Report LLM su Server Lenovo 2x NVIDIA RTX PRO 6000

Framework metodologico per sistemi conversazionali enterprise

Una collaborazione Memori, Lenovo e Araneum e i loro team

a cura di:

Fabio Lecca, Matteo Mastranza, Paolo Mazzitti, Francesco Piccolo, Mario Sebastiani, Massimo Chiriatti.

Marzo 2026

Introduzione

Nel contesto dei precedenti studi condotti sui vantaggi dell'adozione di Large Language Models open source in esecuzione locale, è emerso con chiarezza che tali modelli, a partire da una certa soglia — indicativamente dagli 8 miliardi di parametri in su — sono in grado di soddisfare i principali casi d'uso aziendali, pur presentando limitazioni intrinseche legate alla loro natura e configurazione.

Muovendoci lungo il percorso tracciato nei report precedenti, questo nuovo studio si propone di analizzare le performance degli stessi modelli, oltre ad alcuni più recenti, su una nuova infrastruttura hardware, caratterizzata da capacità computazionali leggermente minori ma meno costose rispetto a quella impiegata in precedenza. L'obiettivo è duplice: da un lato, valutare quantitativamente le performance dei modelli, analizzando il carico di richieste gestibili per unità di tempo in un contesto di risorse più vincolate; dall'altro, identificare lo scenario operativo più idoneo per ciascun modello analizzato, così da orientare scelte di adozione consapevoli e strategicamente fondate.

Indice

1. Introduzione

Obiettivi del test

In questo nuovo ciclo di test, l'obiettivo è quello di valutare le prestazioni della nuova macchina, destinata all'esecuzione locale di LLM, confrontandole con quelle dell'infrastruttura utilizzata nello studio passato. Alla luce dei risultati emersi precedentemente, l'analisi si concentra solo su aspetti prestazionali, legati alla gestione di carico concorrente. In particolare, l'attenzione è rivolta alla capacità del sistema di gestire richieste simultanee provenienti da più utenti, mantenendo tempi di risposta adeguati. Non sono stati quindi ripetuti i test qualitativi sulle risposte generate dai modelli, in quanto i risultati già ottenuti, hanno fornito evidenze sufficienti in merito alla qualità delle risposte prodotte dai modelli utilizzati. Inoltre, i modelli impiegati in questo studio, sono gli stessi, oppure versioni più recenti, che in linea teorica presentano prestazioni qualitative uguali o migliori rispetto ai modelli già analizzati.
Come fatto precedentemente, i test quantitativi sono stati eseguiti utilizzando due metodologie. La prima prevede l'utilizzo del benchmark integrato di vLLM, impiegato per misurare il numero di richieste completate al secondo, assumendo una lunghezza in output di 128 token. La seconda invece, prevede l'utilizzo di Apache JMeter, per simulare scenari di utilizzo reale, con l'obiettivo di determinare il numero massimo di utenti simultanei che il sistema è in grado di gestire, mantenendo prestazioni accettabili.

2. Server Lenovo

Specifiche Hardware

Il server Lenovo ThinkSystem SR650a V4 impiegato per l'esecuzione dei test è configurato con componenti hardware di fascia alta, pensati per sostenere carichi di lavoro intensivi legati all'elaborazione di LLM. Di seguito una panoramica delle principali caratteristiche tecniche.

- **Processore:** il sistema è equipaggiato con 2x Intel Xeon 6737P, per un totale di 64 core fisici e 128 thread. Ogni CPU opera con una frequenza massima boost di 4.0 GHz ed è basata sull'architettura x86_64.
- **Memoria RAM:** il sistema dispone di 1.024 GB di RAM DDR5 ECC con una frequenza di 6400 MT/s, distribuiti su moduli da 64 GB ciascuno prodotti da SK Hynix.
- **GPU:** 2x NVIDIA RTX PRO 6000 Blackwell Server Edition, ognuna con circa 95,6 GB di memoria dedicata e supporto completo a CUDA 13.1. Queste GPU, basate sull'architettura Blackwell di ultima generazione, sono progettate per carichi AI ad altissime prestazioni e operano con il driver versione 590.48.01.
- **Storage:** il sistema è dotato di tre dispositivi di archiviazione ad alte prestazioni. Il disco principale è un volume RAID (RAID B545i-2i) da 446,6 GB, su cui risiede il sistema operativo. A questo si affiancano 2× SSD NVMe KIOXIA KCD8DPUG1T92 da 1,7 TB ciascuno, per una capacità NVMe complessiva di 3,4 TB.

