Ori’s Guide to Self-Hosted LLMs

Ciera Fowler
ML Engineering Lead

September 2024

Agenda

I. Introduction
II. Self-Hosted LLMs
III. Model Selection
IV. Optimising LLM Inference

Learning Outcomes

- Learn when to self-host large language models instead of using APIs providers
- Learn about GPU hardware requirements for LLMs
- Learn how to select the appropriate model for your use case
- Learn LLM optimisation strategies: quantisation, parallelism, and inferencing engines

Source: [Hugging Face]

I. Introduction

Ciera Fowler

Now:	**ML Engineering Lead @ Ori**
	MBA Candidate @ LBS
2023	Staff Data Engineer @ Hinge
2022	Senior Data Architect @ Clear Street
2019 – 2022	Data Engineering Manager @ Ocrolus
2017 – 2019	Consultant @ Element22
2014 – 2017	Data Engineer @ Enso
2010 – 2014	ME in Chemical Engineering @ Cooper Union

Ori is the AI Native Cloud Platform

Deploy AI-optimized GPU instances for training, finetuning and inference workloads. Significantly reduce GPU costs compared to traditional cloud providers. Scale effortlessly from on-demand instances to custom private clouds with bare-metal, virtual machines and Kubernetes GPU instances.

On-Demand Cloud

Get your models running quickly on virtual machine instances of state-of-the-art GPUs with 75% cost savings.

Private Cloud

Achieve the best possible architecture tailored to your needs, with options across data center grade GPUs, networking, storage, and massive-scale GPUs.

Serverless Kubernetes

Blend the scalability and flexibility of Kubernetes with the simplicity of a serverless platform to get your AI models to market faster while optimizing GPU usage.

Managed Kubernetes

Build your custom Kubernetes service for massive scale AI projects with the tools, storage and networking designed to maximize model performance.

II. Self-Hosted LLMs

Why Self Host LLMs?

Ori customers typically say:

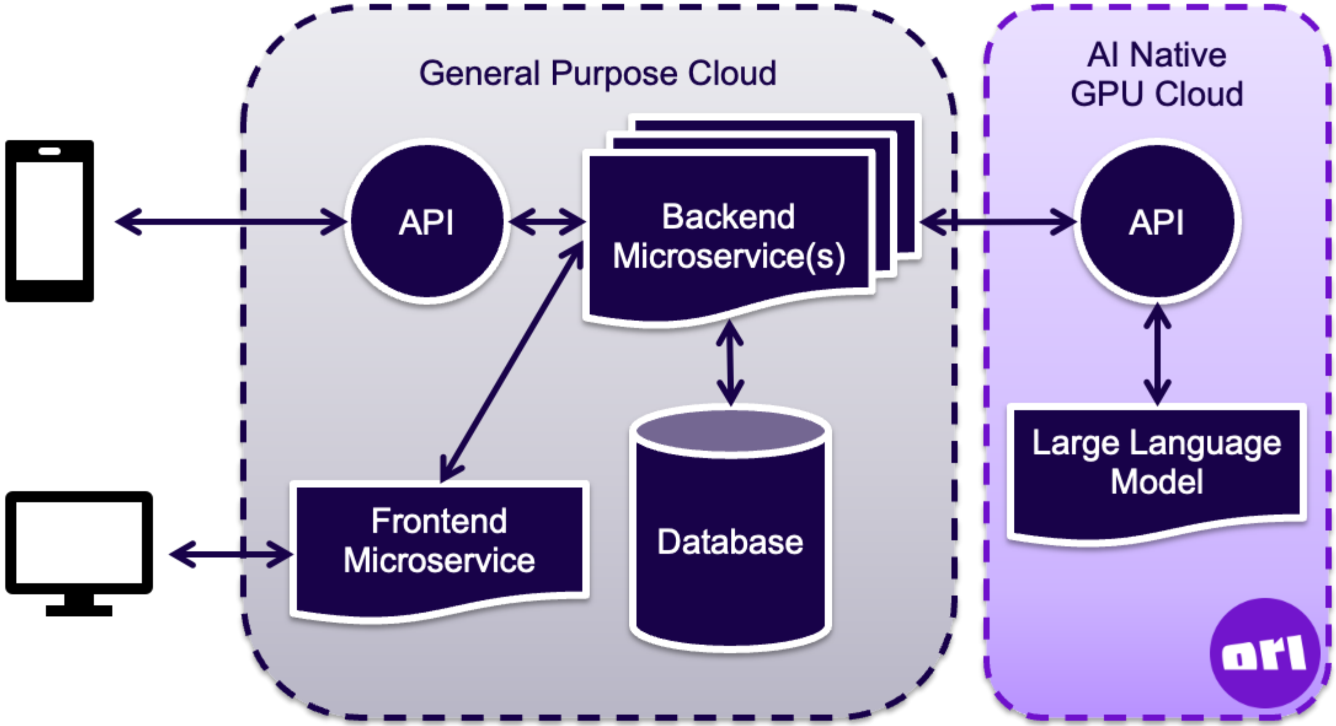
- Data privacy and non-expatriation
- Customise (Optimise?) models
  (e.g. finetunning, quantisation)
- No “noisy neighbor” interference
- Reduce Cost
  (e.g. high utilisation, summarisation use cases)
- Predictable Costs
- Prevent Vendor Lock-in
- Control model versioning/updates

Example use cases:

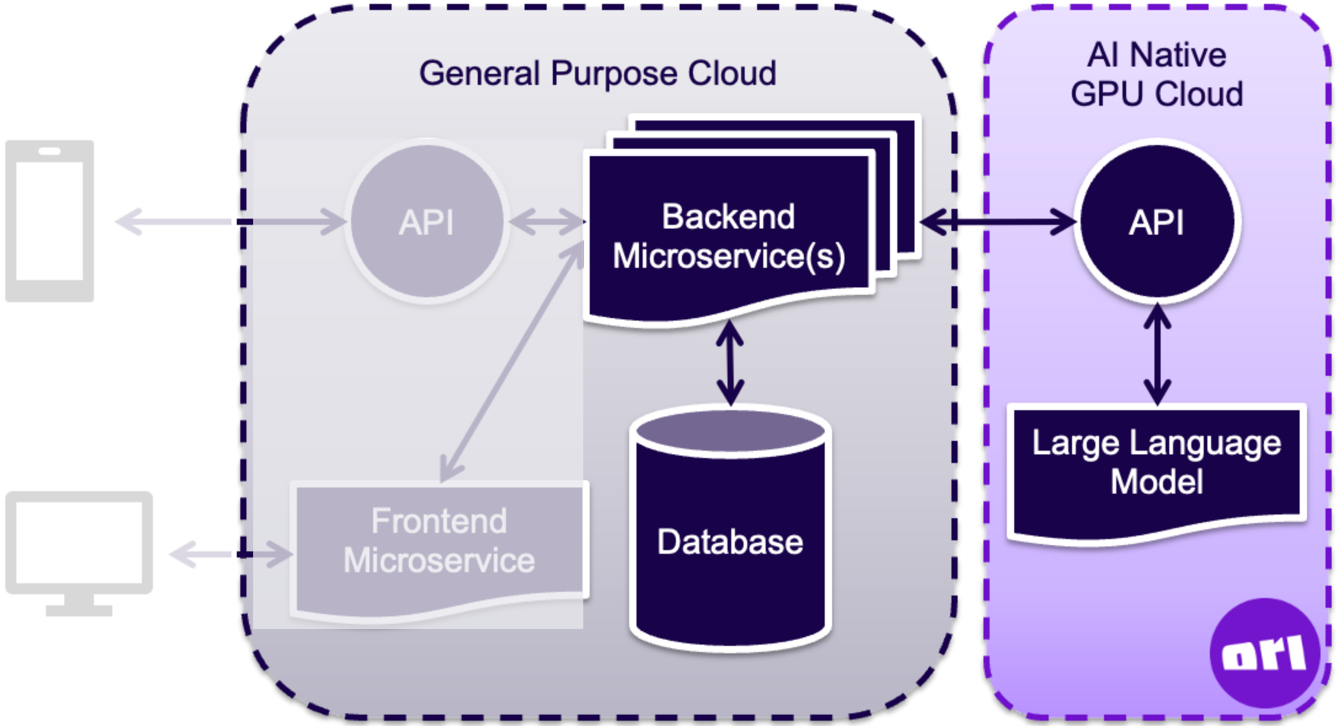
- Research using sensitive healthcare data can self-host to ensure participant confidentiality and compliance with data privacy regulations like GDPR.
- Research summarising/analysing large social media datasets for sentiment analysis may face prohibitive API costs.
- Long-term research projects will want to use the same model version throughout analysis to prevent to prevent inconsistencies in results over time.
- Research that requires knowlege of a regional dialect can fine-tunned a model cost effectively.

