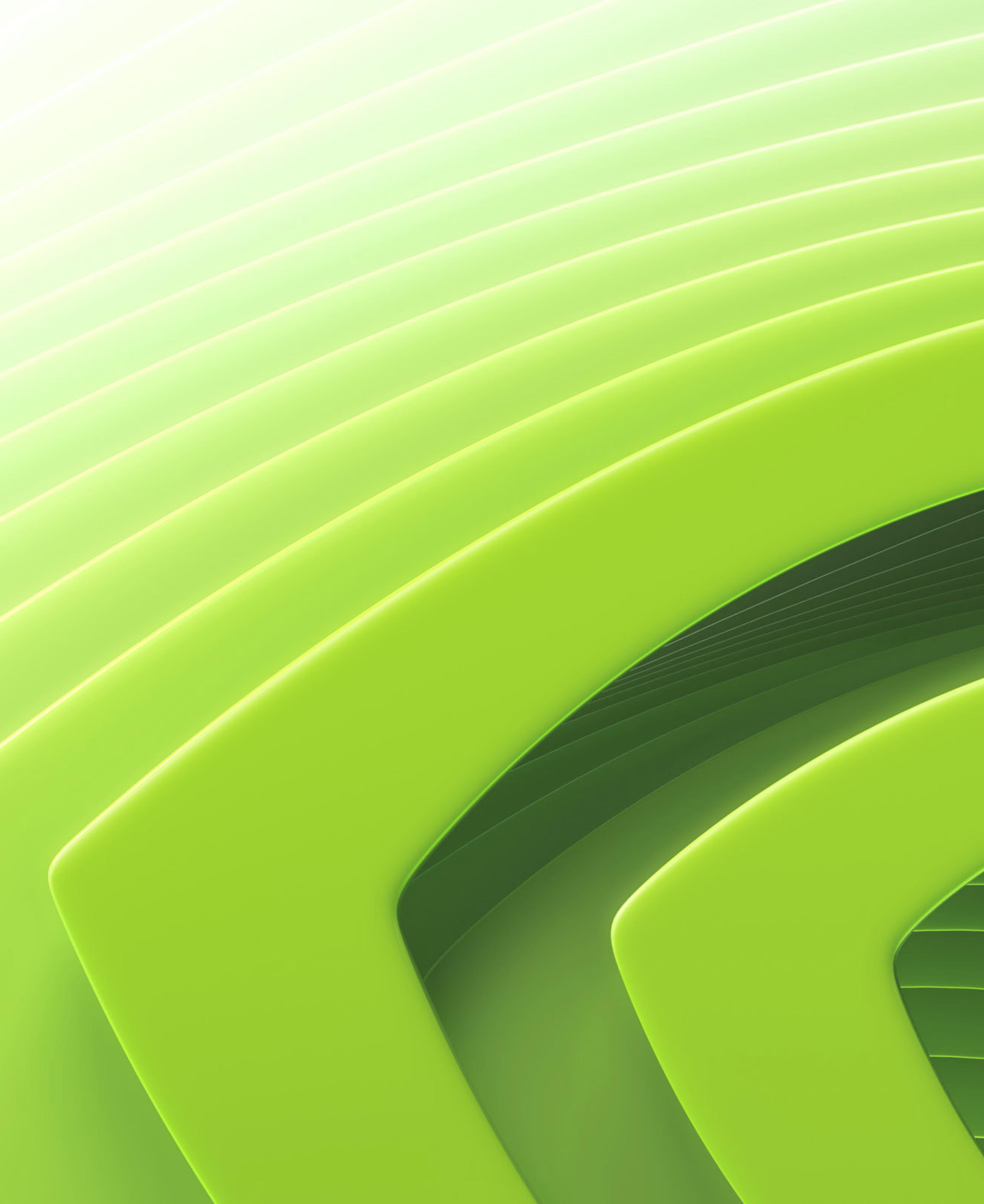




NCHC End-to-end LLM Bootcamp (Day2)

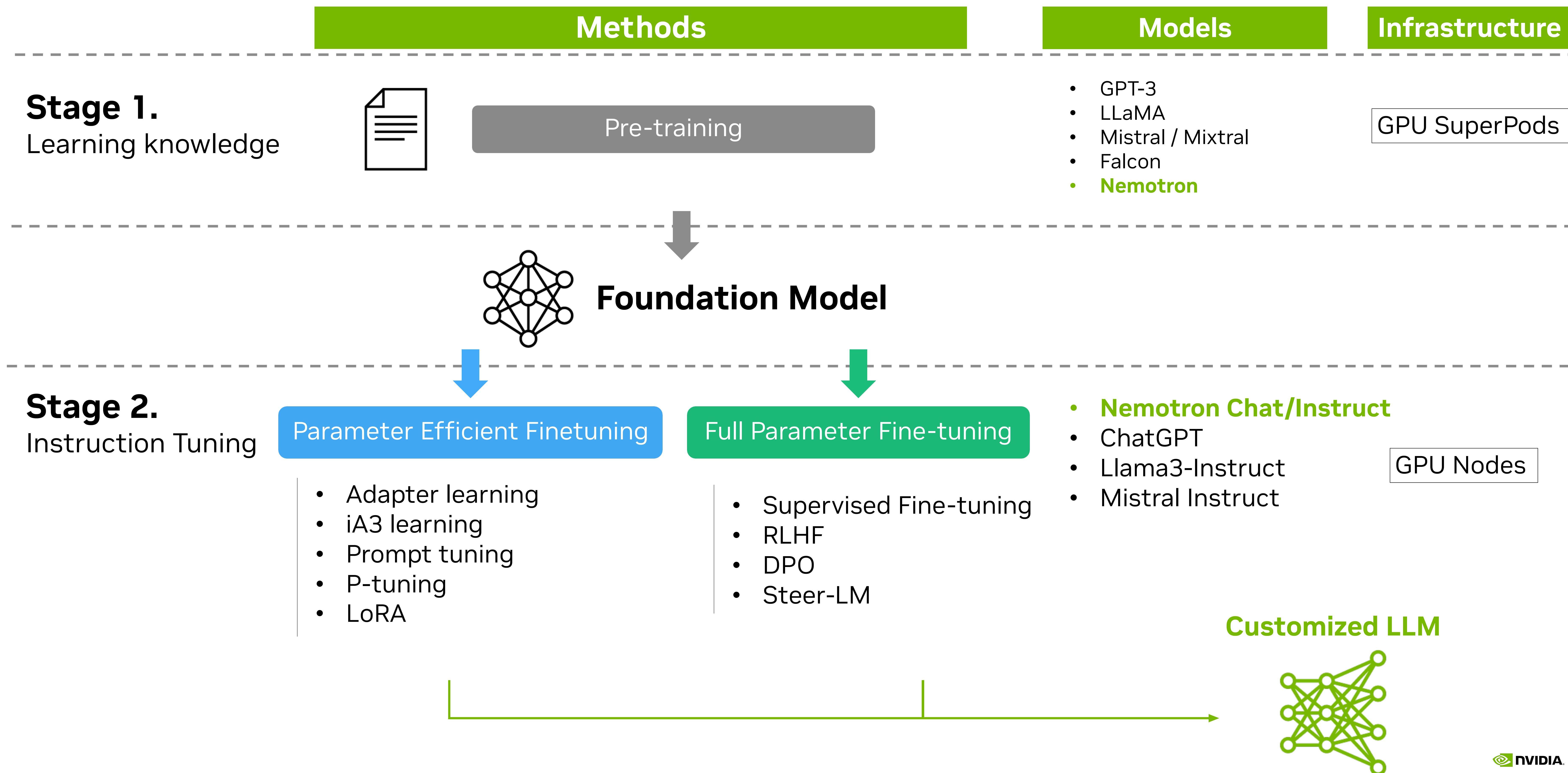
Cliff Chiu, Solution Architect | Aug 07, 2024



Day 2: Wednesday, August 7, 2024 // 09:00 AM - 04:30 PM (In-Person)

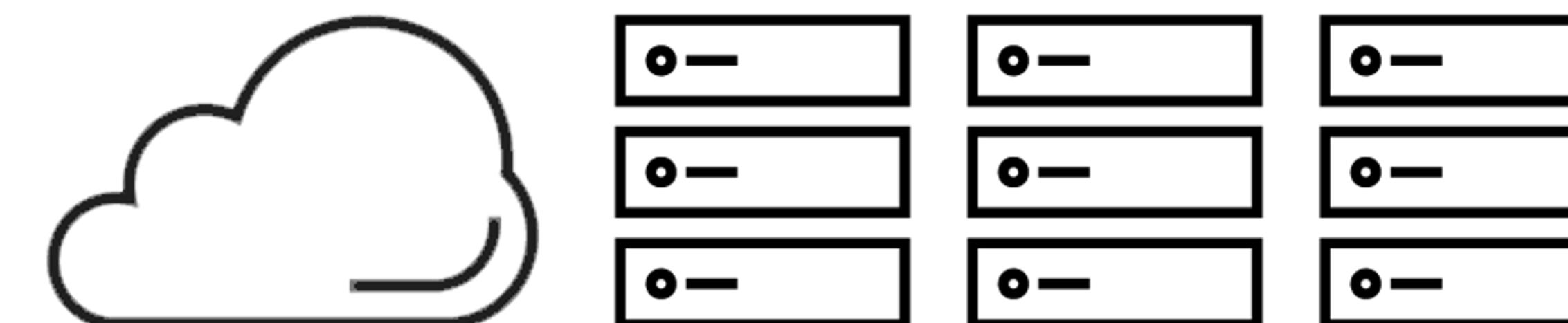
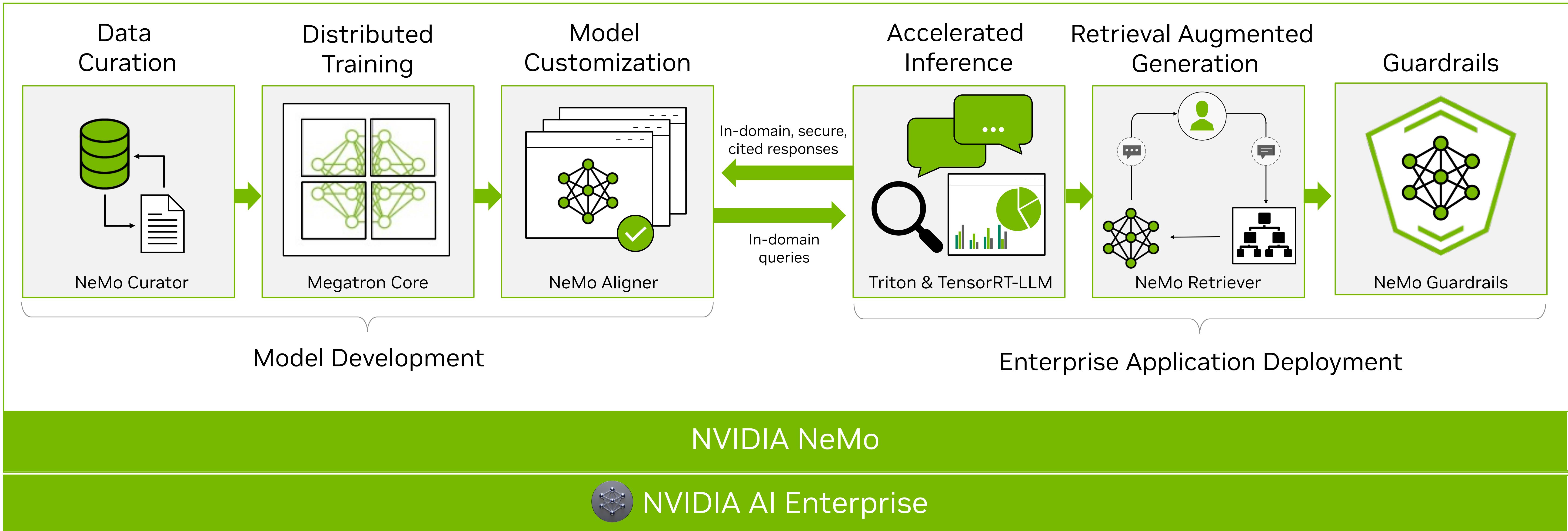
- 09:00 AM - 09:10 AM: Welcome to End to End LLM Bootcamp
- 09:10 AM - 09:30 AM: environment setup and connecting to the cluster
- 09:30 AM – 12:00 PM: Hands-on with NeMo Framework (Lab)
- 12:00 PM – 01:00 PM: Lunch
- 01:00 PM - 02:00 PM: Hands-on with NeMo Framework (Lab)
- 02:00 PM - 02:10 PM: Break
- 02:10 PM - 03:10 PM: Hands-on LLM deployment using TensorRT-LLM (Lab)
- 03:10 PM - 03:20 PM: Break
- 03:20 PM - 04:20 PM: Introduction to NeMo Guardrails (Lecture and Lab)
- 04:20 PM - 04:30 PM: Q&A

