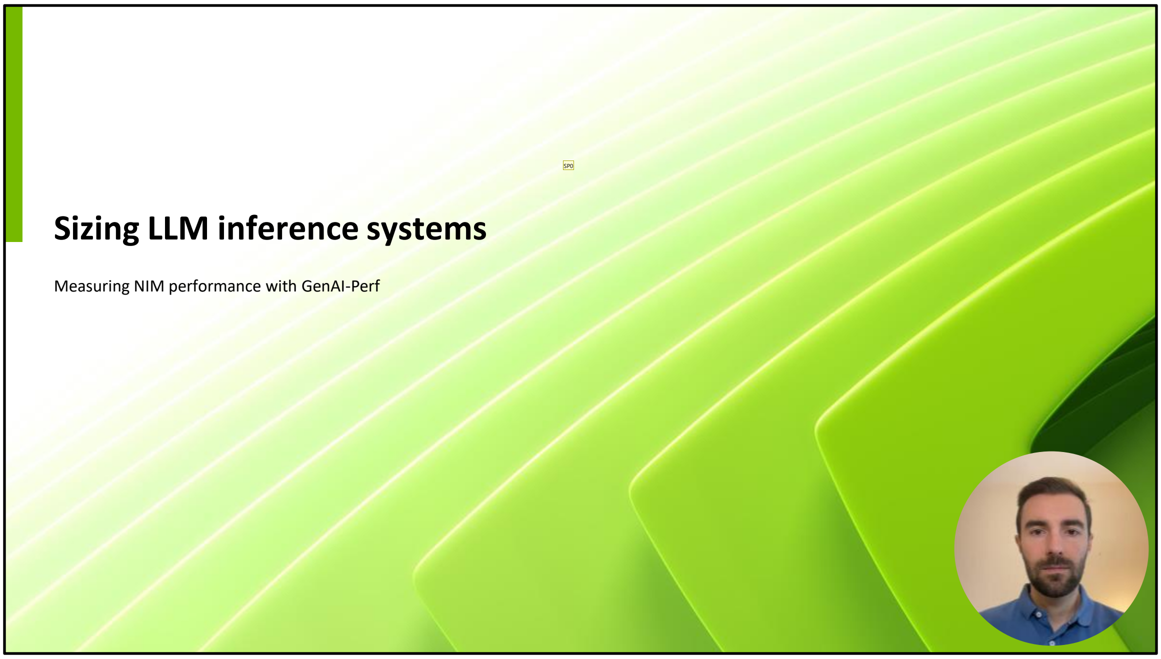


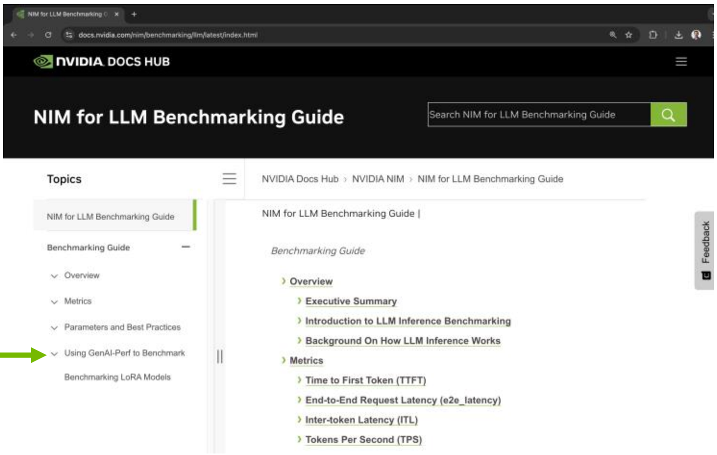
Sizing LLM inference systems

Measuring NIM performance with GenAI-Perf

In this notebook, you will learn how to use the GenAI-Perf tool to measure the latency and throughput of various inference workloads. Let’s cover some basics before you start working on the notebook.