Maintain complete control over the data, model, and environment.

Reference Architecture



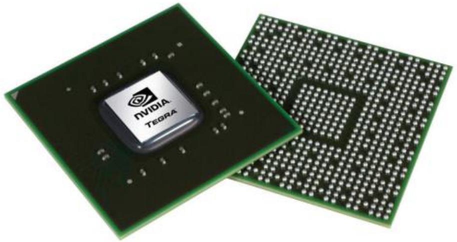
Reference Architecture



GPUs and CPUs

CPU - Central Processing Units

- The CPU is like the brain of the computer. Everything the computer does, like running programs or apps, goes through the CPU.
- It carries out instructions, makes calculations, and completes tasks.
- Good at handling a few complex tasks at a time. It's optimized for sequential tasks where one step is completed before moving to the next.
- Best for general computing tasks, like running word processors, web browsers, or operating systems.



GPU - Graphics Processing Units

- The GPU is specialized for handling tasks that involve processing large amounts of data at once, such as rendering images and videos or performing machine learning computations.
- It excels at parallel processing, meaning it can handle thousands of smaller tasks simultaneously, which makes it ideal for graphics, AI, and other data-intensive workloads.
- Its ability to process multiple operations in parallel is crucial for accelerating tasks that would take much longer on a CPU.

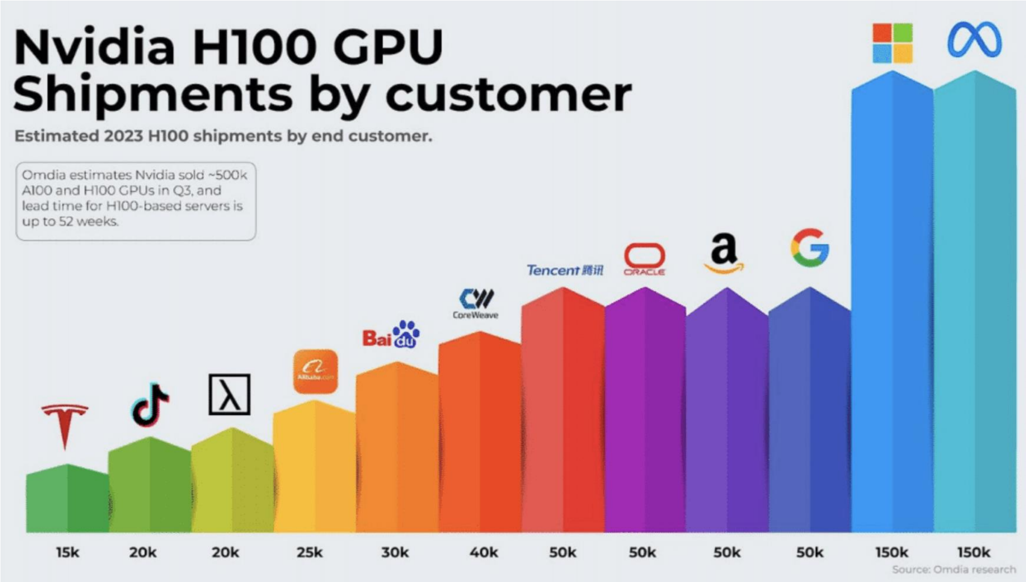
All GPUs are not created equally...

NVIDIA Chips Dominate the Market:

- Industry-leading performance and reliability
- Software ecosystem (CUDA) and developer support

NVIDIA Chips:

- A100 – released 2020 (previous generation)
- H100 – released 2022 (current flagship)
- H200 – expected 2024 (next generation)



H100 wholesale prices are estimated to be at least **US$30,000 each**

Source: Tom's Hardware

Nvidia data centre GPU market share is estimated at **98%**

Source: HPC Wire

GPU Configuration

Key GPU Attributes

- **vCPU** – virtual Central Processing Units
- **RAM** – Random Access Memory, aka "fast memory", temporarily holds data that the computer needs quick access to while running programs.
- **VRAM** - video memory is specialised RAM that helps a computer's graphics card quickly process images and large tasks like LLMs.
- **Storage NVMe** - high-speed storage (faster than SSD) that enables the computer to access and save large files almost instantly.
- **Storage SSD** - Solid-State Drive is "slow memory", that is used to keep files and programs over long time periods.
- **Bandwidth** - How fast data move can into and out of the system.

GPUs	VRAM/GPU	vCPUs	RAM (GB)	Storage SSD	Storage NVMe	Bandwidth (Gbps)	Price (USD)
1 × NVIDIA H100	80	30	380	50	3840	8	3.24/h
2 × NVIDIA H100	160	60	760	50	7680	16	6.48/h
4 × NVIDIA H100	320	120	1520	50	15360	25	12.97/h
1 × NVIDIA A100	80	15	180		300	8	2.74/h
2 × NVIDIA A100	160	30	360		500	16	5.48/h
4 × NVIDIA A100	320	60	720		500	25	10.96/h

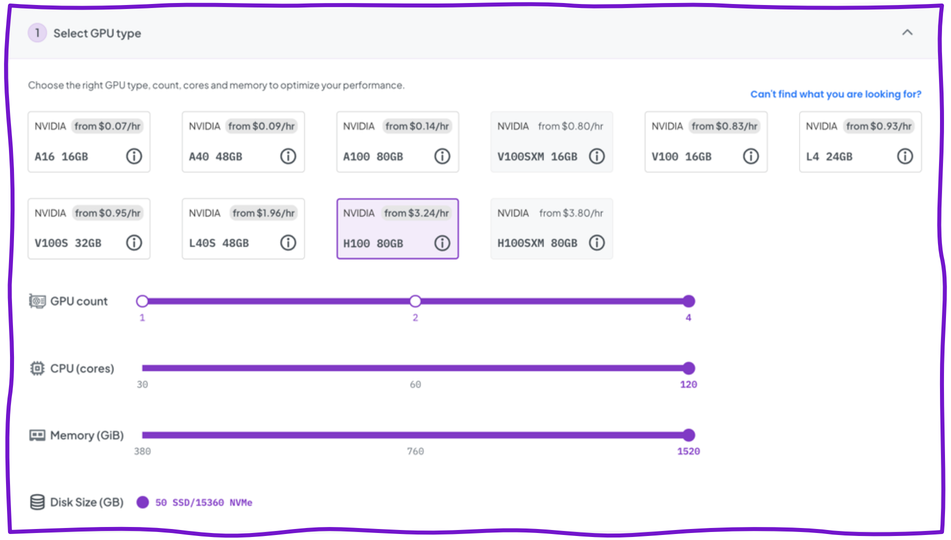
Practical Implications

1. **VRAM** –the entire model needs to be held in memory, based on the number of parameters and precision of the model
2. **(v)CPU** – the number of cores will determine the level of parallelisation you can support
3. **Bandwidth** – for online applications (e.g. chatbots), this can become the bottleneck

Setting up a GPU Configuration on Ori

To Launch a VM on Ori's Public Cloud:

1. Select GPU type & count
   → *See previous slide*
2. Choose a location
   → *Geographical distance impacts latency*
3. Choose an OS image
   → *Typically Ubuntu v22.04 is a safe option*
4. Configure Init Script
5. Set up networking
   → *Allows external network access*
6. Add public SSH key
   → *This is like a "password" to allow access*
7. Name your virtual machine



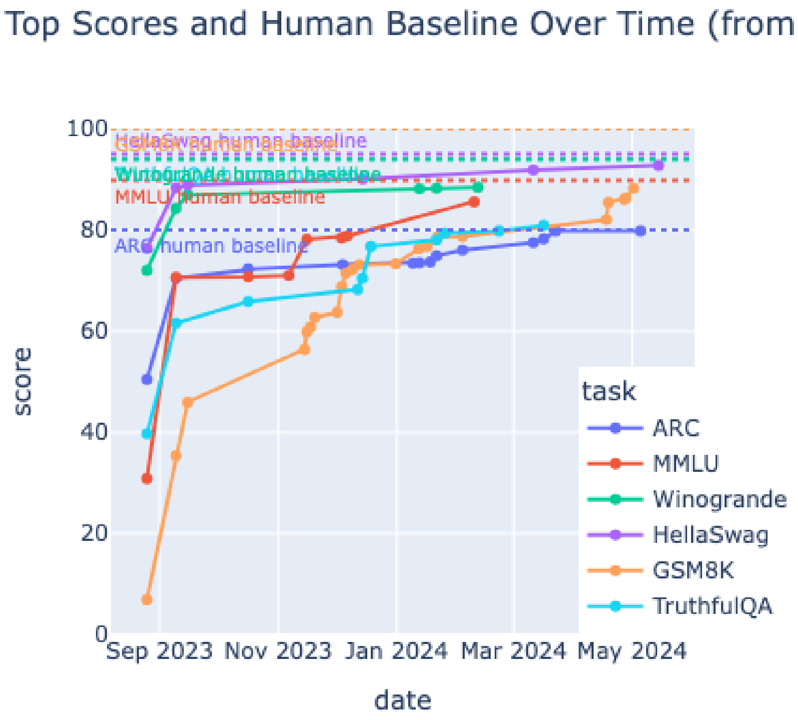
III. Model Selection

Hugging Face

A company and open-source platform for machine learning that provides:

- Tools – leaderboards, datasets
- Libraries – transformers, diffusers, accelerate
- Models

Open LLM Leaderboard V1



Source: Hugging Face

Hugging Face's Open LLM Leaderboard

Leaderboards are used to aggregate and compare model quality benchmarks

- Hugging Face has become the leading source for these open-source models, widely adopted by AI researchers and developers.
- Others are also popular, such as OpenAI's benchmarks for closed models.
- Leading models were converging in performance, raising concerns about a plateau in innovation
- In response, HuggingFace released version 2 of their leaderboard in June 2024

T	Model	Average	IFEval	BBH	MATH Lvl 5	GPQA	MUSR	MMLU-PRO
	MaziyarPanahi/calme-2.4-rys-78b	50.26	80.11	62.16	37.69	20.36	34.57	66.69
	dnhkng/RYS-XLarge	44.75	79.96	58.77	38.97	17.9	23.72	49.2
	MaziyarPanahi/calme-2.1-rys-78b	44.14	81.36	59.47	36.4	19.24	19	49.38
	MaziyarPanahi/calme-2.2-rys-78b	43.92	79.86	59.27	37.92	20.92	16.83	48.73
	MaziyarPanahi/calme-2.1-qwen2-72b	43.61	81.63	57.33	36.03	17.45	20.15	49.05
	MaziyarPanahi/calme-2.2-qwen2-72b	43.4	80.08	56.8	41.16	16.55	16.52	49.27
	dfurman/Qwen2-72B-Orpo-v0.1	43.32	78.8	57.41	35.42	17.9	20.87	49.5
	Qwen/Qwen2-72B-Instruct	42.49	79.89	57.48	35.12	16.33	17.17	48.92
	abacusai/Dracarys-72B-Instruct	42.37	78.56	56.94	33.61	18.79	16.81	49.51
	VAGOsolutions/Llama-3.1-SauerkrautLM-70b-Instruct	42.24	86.56	57.24	29.91	12.19	19.39	48.17
	alpindale/magnum-72b-v1	42.17	76.06	57.65	35.27	18.79	15.62	49.64
	meta-llama/Meta-Llama-3.1-70B-Instruct	41.74	86.69	55.93	28.02	14.21	17.69	47.88

Source: Hugging Face

Why were benchmarks plateauing?

- **Homogeneity in Training Data**: Many models are trained on similar datasets, leading to less variety in outputs.
- **Overfitting on Evaluation Data Sets**: models appeared to be trained on benchmark data or on data very similar to benchmark data.
- **Overemphasis on Scaling**: Simply increasing model size isn't yielding the same performance improvements as before.
- **Benchmark Suitability**: Benchmarks were not pushing models to develop new, real-world capabilities like reasoning or long-term memory.
- **Benchmark Accuracy**: Some benchmarks contained errors, e.g. MMLU was investigated by several groups (MMLU-Redux and MMLU-Pro), which surfaced mistakes in its responses.

Source: Hugging Face

How were New Benchmarks Selected?

1. Evaluation quality:

- Human review of dataset
- Widespread use in the academic and/or open-source community

2. Reliability and fairness of metrics:

- Multichoice evaluations are, in general, fair across models.
- Generative evaluations should either constrain the format very much or use very unambiguous metrics or post-processing to extract the correct answers.

3. General absence of contamination in models:

- Gating
- Newness

4. Measuring model skills that are interesting for the community:

- Correlation with human preferences
- Evaluation of a specific capability we are interested in

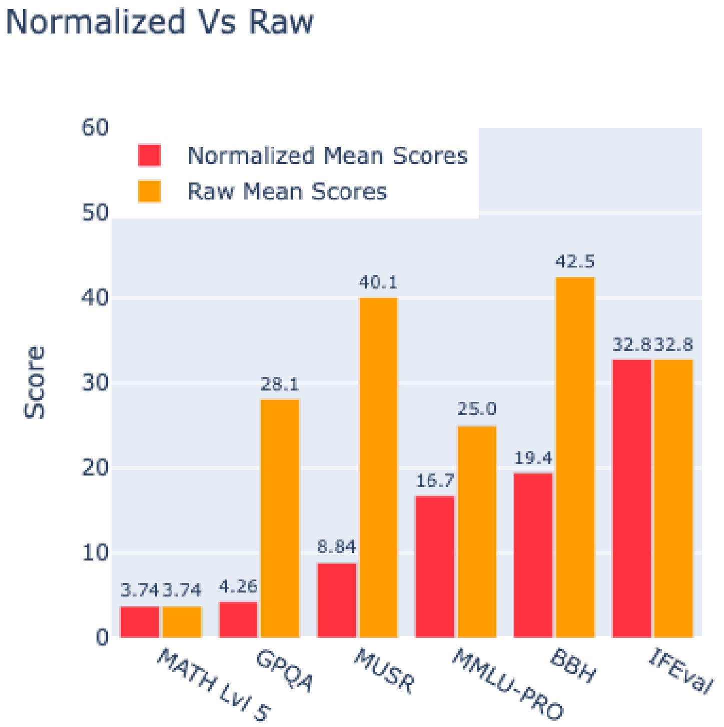
Source: Hugging Face

Change to Use Normalised Scores

In addition to the changing which benchmarks are used, the average ranking now uses normalised scores:

- The random baseline is 0 points
- The maximal possible score is 100 points

For example, in a benchmark containing two choices for each question, a random baseline will get 50 points (out of 100 points). Therefore, the range changed so that a 50 on the raw score is a 0 on the normalized score.



Source: [Hugging Face]

How did the rankings change?

The top 10 models under the new rankings, as of the launch of Open LLM Leadboard V2 (top table). Some models maintained a relatively stable top 10 ranking (**in bold**).

Rank	New Leaderboard Ranking
⭐	**Qwen/Qwen2-72B-Instruct**
2	**meta-llama/Meta-Llama-3-70B-Instruct**
3	*microsoft/Phi-3-medium-4k-instruct*
4	**01-ai/Yi-1.5-34B-Chat**
5	**CohereForAI/c4ai-command-r-plus**
6	**abacusai/Smaug-72B-v0.1**
7	Qwen/Qwen1.5-110B
8	Qwen/Qwen1.5-110B-Chat
9	microsoft/Phi-3-small-128k-instruct
10	01-ai/Yi-1.5-9B-Chat

There was a large backlog of models to be evaluated on the new benchmarks at the time of launch. Many more models have been released and evaluated in the 3 months since Leaderboard V2 was launched and the top 10 have changed.

T	Model	Average
💬	MaziyarPanahi/calme-2.4-rys-78b	50.26
🔶	dnhkng/RYS-XLarge	44.75
💬	MaziyarPanahi/calme-2.1-rys-78b	44.14
💬	MaziyarPanahi/calme-2.2-rys-78b	43.92
💬	MaziyarPanahi/calme-2.1-qwen2-72b	43.61
💬	MaziyarPanahi/calme-2.2-qwen2-72b	43.4
💬	dfurman/Qwen2-72B-Orpo-v0.1	43.32
💬	Qwen/Qwen2-72B-Instruct	42.49
🔶	abacusai/Dracarys-72B-Instruct	42.37
🔶	VAGOsolutions/Llama-3.1-SauerkrautLM-70b-Instruct	42.24

