Optimizing CPU-Based SLM Inference on Lenovo ThinkAgile HX V4 and FX V4 Servers with Nutanix

Planning / Implementation

Generative AI adoption is accelerating across enterprises and SMEs as organizations look for practical ways to improve productivity, automate knowledge workflows, summarize business content, and deploy AI assistants close to enterprise data. Small Language Models (SLMs) provide an opportunity to deliver these capabilities on modern CPU-based infrastructure, reducing dependency on specialized GPU platforms while maintaining a simpler and more cost-effective operational model.

This technical brief establishes proven architectural patterns and a deployment foundation for running Small Language Models (SLMs) on CPU infrastructure powered by Lenovo ThinkAgile HX V4 and FX V4 servers with Nutanix software and Intel Xeon 6 processors. The benchmark translates benchmark data into scalable deployment blueprints, providing data-backed sizing guidance for small and medium enterprises (SMEs) to balance latency and throughput for enterprise-wide use cases.

The benchmark architecture evaluated in this paper optimizes Small Language Model (SLM) deployment via quantized mixed-precision formats (INT4/INT8) executed locally within an isolated, NUMA-bound virtual machine environment. For enterprises evaluating distributed, multi-node scaling of uncompressed model weights utilizing native, up-streamed frameworks, see the companion Lenovo Press technical brief, Optimizing Distributed SLM Inference on Lenovo ThinkAgile VX V4 and FX V4 Servers with VMware Cloud Foundation 9.0.

Lenovo ThinkAgile HX V4 and FX V4 with Intel Xeon 6 Processors

The Lenovo ThinkAgile HX650 V4 and FX650 V4 are 2-socket, 2U systems and the Lenovo ThinkAgile HX630 V4 and FX630 V4 are 2-socket, 1U systems. These hyperconverged systems feature the Intel Xeon 6 two-socket processors (formerly code named "Granite Rapids"). ThinkAgile HX V4 and FX V4 hyperconverged systems are designed for deploying industry-leading hyperconvergence software from Nutanix on Lenovo enterprise platforms.

Figure 1. ThinkAgile HX650 V4 (top) and HX630 V4 (bottom) designed for Nutanix hyperconverged infrastructure

The Nutanix Cloud Platform (NCP) delivers cloud-scale advantages to enterprise applications and databases by combining ThinkAgile HX V4 and FX V4 and public cloud infrastructure. It provides robust enterprise storage, integrated data protection, built-in resilience, unified management, advanced analytics, and end-to-end security and consistent performance for mission-critical workloads.

ThinkAgile FX offers a unique, industry first flexibility for software-defined approach to hyperconvergence, leveraging the ability to move between hypervisors of your choice to deliver computing, storage and management in a tightly integrated software stack and future-proof your investment with seamless HCI software transitions.

Figure 2. ThinkAgile FX650 V4 (top) and FX630 V4 (bottom) designed for flexible hyperconverged infrastructure

Hardware configuration

The hardware configuration represents a typical SME deployment model where generative AI inference can be deployed on Lenovo ThinkAgile HX/FX V4 systems with Nutanix and Intel Xeon 6 Processors.

Table 1. Lenovo ThinkAgile HX650 V4 Cluster Configuration

Item	Description
Server platform	4x Lenovo ThinkAgile HX650 V4
Processor per host	2xIntel Xeon 6767P 64C 350W 2.4GHz Processor
Memory	1 TB (16x ThinkSystem 64GB TruDDR5 6400MHz (2Rx4) RDIMM)
Storage	2x ThinkSystem M.2 7450 PRO 480GB Read Intensive NVMe PCIe 4.0 x4 NHS SSD 8x ThinkAgile HX 2.5" U.2 VA 1.92TB Read Intensive NVMe PCIe 4.0 x4 HS SSD
Network	2x Mellanox ConnectX-6 Lx 10/25GbE SFP28 2-port PCIe Ethernet Adapter
Nutanix Software	AHV 10.3.0.1, AOS 7.3.0.5
CVM configuration	16 vCPUs, 64 GB RAM

The following table lists the virtual machine configuration.

