by following the GitHub link: https://github.com/intel/intel-extension-for-pytorch/tree/v2.1.0%2Bcpu/examples/cpu/inference/python/llm.

Note: The latest release of IPEX 2.2 comes with a Dockerfile; this can be used without you having to build a fresh Dockerfile: https://github.com/intel/intel-extension-for-pytorch/tree/v2.2.0%2Bcpu/examples/cpu/inference/python/llm.

- Create a run directory on the NVMe drive: /dslocaldisk/container.

```
[root@emr-gen-ai ~]# lsblk
NAME          MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda             8:0    0 445.2G  0 disk
nvme1n1       259:0    0   3.5T  0 disk /dslocaldisk
nvme2n1       259:1    0   3.5T  0 disk
nvme0n1       259:2    0 372.6G  0 disk
|-nvme0n1p1   259:3    0   600M  0 part /boot/efi
|-nvme0n1p2   259:4    0     1G  0 part /boot
`-nvme0n1p3   259:5    0   371G  0 part
  |-rhel-root 253:0    0    70G  0 lvm  /
  |-rhel-swap 253:1    0     4G  0 lvm  [SWAP]
  `-rhel-home 253:2    0   297G  0 lvm  /home
```

- Git clone the repository into this container folder:

  git clone --depth=1 -b v2.1.0+cpu https://github.com/intel/intel-extension-for-pytorch.git.

```
[root@emr-gen-ai container]# ls -ltr
drwxr-xr-x. 16 root root 4096 Dec 15 09:01 intel-extension-for-pytorch
-rw-r--r--.  1 root root  223 Dec 15 09:02 huggingface_login.py
-rw-r--r--.  1 root root 3260 Dec 23 02:01 Dockerfile
-rw-r--r--.  1 root root 9180 Feb 19 07:31 benchmark.sh
```

- Execute the benchmark script (composed of the varying hyperparameters and precisions) using the following command:
  - bash benchmark.`sh --num_runs <#> --model_name meta-llama/< Llama-2-7b-hf/ Llama-2-13b-hf> --precision <bfloat16/int8>`

This script is designed to sweep through batch sizes and input/output tokens, performing greedy search for varying precisions (bfloat16 and weight-only-quantization int8). Using the --token-latency parameters for IPEX, the first token, 2nd token, and total inference times can be varied for different configurations.

As for typical real-time chatbot use cases, the 2nd token latency is a critical metrics, which is discussed in the next section.

Performance results

Meta Llama 2 inference results on 5th Gen Intel Xeon Scalable processors

We at Cisco conducted performance characterization of `meta-llama/Llama-2-7b-hf and meta-llama/Llama-2-13b-hf`. The 5th Gen Intel Xeon Scalable Processor's higher memory capacity enables low-latency LLM execution with DeepSpeed on a single node, which is applicable for conversational AI and text summarization applications. This evaluation highlights the latency captured while executing the above-mentioned Llama 2 models on a single node with two sockets for bfloat16 and int8 precisions. Intel Extension for PyTorch has enabled support for SmoothQuant to secure good accuracy with int8 precision models.

Considering that LLM applications need to generate tokens fast enough to satisfy the reading speed of a fast reader, Intel chose token latency (time to generate each token) as the major performance metric to report, and, as a reference, the reading speed of a fast human reader, which translates to ~100ms per token. Hence ~100ms is treated as an industry-accepted latency, and various tests performed on Cisco UCS X210c Compute Nodes with 8568Y+ CPU adhere to <100ms latency for `meta-llama/Llama-2-7b-hf` and `meta-llama/Llama-2-13b-hf` on bfloat18 and weight-only-quantized int8 precisions. The scripts cover model generation inference with low-precision cases for different models with best performance and accuracy.

Note: Though the Intel IPEX with DeepSpeed benchmark script supports different models, we have chosen Meta Llama 2 models for this white paper, because Meta Llama 2 has emerged as an appealing substitute for the ChatGPT LLM, with optimized variations, heightened safety measures, and a complimentary licensing agreement.

While running the DeepSpeed benchmark script, we have kept the output tokens parameter constant at 256 though all the tests. We have captured the 2nd token latency on the models: "`meta-llama/Llama-2-7b-hf`" and "`meta-llama/Llama-2-13b-hf`" for bfloat16 and int8 precisions by varying batch sizes ranging from 1 to 16 (1, 4, 8, 16) and input tokens from 256 to 1024. The following graphs shows the latencies captured while performing greedy search for varying precisions and benchmark parameters using the DeepSpeed script. On Cisco UCS X210c Compute Nodes with 5th Gen Intel Xeon Scalable processors (Intel Xeon Platinum 8568Y+ processors).