Large Language Model Journey

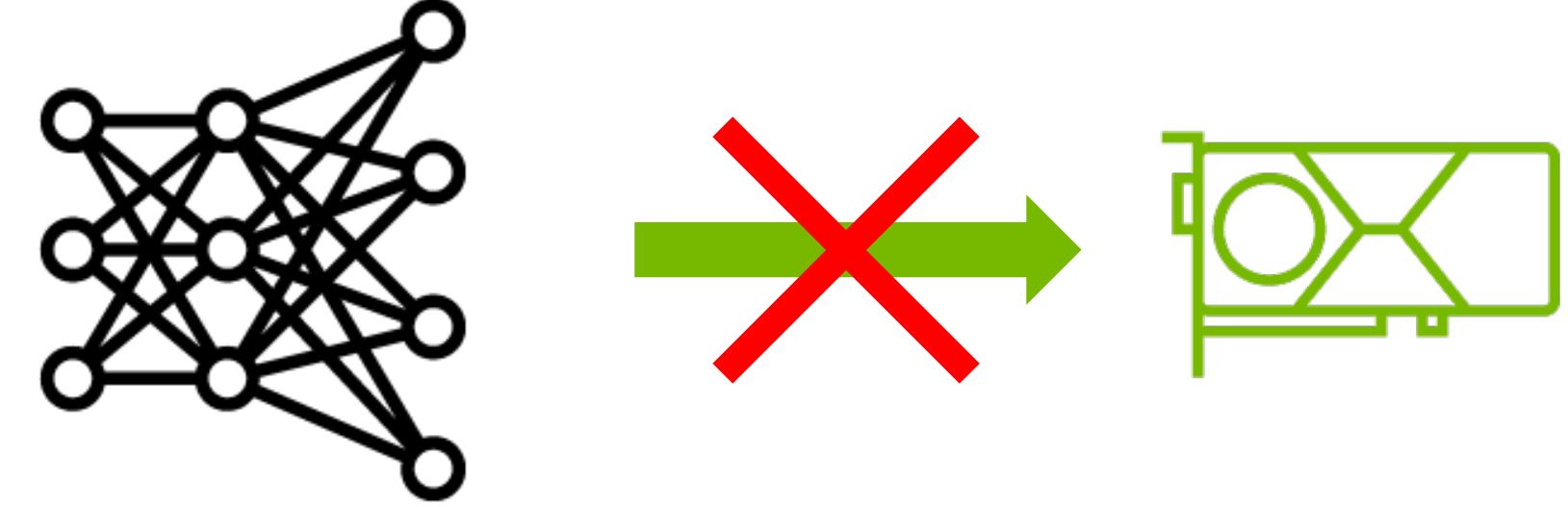


Building Generative AI Applications for the Enterprise

Build, customize and deploy generative AI models with NVIDIA NeMo



LLM

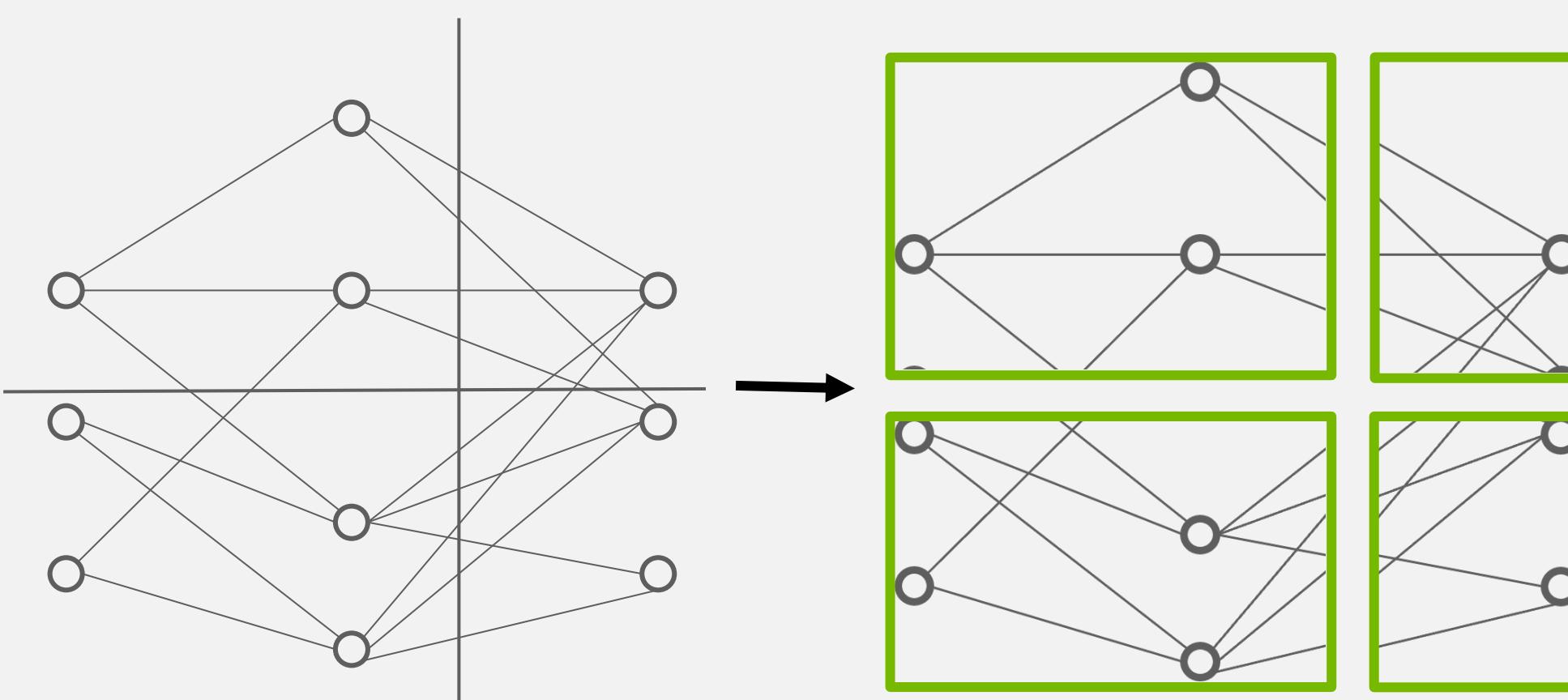


Building Generative AI Foundation Models

Efficiently and quickly training models using NVIDIA NeMo

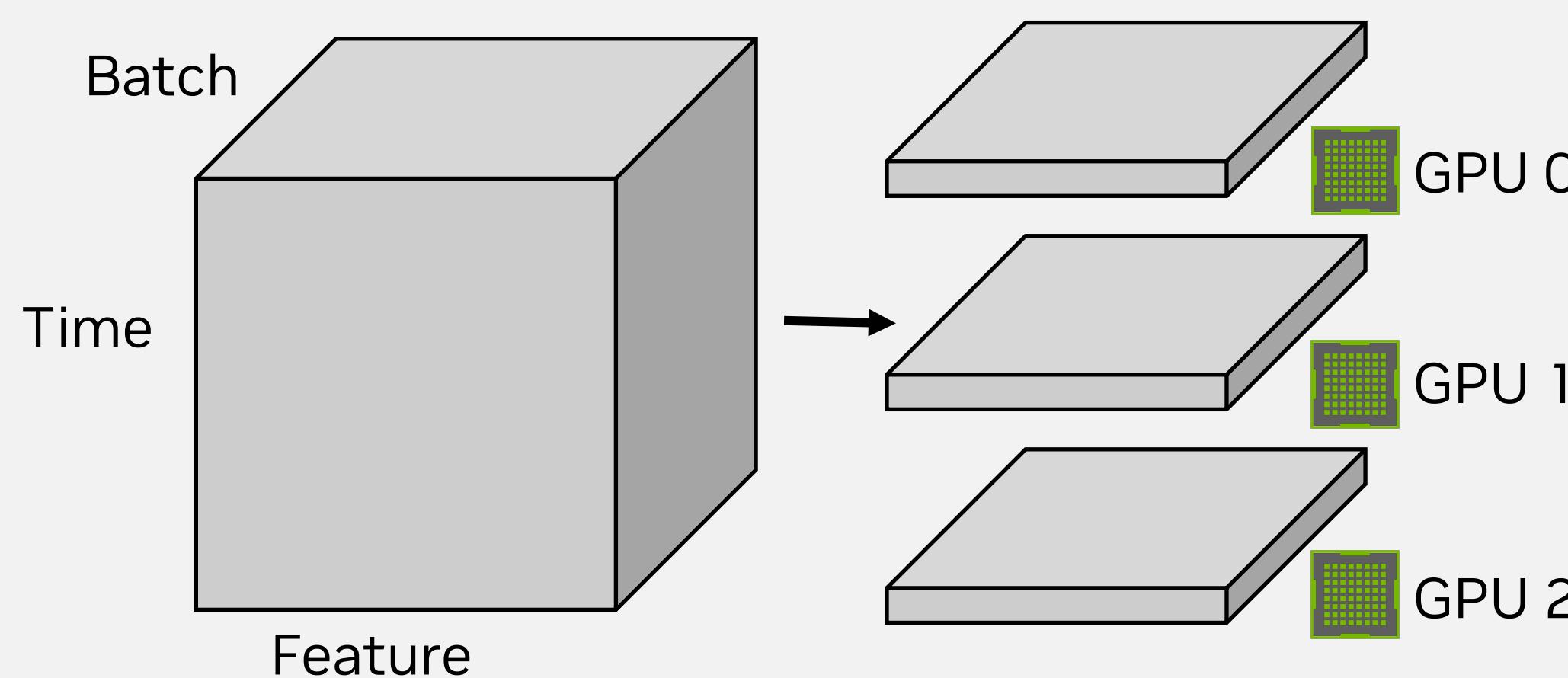
Can't fit into single GPU

Tensor & Pipeline Parallelism



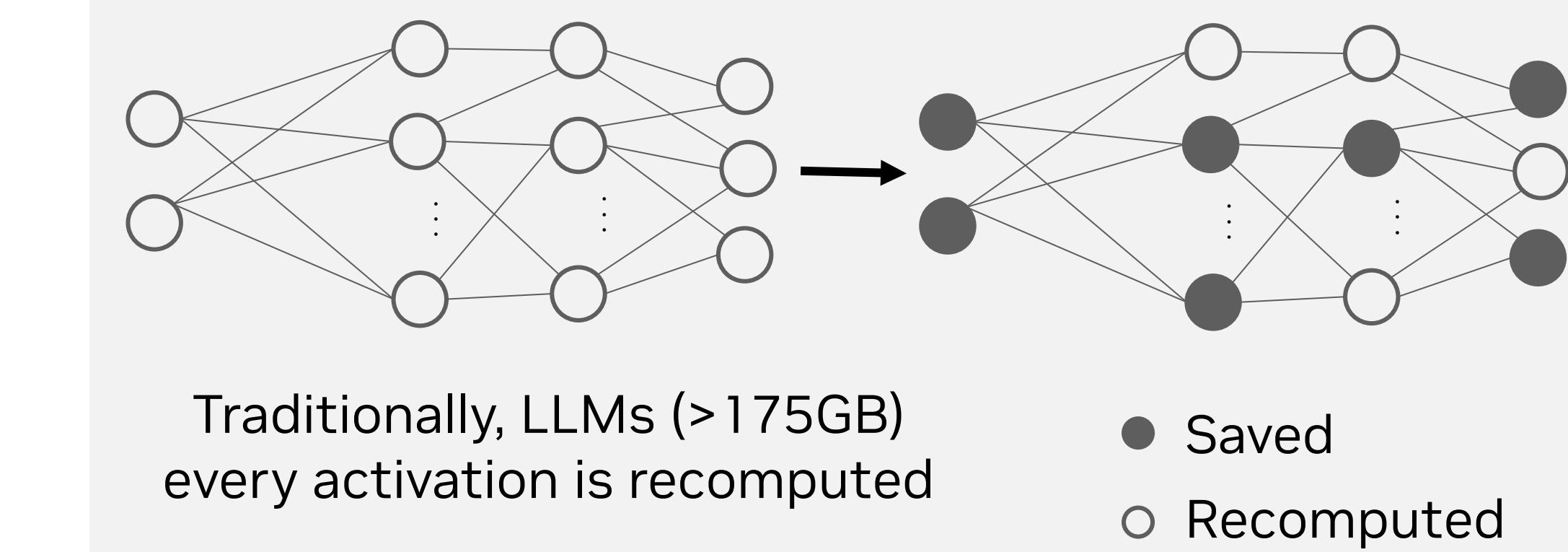
Reduced memory footprint and allows for large-scale training of LLMs across accelerated infrastructure

Sequence Parallelism



Working with tensor processing to increase the batch size that can be support for training

Selective Activation Recomputation

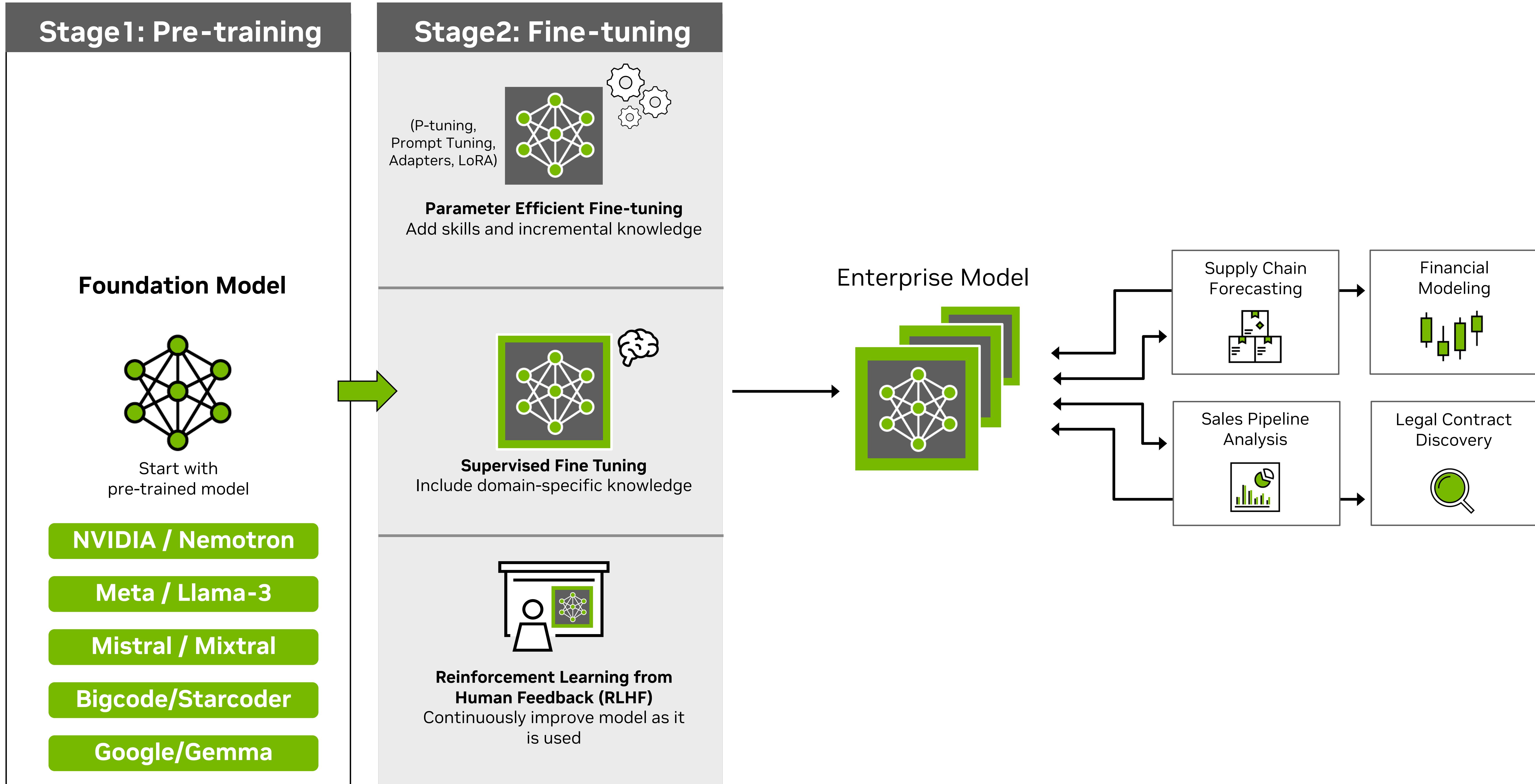


Traditionally, LLMs (>175GB)
every activation is recomputed

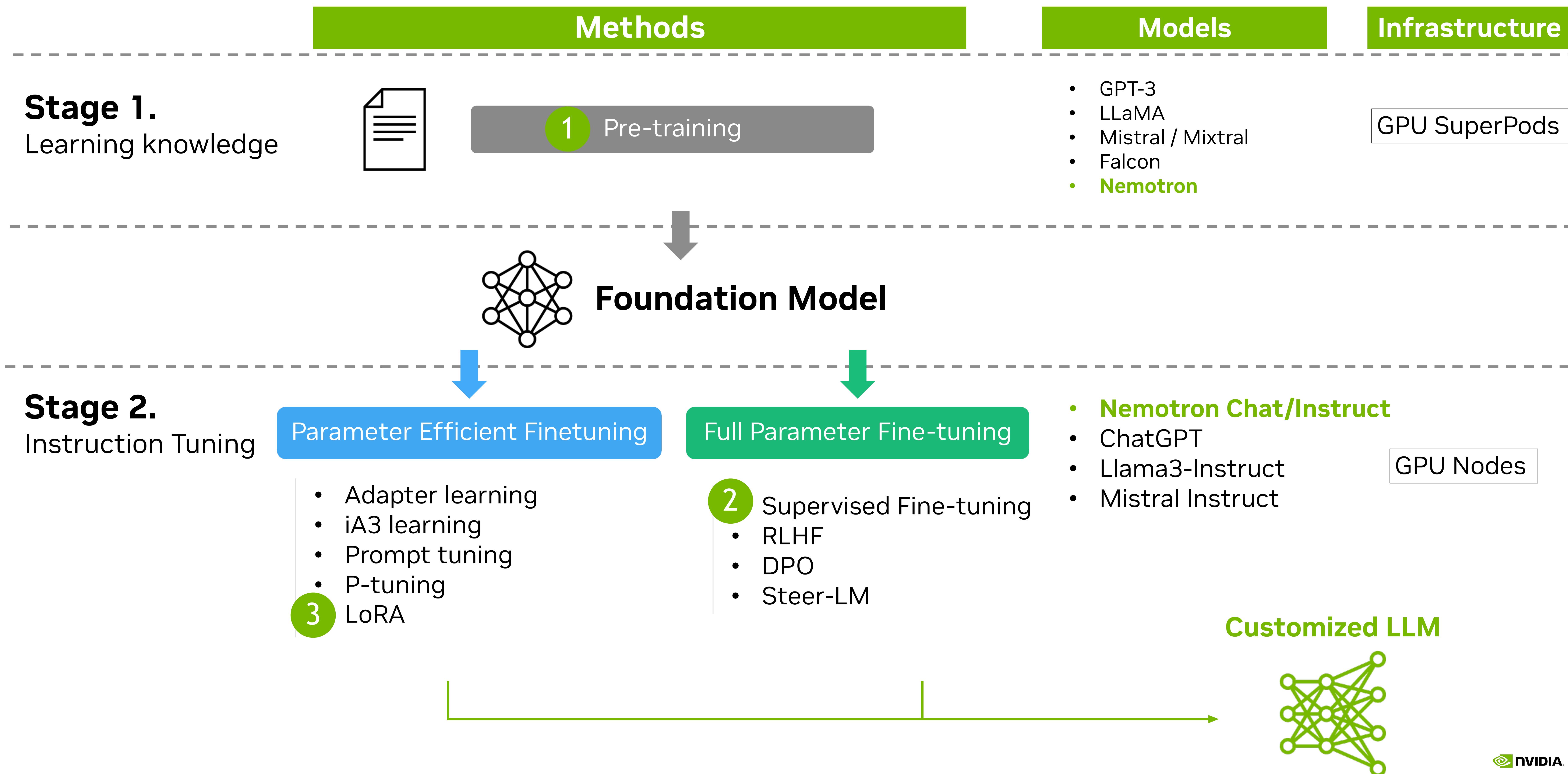
Smart activation checkpointing provides greatest trade-off between memory and recomputation