- **Sistema operativo:** il server esegue Ubuntu 24.04.4 LTS a 64 bit, una distribuzione Linux stabile e compatibile con ambienti HPC e AI.

3. Modelli usati per i test

Nello studio sono stati analizzati diversi LLM open source per identificarne l'efficacia nel nostro caso di studio

LLM usati

Modello	Azienda	Context window (Tokens)	Memoria occupata (approx)
Qwen3-32B	Qwen	128k*	30 GB
Qwen3-30B-A3B	Qwen	128k*	28 GB
Qwen3-14B	Qwen	128k*	27.5 GB
Qwen3-8B	Qwen	128k*	7.64 GB
Qwen3-4B	Qwen	128k*	3.8 GB
Qwen3-1.7B	Qwen	32k	1.6GB
Qwen3-0.6B	Qwen	32k	523MB
granite-3.1-8b-instruct	ibm-granite	32k*	7.61GB
granite-3.3-8b-instruct	ibm-granite	128k	7.61GB
granite-3.3-2b-instruct	ibm-granite	128k	2.36GB
gemma-3-27b-it	google	128k	17GB
Llama-3.1-8B-Instruct	meta-llama	128k	4.5GB
Llama-3.3-70B-Instruct	meta-llama	128k	43GB
Llama-3.2-3B	meta-llama	128k	2GB

*I modelli Qwen3 supportano nativamente una context length di 32k ma arrivano a 128k token utilizzando la libreria YaRN supportata da vLLM.

Modelli più recenti

Per fare un confronto diretto con i report precedenti, i benchmark sono stati condotti sugli stessi modelli già usati. Inoltre, è stato scelto anche di includere dei modelli più recenti, con l'obiettivo di valutare il comportamento dell'infrastruttura su architetture e LLM più aggiornati.

Modello	Azienda	Context window	Memoria occupata

		(Tokens)	(approx)
gpt-oss-20b	OpenAI	128k	16 GB
gpt-oss-120b	OpenAI	128k	60 GB
Qwen3-Next-80B-A3B-Instruct	Qwen	256k	80 GB
NVIDIA-Nemotron-3-Nano-30B-A3B-BF16	Nvidia	1M	60 GB
phi-4 (14 B)	Microsoft	16k	12 GB

Proprietà dei modelli

Questi modelli hanno delle proprietà che li caratterizzano:

- **Dimensione e memoria**: modelli più grandi offrono prestazioni migliori ma richiedono molta più memoria. I modelli più piccoli sono meno esigenti ma adatti a compiti più semplici.
- **Finestra di contesto**: maggiore è la finestra di contesto, più le richieste possono essere articolate. Tuttavia, l'elaborazione di input molto lunghi può ridurre la qualità delle risposte.

Con le risorse hardware a disposizione, non è stato possibile testare modelli di dimensioni superiori, così da garantire un livello adeguato di concorrenza nell'ambiente di esecuzione previsto.

4. Benchmark vLLM: token/sec

Questo capitolo fornisce un'analisi quantitativa delle performance dei modelli testati, utilizzando un dataset di circa **53k conversazioni ShareGPT** derivate da un dataset iniziale di ~100 k interazioni presente su Hugging Face:
https://huggingface.co/datasets/anon8231489123/ShareGPT_Vicuna_unfiltered/blob/main/ShareGPT_V3_unfiltered_cleaned_split.json

vLLM

vLLM è una libreria open-source progettata per l'inferenza e il servizio di modelli di linguaggio di grandi dimensioni (LLM) in modo efficiente e ad alte prestazioni. Con il comando *serve* si possono istanziare i LLM installati per interrogarli e valutarne le prestazioni.

Metodologia di benchmark

I test sono stati condotti utilizzando input standardizzati e un limite massimo di token per la risposta (**Output-len**) impostato a 128 token. Per ogni modello, sono stati raccolti i seguenti indicatori di performance:

- **throughput (requests/s)**: il numero di richieste completate al secondo, che rappresenta un indicatore diretto della capacità del modello di gestire carichi multiutente.
- **total tokens/s**: il totale dei token processati al secondo, inclusi input ed output.
- **output tokens/s**: il numero di token generati dal modello al secondo, un indicatore dell'efficienza del modello nella produzione di risposte.