Table 2. Virtual Machine Configuration

Item	Description
Guest OS	Ubuntu 22.04.5 LTS
VM Config	96 vCPUs, 384 GB memory and 600GB disk
NUMA topology	4 NUMA nodes, 24 vCPUs per NUMA node
Inference execution model	4 OpenVINO workers, 1 worker pinned to each NUMA node
NUMA affinity	CPU and memory pinned locally per worker
Threads per worker	24
Streams per worker	1

Software configuration

The inference stack was based on OpenVINO GenAI, which provides optimized generative AI pipelines for Intel hardware for CPU based inference.

Table 3. Software Configuration

Category	Component	Specification / Version
Virtualization platform	Nutanix AHV	10.3.0.1
HCI software	Nutanix AOS	7.3.0.5
Guest operating system	Ubuntu	Ubuntu 22.04.5 LTS
Python environment	Conda environment	phi3-intel
Inference framework	OpenVINO GenAI	OpenVINO GenAI Python API
Runtime device	OpenVINO CPU device	CPU
Model execution API	OpenVINO GenAI	openvino_genai.LLMPipeline
Performance metrics API	OpenVINO GenAI	PerfMetrics
CPU and memory affinity	Linux numactl	CPU node binding and memory node binding
Benchmark orchestration	Custom Python and Bash scripts	Warm-up, measured runs, NUMA-local execution
Output metrics	TTFT, TPOT, throughput	Reported from OpenVINO GenAI metrics

The following table lists the model and runtime configuration.

Table 4. Model and Runtime Configuration

Item	Configuration
Model under test	Microsoft Phi-3 Mini 128K Instruct
Precision formats tested	INT4, INT8, FP16
Input token sizes	128, 256, 512, 1024
Output token size	256 generated tokens
Batch sizes per NUMA worker	1, 4, 8, 16, 32
OpenVINO device	CPU
OpenVINO performance hint	Latency
CPU pinning	Enabled
NUMA memory binding	Enabled
Warm-up runs	1
Measured runs	5

Testing methodology

The inference framework used for this validation was OpenVINO GenAI, which provides optimized generative AI pipelines on top of the OpenVINO Runtime. OpenVINO GenAI exposes the LLMPipeline API for running language models on devices such as CPU and provides native performance metrics through PerfMetrics (openvinotoolkit.github.io). The model selected for testing was Microsoft Phi-3 Mini 128K Instruct, a small language model from the Microsoft Phi-3 family (huggingface.co) suitable for use cases such as knowledge assistance, summarization, document interaction, and workflow automation.

In the results, total throughput is summed across the four workers because each worker generates tokens independently. TTFT and TPOT are not summed up; they are interpreted as per-worker responsiveness metrics under concurrent execution. This distinction allows the analysis to separate full-system capacity from per-request latency behavior.

Benchmark Configuration

Each test point was executed by launching four independent OpenVINO GenAI workers, with one worker pinned to each NUMA node and each worker used 24 inference threads, one stream, one warm-up run, and five measured runs. The warm-up run was used to reduce cold-start effects before collecting measured results. Values reported represent the mean across 5 measured execution runs following 1 warmup run.

The benchmark collected the following OpenVINO GenAI PerfMetrics.

- **TTFT (Time to First Token)** - Measures how quickly the first token is produced; key for interactive responsiveness (response-start latency).

  The metric is extracted via OpenVINO GenAI's PerfMetrics API for a single batched generate() call. Empirical verification confirmed prefill executes as a parallel batched operation: wall-clock completion of first-token generation for all sequences matches the reported mean TTFT (within 1%) at Batch 32, the largest tested configuration, confirming reported TTFT represents true per-request first-token latency.

- **TPOT (Time per Output Token)** - Measures sustained generation speed after the first token (streaming generation speed).

  Represents aggregate worker throughput efficiency across the concurrent processing pipeline. Per-user streaming latency for an individual request inside a batch is calculated as Batch Size × System TPOT.

- **Throughput** - Measures generated tokens per second
- **Generate duration** - Shows model generation time for the fixed 256-token response

The goal was to measure peak throughput and understand how model precision, batch size, input token length, and NUMA-local execution affect responsiveness and system capacity.

The results are interpreted relevant to architecture patterns for deploying AI models:

- **Interactive (Realtime two-way user conversation)** - Ultra low TTFT and predictable benchmark elapsed time.
- **Streaming (Continuous one-way ingestion or output) -** low TPOT and stable generation behavior.
- **Shared services (Maximize infrastructure usage and reduce costs)** - balance between latency and aggregate throughput.
- **Throughput-oriented workloads** – Maximum processing volume and system throughput and batch efficiency.