NIM for LLM Benchmarking Guide

https://docs.nvidia.com/nim/benchmarking/llm/latest/index.html


NIM for LLM Benchmarking G
docs.nvidia.com/nim/benchmarking/llm/latest/index.html
NVIDIA DOCS HUB
Search NIM for LLM Benchmarking Guide
Topics
NIM for LLM Benchmarking Guide
Benchmarking Guide
Overview
Metrics
Parameters and Best Practices
Using GenAI-Perf to Benchmark
Benchmarking LoRA Models
Feedback

NIM for LLM Benchmarking Guide

NVIDIA Docs Hub › NVIDIA NIM › NIM for LLM Benchmarking Guide

NIM for LLM Benchmarking Guide |

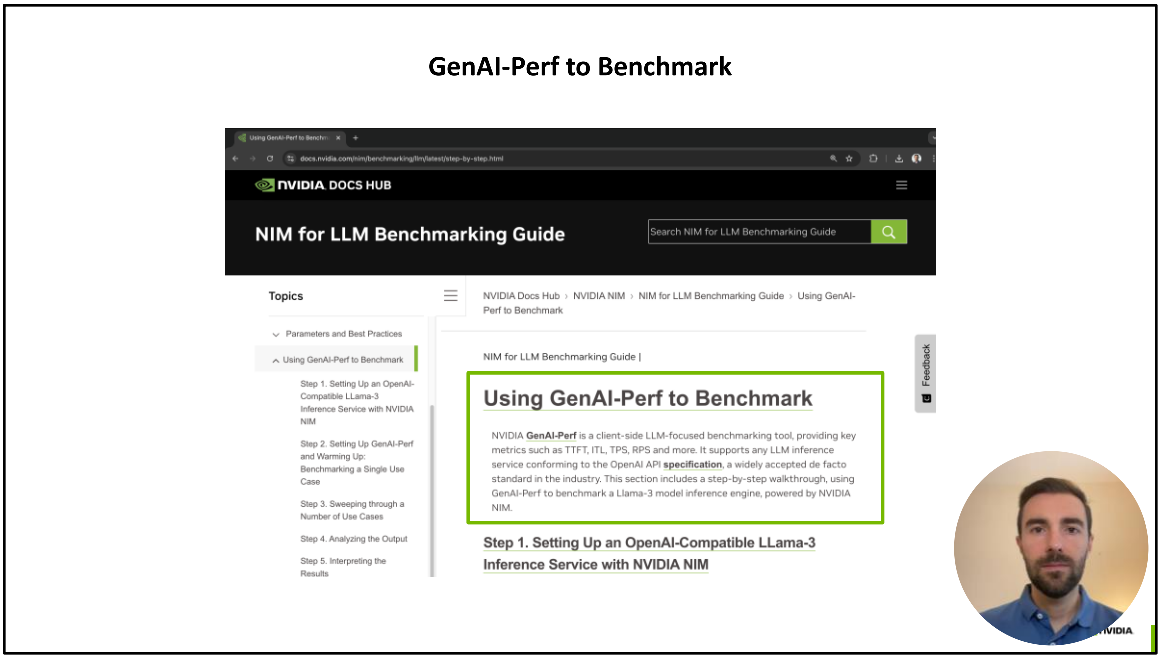
Benchmarking Guide

- Overview
  - Executive Summary
  - Introduction to LLM Inference Benchmarking
  - Background On How LLM Inference Works
- Metrics
  - Time to First Token (TTFT)
  - End-to-End Request Latency (e2e_latency)
  - Inter-token Latency (ITL)
  - Tokens Per Second (TPS)

GenAI-Perf to benchmark LLMs

The GenAI-Perf tool has been developed by the Triton team at NVIDIA, and it’s the recommended tool to measure inference performance no matter the inference endpoint. In our NIM for LLM Benchmarking Guide, we make use of GenAI-Perf to measure the performance of NIM. I recommend you that you check out the guide, it covers many of the concepts that you learned in previous notebooks.

GenAI-Perf to Benchmark


Using GenAI-Perf to Benchm
docs.nvidia.com/nim/benchmarking/llm/latest/step-by-step.html
NVIDIA DOCS HUB
Search NIM for LLM Benchmarking Guide
Feedback
NVIDIA

NIM for LLM Benchmarking Guide

Topics

- Parameters and Best Practices
- Using GenAI-Perf to Benchmark
  - Step 1. Setting Up an OpenAI-Compatible LLama-3 Inference Service with NVIDIA NIM
  - Step 2. Setting Up GenAI-Perf and Warming Up: Benchmarking a Single Use Case
  - Step 3. Sweeping through a Number of Use Cases
  - Step 4. Analyzing the Output
  - Step 5. Interpreting the Results

NVIDIA Docs Hub › NVIDIA NIM › NIM for LLM Benchmarking Guide › Using GenAI-Perf to Benchmark