We already know that quantization is a technique to reduce the computational and memory costs of running inference by representing the weights and activations with low-precision datatypes. Reducing the number of bits directly implies that the resulting model requires less memory storage and consumes less energy (in theory), and operations such as matrix multiplication can be performed much faster with integer arithmetic. As LLMs become more prevalent, the need for quantization methods is also crucial. To maintain accuracy while reducing computational costs, Weight-Only Quantization (WOQ) is a better tradeoff to balance performance and accuracy.

In Figure 9, we have compared the results obtained while running the DeepSpeed benchmark script on two of the Meta Llama 2 models with 7b and 13b parameters for bfloat16 and int8 precisions. In this comparison study, input tokens and output tokens were set to 256, and the 2nd token latency was captured by varying batch sizes: 1, 4, 8, and 16. Varying batch sizes can be loosely tied to the number of concurrent users' requests. From the graph, we see that the 2nd token latency increases linearly with batch size. This linear graph shows that the CPU can process the requests as expected, and all the requests, typically measured with 2nd token latency, are well under 100ms, which is considered an accepted latency industry-wide.

The comparison study showed that the weight-only-quantized int8 performed slightly better than with bfloat16, adhering to the expectations of a quantized model.

We have also run accuracy tests to verify the accuracy of the Meta Llama 2 models. For the models "`meta-llama/Llama-2-7b-hf`," the accuracy obtained was 73.59 percent and "`meta-llama/Llama-2-13b-hf`" the accuracy obtained was 76.69 percent, meeting typical expectations for accuracy.