Test results and analysis

INT4 delivered the strongest overall CPU inference performance for Phi-3, leading in Time to First Token (TTFT), Time per Output Token (TPOT), and total system throughput across the full workload matrix. The tradeoff rule is that larger batch sizes increase aggregate throughput and improve TPOT but also increase TTFT and total response completion time.

- Throughput
- Batch size
- Input token length

Throughput

The following table shows INT4 provides the fastest response start, sustained token generation, and highest total throughput for the most latency-sensitive test case. INT4 provided the best balance of responsiveness, sustained generation speed, and total system throughput. INT8 remained a viable compressed alternative, but it did not outperform INT4 in the benchmark. FP16 provided the lowest throughput and slowest sustained generation in this workload, making it more appropriate as a higher-precision reference configuration than as the default performance choice.

The table shows mathematical averages across all tested batch sizes and token lengths to illustrate macro-level throughput scaling across FP16, INT8, and INT4 precisions. They do not represent a single operational batch profile.

Table 5. Aggregate Precision Summary Across Full Execution Matrix (Relative Format Comparison)

Precision	Avg. TTFT (ms)	Avg. TPOT (ms/token)	Avg. total throughput (tokens/sec)	Avg. benchmark elapsed time (sec)
INT4	56.11	9.42	752.52	13.52
INT8	68.39	15.79	658.87	16.76
FP16	76.81	18.57	573.54	19.66

The following two figures show that INT4 achieved the lowest average TTFT and highest throughput across the workload matrix, indicating the fastest response-start behavior.

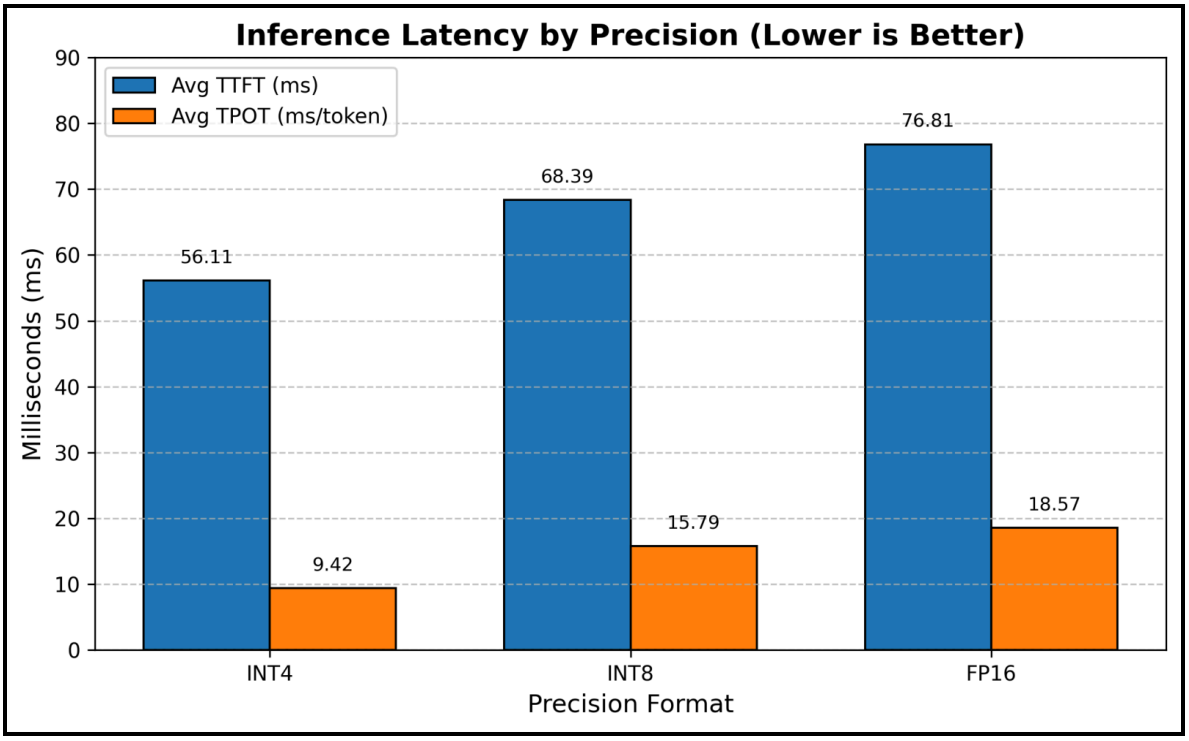

Figure 3. Inference Latency by Precision (Lower is Better)