Per avviare i test è stato predisposto un container Docker con l'immagine di NVIDIA vLLM *nvcr.io/nvidia/vllm:25.11-py3,* per garantire un ambiente di esecuzione riproducibile e ottimizzato per le GPU a disposizione. Il container ha accesso diretto a tutte le GPU ed è stato configurato con i parametri di sistema necessari per gestire carichi di lavoro ad alta intensità di memoria. Il benchmark è stato eseguito tramite il comando *vllm bench throughput*, che sottopone il modello all'intero dataset di conversazioni, misurando le metriche di throughput descritte in precedenza.

Comando utilizzato per far partire i test:

```
docker run -it --rm   --gpus all   --ipc=host   --ulimit memlock=-1
--ulimit stack=67108864  -e HF_TOKEN="$TOKEN"
-v ~/test:/tmp/dataset   -v
~/.cache/huggingface:/root/.cache/huggingface
nvcr.io/nvidia/vllm:25.11-py3   vllm bench throughput
--dataset /tmp/dataset/ShareGPT_V3_unfiltered_cleaned_split.json --
output-len 128     --model $MODEL_NAME
```

Modello	Requests/s	Total tokens/s	Output tokens/s
ibm-granite/granite-3.1-8b-instruct	40,05	15.273,94	5.125,84
ibm-granite/granite-3.3-2b-instruct	90,72	34.600,06	11.611,58
ibm-granite/granite-3.3-8b-instruct	39,89	15.215,16	5.106,12
Qwen/Qwen3-0.6B	142,59	50.952,47	18.251,60
Qwen/Qwen3-1.7B	109,38	39.086,31	14.001,04
Qwen/Qwen3-4B	65,98	23.576,04	8.445,13
Qwen/Qwen3-8B	45,74	16.343,03	5.854,21
Qwen/Qwen3-14B	29,40	10.505,99	3.763,33
Qwen/Qwen3-32B	9,16	3.272,90	1.172,38
Qwen/Qwen3-30B-A3B	38,47	13.745,45	4.923,73
google/gemma-3-27b-it	8,06	2.992,63	1.031,30
meta-llama/Llama-3.2-3B	85,57	30.148,11	10.952,68
meta-llama/Llama-3.3-70B-Instruct	7,66	2.699,42	980,69

meta-llama/Meta-Llama-3.1-8B-Instruct	49,09	17.294,79	6.283,12

Nota: per il modello più grande, meta-llama/Llama-3.3-70B-Instruct, è stato necessario includere due parametri aggiuntivi per consentirne l'esecuzione: --tensor-parallel-size 2, che distribuisce il modello sulle due GPU in parallelo, e --gpu-memory-utilization 0.95, che permette di sfruttare la memoria della GPU fino al 95%. Senza questi accorgimenti, il modello non poteva essere eseguito, date le sue grandi dimensioni.

Test con parallelizzazione

Per valutare i benefici dell'utilizzo simultaneo di entrambe le GPU, i benchmark sono stati ripetuti per tutti i modelli aggiungendo il parametro --tensor-parallel-size 2. I risultati mostrano comportamenti differenti a seconda della dimensione del modello.

Per i modelli più piccoli (fino a 8B parametri), lo speedup è risultato impercettibile, inferiore al 10%, e in diversi casi si è osservata addirittura una leggera degradazione delle performance. Questo è attribuibile all'overhead di comunicazione introdotto dalla distribuzione del modello tra le due GPU, che in questo caso supera il guadagno computazionale ottenibile.

Per i modelli più grandi, invece, il beneficio è stato significativo: nel caso di Qwen 32B è stato registrato un miglioramento fino al doppio delle richieste al secondo rispetto alla configurazione con singola GPU. Modelli di queste dimensioni tendono a saturare la memoria di una singola GPU, costringendo il sistema a compromessi quali batch ridotti e swap di memoria, che penalizzano sensibilmente le performance. La parallelizzazione elimina questi colli di bottiglia, rendendo la doppia GPU conveniente non solo nei casi in cui il modello non possa essere caricato su una singola scheda, ma anche quando le dimensioni del modello sono tali da comprometterne l'efficienza operativa.

Per i modelli più piccoli, un approccio alternativo e più efficace consiste nel lanciare due o più istanze indipendenti di vLLM, una per GPU, massimizzando così l'utilizzo dell'hardware disponibile senza introdurre overhead di comunicazione.