Source: HuggingFace

Model Selection Best Practice

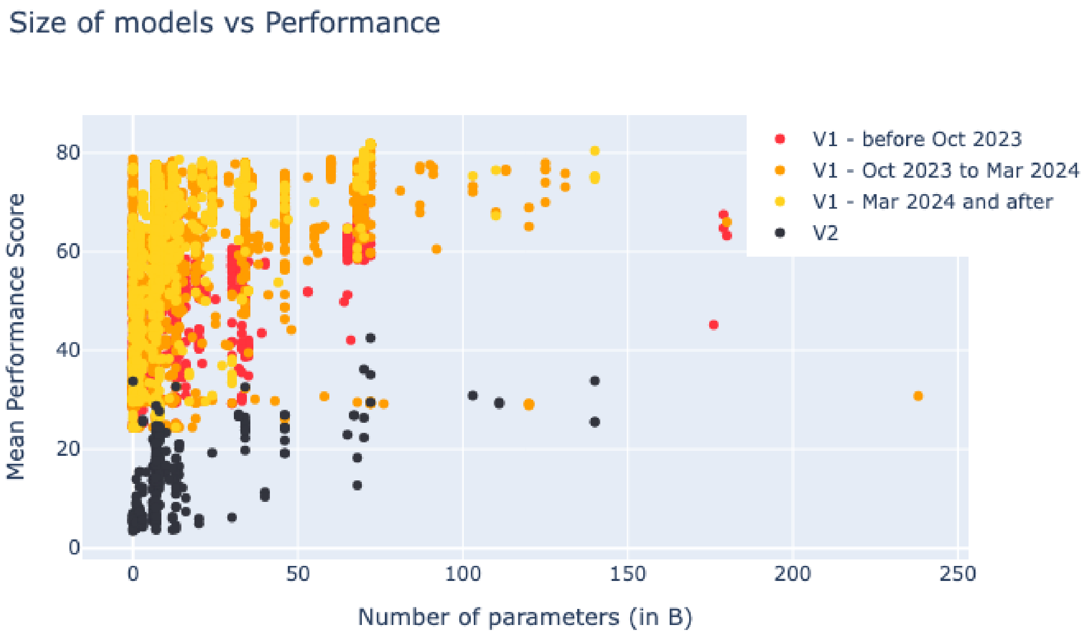
With the rapid rate of change in the leaderboard, it may seem futile to select the top-ranking model for a project when it soon will fall in the rankings anyway.

Best practice is to take a more holistic approach to model selection:

- Limit your selection to models below a certain size. There is a tradeoff between model size and resource costs. We typically expect bigger models to produce higher quality results, but sometimes the incremental improvement is not worth the cost.
- Rank the remaining options by the subset of benchmarks relevant to your use case, rather than the average of all Hugging Face selected benchmarks.
- If you have the resources, develop your own “benchmark” or set of prompts and acceptable responses to test the top-ranking models from the previous criteria.

Model Size and Performance Trade Offs

The evolution of all the 7400 evaluated models on the Open LLM Leaderboard V1 (red, yellow, orange dots) and V2 (black dots) through time reveal a strong trend going from larger (red dots) models to smaller (yellow dots) models while at the same time improving performance.



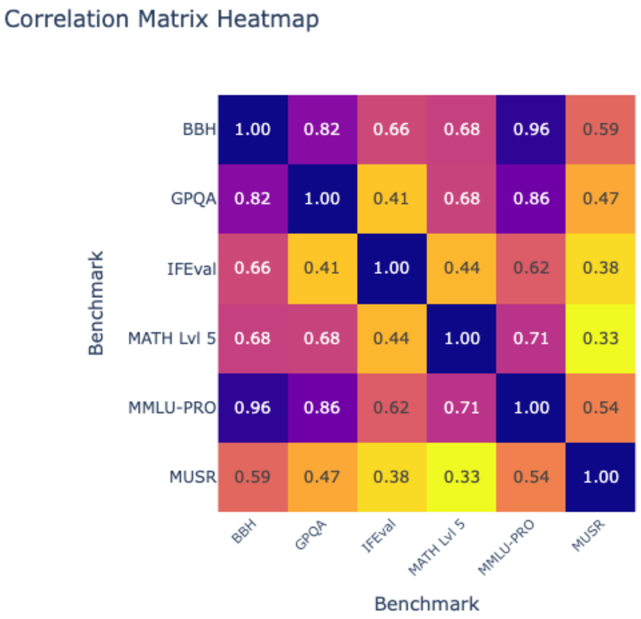
Source: [HugginFace](HuggingFace)

The 6 Leaderboard V2 Benchmarks

- **MMLU-Pro**: Harder, refined question set with 10 multichoice answers each, improved quality and noise reduction. Focuses on reasoning.
- **GPQA**: Expert-designed, PhD questions aimed at preventing model contamination by gating the questions.
- **MuSR**: Complex, multi-step reasoning problems like murder mysteries or team allocation.
- **MATH-Lvl5** : High-school-level competition math problems, focusing on strict output formats.
- **IFEval**: tests the capability of models to clearly follow explicit instructions, such as “include keyword x”, rather than the actual contents generated.
- **BBH**: 23 challenging tasks testing logic, language understanding, and world knowledge. It is reportedly difficult for both models and humans, and strongly correlates with human preferences.

Source: HuggingFace

Correlation of Benchmark Results



Different evaluation results are not always correlated with one another:

- **MMLU-Pro** and **BBH** are rather well correlated. These benchmarks are also quite correlated with human preference (i.e., they tend to align with human judgment)
- **IFEval** targets chat capabilities. It investigates whether models can follow precise instructions, so it tends to favour chat and instruction-tuned models, with pretrained models having a harder time reaching high performances.
- **MMLU-Pro** and **GPQA** provide the best performance on model knowledge rather than alignment or chat capabilities.
- **MATH-Lvl5** is obviously interesting for people focusing on math capabilities.

Source: HuggingFace

II. Optimising LLM Inference

Ori's Holistic Approach to GPU Performance for AI

Use Case(s) Definition

Are you doing training? Inference? What models are you using? Understanding the workloads that matter most will help focus on the relevant optimization areas. This is particularly important in the context of complex, multimodal workflows.

Model Quantization

The size of inference models (and therefore their efficiency) may be improved through quantization, or pruning (for instance).

Parallelism

Finding the right balance between data, pipeline, and tensor parallelism may yield the largest efficiency gains.

Data Management

Ensuring data locality, and establishing a steady data flow is particularly critical for certain types of AI workloads (e.g. LLMs training).

Intra-GPUs Optimization

Choosing the right type of GPUs and their optimized software stack (making use of mixed precision, optimized libraries, etc.) will yield the last extra bits of performance.

Mission Accomplished!

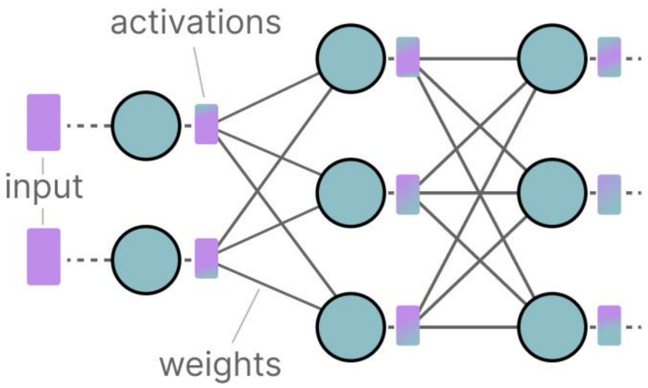
Going through these stages will remove the most obvious pitfalls that can harm GPUs productivity, and help you get the best bang for your bucks!

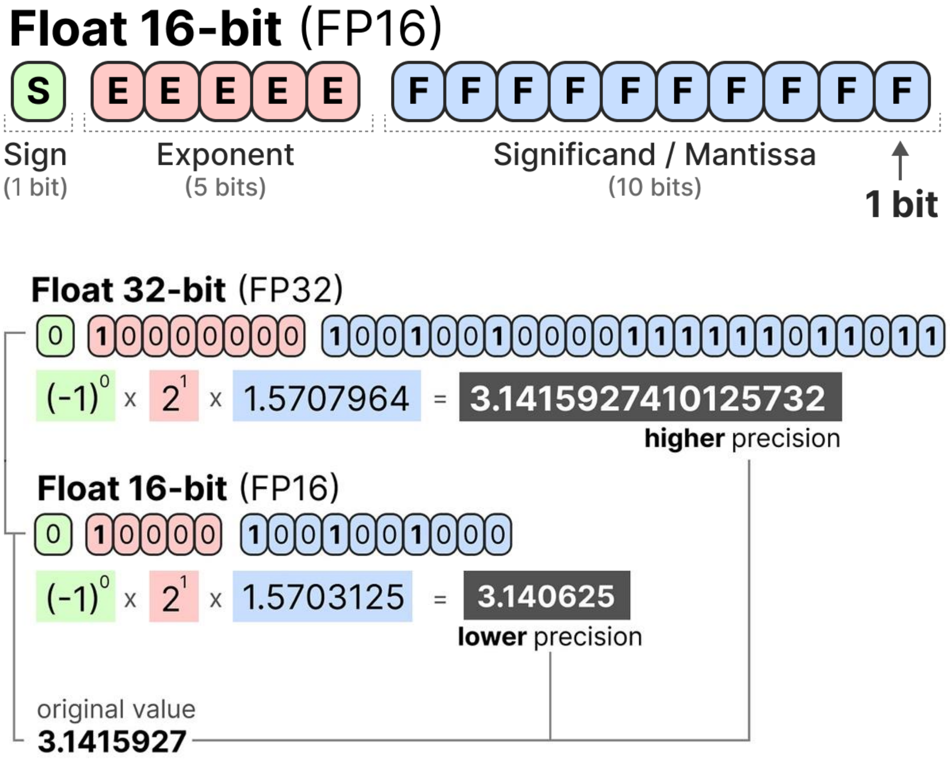
Source: Patrick Wohlschlegel

LLM Precision

Model parameters are mostly *weights*, which can be quite expensive to store.