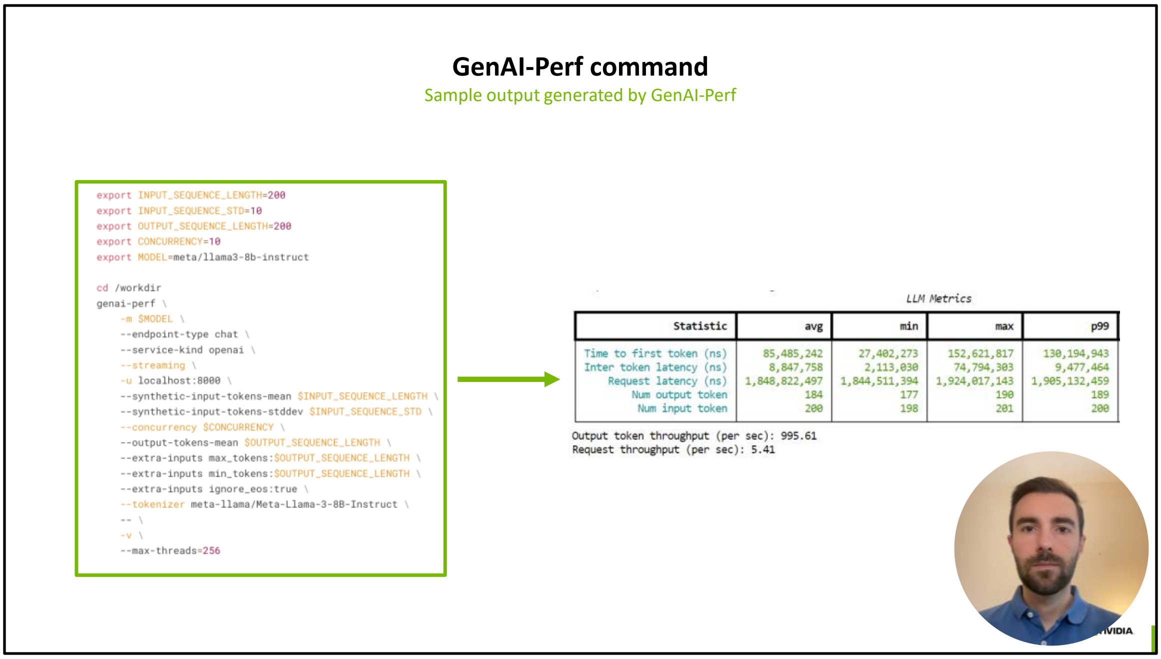
NIM for LLM Benchmarking Guide |

Using GenAI-Perf to Benchmark

NVIDIA **GenAI-Perf** is a client-side LLM-focused benchmarking tool, providing key metrics such as TTFT, ITL, TPS, RPS and more. It supports any LLM inference service conforming to the OpenAI API **specification**, a widely accepted de facto standard in the industry. This section includes a step-by-step walkthrough, using GenAI-Perf to benchmark a Llama-3 model inference engine, powered by NVIDIA NIM.

Step 1. Setting Up an OpenAI-Compatible LLama-3 Inference Service with NVIDIA NIM

The GenAI-Perf tool is a client-side LLM-focused benchmarking took. It provides metrics like time to first token, inter token latency, tokens per second, requests per second and more. You can use it to measure performance for any LLM inference endpoint that satisfies the OpenAI specification. So you can compare the performance of NIMs versus your favourite Generative AI managed services.