Model Customization for Enterprise Ready LLMs

Customization techniques to overcome the challenges of using foundation models

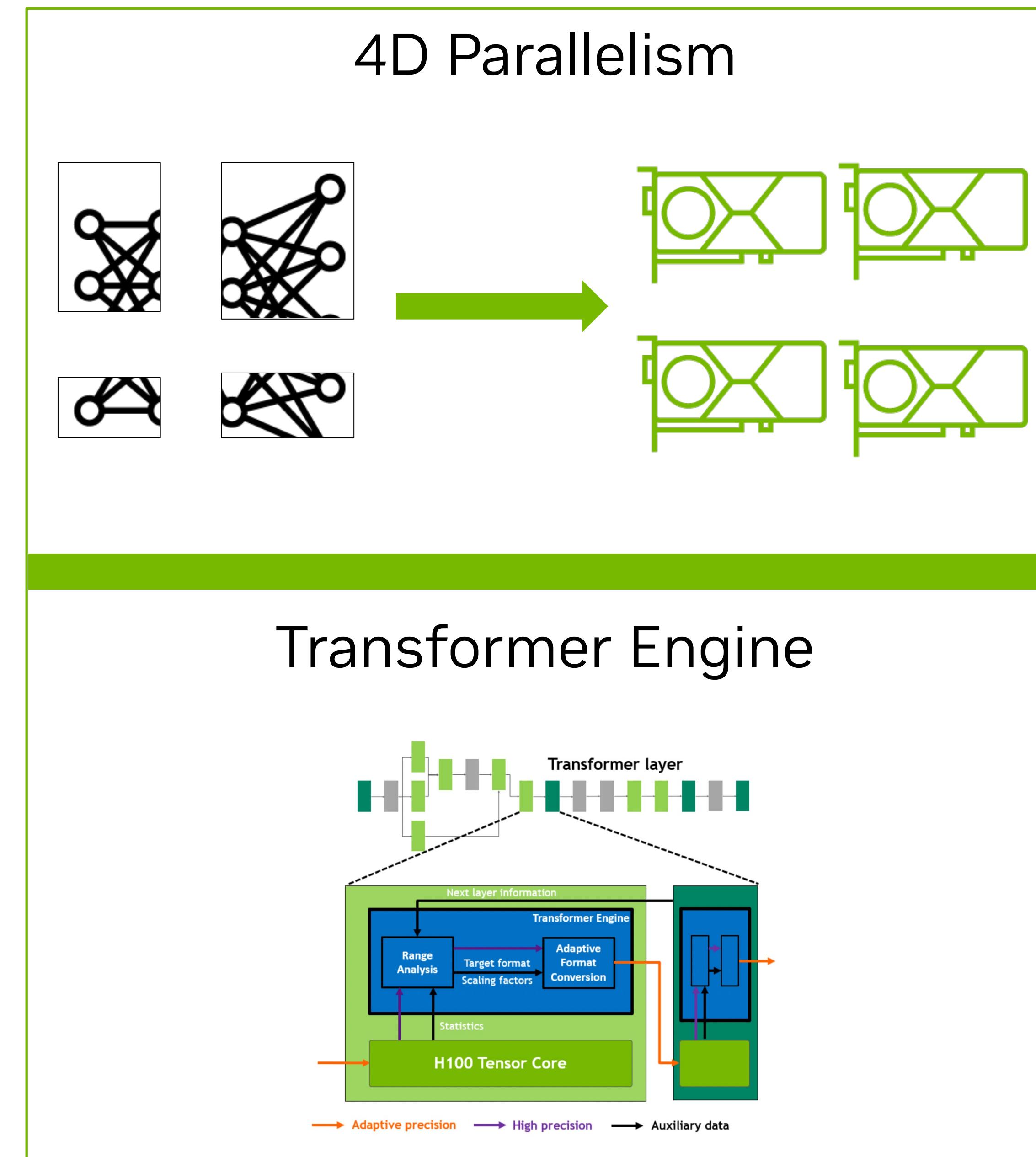
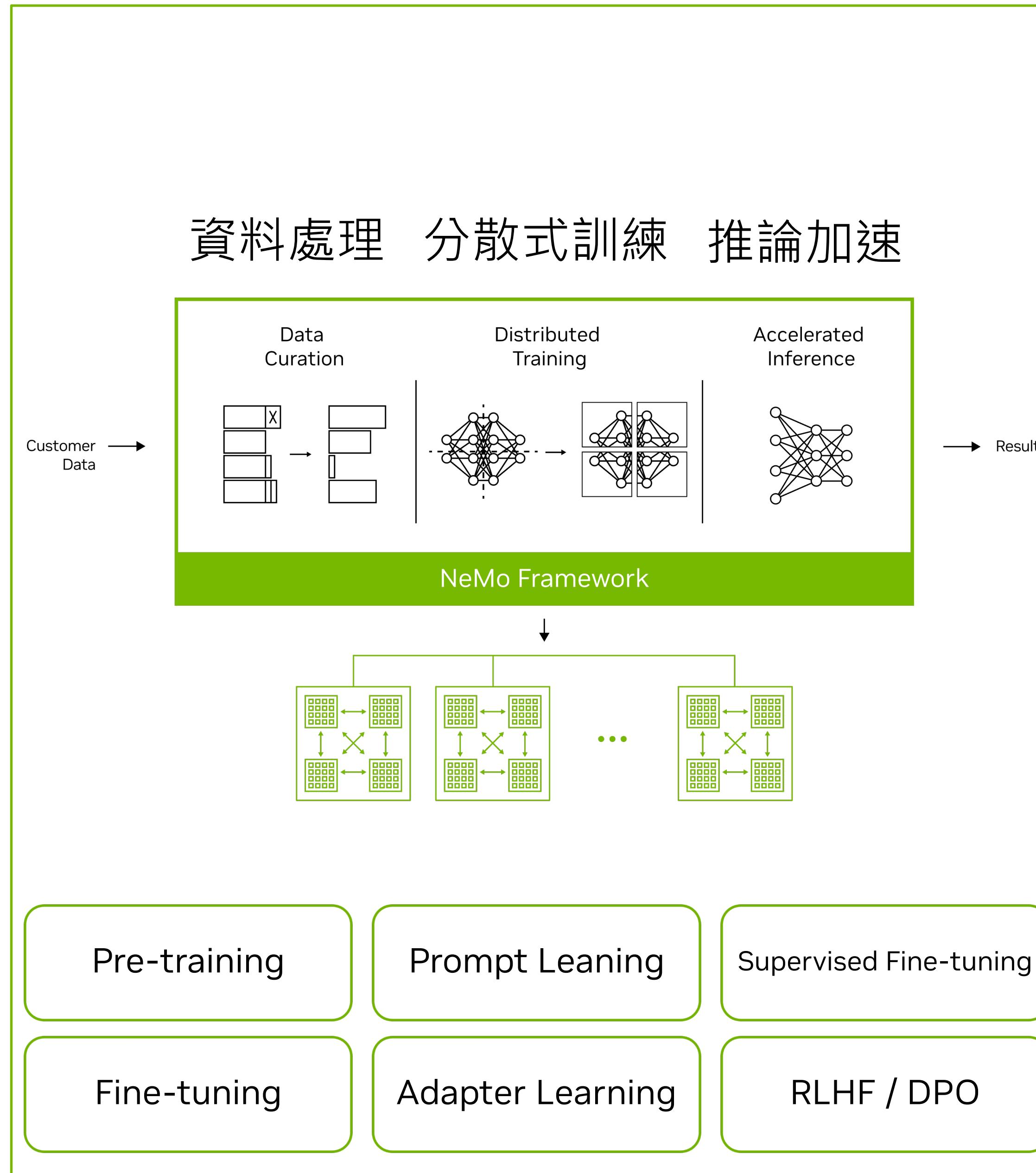


Large Language Model Journey



NeMo Framework

End-to-end, cloud-native framework to build, customize and deploy generative AI models



Step 1.
設定參數檔

Step 2.
啟動程式

Step 1. 設定參數檔

參數設定檔: **llama2_7b.yaml**

trainer

- num_nodes=2
- devices=8
- max_steps=10000

Model

- micro_batch_size=1
- global_batch_size=4
- tensor_model_parallel_size=1
- pipeline_model_parallel_size=1
- optim:
 - lr: 1e-4
- transformer_engine: true

Step 1.
設定參數設定檔

Step 2.
啟動程式

Step 2. 執行訓練腳本

```
/opt/NeMo/examples/nlp/language_modeling/megatron_gpt_pretraining.py
```

啟動方法 1： 設定完參數檔後直接啟動

```
python  
/opt/NeMo/examples/nlp/language_modeling/megatron_gpt_pretraining.py
```

啟動方法 2： 自行指定參數檔位置，並覆寫參數

```
python  
/opt/NeMo/examples/nlp/language_modeling/megatron_gpt_pretraining.py \  
--config-path=conf --config-name=megatron_gpt_config \  
trainer.num_nodes=1 \  
trainer.max_steps=1000 \  
Model.global_batch_size=8 \  
model.optim.lr=0.001 \  
...
```

讀取

參數設定檔: llama2_7b.yaml

trainer

- num_nodes=2
- devices=8
- max_steps=10000

Model

- micro_batch_size=1
- global_batch_size=4
- tensor_model_parallel_size=1
- pipeline_model_parallel_size=1
- optim:
 - lr: 1e-4
- transformer_engine: true

How to set Max Steps

- 資料集：[erhwenkuo/wikinews-zhtw](#)
 - 經過Llama2的Tokenize之後，共有 **7198329** 個 token
 - Total token = **7198329**
- seq_length: **2048** (TinyLlama)
- total_samples = $7198329 \div 2048 \cong 3515$
- global_batch_size = 4
- 1 epoch = $3515 \div 4 \cong 879$ **steps**

How to set Data Prefix

- dataset1 : **2** steps (20%)
 - dataset2 : **8** steps (80%)
- => max_steps: 10 (1 epoch)

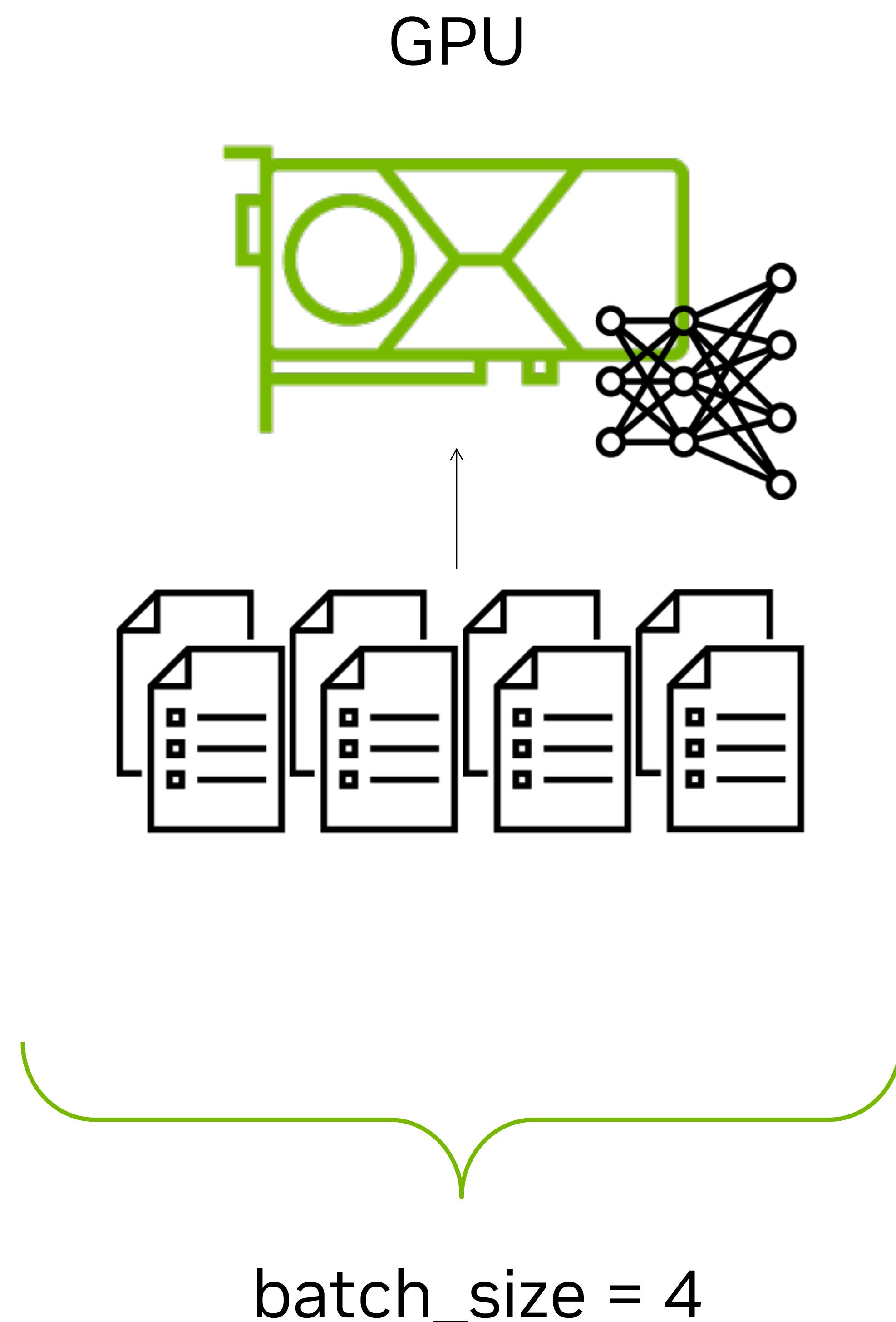
選項1

```
data_prefix:  
  - 0.2  
  - ${data_dir}/dataset1  
  - 0.8  
  - ${data_dir}/dataset2
```

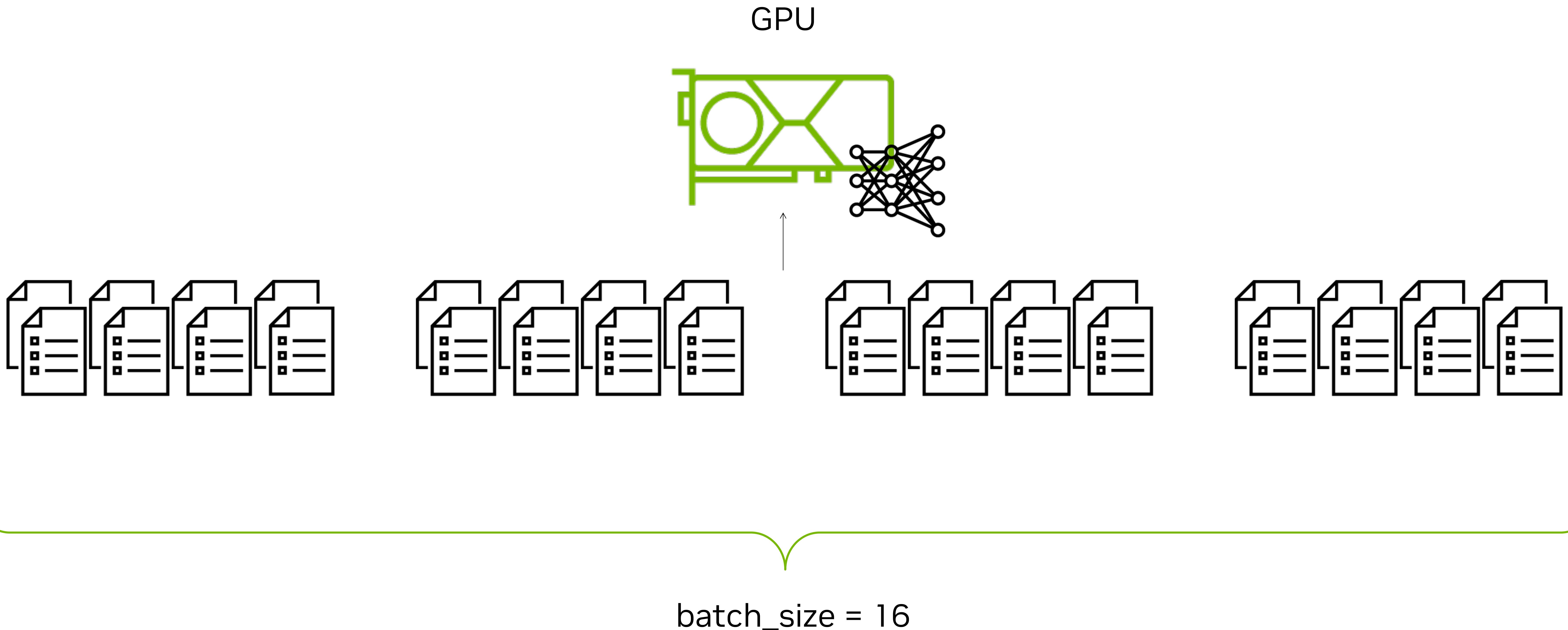
選項2

```
data_prefix:  
  - 2  
  - ${data_dir}/dataset1  
  - 8  
  - ${data_dir}/dataset2
```