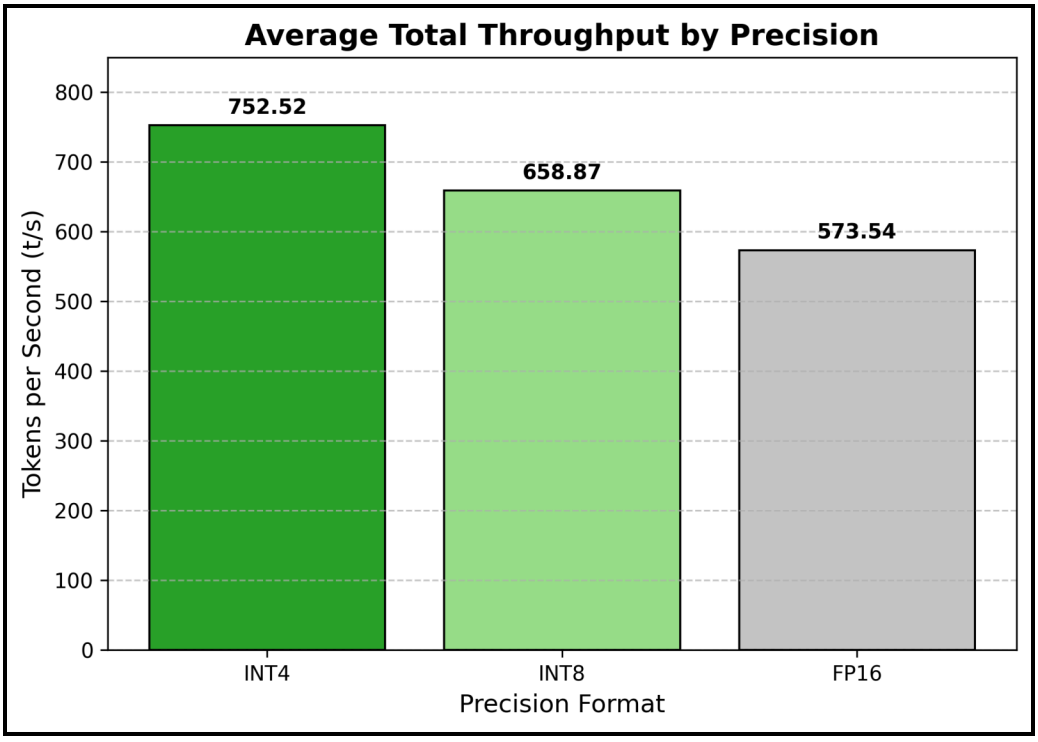

Figure 4. Average Total Throughput by Precision

Batch size

Larger batch size impacts total system throughput and latency across all tested precision formats. As batch size increased, total throughput increased sharply and TPOT decreased, indicating better sustained decoding efficiency. At the same time, TTFT increased, which means larger batches improved capacity but reduced response-start responsiveness.

The following table and figure show TTFT increases as batch size grows, indicating that higher-throughput configurations introduce additional response-start latency. INT4 maintains the lowest TTFT across the tested batch-size range, while INT8 and FP16 show higher response-start latency. TPOT decreases as batch size increases, indicating improved sustained token generation efficiency at higher concurrency levels. INT4 maintains the lowest TPOT across the tested batch-size range, followed by INT8 and FP16.