During inference, activations are created as a product of the input and the weights, which similarly can be quite large.





Source: Exploring Language Models

Computing LLM VRAM from Precision

The memory required for an LLM can be calculated from the precision and number of parameters.

Taking Meta-Llama 3.1 70B as an example, we can calculate the VRAM requirement at different precisions:

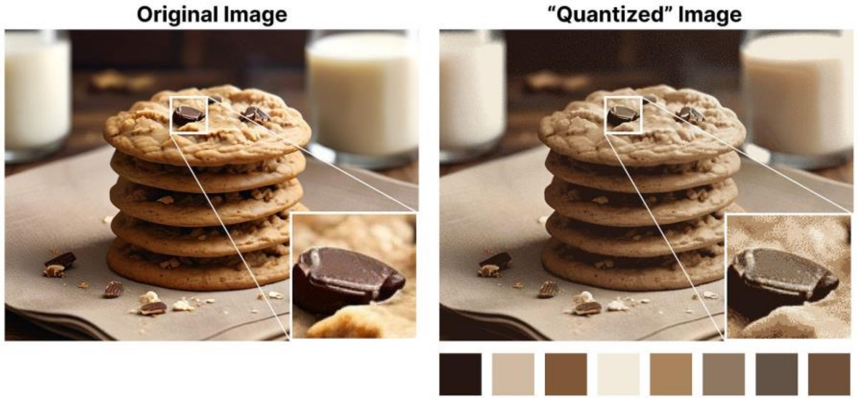
$$\textbf{64-bits} = \frac{64}{8} \times 70\text{B} \approx \textbf{560}\ \text{GB}$$

"Full Precision"

$$\textbf{32-bits} = \frac{32}{8} \times 70\text{B} \approx \textbf{280}\ \text{GB}$$

$$\textbf{16-bits} = \frac{16}{8} \times 70\text{B} \approx \textbf{140}\ \text{GB}$$

$$\text{memory} = \frac{\text{nr\_bits}}{8} \times \text{nr\_params}$$

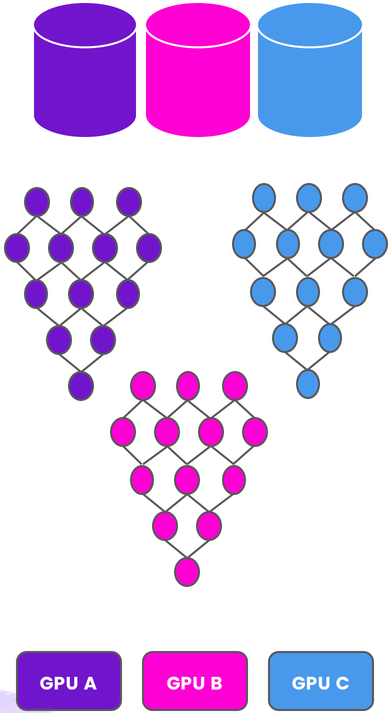


NOTE: *In practice, more things relate to the amount of (V)RAM you need during inference, like the context size and architecture.*

Source: [Exploring Language Models]

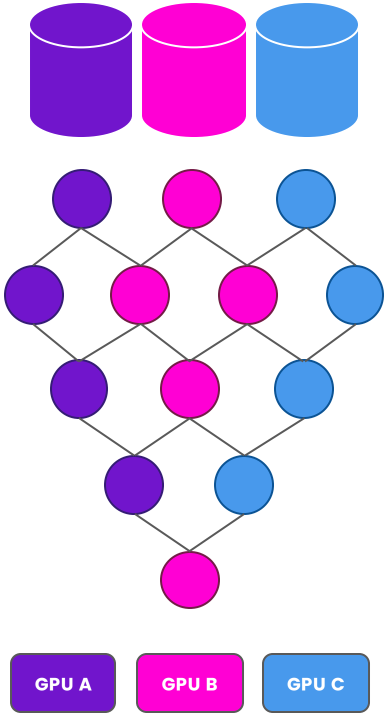
LLM Parallelisation Strategies

Data Parallelism



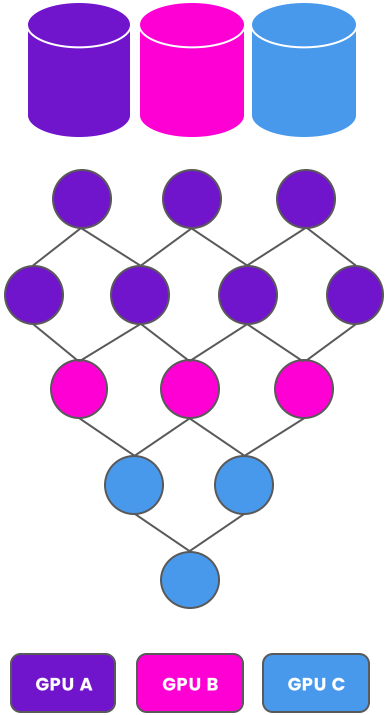
100% Utilization | 100% Utilization | 100% Utilization

Model & Tensor Parallelism



80% Utilization | 100% Utilization | 80% Utilization

Pipeline Parallelism

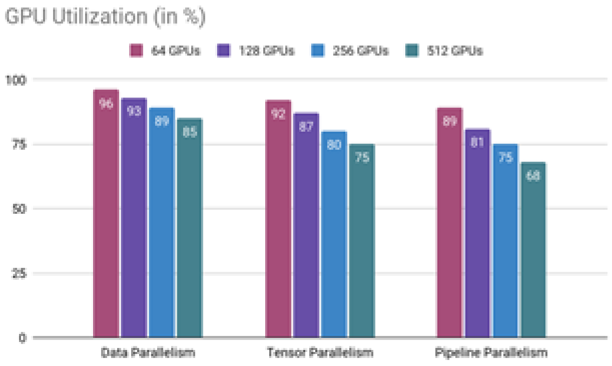


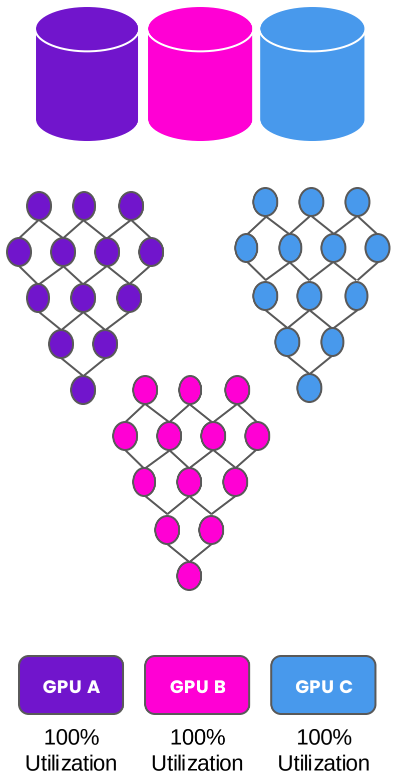
100% Utilization | 42% Utilization | 42% Utilization

Source: Patrick Wohlschlegel

Data Parallelism

- The same model is copied across GPUs, and each processes different batches of data in parallel.
- Data parallelism works well for simple data sets, but performance tends to degrade as data size exceeds GPUs memory capacity.