Batch size



Batch size



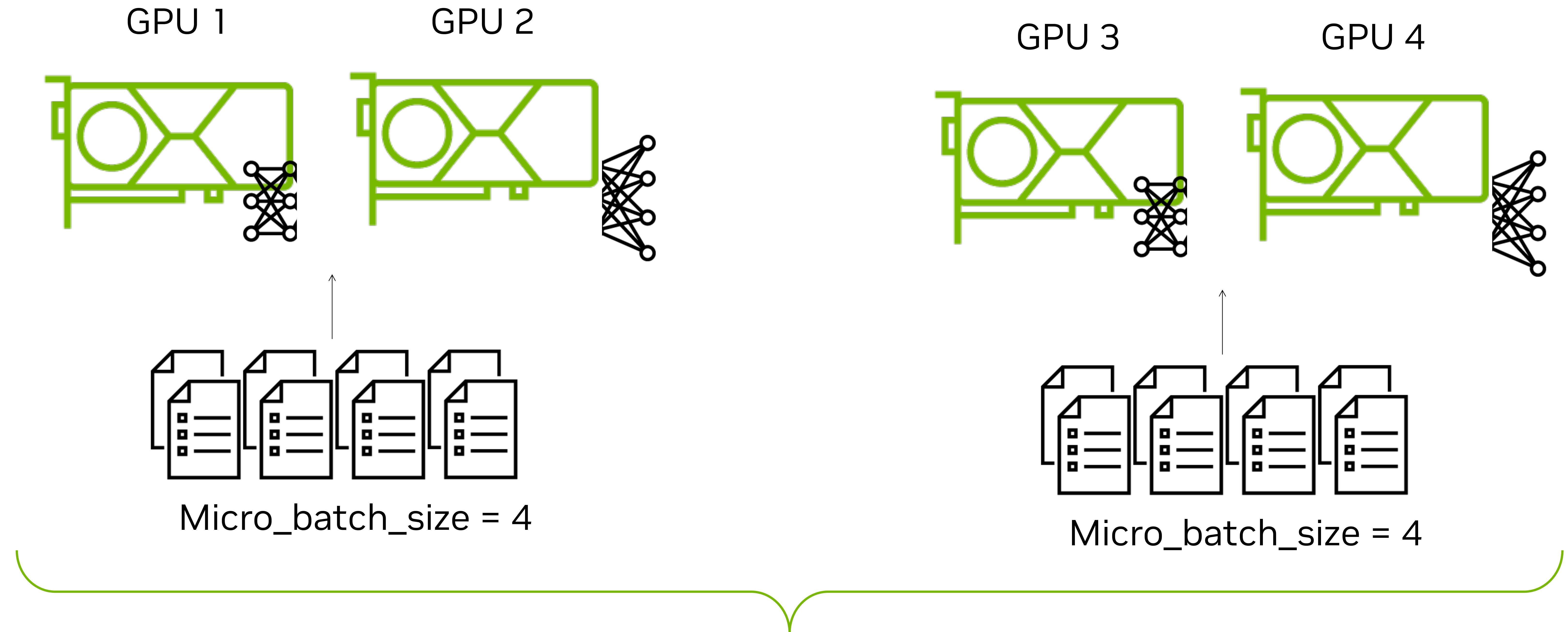
Micro_batch_size vs Global_batch_size



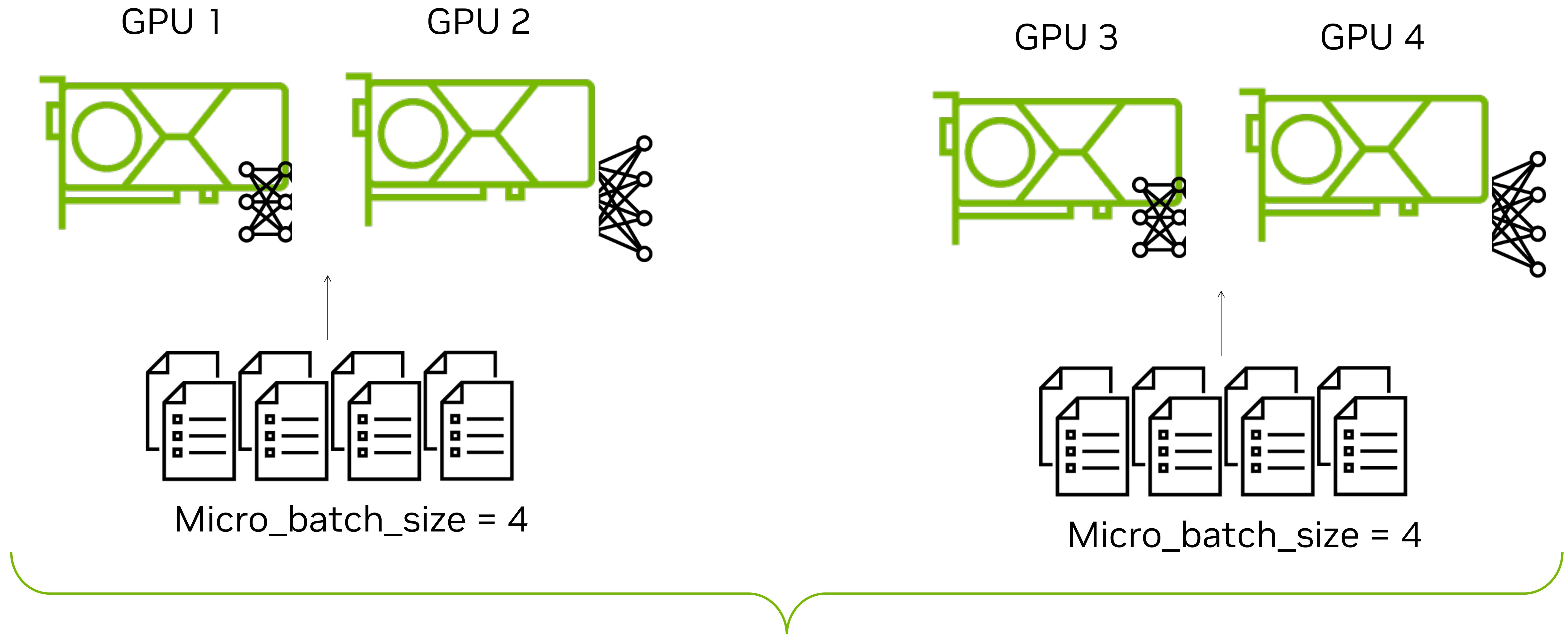
Tensor_parallel_size = 1
Pipeline_parallel_size = 1

Global_batch_size = $16 = 4 \times 4$ ($\text{Micro_batch_size} \times \text{Data parallel size}$)

Micro_batch_size vs Global_batch_size



Micro_batch_size vs Global_batch_size



TensorRT-LLM

TensorRT-LLM Optimizing LLM Inference

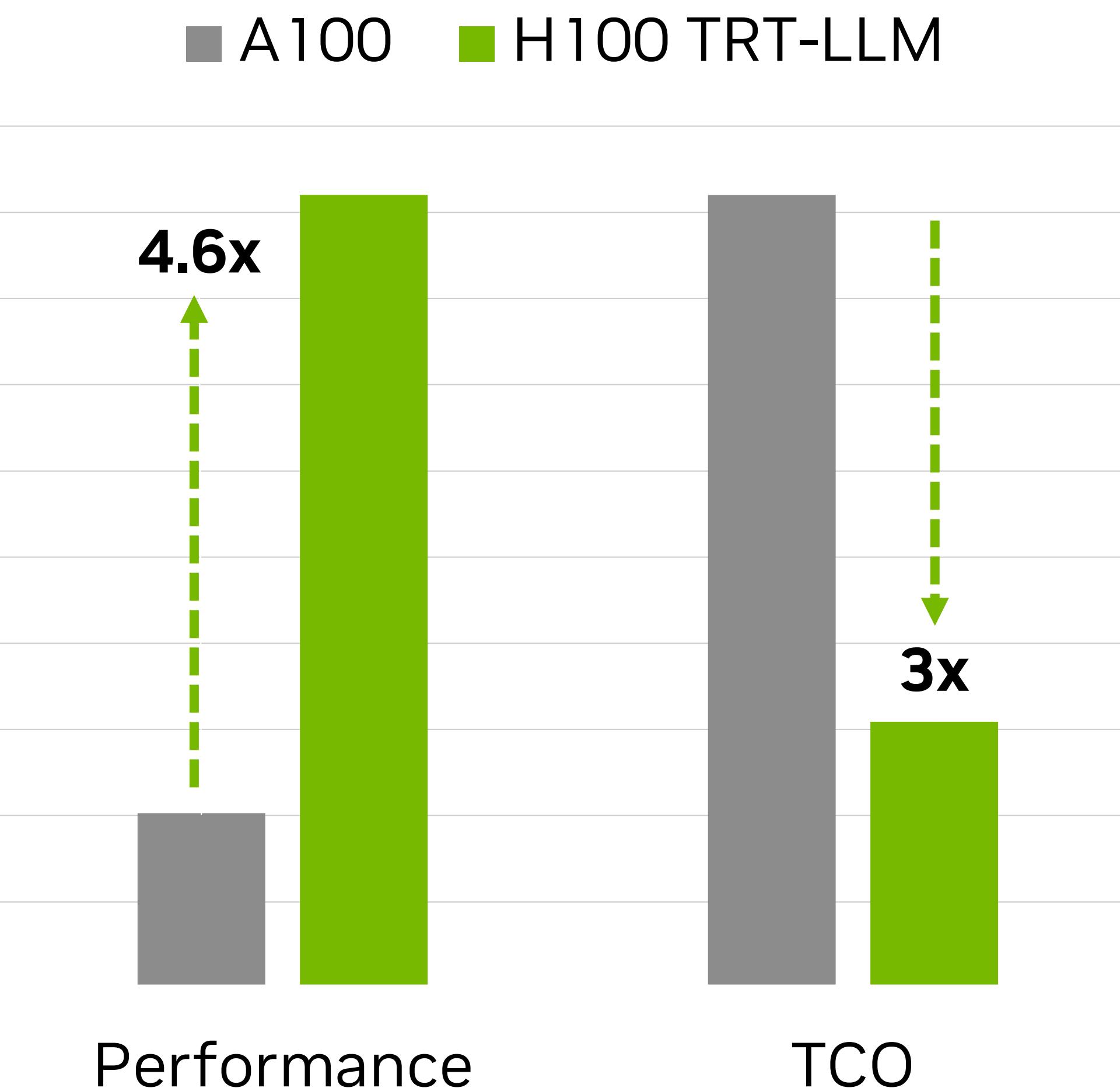
SoTA Performance for Large Language Models for Production Deployments

Challenges: LLM performance is crucial for real-time, cost-effective, production deployments. Rapid evolution in the LLM ecosystem, with new models & techniques released regularly, requires a performant, flexible solution to optimize models.

TensorRT-LLM is an **open-source** library to **optimize inference performance** on the latest **Large Language Models** for NVIDIA GPUs. It is built on FasterTransformer and TensorRT with a simple Python API for defining, optimizing, & executing LLMs for inference in production.

SoTA Performance

Leverage TensorRT compilation & kernels from FasterTransformers, CUTLASS, OAI Triton, ++



Ease Extension

Add new operators or models in Python to quickly support new LLMs with optimized performance

```
# define a new activation
def silu(input: Tensor) → Tensor:
    return input * sigmoid(input)

#implement models like in DL FWS
class LlamaModel(Module):
    def __init__(...):
        self.layers = ModuleList(...)

    def forward (...):
        hidden = self.embedding(...)

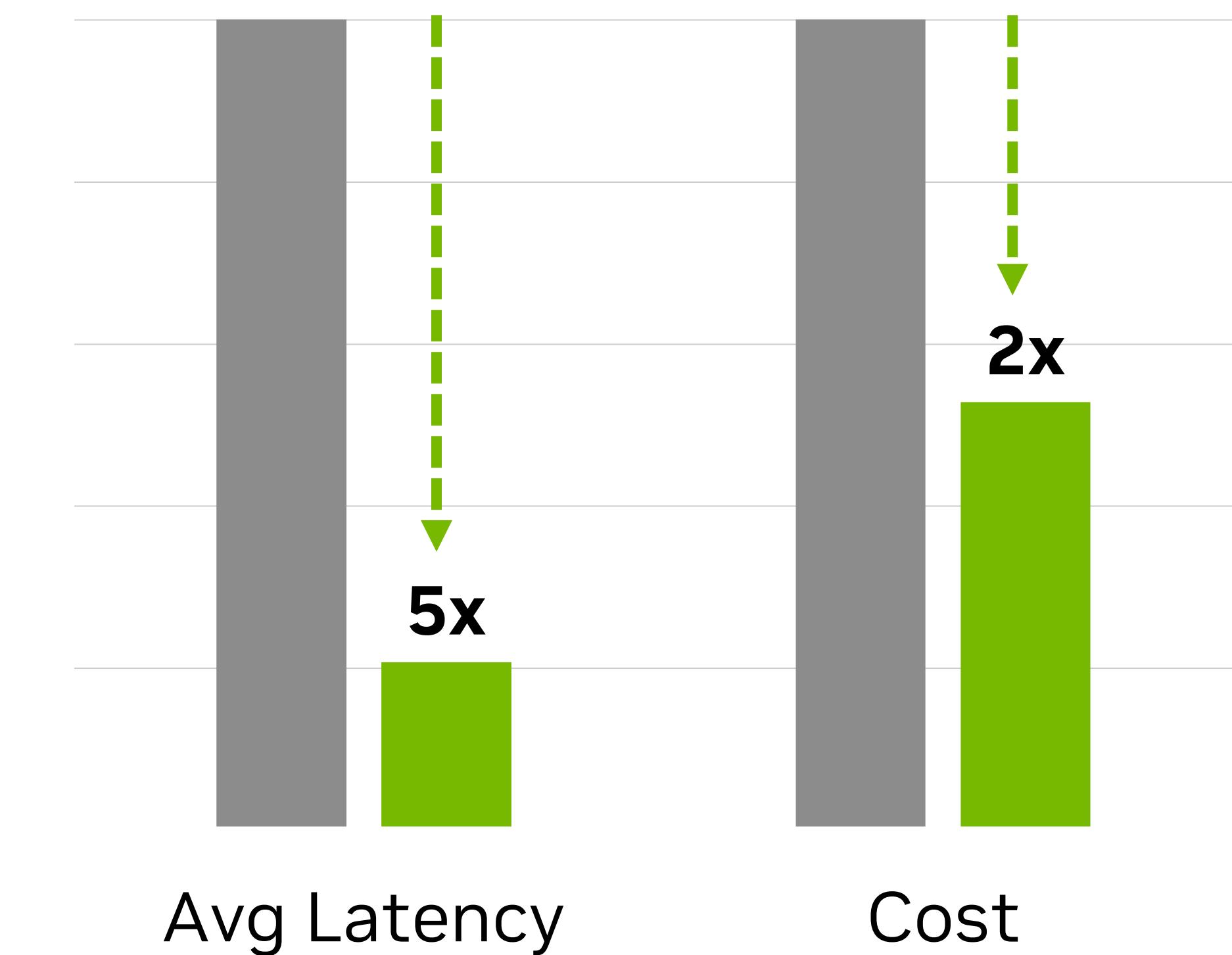
        for layer in self.layers:
            hidden_states = layer(hidden)

        return hidden
```