GenAI-Perf command

Sample output generated by GenAI-Perf

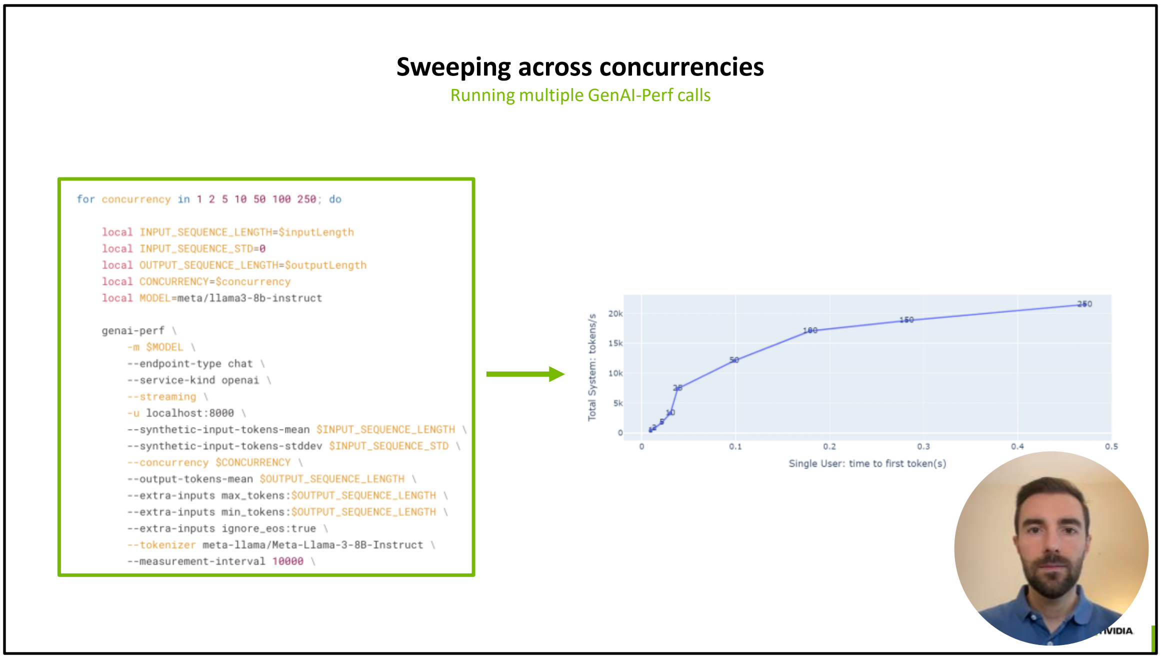
```
export INPUT_SEQUENCE_LENGTH=200
export INPUT_SEQUENCE_STD=10
export OUTPUT_SEQUENCE_LENGTH=200
export CONCURRENCY=10
export MODEL=meta/llama3-8b-instruct

cd /workdir
genai-perf \
    -m $MODEL \
    --endpoint-type chat \
    --service-kind openai \
    --streaming \
    -u localhost:8000 \
    --synthetic-input-tokens-mean $INPUT_SEQUENCE_LENGTH \
    --synthetic-input-tokens-stddev $INPUT_SEQUENCE_STD \
    --concurrency $CONCURRENCY \
    --output-tokens-mean $OUTPUT_SEQUENCE_LENGTH \
    --extra-inputs max_tokens:$OUTPUT_SEQUENCE_LENGTH \
    --extra-inputs min_tokens:$OUTPUT_SEQUENCE_LENGTH \
    --extra-inputs ignore_eos:true \
    --tokenizer meta-llama/Meta-Llama-3-8B-Instruct \
    -- \
    -v \
    --max-threads=256
```

```
                                        LLM Metrics
┏━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃                Statistic ┃           avg ┃           min ┃           max ┃           p99 ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ Time to first token (ns) │    85,485,242 │    27,402,273 │   152,621,817 │   130,194,943 │
│ Inter token latency (ns) │     8,847,758 │     2,113,030 │    74,794,303 │     9,477,464 │
│     Request latency (ns) │ 1,848,822,497 │ 1,844,511,394 │ 1,924,017,143 │ 1,905,132,459 │
│        Num output token  │           184 │           177 │           190 │           189 │
│         Num input token  │           200 │           198 │           201 │           200 │
└──────────────────────────┴───────────────┴───────────────┴───────────────┴───────────────┘
Output token throughput (per sec): 995.61
Request throughput (per sec): 5.41
```

Here you can inspect a typical command of GenAI-Perf. Check for example that I'm passing the number of input and output tokens, in addition to other flags that you can read about in the documentation.

The output of the command is displayed at the right. It contains statistics of important metrics like time to first token, inter token latency and request latency (which is equivalent to end to end latency). It also counts the number of input and output tokens, so that you can compute the throughput from them based on the latencies.