Modello	Requests/s	Total tokens/s	Output tokens/s	Incremento
ibm-granite/granite-3.1-8b-instruct	44,46	16.956,54	5.690,52	0,11
ibm-granite/granite-3.3-2b-instruct	87,32	33.306,60	11.177,50	-0,04
ibm-granite/granite-3.3-8b-instruct	44,43	16.946,61	5.687,18	0,11
Qwen/Qwen3-0.6B	138,53	49.501,74	17.731,93	-0,03
Qwen/Qwen3-1.7B	102,07	36.474,40	13.065,43	-0,07
Qwen/Qwen3-4B	69,20	24.728,99	8.858,13	0,05
Qwen/Qwen3-8B	49,15	17.561,83	6.290,79	0,07
Qwen/Qwen3-14B	34,01	12.152,55	4.353,14	0,16
Qwen/Qwen3-32B	18,16	6.489,71	2.324,67	0,98

Qwen/Qwen3-30B-A3B	55,24	19.738,85	7.070,62	0,44
google/gemma-3-27b-it	14,61	5.428,40	1.870,69	0,81
meta-llama/Llama-3.2-3B	82,33	29.005,72	10.537,65	-0,04
meta-llama/Meta-Llama-3.1-8B-Instruct	54,16	19.082,03	6.932,42	0,10

Test modelli più recenti

Come già specificato in precedenza, i test sono stati condotti anche su dei modelli più recenti e di seguito ci sono i risultati ottenuti.

Modello	Requests/s	Total tokens/s	Output tokens/s	Note	Output-len
openai/gpt-oss-20b	69,65	24.712,98	8.915,82		128
openai/gpt-oss-120b	37,51	13.308,23	4.801,27		128
openai/gpt-oss-120b	46,47	16.487,32	5.948,21	Con parallelizzazione	128
Qwen/Qwen3-Next-80B-A3B-Instruct	11,98	4.279,79	1.533,06	Con parallelizzazione	128
nvidia/NVIDIA-Nemotron-3-Nano-30B-A3B-BF16	13,48	4.907,80	1.725,65	Con versione più recente di vllm	128
microsoft/phi-4	24,58	8.703,17	3.146,00		128

Comparazione con la precedente infrastruttura

Per capire la potenza di questa nuova macchina, sono stati comparati i benchmark ottenuti con il server precedente. A differenza della scorsa analisi, i test sono stati eseguiti utilizzando gli stessi LLM, per garantire una comparazione omogenea.

	Macchina	Request/s	Performance Ratio
ibm-granite/granite-3.1-8b-instruct	H200	85,46	0,52
	RTX 6000	44,46	
ibm-granite/granite-3.3-2b-instruct	H200	115,08	0,76
	RTX 6000	87,32	
ibm-granite/granite-3.3-8b-instruct	H200	82,42	0,54
	RTX 6000	44,43	

Qwen/Qwen3-0.6B	H200	150,43	0,92
	RTX 6000	138,53	
Qwen/Qwen3-1.7B	H200	134,34	0,76
	RTX 6000	102,07	
Qwen/Qwen3-4B	H200	108,78	0,64
	RTX 6000	69,20	
Qwen/Qwen3-8B	H200	90,91	0,54
	RTX 6000	49,15	
Qwen/Qwen3-14B	H200	65,5	0,52
	RTX 6000	34,01	
Qwen/Qwen3-32B	H200	35,24	0,52
	RTX 6000	18,16	
Qwen/Qwen3-30B-A3B	H200	75,35	0,73
	RTX 6000	55,24	
google/gemma-3-27b-it	H200	28,24	0,52
	RTX 6000	14,61	
meta-llama/Llama-3.2-3B	H200	129,53	0,64
	RTX 6000	82,33	
meta-llama/Llama-3.3-70B-Instruct	H200	18,77	0,41
	RTX 6000	7,66	
meta-llama/Llama-3.1-8B-Instruct	H200	88,73	0,61
	RTX 6000	54,16	

Risultati

Dai confronti emerge che, per i modelli più potenti (>= 8B), le prestazioni della nuova macchina risultano circa dimezzate rispetto alla precedente, ma a fronte di un costo più contenuto e sostenibile per le aziende. Per i modelli di dimensioni inferiori, generalmente meno performanti, le differenze sono più contenute, ma comunque significative.