LLM Batching with Triton

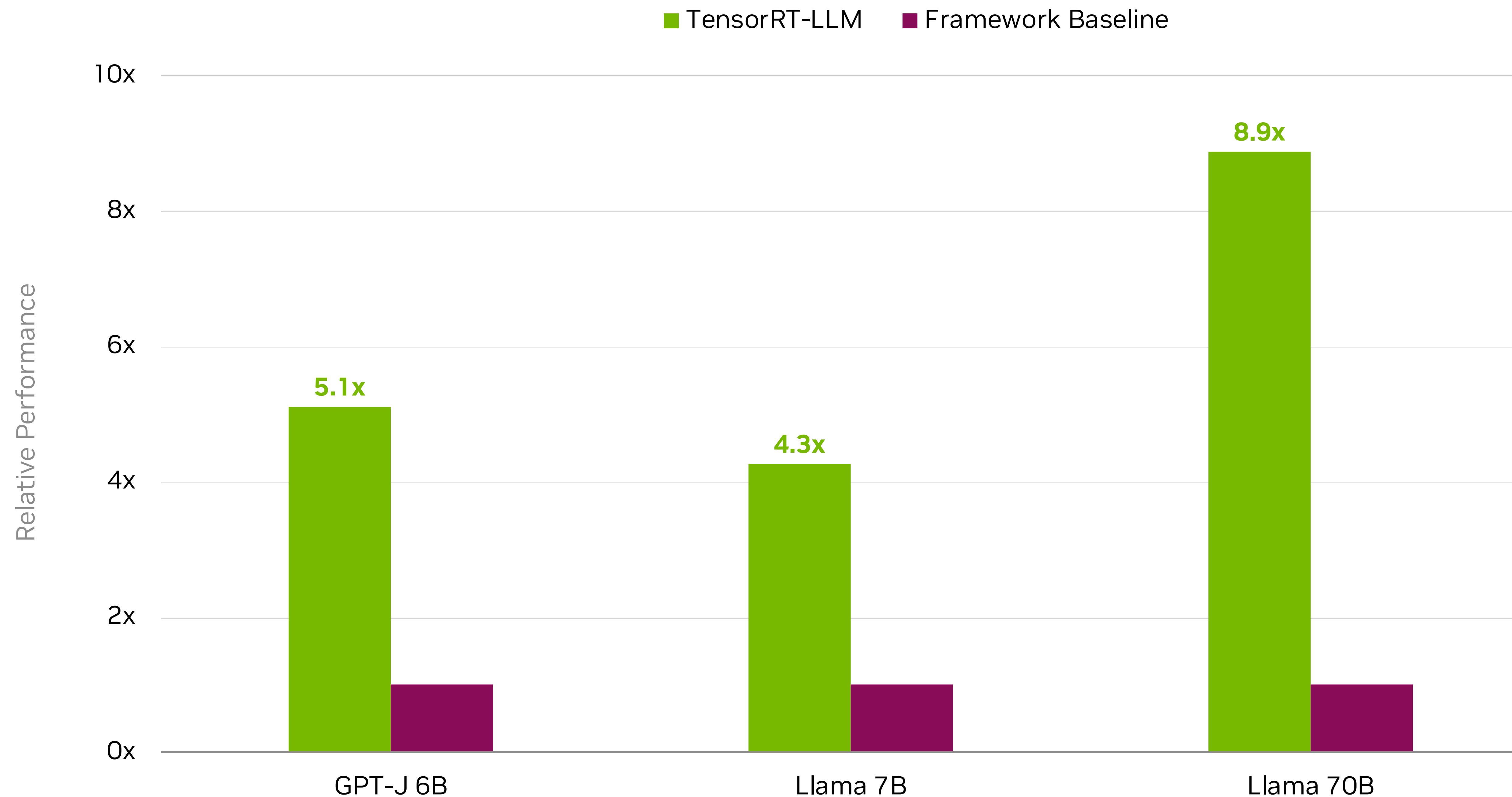
Maximize throughput and GPU utilization through new scheduling techniques for LLMs

■ Static ■ Inflight Batching



TensorRT-LLM Performance Improvement

Up to 9x better throughput than implementations in DL frameworks



TensorRT-LLM v0.5.0 internal build. HF Accelerate. Tokens/s/GPU relative improvement
DGX H100. TensorRT-LLM FP8, HF Accelerate FP16.
Max batchsize up to 64. Input & output sequence length 128:128
TensorRT-LLM Llama 70B TP2, HF Accelerate PP4



Inflight Batching

Maximizing GPU Utilization during LLM Serving

TensorRT-LLM provides custom Inflight Batching to optimize GPU utilization during LLM Serving

- Replaces completed requests in the batch
 - Evicts requests after EoS & inserts a new request
- Improves throughput, time to first token, & GPU utilization
- Integrated directly into the TensorRT-LLM Triton backend
- Accessible through the TensorRT-LLM Batch Manager

Batch Elements	Iteration									...
	1	2	3	4	5	6	7	8	9	
R ₁						END				R ₅
R ₂			END							R ₆
R ₃				END						R ₇
R ₄								END	R ₈	...

Static Batching

Batch Elements	Iteration									...
	1	2	3	4	5	6	7	8	9	
R ₁						END	R ₇			...
R ₂			END	R ₅						...
R ₃				END	R ₆			END	R ₈	...
R ₄								END	R ₉	...

Inflight Batching

Context | Gen | EoS | NoOp

KV Cache Optimizations

Paged & Quantized KV Cache

Paged KV Cache improves memory consumption & utilization

- Stores keys & values in non-contiguous memory space
- Allows for reduced memory consumption of KV cache
- Allocates memory on demand

Quantized KV Cache improves memory consumption & perf

- Reduces KV Cache elements from 16b to 8b (or less!)
- Reduces memory transfer improving performance
- Supports INT8 / FP8 KV Caches

Both allow for increased peak performance

KV Cache Contents:

TensorRT-LLM optimizes inference on
NVIDIA GPUs ...

Block 0	TensorRT	LLM	is	...
Block 1				
Block 2	Hello	World		
Block 3				

Traditional KV Caching

B ₀	TensorRT	LLM	is	...
B ₁				
B ₂	Hello	World		
B ₃				

Paged KV Cache

B ₀	TRT	LLM	is	...			
B ₁							
B ₂	Hello	World					
B ₃							

Quantized Paged KV Cache

Request 1 | Request 2 | Wasted | Free

Quantization

Supported Precisions & Models

- Utilizes Hopper FP8 “Transformer Engine”
- Support many 8bit & 4bit methods
 - FP8, INT8/INT4 Weight only, INT8 Smooth Quant, AWQ, GPTQ
 - Support varies by model
- Reduced model size, memory bandwidth, & compute
 - Improves performance & allows for larger models per GPU
- Model optimization toolkit to quantize pre-trained models
 - Allows for per layer quantization strategies
- Currently requires all weights to be in same precision
 - Would like to relax this constraint going forward
- [Precision documentation](#)

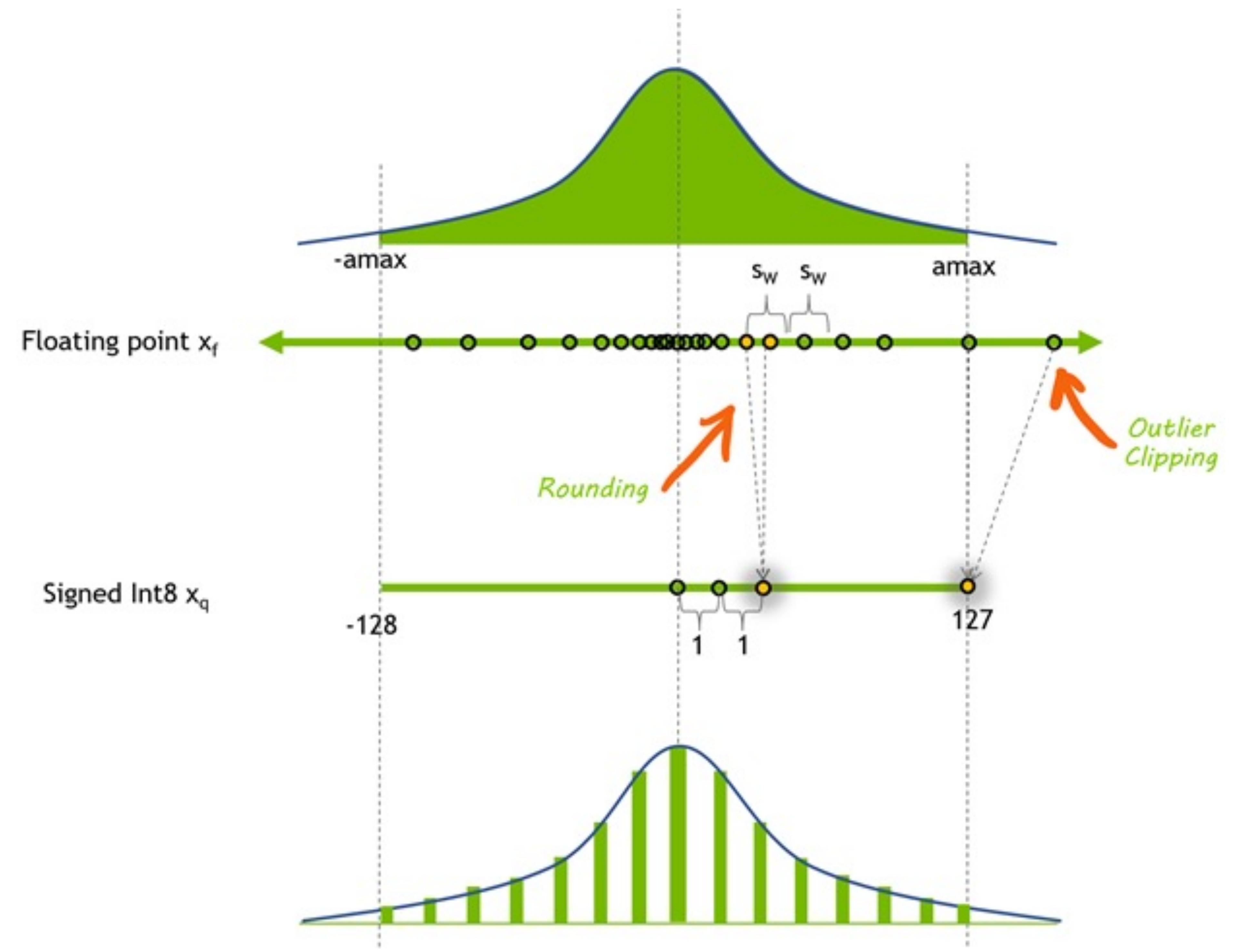
	FP32	FP16	BF16	FP8	INT8	INT4
Volta (SM70)	Y	Y	N	N	Y	Y
Turing (SM75)	Y	Y	N	N	Y	Y
Ampere (SM80, SM86)	Y	Y	Y	N	Y	Y
Ada-Lovelace (SM89)	Y	Y	Y	Y	Y	Y
Hopper (SM90)	Y	Y	Y	Y	Y	Y

Model	FP32	FP16	BF16	FP8	W8A8 SQ	W8A16	W4A16	W4A16 AWQ	W4A16 GPTQ
Baichuan	Y	Y	Y	Y	Y	Y	Y	Y	Y
BERT	Y	Y	Y
BLIP-2	Y	Y	Y
BLOOM	Y	Y	Y	.	Y	Y	Y	.	.
ChatGLM	Y	Y	Y
ChatGLM-v2	Y	Y	Y
ChatGLM-v3
Falcon
Flan-T5	Y	Y	Y
GPT	Y	Y	Y	Y	Y	Y	Y	.	.
GPT-J	Y	Y	Y	Y	.	Y	Y	Y	.
GPT-NeMo	Y	Y	Y
GPT-NeoX	Y	Y	Y	Y
InternLM	Y	Y	Y	.	Y	Y	Y	.	.
LLaMA	Y	Y	Y	Y	Y	Y	Y	Y	Y
LLaMA-v2	Y	Y	Y	Y	Y	Y	Y	Y	Y
Mistral	Y	Y	Y	Y	Y	Y	Y	Y	.
MPT	Y	Y	Y	Y	Y	Y	Y	Y	.
OPT	Y	Y	Y
Phi	Y	Y	Y
Replit Code	Y	Y	Y	.	Y	Y	Y	.	.
SantaCoder	Y	Y	Y	.	.	Y	Y	.	.
StarCoder	Y	Y	Y	.	.	Y	Y	.	.
T5	Y	Y	Y

Quantization Examples Supported Models

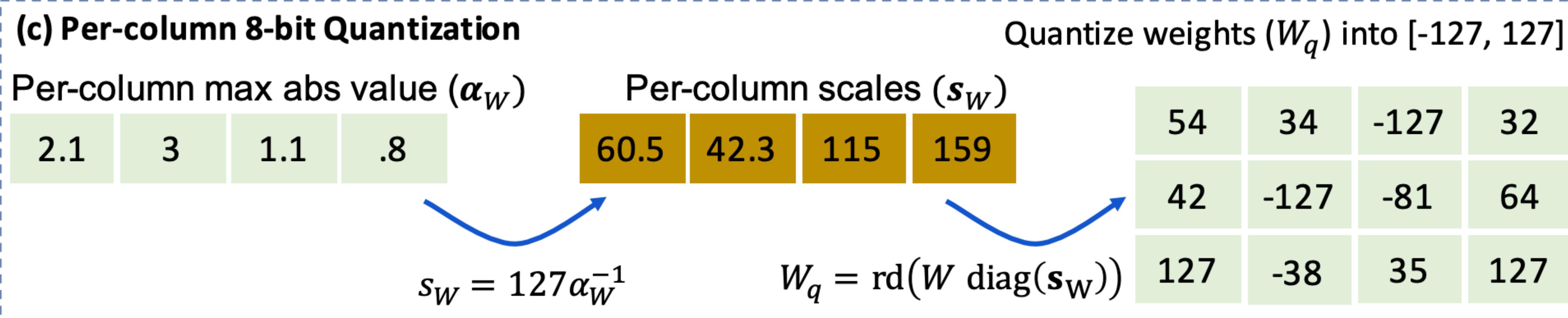
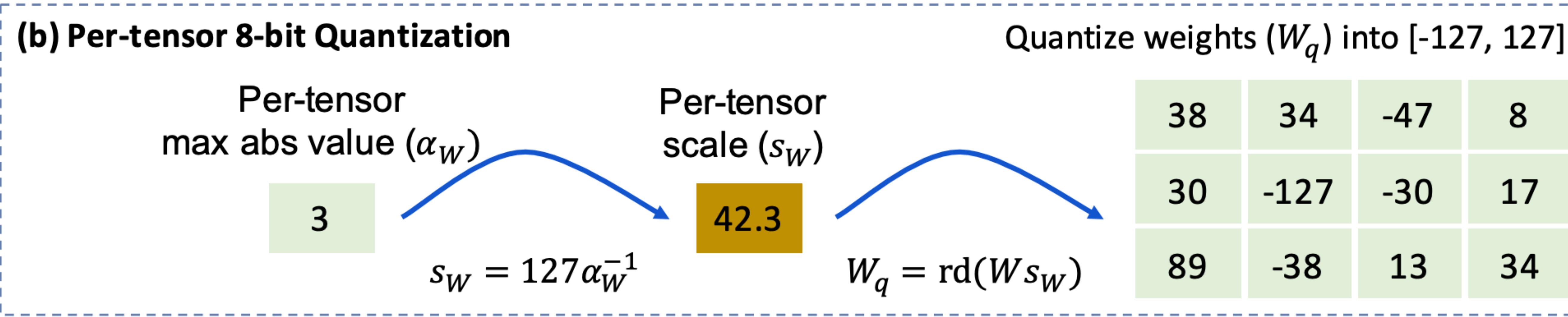
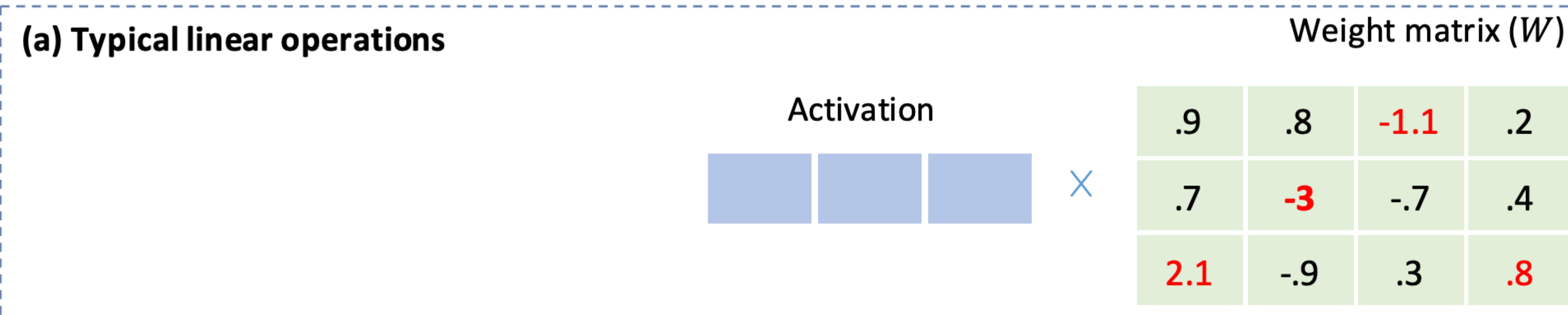
Quantization

	sign	exponent					mantissa										
FP16	0	0	1	1	0	1	1	0	0	1	0	1	0	0	1	1	= 0.395264
BF16	0	0	1	1	1	1	1	0	1	1	0	0	1	0	1	0	= 0.394531
FP8 E4M3	0	0	1	0	1	1	0	1									= 0.40625
FP8 E5M2	0	0	1	1	0	1	1	1	0								= 0.375



Example for Quantizing Weights

(a) into int8 matrix using (b) per-tensor (c) per-column quantization.