Source: Patrick Wohlschlegel

Model & Tensor Parallelism

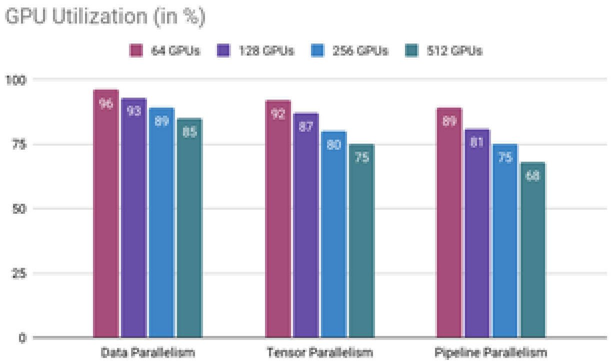
Model Parallelism:

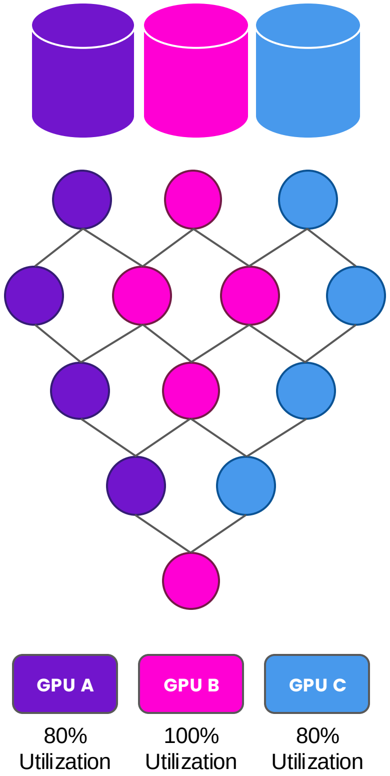
- The model itself is split across multiple GPUs, with each GPU handling different parts of the model.
- Useful for large models that don't fit on a single GPU.

Tensor Parallelism:

- Splits individual tensors (data structures) across GPUs.
- Each GPU processes a portion of the tensor simultaneously.

Model & Tensor parallelism reduces memory requirement per GPU but introduces synchronization penalties.

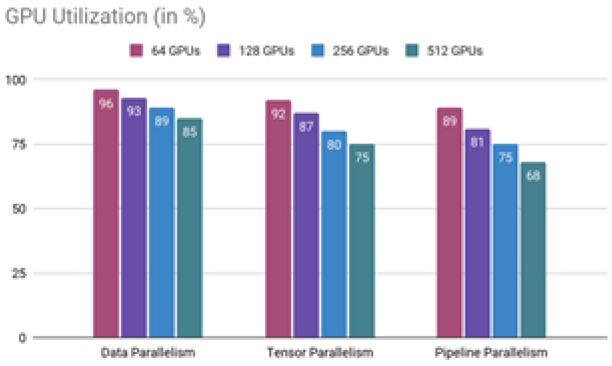


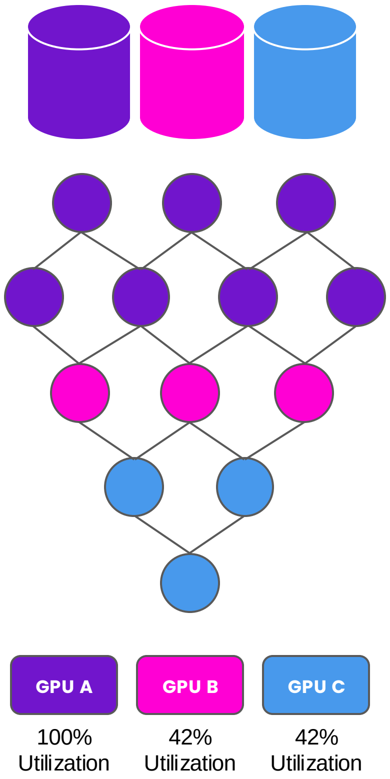


Source: Patrick Wohlschlegel

Pipeline Parallelism

- Model layers are split into stages, with each GPU handling a different stage.
- Inputs move through the pipeline, allowing simultaneous processing across stages.
- Pipeline parallelism can be very effective but may cause low GPU utilization due to data dependencies between layers placed on different GPUs.





Source: Patrick Wohlschlegel

Inferencing Engines Leverage Parallelism

- **TensorRT:** Uses data parallelism and tensor-level parallelism to optimize inference, distributing data across GPUs and reducing redundant computations within tensors.
- **vLLM:** Employs model parallelism and pipeline parallelism, splitting models across GPUs and processing different parts simultaneously, enhancing efficiency for large language models.
- **Grok:** Implements model parallelism and fine-grained tensor parallelism, enabling large-scale models to be split across multiple devices while maintaining high computational efficiency.

Our customers needed help benchmarking self-hosted AI models across different chips.

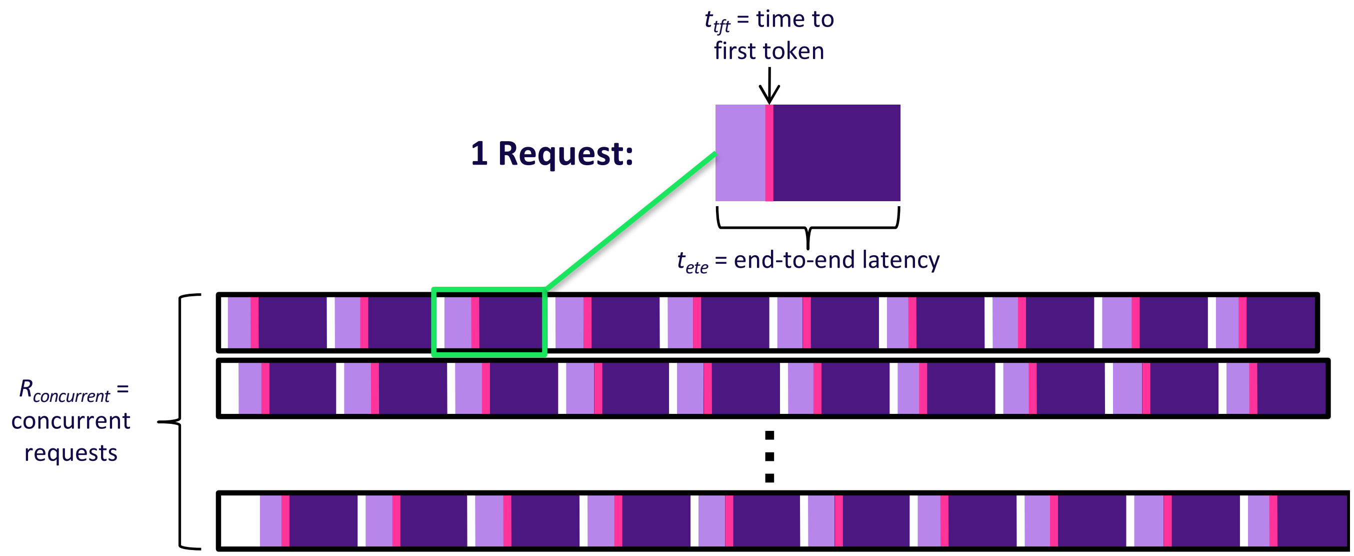
Why leave the selection of the optimal hardware to chance?

There's no simple heuristic, but you will need:

- sufficient **RAM** to hold billions of parameters in memory
- sufficient **bandwidth** to get your prompts and inferences to and from users
- sufficient **compute** to handle parallel requests at scale

BeFOri addresses this gap in the MLOps cycle.

Visualisation of BeFOri Benchmarking



BeFOri Provides 4 Metrics

Time to First Token

Lower is Better

Inter-Token Latency

Lower is Better

End-to-End Latency

Lower is Better

Token Throughput

Higher is Better

Supported Models

Self-hosted LLM models through the Hugging Face transformers library:

- ✓ **Llama Family**
- ✓ **DBRX**
- ✓ **Microsoft Phi**
- ✓ **BERT**
- ❑ Mistral
- ❑ Qwen
- ❑ Many others...

API Provided LLM models:

- OpenAI Compatible APIs
- Anthropic
- TogetherAI
- HuggingFace API
- LiteLLM
- Vertex AI
- SageMaker
- Local Vllm

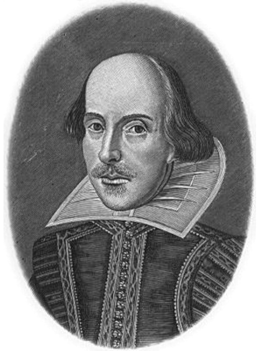
LLM Inference Benchmarking Study

GPU

NVIDIA H100
NVIDIA V100S [2X]