Table 6. Performance with different batch sizes at 128 input tokens

Precision	Batch per NUMA worker	Effective system batch	TTFT (ms)	TPOT (ms/token)	Total throughput (tokens/sec)
INT4	1	4	24.01	23.44	170.78
INT4	4	16	35.32	8.54	468.58
INT4	8	32	48.88	5.85	684.95
INT4	16	64	61.65	3.59	1,115.22
INT4	32	128	80.54	2.27	1,764.35
INT8	1	4	47.69	47.87	83.56
INT8	4	16	57.38	14.37	278.31
INT8	8	32	55.84	6.65	602.59
INT8	16	64	66.14	3.90	1,026.23
INT8	32	128	85.53	2.43	1,648.26
FP16	1	4	58.73	58.66	68.89
FP16	4	16	59.02	14.70	274.97
FP16	8	32	63.44	7.93	509.53
FP16	16	64	65.06	4.21	959.94
FP16	32	128	86.68	2.71	1,499.07

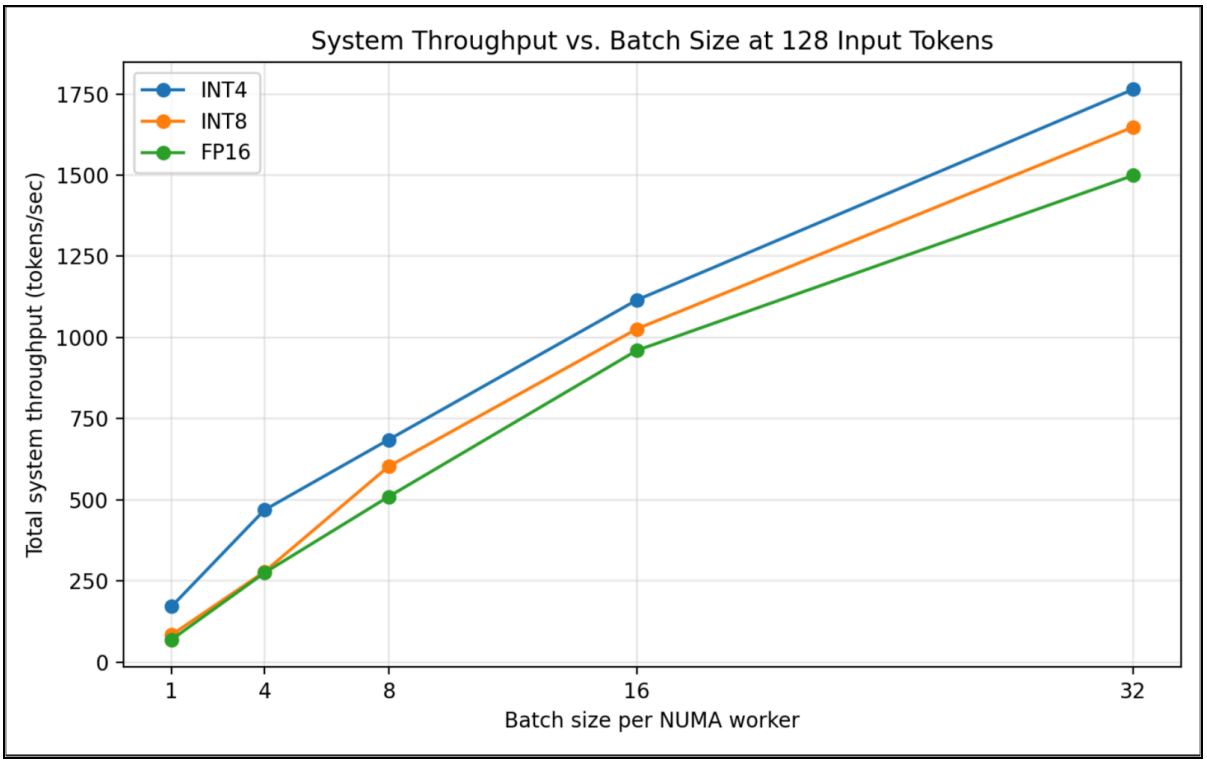

Figure 5. Total system throughput vs. batch size at 128 input tokens

The key interpretation is that batch size should be selected based on deployment intent:

- Batch 1-4 per NUMA worker is best suited to interactive or latency-sensitive use cases.
- Batch 8-16 per NUMA worker provides a balanced operating range for shared SME services.
- Batch 16-32 per NUMA worker is best suited to throughput-oriented or queued workloads.

Larger batches improve system throughput and token generation efficiency, but they also increase response-start latency.

Input token length

Longer prompts increased the amount of work required during prompt processing, which increased TTFT and benchmark elapsed time while reducing total system throughput. This effect was most visible at higher batch sizes, where more prompt tokens were processed concurrently across the four NUMA-local workers.