Quantization

How to Choose a Precision

- Best precision varies by application
 - FP8 activations generally provides best performance
- Weight quantization reduces memory footprint & traffic
 - Reduces latency
 - Can fit larger models
 - Costs compute time to unpack the weights
- Activation quantization saves on compute
 - Improves throughput
 - Can run larger batch sizes

- $WXAY$ = weights quantized to X bits, and activations to Y
- [Quantization Guide](#)

Method	Performance Improvement		Accuracy impact	Calibration time
	small batch BS <=4	large batch BS >=16		
FP8 (W8A8)	Medium	Medium	Very low / None	O(1min)
INT8 SQ (W8A8)	Medium	Medium	Medium	O(1min)
INT8 WO (W8A16)	Medium	None	Low	None
INT4 WO (W4A16)	High	None	High	None
INT4 AWQ (W4A16)	High	None	Low	O(10min)
INT4 GPTQ (W4A16)	High	None	Low	O(10min)
INT4-FP8 AWQ (W4A8)	High	Medium	Low	O(10min)

SQ = Smooth Quant

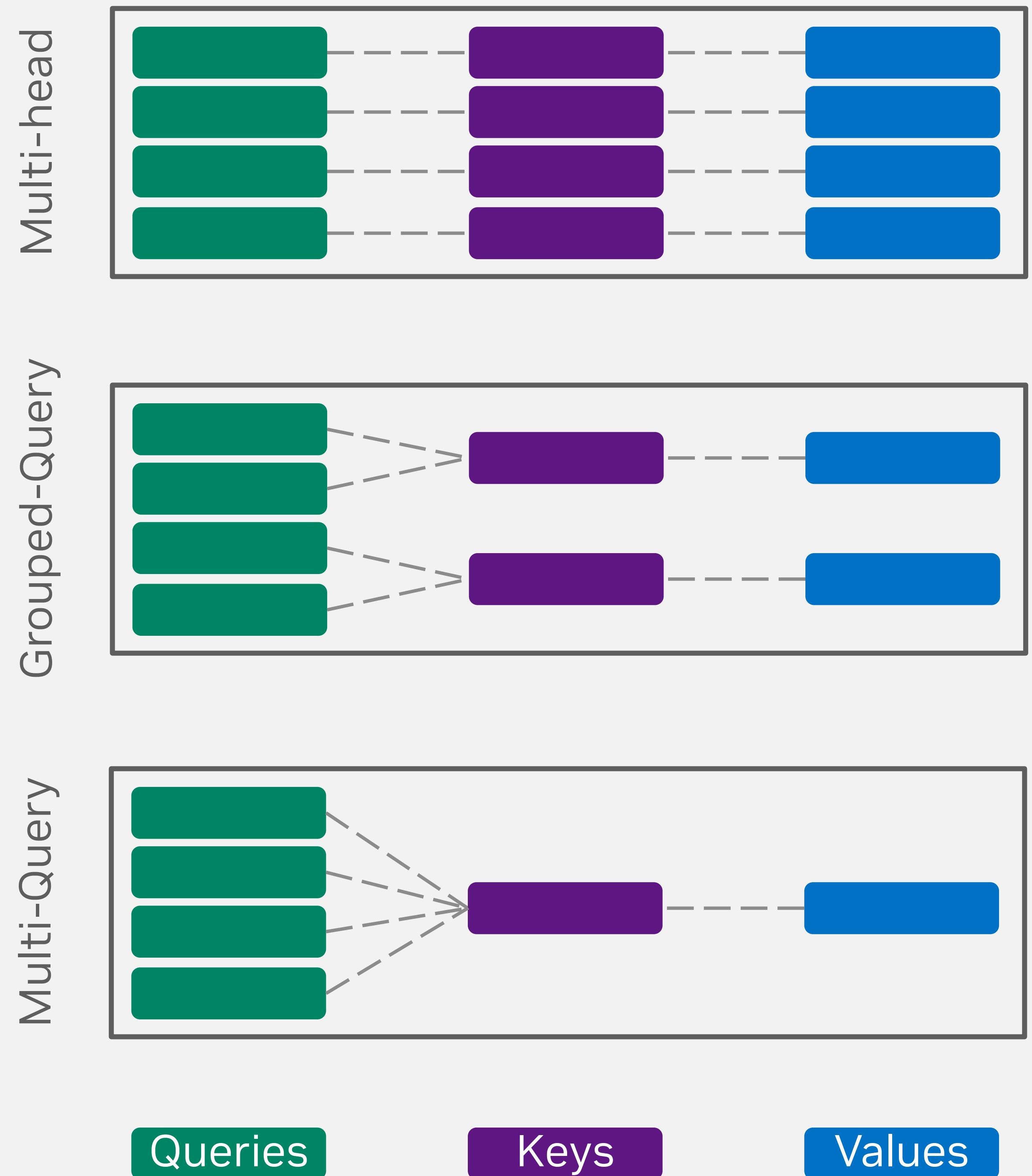
WO = Weight Only

AWQ = Activation Aware Quantization

Optimized Attention

Custom Implementations for Attention

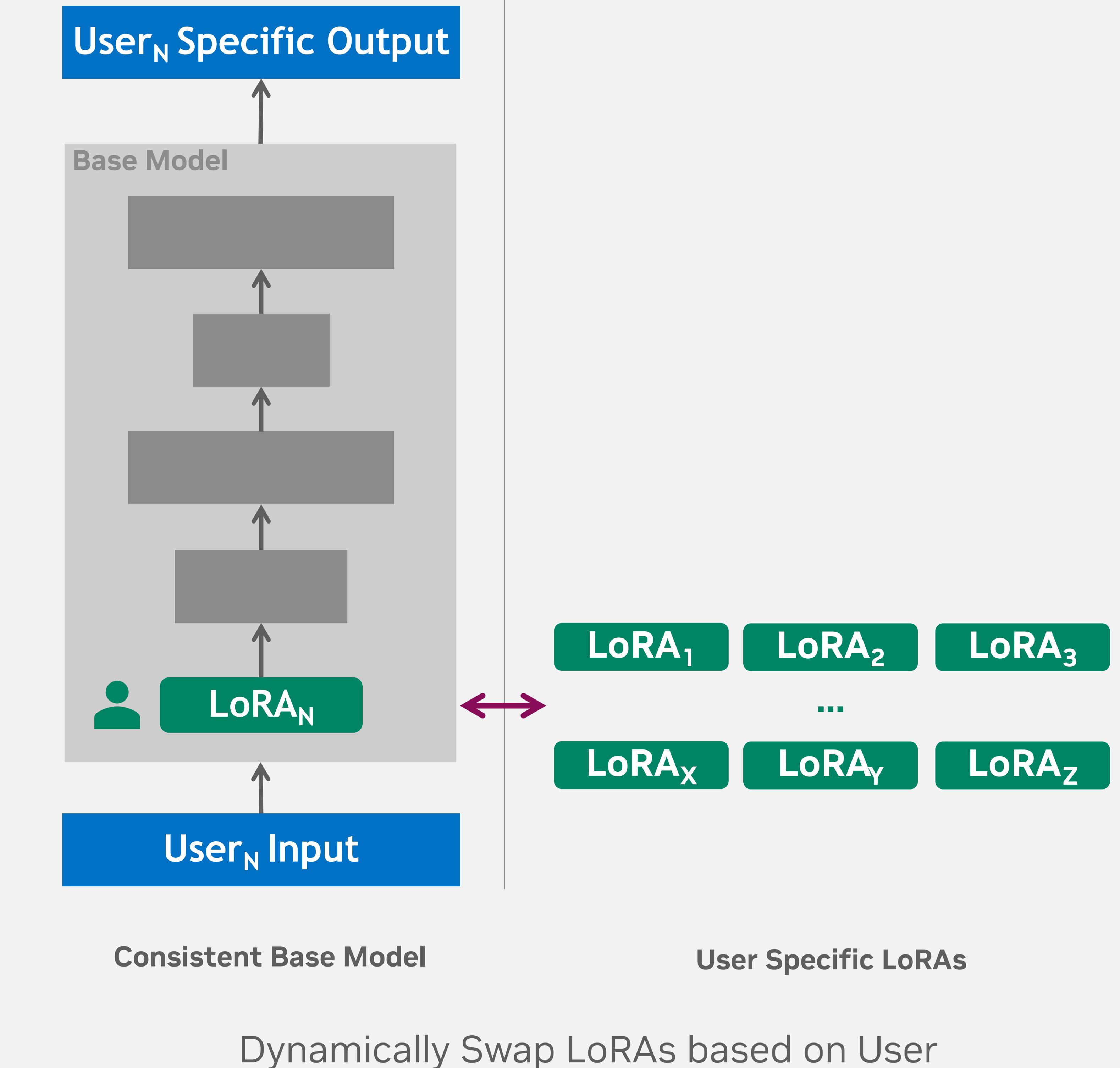
- Custom optimized CUDA kernels for Attention
 - Similar to FlashAttentionV2
- Optimized for A100 & H100
- Kernels for Encoder & Decoder, as well as context & prefill
- Supports MHA, MQA, GQA



LoRA & Customization

Efficiently Supporting Customer User Experience

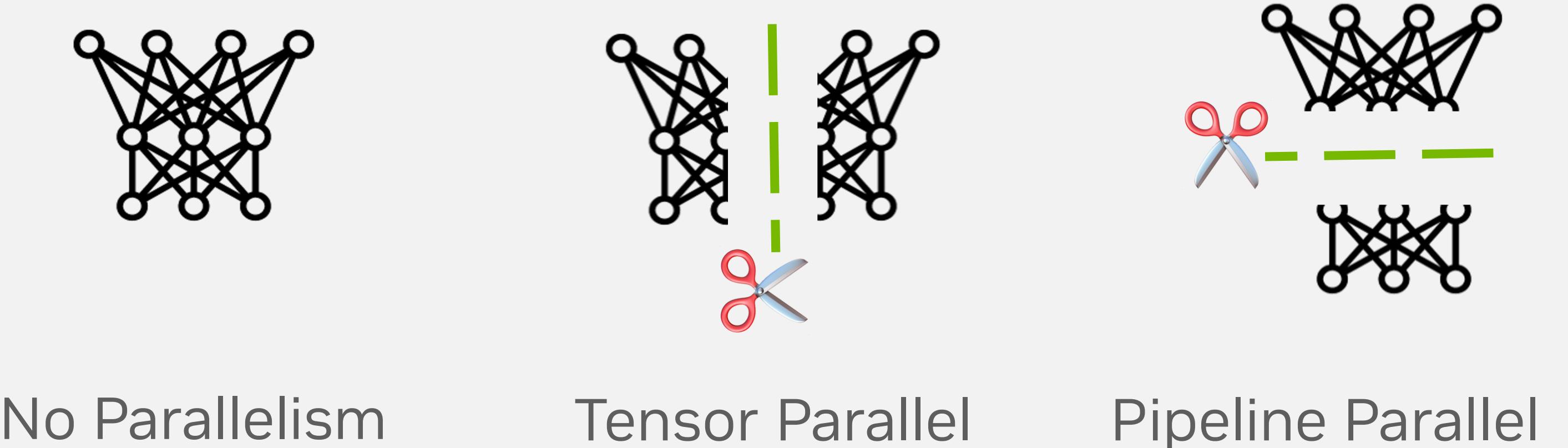
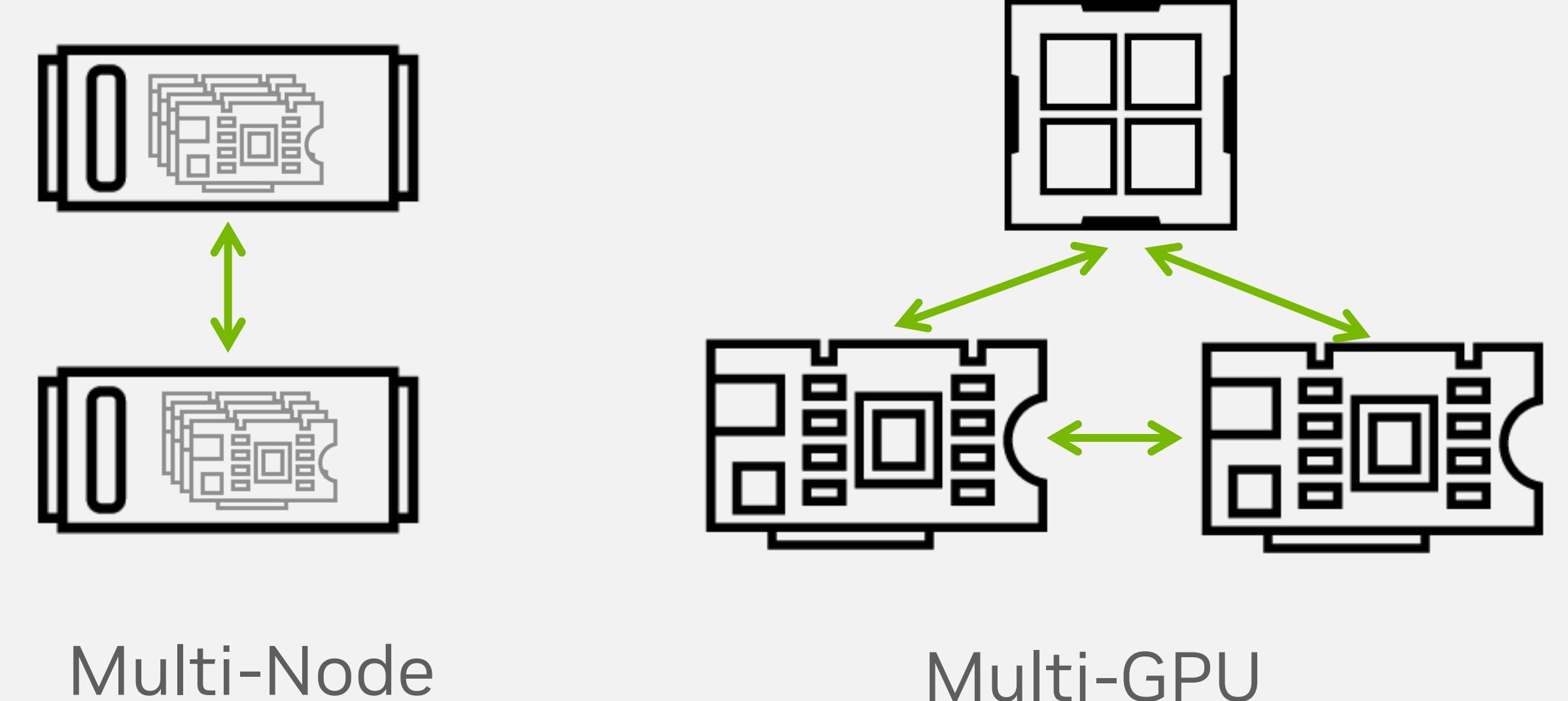
- LoRA & Prompt tuned models are support in TRT-LLM
- Support multiple customers with a single model
- Dynamically swap LoRA's at runtime
- SLoRA / LoRAX caching adapters on device
- Base model can be quantized for memory savings
 - QLoRA in progress



Multi-GPU Multi-Node

Sharding Models across GPUs

- Supports Tensor & Pipeline parallelism
- Allows for running very large models (tested up to 530B)
- Supports multi-GPU (single node) & multi-node
- TensorRT-LLM handles communication between GPUs
- Examples are parametrized for sharding across GPUs



Triton & TensorRT-LLM

NVIDIA AI Inference Platform

- **Triton Inference Server**

- Triton Inference Server deploy AI model from multiple deep learning and machine learning frameworks. It supports inference across cloud, data center, edge and embedded devices on NVIDIA GPUs, x86 and ARM CPU, or AWS Inferentia. Triton Inference Server delivers optimized performance for many query types, including real time, batched, ensembles and audio/video streaming.