Models

Llama2 7B Chat
Llama3 8B



Prompts

Sample of
Shakespear’s Sonnets

Nvidia H100 Vs. 2 X V100S

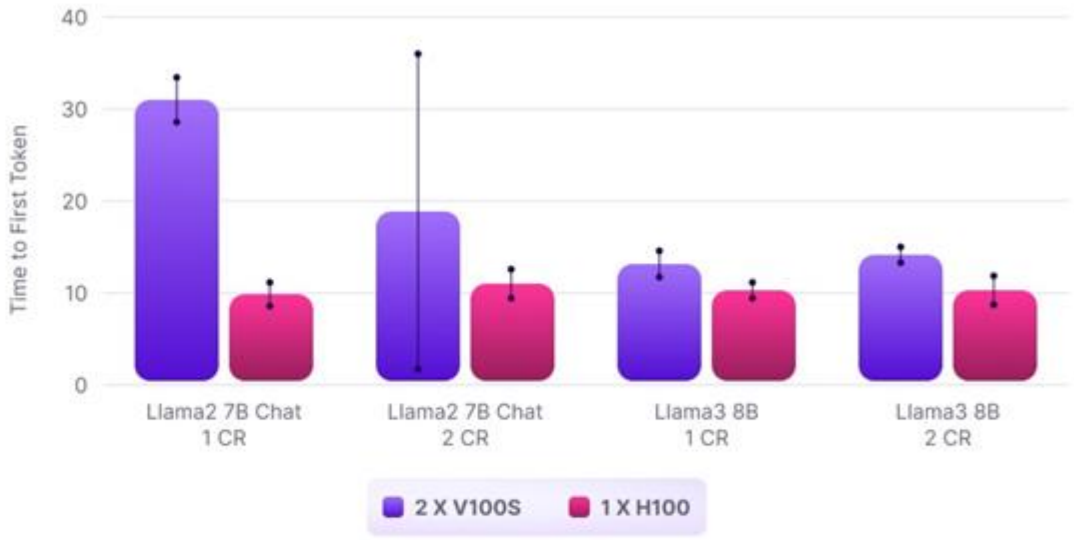
Today you can rent one NVIDIA H100 on Ori Cloud for $3.24/h and two Nvidia V100S for $1.91/h, which will give you the following:

Chip	VRAM / GPU	vCPUs	RAM (GB)	Storage SSD	Storage NVMe	Band-width (GBPs)
1 X H100	64	30	90	500		4
2 X V100S	80	30	380	50	3840	8

Nvidia H100 Vs. 2 X V100S

Time to First Token

Lower is Better

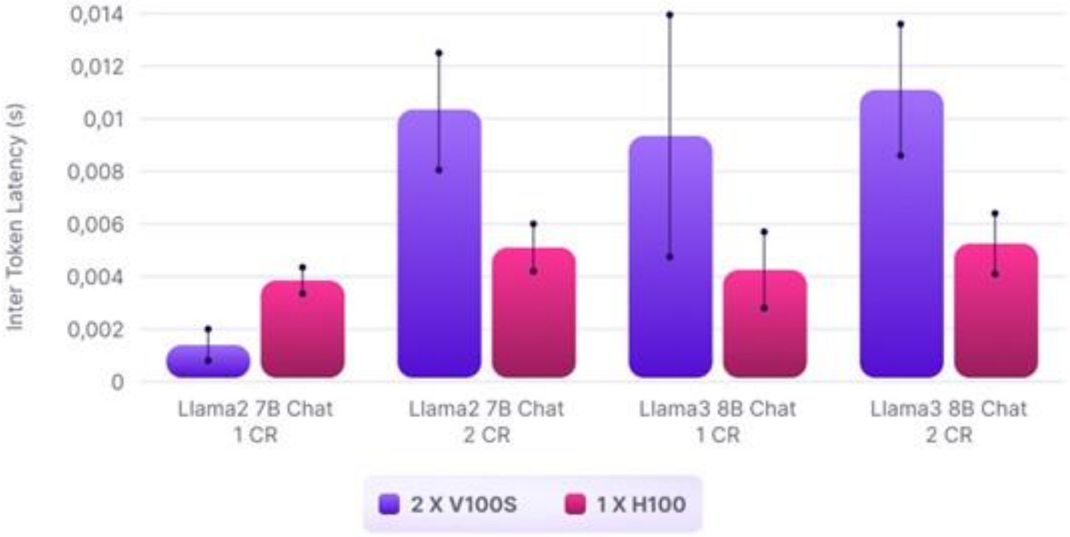


For all configurations, the H100 chip decreased TTFT by an average of **40.9%.**

Nvidia H100 Vs. 2 X V100S

Inter-Token Latency

Lower is Better

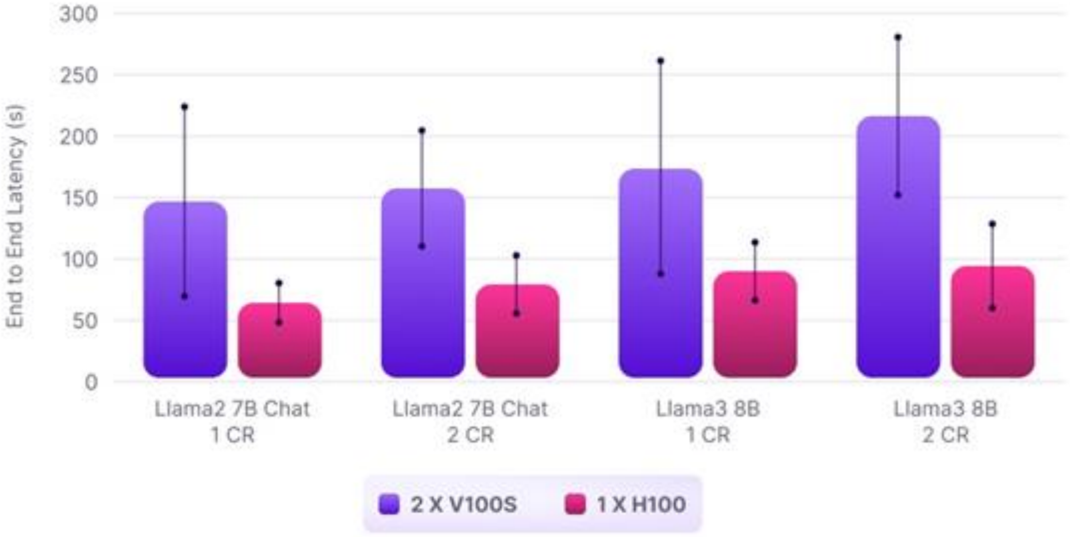


With the exception of Llama2 7B Chat with 1 concurrent request, the H100 chip provided an average of **52.0%** decrease in ITL over 2 X V100S.

Nvidia H100 Vs. 2 X V100S

End-to-End Latency

Lower is Better



For all configurations, the H100 chip decreased ETEL by an average of **53.7%**.

Nvidia H100 Vs. 2 X V100S

Token Throughput

Higher is Better

With the exception of Llama2 7B Chat with one concurrent request, the H100 chip increased token throughput by an average of **.83** tokens per second.

Llama2 Vs. Llama3

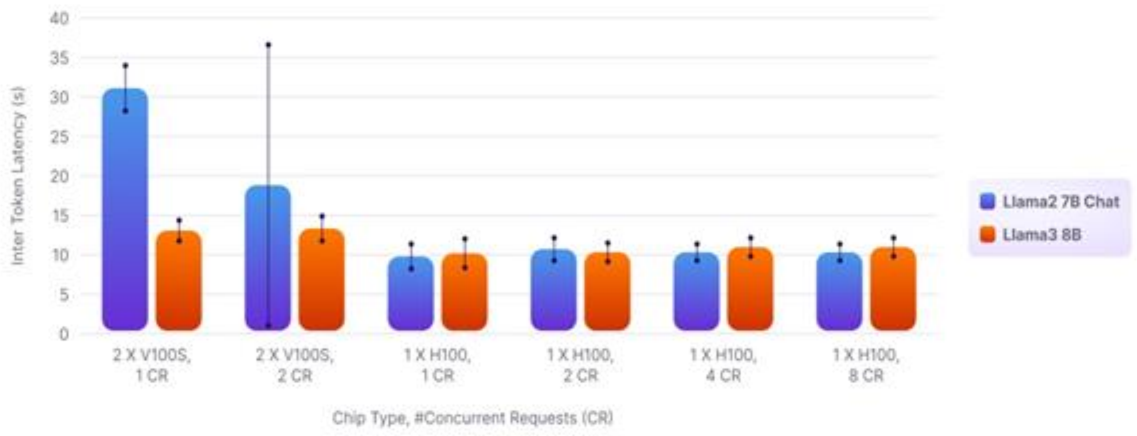
Llama3 was released on 18 April, 2024 about **9 months** after Llama2. We know Meta has the compute equivalent of **600,000 NVIDIA H100 GPUs**. So, it’s safe to say expectations were high!