5. Test di prestazioni: utenti simultanei

L'obiettivo principale del test è identificare il giusto compromesso tra tre fattori fondamentali:

- **Qualità delle risposte**: ovvero la capacità del modello di fornire output coerenti, rilevanti e precisi rispetto alla richiesta
- **Tempo di risposta**: parametro cruciale in applicazioni interattive o real-time, che misura la reattività percepita dal punto di vista dell'utente
- **Numero di richieste gestibili nell'unità di tempo**: indicatore diretto della scalabilità del sistema, fondamentale per mantenere performance accettabili anche in scenari ad alto traffico.

Questi tre elementi sono correlati tra loro: migliorare la qualità richiede modelli più complessi e pesanti, che però aumentano i tempi di risposta e riducono il throughput complessivo. Al contrario, l'utilizzo di modelli più leggeri migliora scalabilità e latenza, ma può compromettere la qualità delle risposte. L'equilibrio tra questi fattori è quindi essenziale per garantire un'esperienza d'uso ottimale all'interno di un contesto come quello analizzato.

Per arrivare a quantificare un numero di utenti simultanei il più possibile corrispondente con un caso d'uso reale, sono state effettuate le seguenti considerazioni:

- la "Qualità delle risposte" è stata assicurata selezionando un modello di riferimento per i test, ovvero openai/gpt-oss-120b, che è molto recente e ha dato risultati qualitativi molto buoni
- è stata definita una soglia di tempo di risposta accettabile (100 secondi al massimo per ottenere una risposta complessa), in modo da garantire la qualità percepita dall'utente
- è stato quindi eseguito un insieme di test aumentando via via il numero di utenti simultanei, in modo da arrivare al numero massimo di richieste simultanee che rispettano il parametro del tempo di attesa (cosiddetto "punto di rottura"),
- è stato utilizzato un prompt tipico di AIsuru costituito da 50.000 caratteri prevalentemente in lingua italiana (circa 12.500 token), utilizzando tale dimensione come valore medio, tenendo conto che alcune richieste possono essere più semplici ed altre più complesse e richiedendo più iterazioni, come ad esempio quelle che prevedono il *tool calling.*

Test Apache JMeter

In quest'ottica, il test di performance è stato progettato con carichi progressivi, fino a raggiungere un punto di effettivo stress dell'infrastruttura, identificato nel carico di un centinaio di utenti simultanei.

Modello	GPU Mem	Num thread	Tempo totale	Throughput	Media	Min	Max
openai/gpt-oss-120b	88,17Gb	1	00:00:06	0.2 req/s	5718	5718	5718
openai/gpt-oss-120b	88,17Gb	2	00:00:12	0.2 req/s	9582	7186	11978

openai/gpt-oss-120b	88,17Gb	4	00:00:13	0.3 req/s	10346	8863	12930
openai/gpt-oss-120b	88,17Gb	8	00:00:17	0.5 req/s	13638	12773	16139
openai/gpt-oss-120b	88,17Gb	16	00:00:25	0.6 req/s	19813	17728	23877
openai/gpt-oss-120b	88,17Gb	32	00:00:38	0.8 req/s	30309	27103	38102
openai/gpt-oss-120b	88,17Gb	64	00:01:02	1.0 req/s	49020	42106	61451
openai/gpt-oss-120b	88,17Gb	128	00:02:03	1.0 req/s	90092	53011	122705

Il test è stato ripetuto istanziando lo stesso modello sulle due GPU e avendo un bilanciatore hproxy che bilanciava due istanze di modello che rispondevano a porte diverse, ed è emerso che con questa configurazione la doppia GPU scala linearmente nel numero di thread simultanei.

6. Modelli NIM di Nvidia

La licenza AI Enterprise di Nvidia consente di accedere al repository NGC (NVIDIA Container Registry), che contiene immagini Docker certificate che espongono modelli e motori di inferenza ottimizzati per le architetture NVIDIA con servizi web compatibili con OpenAI mediante i NIM (Nvidia Inference Microservices), usufruendo di un supporto costante che prevede anche aggiornamenti tempestivi e verifiche di sicurezza delle immagini.
Per verificare le prestazioni dei modelli NIM è stato quindi condotto un test specifico misurando le prestazioni dell'inferenza a parità di modello nelle versioni NIM e non-NIM (servite da un motore open source vllm o sglang).