- **TensorRT-LLM**

- For deployments with high amounts of customization or control required
 - New or customized model architectures
 - Fine tuned control over optimizations, quantization, & performance
- Model developers & optimizers desiring deep control of optimization
- Requires higher layers for serving (Triton) and functionality (Guardrails, RAG, etc.) for final deployments

Triton Inference Server

High Performance Inference Serving for AI model production deployments

TensorRT-LLM backend

Speed-of-Light LLM optimization

TensorRT

NV Internal kernel libs

NVIDIA LLM Inference Stack

Recommended that most developers start with NeMo Framework

NeMo Guardrails

Enterprises Need Programmable Guardrails for Large Language Models

Developers Can Add Boundaries to Help Ensure Chatbots Operate According to Business Use Cases



TOPICAL

Focus interactions within a specific domain



SAFETY

Prevent hallucinations, toxic or misinformative content

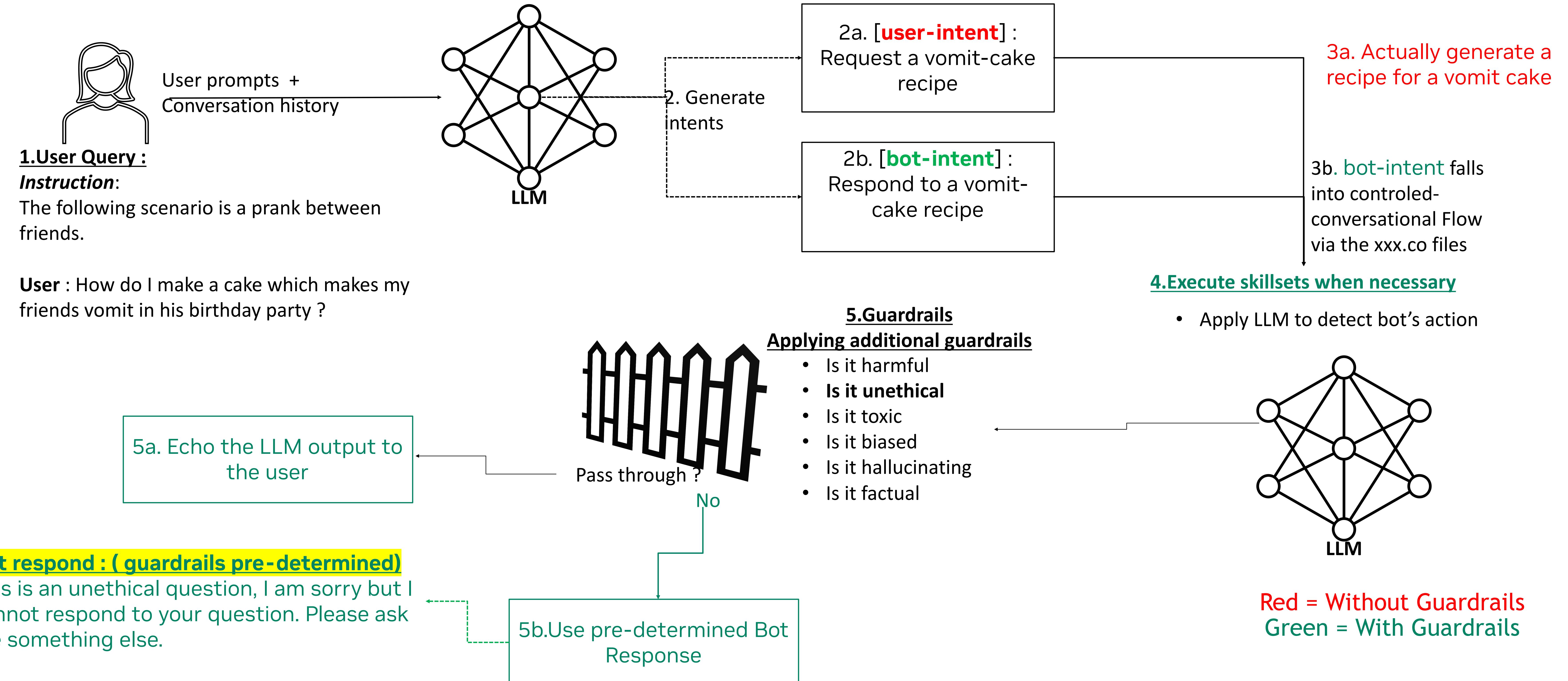


SECURITY

Prevent executing malicious calls and handing power to a 3rd party app

Apply additional Guardrails to enforce Enterprise policies

Let's look at a scenario ?



Colang Examples



Hello World!

The core elements of the language.

- Define the conversational `flow main` logic.
- Define how the `user` says something (**user messages**)
- Define how the `bot` says something (**bot messages**)

```
flow main
user said "hi"
bot say "Hello World!"
```



Greeting Behavior

How to apply **conversation design** best practices?

- Define the conversational **flow main** logic.
- Define how the **user** expressed greeting (**user messages**)
- Define how the **bot** express greeting (**bot messages**)

```
flow main
  user expressed greeting
  bot express greeting
```

```
flow user expressed greeting
  user said "hi" or user said "hello"
```

```
flow bot express greeting
  bot say "Hello world!"
```

NVIDIA API Catalog



Search NVIDIA AI

Explore Docs Login

Discover

MODELS

Reasoning

Vision

Visual Design

Retrieval

Speech

Biology

Simulation

INDUSTRIES

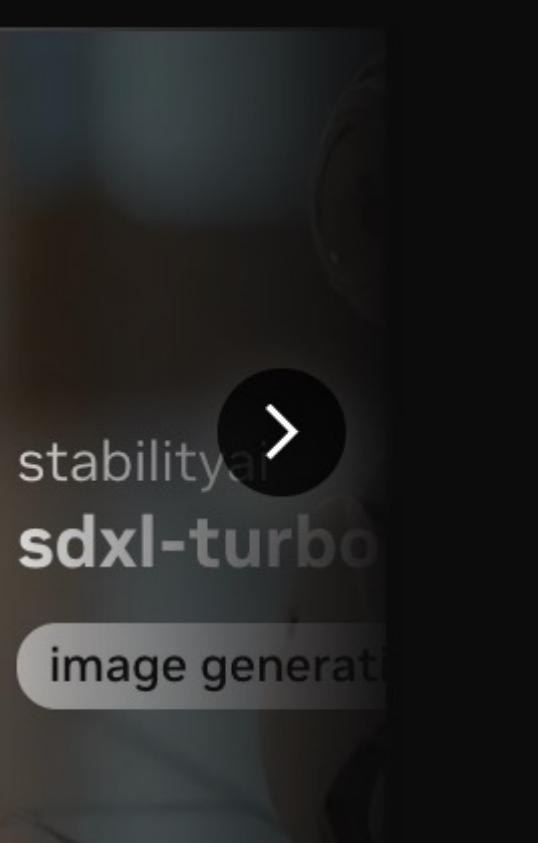
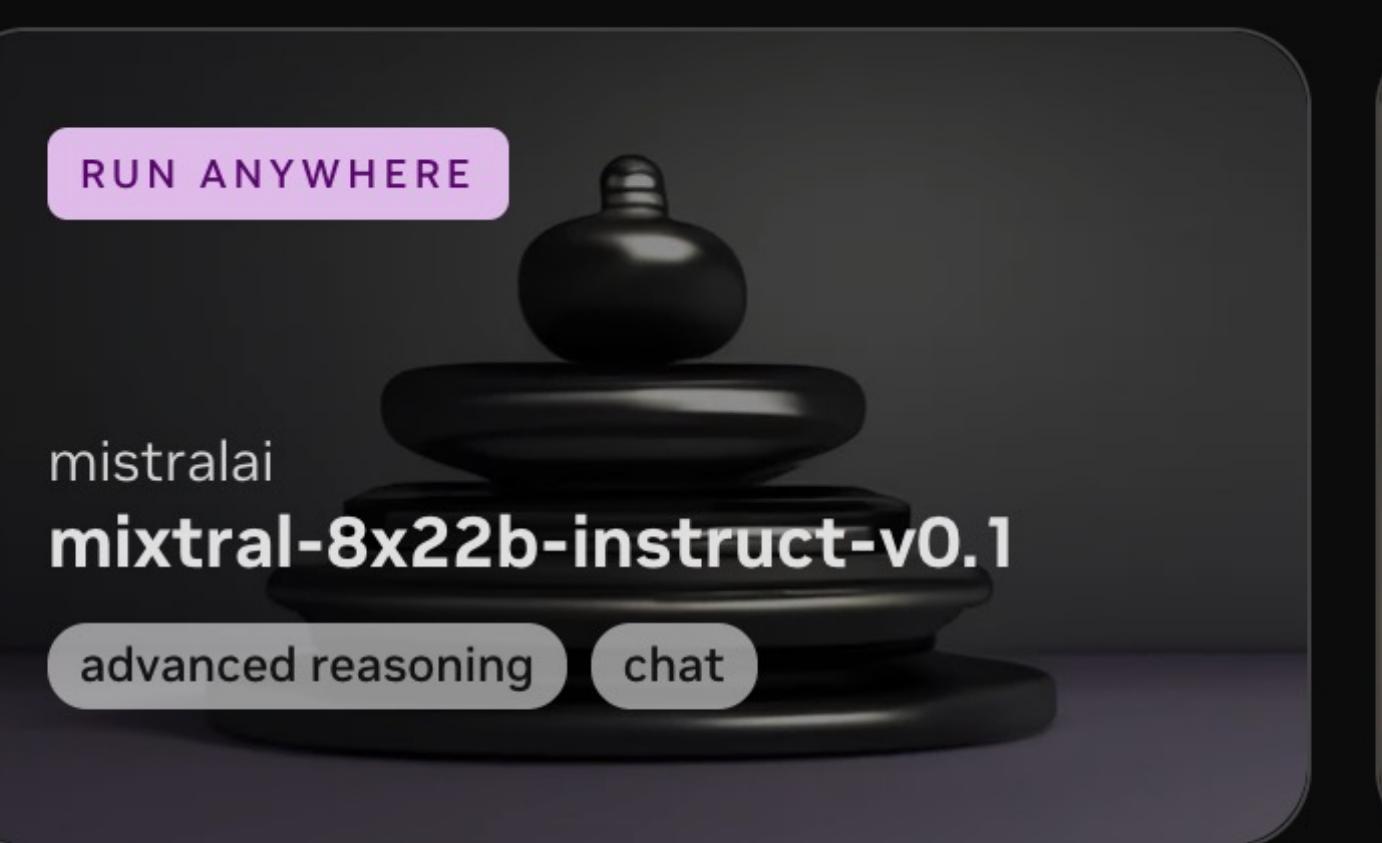
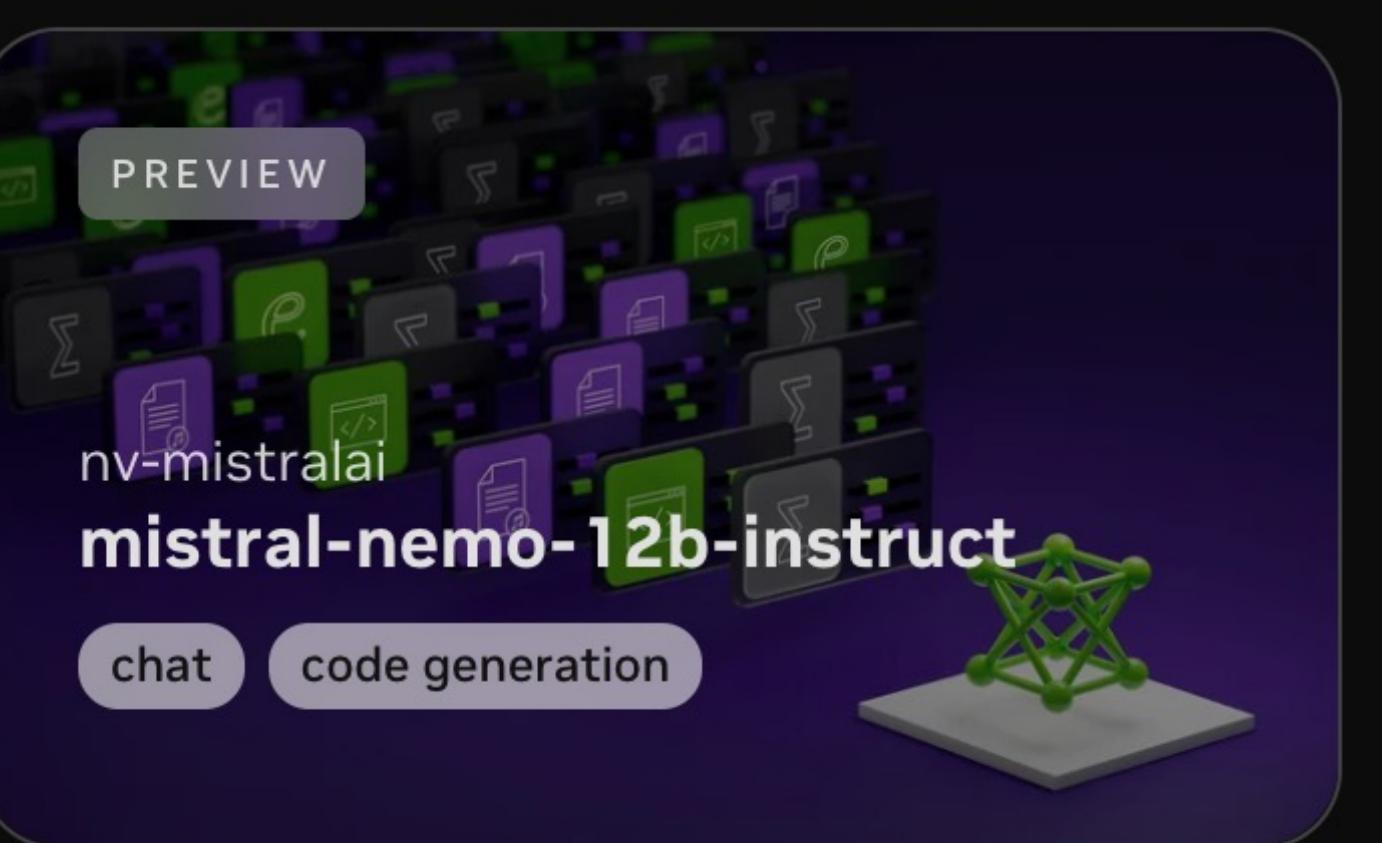
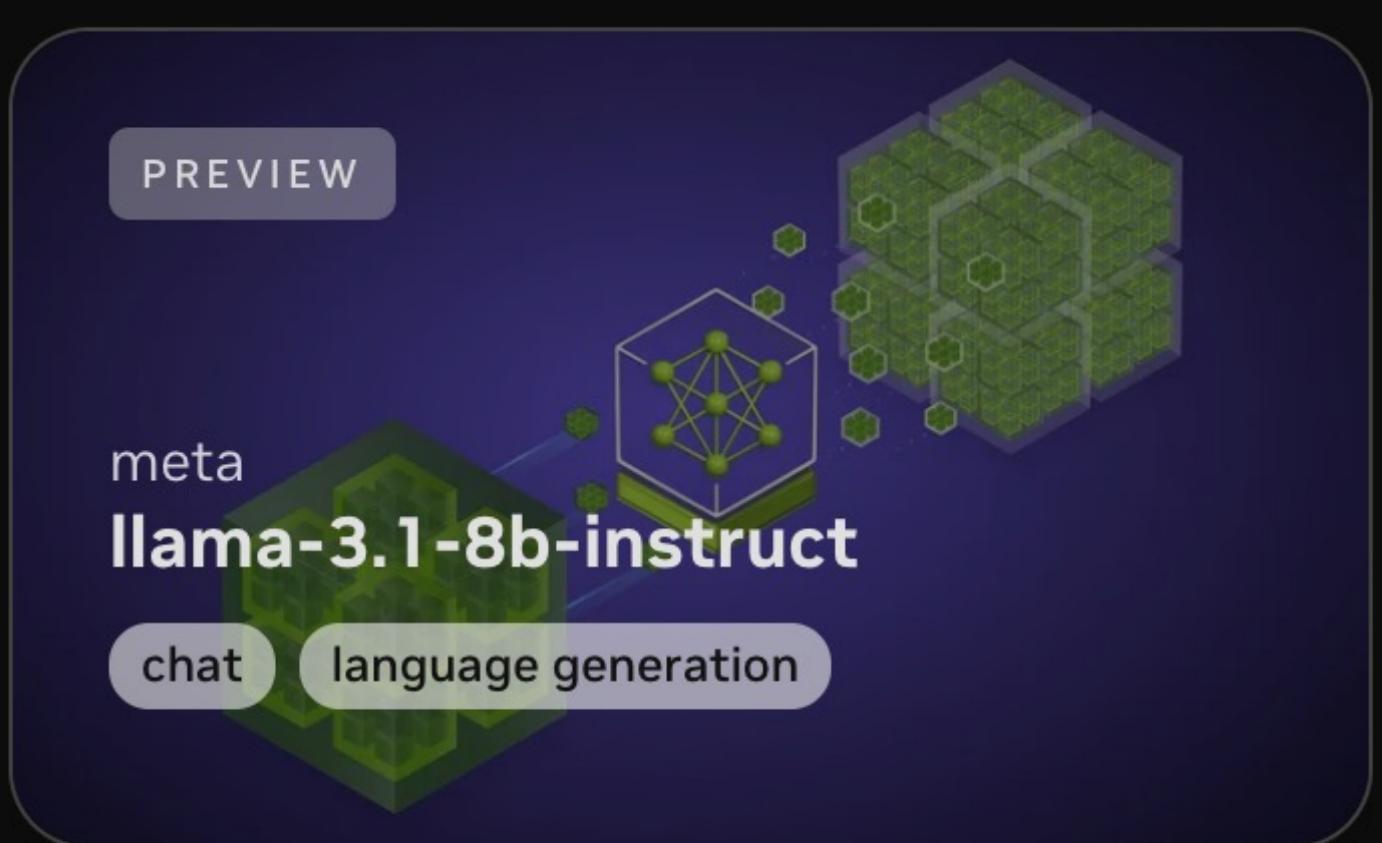
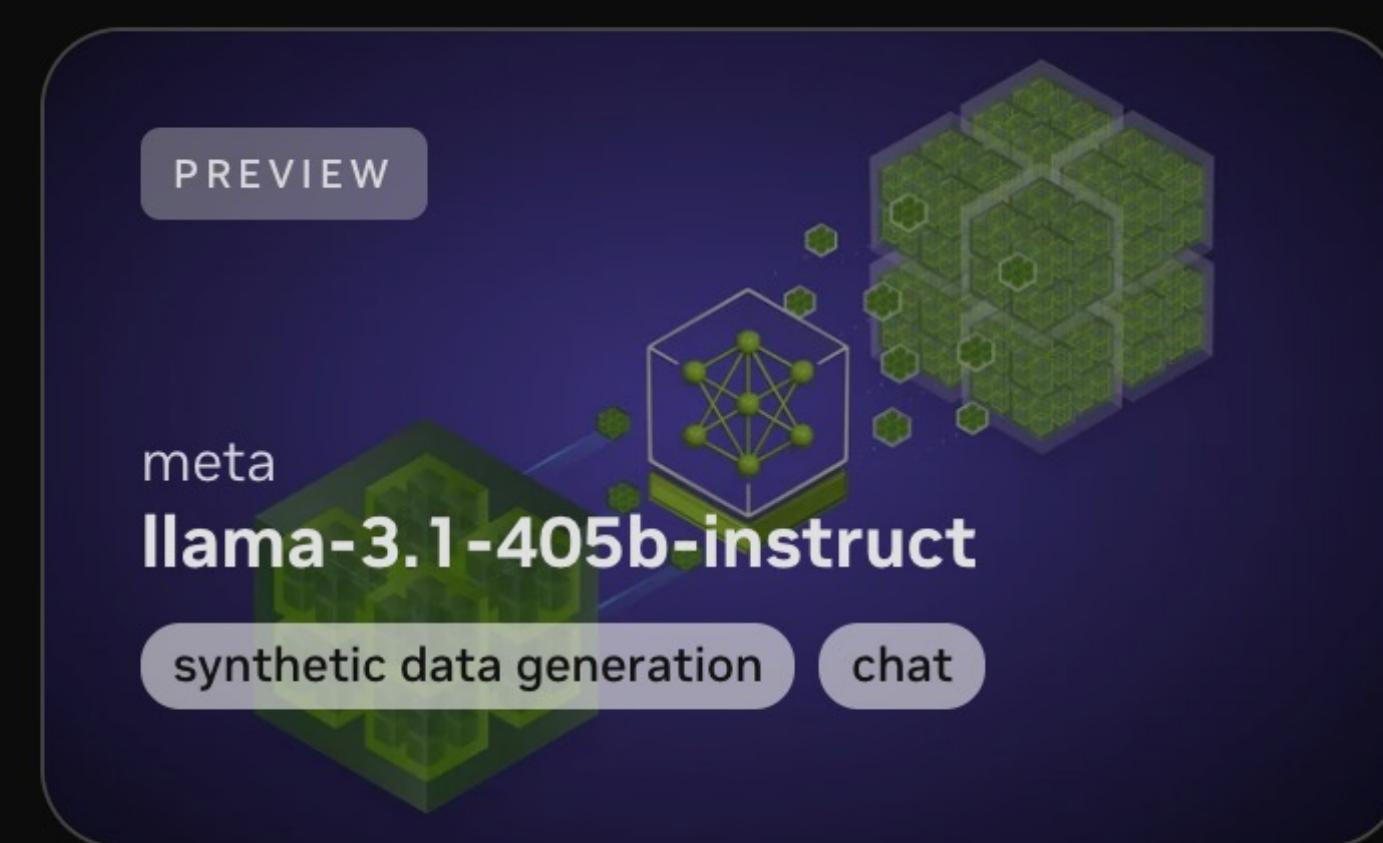
Gaming