The following table summarizes the effect of input token length at batch 16 per NUMA worker. This batch size was selected because it represents a balanced shared-service operating point: high enough to show system capacity behavior, but not as throughput-oriented as batch 32.

Table 7. Performance with different input token length at batch 16 per NUMA worker

Precision	Input tokens	Effective system batch	TTFT (ms)	TPOT (ms/token)	Total throughput (tokens/sec)	Benchmark elapsed time (sec)
INT4	128	64	61.65	3.59	1,115.22	14.70
INT4	256	64	64.01	3.77	1,063.16	15.42
INT4	512	64	69.68	4.11	972.54	16.86
INT4	1024	64	84.08	4.90	817.16	20.06
INT8	128	64	66.14	3.90	1,026.23	15.98
INT8	256	64	68.50	4.06	985.92	16.63
INT8	512	64	75.06	4.45	899.15	18.23
INT8	1024	64	87.03	5.21	767.87	21.35
FP16	128	64	65.06	4.21	959.94	17.25
FP16	256	64	71.55	4.59	877.12	18.81
FP16	512	64	83.26	5.22	770.10	21.38
FP16	1024	64	110.66	6.92	579.06	28.35

The following figure shows total system throughput as input token length increases from 128 to 1024 tokens.

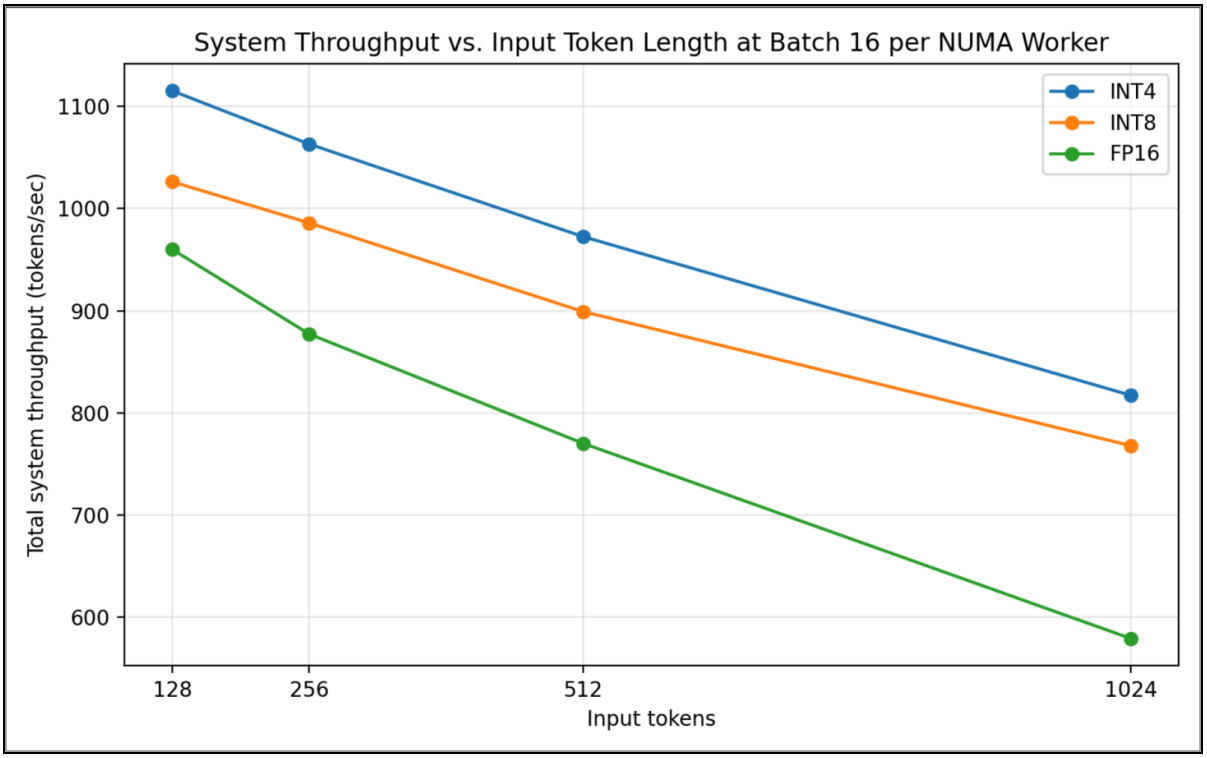

Figure 6. System throughput vs. input token length at batch 16 per NUMA worker

The total system throughput decreases as input token length increases. INT4 maintains the highest throughput across all input lengths, followed by INT8 and FP16. The decline is expected because longer prompts require more prompt-processing work before and during generations.