Other model performance benchmarks have shown a strong improvement, however **we found Llama3 did not perform as quickly as Llama2 using BeFOri.**

Meta’s Llama2 Vs. Llama3

Time to First Token

Lower is Better

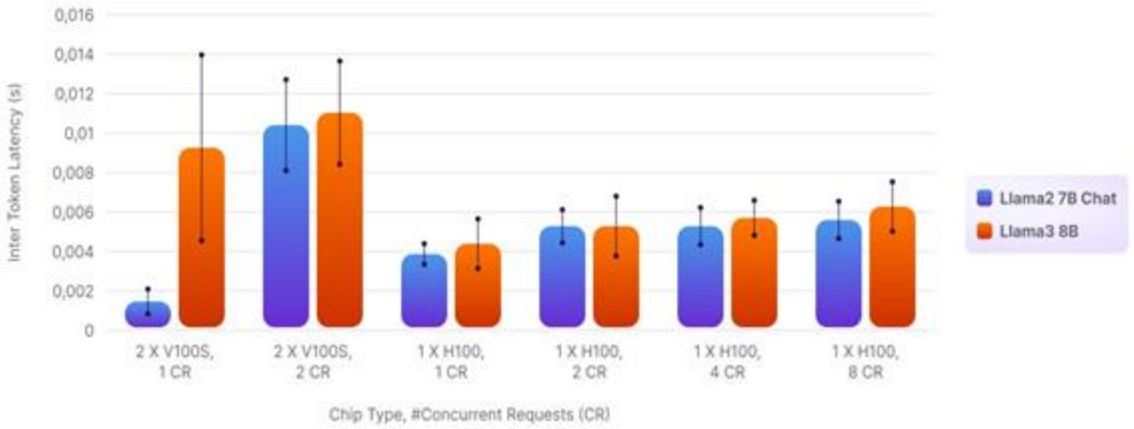


Llama3 8B performed much better than Llama2 7B for TTFT on 2 X V100S, but performance was about the same on the H100 chip.

Meta's Llama2 Vs. Llama3

Inter-Token Latency

Lower is Better

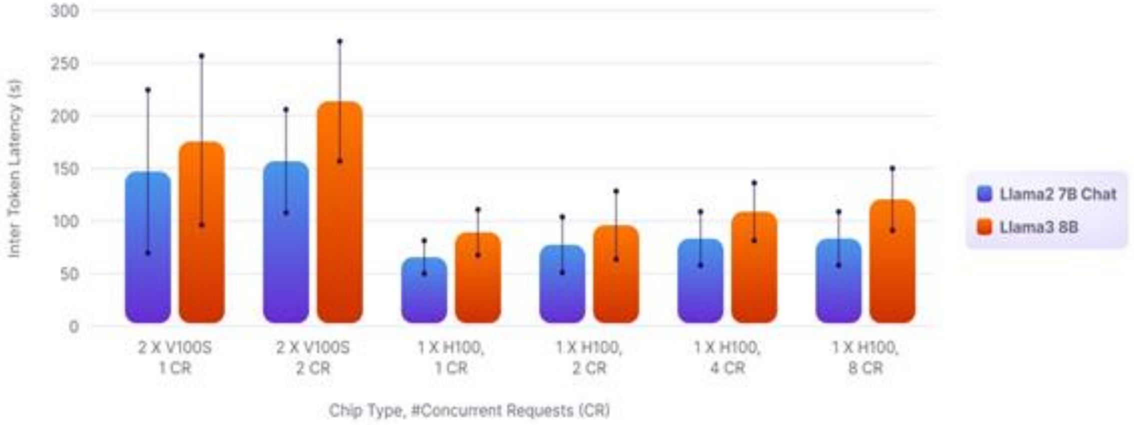


With the exception of one concurrent request on 2 X V100S chips, Llama3 8B was on average **7.3%** slower than Llama2 7B.

Meta's Llama2 Vs. Llama3

End-to-End Latency

Lower is Better

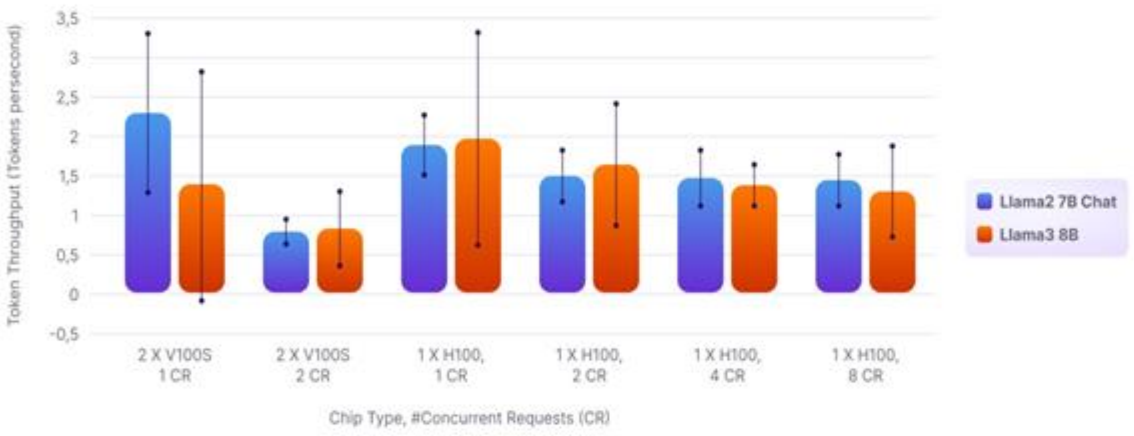


Llama3 8B was slower than Llama2 7B Chat for every configuration we tested, by an average of **31.7%** for ETEL.

Meta’s Llama2 Vs. Llama3

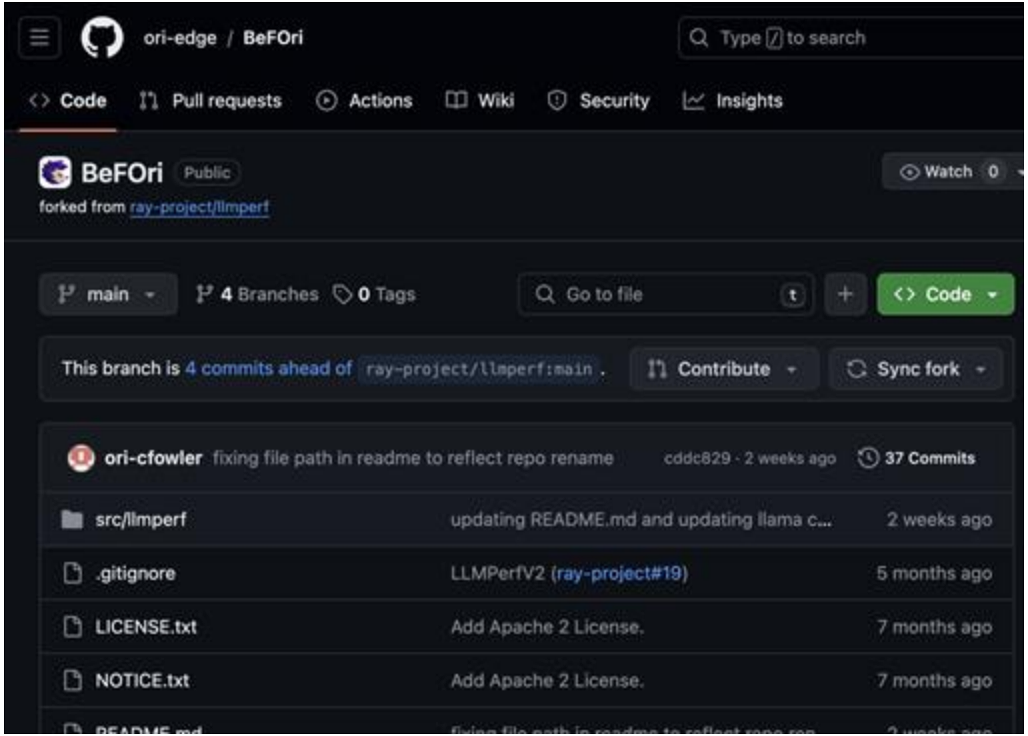
Token Throughput

Higher is Better



The results for TT are mixed with the performance of Llama2 and Llama3 falling with one standard deviation of each other.

Get Started with BeFOri Today!




```
git clone https://github.com/ori-edge/BeFOri.git
cd ./BeFOri
pip install -r requirements.txt
export PYTHONPATH="/PATH/TO/ori-llmperf/src/"
```

Recommended Reading

- **Building LLMs for Production: Enhancing LLM Abilities and Reliability with Prompting, Fine-Tuning, and RAG by Louis-François Bouchard & Louie Peters**
- A Visual Guide to Quantization by Maarten Grootendorst
- Performances are plateauing, let's make the leaderboard steep again
- An LLM agent for assisting machine learning research
  - [Blog] Sakana.ai
  - [arXiv Paper] The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery

Questions?

Thanks!

To get started on Ori Public Cloud, give us a shout…!

ML Engineer Lead
ciera.fowler@ori.co

LBS MBA Class of 2025
clowe.mba2025@london.edu