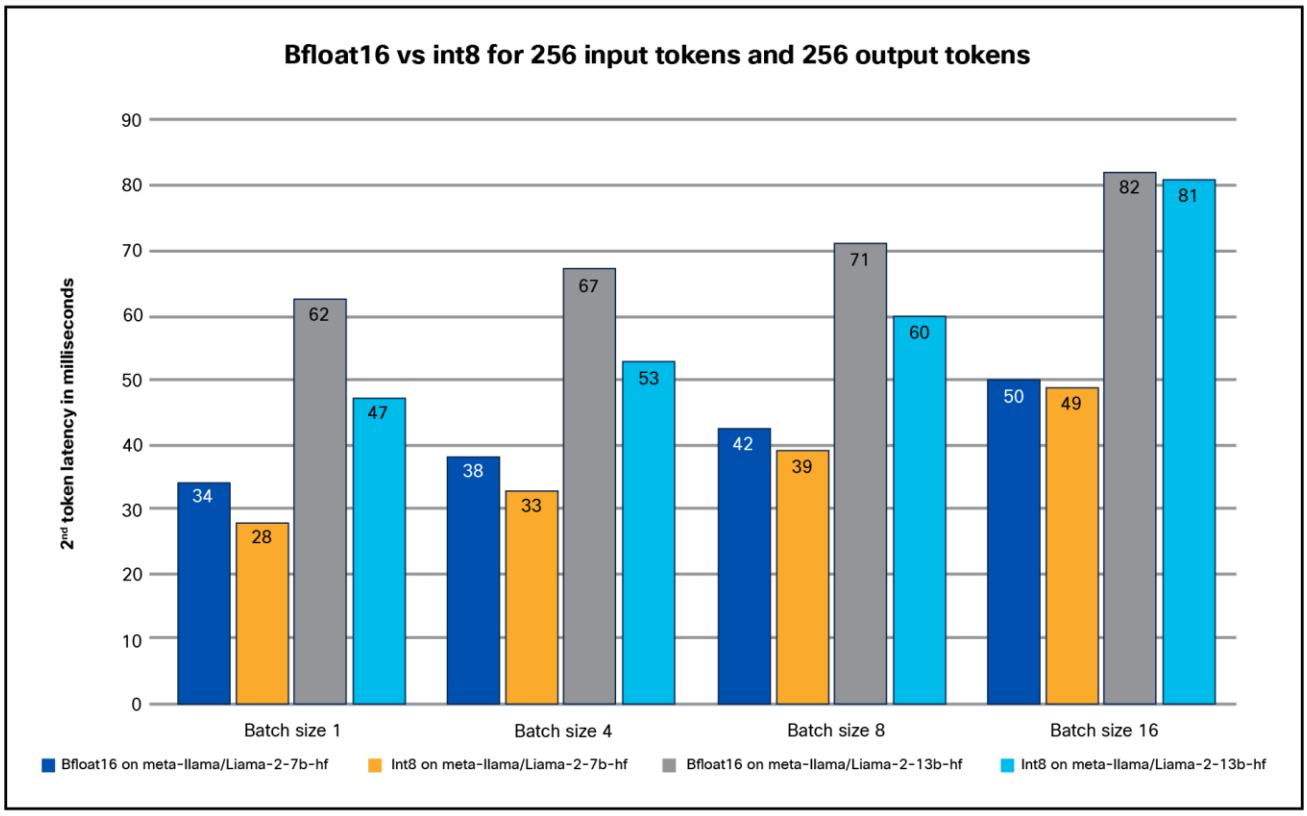


Figure 9.
Latency measurements for Llama 2 models for bfloat16 and int8 precisions – 256 input tokens

Figure 10 shows the comparison study for running the DeepSpeed benchmark script on two Meta Llama 2 models with 7b and 13b parameters for bfloat16 and int8 precisions. In this comparison study, the input tokens were increased from 256 to 1024, and the output tokens were set to 256; the 2nd token latency was captured by varying batch sizes: 1, 4, 8, and 16. The comparison study showed that the weight-only-quantized int8 performed slightly better than with bfloat16, adhering to the expectations of a quantized model. Results show that, with varying batch sizes and input tokens, the 2nd token latency remained less than 100ms.