Sweeping across concurrencies

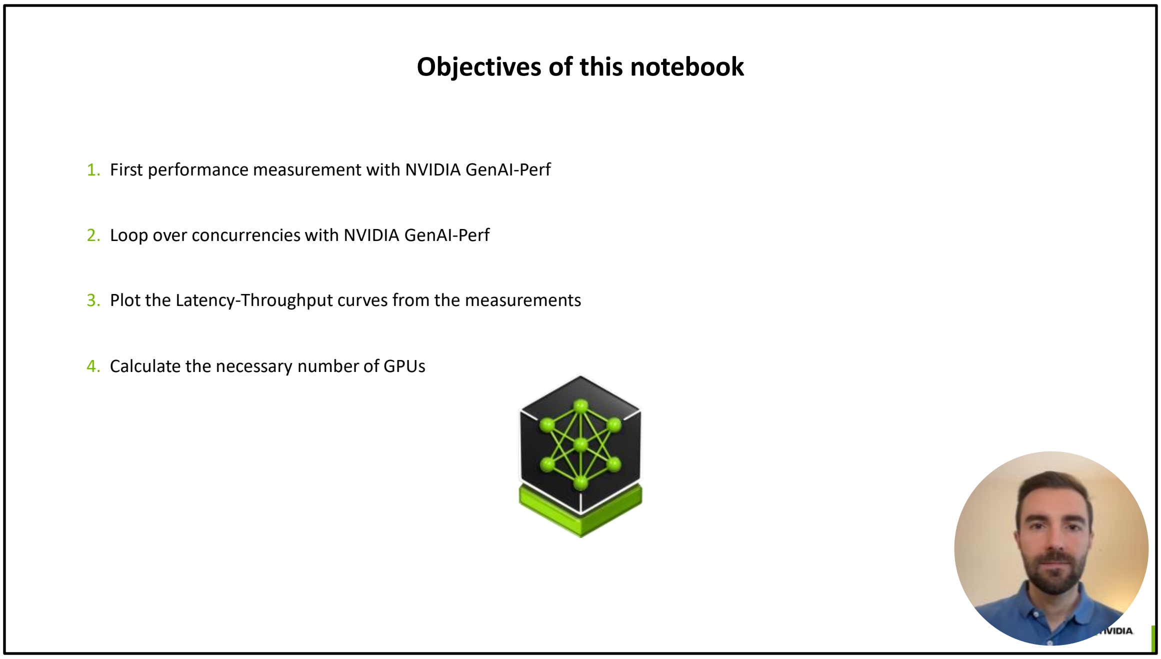
Running multiple GenAI-Perf calls

```
for concurrency in 1 2 5 10 50 100 250; do

    local INPUT_SEQUENCE_LENGTH=$inputLength
    local INPUT_SEQUENCE_STD=0
    local OUTPUT_SEQUENCE_LENGTH=$outputLength
    local CONCURRENCY=$concurrency
    local MODEL=meta/llama3-8b-instruct

    genai-perf \
        -m $MODEL \
        --endpoint-type chat \
        --service-kind openai \
        --streaming \
        -u localhost:8000 \
        --synthetic-input-tokens-mean $INPUT_SEQUENCE_LENGTH \
        --synthetic-input-tokens-stddev $INPUT_SEQUENCE_STD \
        --concurrency $CONCURRENCY \
        --output-tokens-mean $OUTPUT_SEQUENCE_LENGTH \
        --extra-inputs max_tokens:$OUTPUT_SEQUENCE_LENGTH \
        --extra-inputs min_tokens:$OUTPUT_SEQUENCE_LENGTH \
        --extra-inputs ignore_eos:true \
        --tokenizer meta-llama/Meta-Llama-3-8B-Instruct \
        --measurement-interval 10000 \
```

To produce the latency versus throughput plots that you explored in the previous notebook, you can set up a sweep across concurrencies. From the measurements, you can produce the plot that I display at the right. Then you can use all the analysis that you studied in the previous notebook with Dmitry about selecting the most optimal dot in the plot.



Objectives of this notebook

1. First performance measurement with NVIDIA GenAI-Perf
2. Loop over concurrencies with NVIDIA GenAI-Perf
3. Plot the Latency-Throughput curves from the measurements
4. Calculate the necessary number of GPUs

That’s all from my side. Let me conclude by summarizing what you are going to learn in this notebook.

You will start by completing your first performance measurement with NVIDIA GenAI-Perf. Next you will set a sweep of concurrencies to get various measurements of performance.

From those measurements, you will be able to produce plots like the ones showed in the previous notebook.

And finally, you will be able to select the most optimal dot in that plot and compute the number of required GPUs from it.

Now it is your chance to start experimenting with GenAI-Perf!