Key Findings

The benchmark supports the following conclusions:

- While INT4 quantization yields up to ~2.5x higher throughput than FP16 at low batch sizes (and ~1.3x on average across the workload matrix), enterprise deployments should conduct task-specific accuracy evaluations prior to production rollout. Public evaluations indicate that modern 4-bit weight-only quantization typically preserves most benchmark accuracy, though degradation is method- and task-dependent.
- INT8 and FP16 also delivered acceptable response-start latency for the tested workloads. Across the full-system benchmark dataset, average TTFT remained below 200 ms for all three precision formats, which supports a responsive user experience for many interactive and shared-service SLM use cases.
- Larger batch sizes improved throughput and reduced TPOT, but increased TTFT. This confirms the expected trade-off between system capacity and response-start latency. For most deployments, batch sizes between 1 and 16 per NUMA worker provide the most practical balance between responsiveness and capacity.
- Longer input prompts reduced throughput and increased benchmark elapsed time, especially at higher batch sizes, confirming that prompt length should be included in sizing assumptions.

Conclusion

The validation of Small Language Model (SLM) inference on Lenovo ThinkAgile HX V4 and FX V4 systems demonstrates that generative AI workloads can be effectively delivered on CPU-based hyperconverged infrastructure without requiring dedicated GPU platforms for every deployment scenario. Lenovo ThinkAgile HX V4 and FX V4 provide an enterprise-ready foundation for deploying SLM-based AI inference alongside business data, applications, and operational workflows. This enables broader adoption of generative AI across diverse enterprise use cases, while maintaining simplicity, scalability, and cost efficiency.

Bill of Materials: ThinkAgile HX650 V4

The following table lists the Bill of Materials for a single ThinkAgile HX650 V4 node in the configuration validated in this paper. Four identically configured nodes formed the Nutanix cluster described in Table 1.

Table 8. Bill of Materials: ThinkAgile HX650 V4

Part number	Product Description	Quantity
7DG4CTO1WW	Server: Lenovo ThinkAgile HX650 V4 Hyperconverged System	1
C6TQ	ThinkAgile HX650 V4 Base	1
B15S	Nutanix Software Stack on Nutanix AHV	1
BVKV	Nutanix Cloud Platform (NCP) Pro Software License with Mission Critical Support	1
C5QY	Intel Xeon 6767P 64C 350W 2.4GHz Processor	2
C0TQ	ThinkSystem 64GB TruDDR5 6400MHz (2Rx4) RDIMM	16
C26V	ThinkSystem M.2 RAID B545i-2i SATA/NVMe Adapter	1
C46P	ThinkSystem 2U V4 8x2.5" NVMe Backplane	2
B0SW	Nutanix Flash Node Config	1
C1AC	ThinkAgile HX 2.5" U.2 VA 1.92TB Read Intensive NVMe PCIe 4.0 x4 HS SSD	8
C286	ThinkSystem M.2 VA 480GB Read Intensive NVMe NHS SSD	2
BE4T	ThinkSystem Mellanox ConnectX-6 Lx 10/25GbE SFP28 2-port OCP Ethernet Adapter	1
C0UD	ThinkSystem 3200W 230V Titanium CRPS Premium Hot-Swap Power Supply	2
6252	2.5m, 16A/100-250V, C19 to C20 Jumper Cord	2
C2DJ	ThinkSystem Advanced Toolless Slide Rail Kit V4	1
C3RG	ThinkSystem SR650 V4 Left Rack Latch with USB/MiniDP	1
C3RD	ThinkSystem 2U 6056 20K Performance Fan Module	6

Resources

For more information, see these resources:

- Lenovo ThinkAgile HX650 V4 Hyperconverged System
  https://lenovopress.lenovo.com/lp2133-lenovo-thinkagile-hx650-v4-hyperconverged-system
- Lenovo ThinkAgile HX630 V4 Hyperconverged System
  https://lenovopress.lenovo.com/lp2132-lenovo-thinkagile-hx630-v4-hyperconverged-system
- Lenovo ThinkAgile FX650 V4 Hyperconverged System
  https://lenovopress.lenovo.com/lp2338-lenovo-thinkagile-fx650-v4-hyperconverged-system
- Lenovo ThinkAgile FX630 V4 Hyperconverged System
  https://lenovopress.lenovo.com/lp2337-lenovo-thinkagile-fx630-v4-hyperconverged-system
- OpenVINO GenAI Documentation — Performance Metrics
  https://openvinotoolkit.github.io/openvino.genai/docs/guides/performance-metrics/
- OpenVINO API Documentation — openvino_genai.PerfMetrics
  https://docs.openvino.ai/2025/api/genai_api/_autosummary/openvino_genai.PerfMetrics.html
- OpenVINO API Documentation — openvino_genai.LLMPipeline
  https://docs.openvino.ai/2025/api/genai_api/_autosummary/openvino_genai.LLMPipeline.html
- OpenVINO GenAI Documentation
  https://openvinotoolkit.github.io/openvino.genai/
- Microsoft Phi-3 Model Collection — Hugging Face
  https://huggingface.co/collections/microsoft/phi-3
- Microsoft Phi-3 Mini 128K Instruct — Hugging Face Model Card
  https://huggingface.co/microsoft/Phi-3-mini-128k-instruct
- OpenVINO Phi-3 Mini 128K Instruct INT4 Model — Hugging Face
  https://huggingface.co/OpenVINO/Phi-3-mini-128k-instruct-int4-ov
- Intel AI Development Software
  https://www.intel.com/content/www/us/en/developer/topic-technology/artificial-intelligence/development-software.html

Author

Cristian Ghetau is an Advisory Engineer for Lenovo in Romania and has experience in Cloud Infrastructure technologies. He has had more than 13 years of experience working with virtual environments from VMware, Microsoft, Oracle, and Linux.

Related product families

Product families related to this document are the following:

- Artificial Intelligence
- ThinkAgile FX Series
- ThinkAgile HX Series for Nutanix

Notices

Lenovo may not offer the products, services, or features discussed in this document in all countries. Consult your local Lenovo representative for information on the products and services currently available in your area. Any reference to a Lenovo product, program, or service is not intended to state or imply that only that Lenovo product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any Lenovo intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any other product, program, or service. Lenovo may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

Lenovo (United States), Inc.
8001 Development Drive
Morrisville, NC 27560
U.S.A.
Attention: Lenovo Director of Licensing

LENOVO PROVIDES THIS PUBLICATION ”AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. Lenovo may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

The products described in this document are not intended for use in implantation or other life support applications where malfunction may result in injury or death to persons. The information contained in this document does not affect or change Lenovo product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of Lenovo or third parties. All information contained in this document was obtained in specific environments and is presented as an illustration. The result obtained in other operating environments may vary. Lenovo may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any references in this publication to non-Lenovo Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this Lenovo product, and use of those Web sites is at your own risk. Any performance data contained herein was determined in a controlled environment. Therefore, the result obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

© Copyright Lenovo 2026. All rights reserved.

This document, LP2454, was created or updated on July 14, 2026.

Send us your comments in one of the following ways:

- Use the online Contact us review form found at:
  https://lenovopress.lenovo.com/LP2454
- Send your comments in an e-mail to:
  comments@lenovopress.com

This document is available online at https://lenovopress.lenovo.com/LP2454.

Trademarks

Lenovo and the Lenovo logo are trademarks or registered trademarks of Lenovo in the United States, other countries, or both. A current list of Lenovo trademarks is available on the Web at https://www.lenovo.com/us/en/legal/copytrade/.

The following terms are trademarks of Lenovo in the United States, other countries, or both:
Lenovo®
ThinkAgile®
ThinkSystem®

The following terms are trademarks of other companies:

Intel®, the Intel logo, OpenVINO®, and Xeon® are trademarks of Intel Corporation or its subsidiaries.

Linux® is the trademark of Linus Torvalds in the U.S. and other countries.

Microsoft® is a trademark of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.