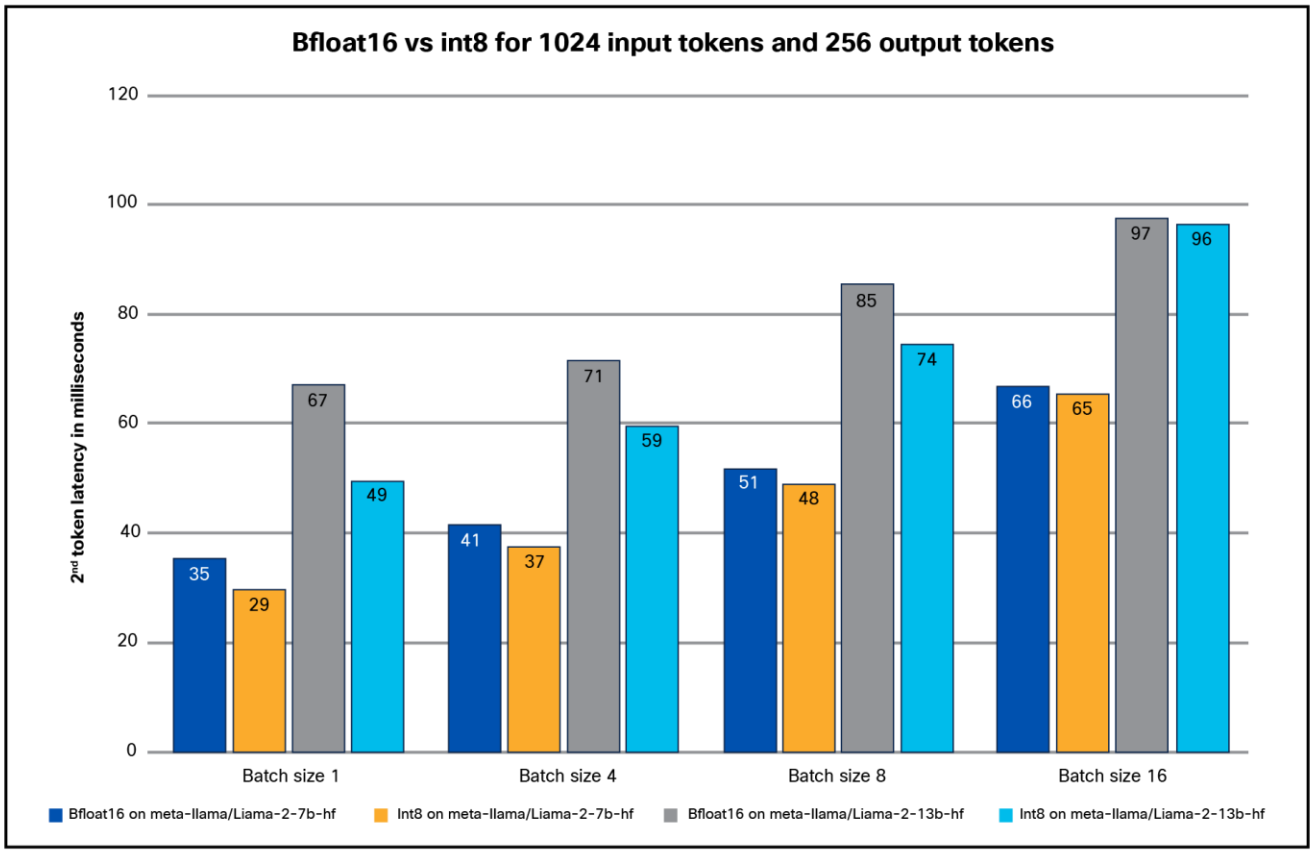


Figure 10.
Latency measurements for Llama 2 models for bfloat16 and int8 precisions - 1024 input tokens

Conclusion

Using IPEX tutorial (https://pytorch.org/tutorials/recipes/recipes/intel_extension_for_pytorch.html) for custom models and IPEX release (https://github.com/intel/intel-extension-for-pytorch/tree/v2.2.0%2Bcpu/examples/cpu/inference/python/llm), which has existing support for models with DeepSpeed auto tensor parallelism and ease-of-use APIs, helps customers to enable Intel optimization in a few steps. This document provides the necessary guidance for Generative AI inferencing use cases to run efficiently on a Cisco UCS platform.

Cisco experts have tested Meta Llama 2 models "`meta-llama/Llama-2-7b-hf`" and "`meta-llama/Llama-2-13b-hf`," an LLM with 7 billion and 13 billion parameters, with 5th Gen Intel Xeon Scalable processors using IPEX with DeepSpeed inference benchmarking methods, which test the text generation of LLMs. These computations were done using bfloat16, and int8 precisions.

All the benchmark tests performed at Cisco on Cisco UCS X-Series X210c Compute Nodes demonstrated seamless and linear scale with varying benchmark parameters. The measured 2nd token latency response was well within the recommended 100ms through all the tests.

Such low latency for a Generative AI workload on a Cisco UCS X-Series System means you can cost-effectively use Cisco UCS servers with 5th Gen Intel Xeon Scalable processors for demanding AI workloads without adding discrete accelerators to the servers.

References

For more information on Cisco UCS servers, Cisco Intersight, and Generative AI on 5th Gen Intel Xeon Scalable processors, refer to the following links:

- **Unleashing Creativity with Generative AI At-a-Glance:** https://www.cisco.com/c/en/us/products/collateral/servers-unified-computing/ucs-x-series-modular-system/unleashing-creativity-generative-ai-aag.html.
- **Mainstreaming Generative AI Inference Operations with 5th Gen Intel Xeon Scalable processors in Cisco UCS At-a-Glance:** https://www.cisco.com/c/en/us/products/collateral/servers-unified-computing/ucs-x-series-modular-system/mg-ai-i-ops-5thgen-intel-xeon-ucs-aag.html.
- **Cisco UCS X- Series Modular System:** https://www.cisco.com/c/en/us/products/collateral/servers-unified-computing/ucs-x-series-modular-system/solution-overview-c22-2432175.html?ccid=cc002456&oid=sowcsm025665.
- **Cisco Intersight Manage Mode Configuration Guide:** https://www.cisco.com/c/en/us/td/docs/unified_computing/Intersight/b_Intersight_Managed_Mode_Configuration_Guide.html.
- **Intel Extension for PyTorch (IPEX) for CPU Inferencing:** https://github.com/intel/intel-extension-for-pytorch/tree/v2.2.0%2Bcpu/examples/cpu/inference/python/llm.
- **Demystifying Tokens in LLMs: Understanding the Building Blocks of Large Language Models:** https://www.linkedin.com/pulse/demystifying-tokens-llms-understanding-building-blocks-lukas-selin/.
- **Intel Extension for PyTorch (IPEX):** https://pytorch.org/tutorials/recipes/recipes/intel_extension_for_pytorch.html.
- **Hugging Face Meta Llama 2 models:** https://huggingface.co/docs/transformers/en/model_doc/llama2.
- **Intel DeepSpeed:** https://www.intel.com/content/www/us/en/developer/articles/technical/llama-2-on-xeon-scalable-processor-with-deepspeed.html.
- **Intel AMX:** https://www.intel.com/content/www/us/en/products/docs/accelerator-engines/what-is-intel-amx.html.

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV Amsterdam,
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **https://www.cisco.com/go/offices**.

Printed in USA C11-4244801-00 03/24