Il test è stato ripetuto più volte ottenendo dei valori medi per le principali metriche: token/sec e TTFT (Time To First Token). Si riporta un confronto tra due esecuzioni tipiche:

OpenAI gpt-oss-120b - NIM	OpenAI gpt-oss-120b - sglang
============ Serving Benchmark Result ============ Successful requests: 300 Request rate configured (RPS): 10.00 Benchmark duration (s): **48.08** Total input tokens: 63107 Total generated tokens: 63992 Request throughput (req/s): 6.24 Output token throughput (tok/s): **1330.84** Peak output token throughput (tok/s): **2104.00** Peak concurrent requests: 125.00 Total Token throughput (tok/s): **2643.28** ----------Time to First Token---------- Mean TTFT (ms): **116.36** Median TTFT (ms): 111.84 P99 TTFT (ms): 203.43 -----Time per Output Token (excl. 1st token)------ Mean TPOT (ms): 52.08 Median TPOT (ms): 53.17 P99 TPOT (ms): 74.86 ----------Inter-token Latency---------- Mean ITL (ms): 50.94	============ Serving Benchmark Result ============ Successful requests: 300 Request rate configured (RPS): 10.00 Benchmark duration (s): **62.61** Total input tokens: 63107 Total generated tokens: 64496 Request throughput (req/s): 4.79 Output token throughput (tok/s): **1030.15** Peak output token throughput (tok/s): **2491.00** Peak concurrent requests: 182.00 Total Token throughput (tok/s): **2038.12** ----------Time to First Token---------- Mean TTFT (ms): **480.36** Median TTFT (ms): 231.41 P99 TTFT (ms): 1713.48 -----Time per Output Token (excl. 1st token)------ Mean TPOT (ms): 116.99 Median TPOT (ms): 95.63 P99 TPOT (ms): 545.90 ----------Inter-token Latency---------- Mean ITL (ms): 84.04

Median ITL (ms): 47.28 P99 ITL (ms): 111.95 ==	Median ITL (ms): 57.95 P99 ITL (ms): 563.67 ==

Dai test condotti si evince che la versione NIM di openai/gpt-oss-120b rispetto alla versione con sglang ha un incremento di velocità fino al 30% sul throughput totale in token/sec e ha un TTFT più costante (gli altri motori iniziano con tempi maggiori, anche se poi migliorano con il tempo arrivando a prestazioni paragonabili).

7. Dimensionamento di RAM e CPU

Il dimensionamento della RAM e della CPU di un server è sempre un tema molto importante nel caso di applicazioni classiche per determinare il numero massimo di utenti che possono utilizzare il sistema. Per questo motivo, durante tutti i test è stato monitorato l'utilizzo di RAM e CPU per capire se l'inferenza porta con sé anche un consumo di risorse significativo, tale da richiedere una correzione nei parametri di dimensionamento.
In realtà durante i test, proprio perchè l'elaborazione dell'inferenza è totalmente a carico della GPU, si è apprezzato un consumo di risorse molto limitato, inferiore al 10% della CPU, indipendentemente dal numero degli utenti simultanei che eseguivano inferenze, ed un consumo di RAM "classica" molto limitato, dell'ordine dei 50Mb per utente. Per questo motivo possiamo affermare che le applicazioni che fanno largo uso dei motori LLM per l'inferenza non risentono di richieste aggiuntive di RAM e CPU rispetto alla normale occupazione delle risorse per lo stesso numero di utenze.

8. Conclusioni

Lo studio corrente è stato completato a pochi mesi di distanza dal precedente, e nel frattempo abbiamo avuto a disposizione un'infrastruttura notevolmente più performante rispetto al costo dell'hardware, nonché una generazione successiva per tutti i modelli open source che erano stati testati a gennaio 2025. Questi due fattori lasciano ben sperare sul fatto che le soluzioni LLM on premise saranno via via sempre più utilizzabili in ambiti lavorativi dove la privacy e la disponibilità dei dati aziendali sia un fattore determinante per la scelta di tale soluzione.

La nuova infrastruttura, pur mostrando delle performance ridotte rispetto alla precedente (costituita dalle schede H200), in realtà democratizza l'accesso ai modelli LLM locali in quanto il costo di accesso è notevolmente minore, inoltre il taglio della RAM da 96Gb e la possibilità di montare due GPU consente l'utilizzo di modelli con maggior numero di parametri. Sono stati infatti identificati dei modelli open source che si integrano perfettamente con la piattaforma AISuru, forniscono una qualità delle risposte più che soddisfacente (in particolare Nemotron-3-super, gpt-oss-120b e Qwen3.5-35B-A3B) e consentono anche un alto grado di parallelismo delle richieste, in modo da consentire a più utenti aziendali simultanei (nell'ordine delle decine) di interagire correttamente con gli agenti AIsuru.