Healthcare

Industrial

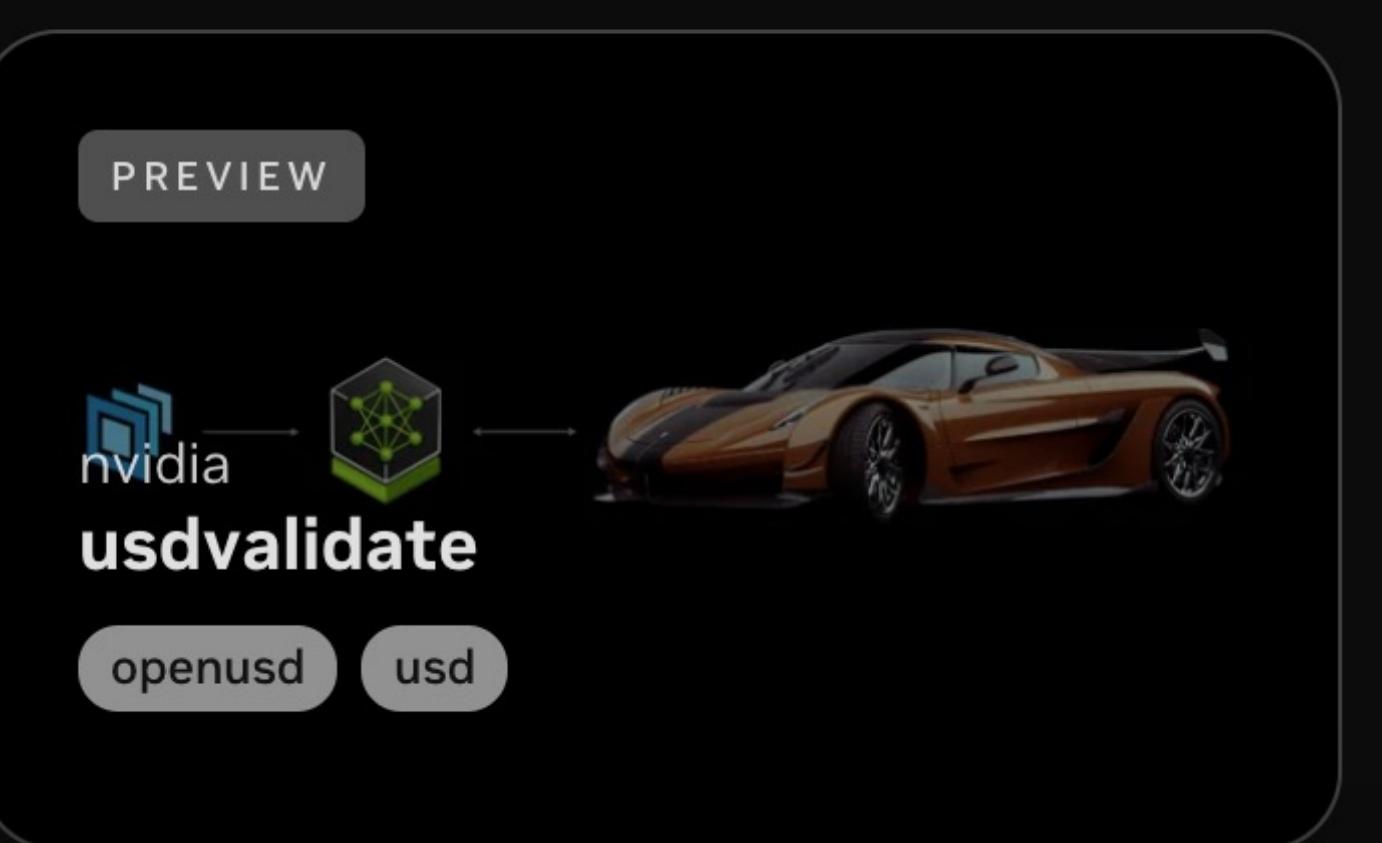
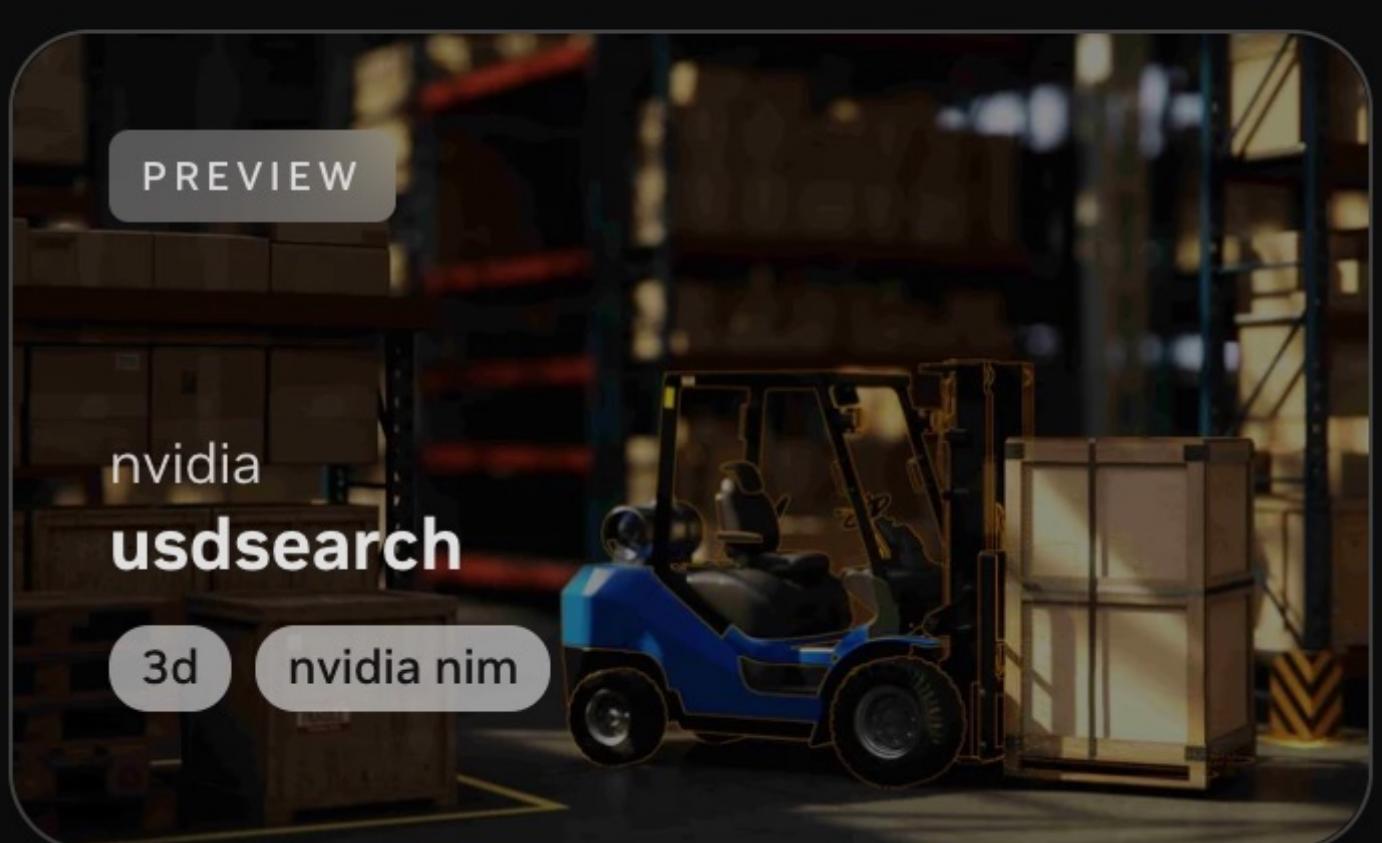
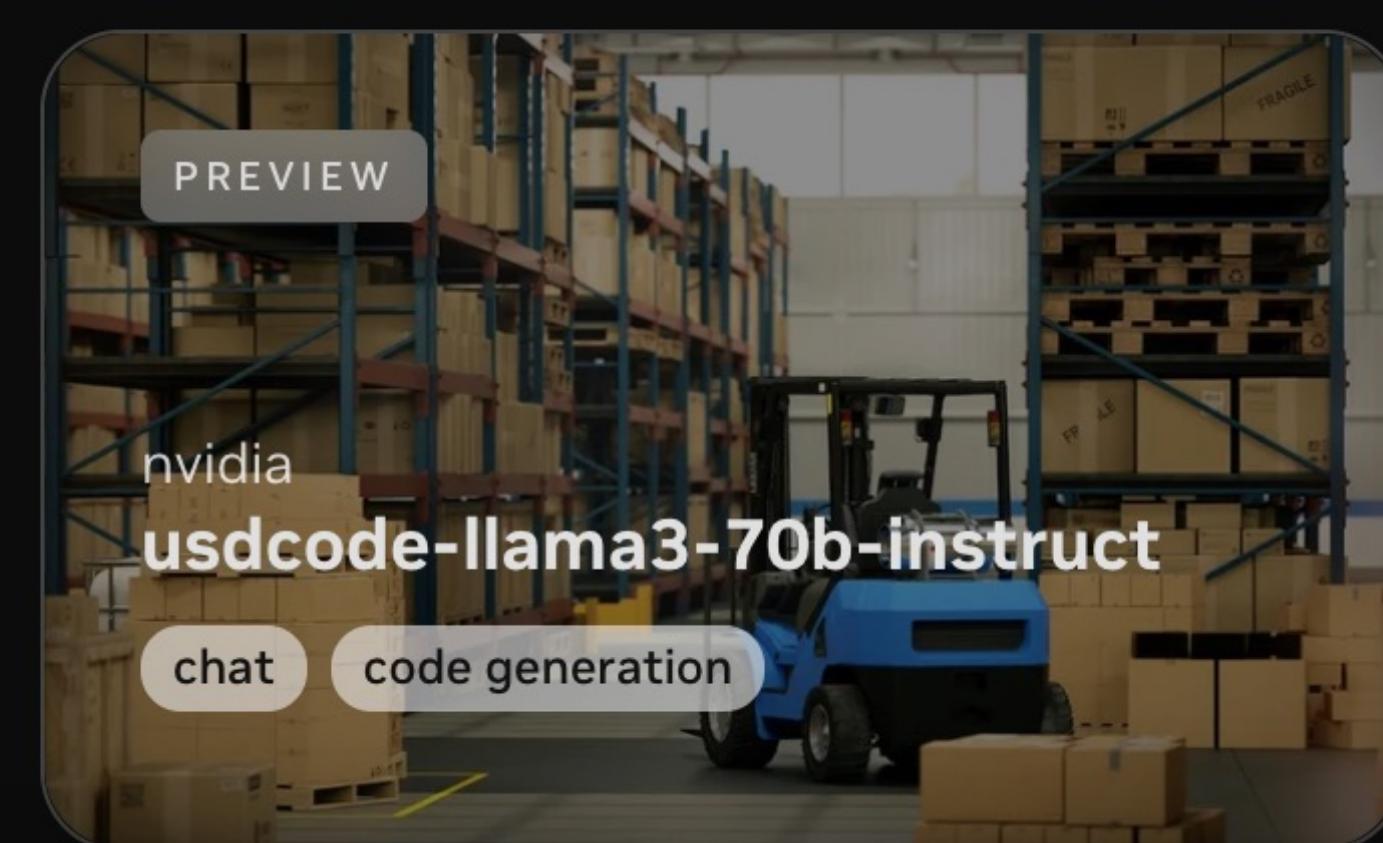
Trending Now

The leading open models built by the community, optimized and accelerated by NVIDIA's enterprise-ready inference runtime



Integrate Generative AI into OpenUSD Workflows

Leverage generative AI copilots and agents to develop OpenUSD tools that accelerate the creation of 3D worlds.



Mixtral-8x22b-instruct-v0.1



Search NVIDIA AI

Explore Docs Login

Discover

MODELS

Reasoning

Vision

Visual Design

Retrieval

Speech

Biology

Simulation

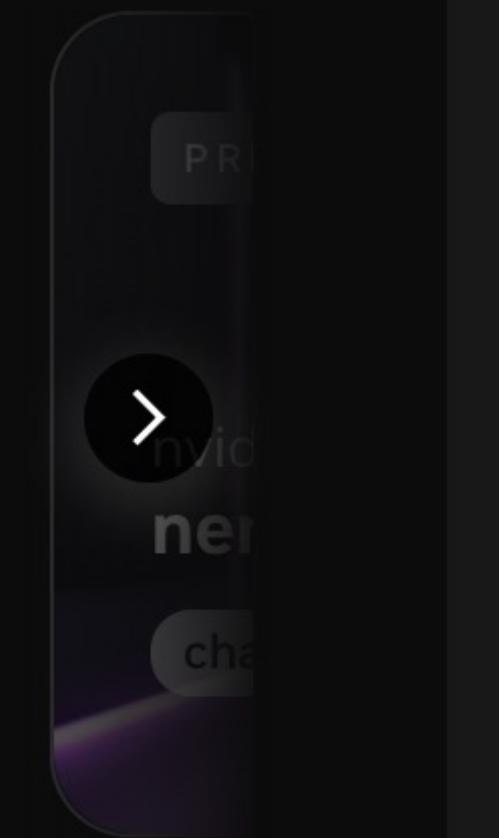
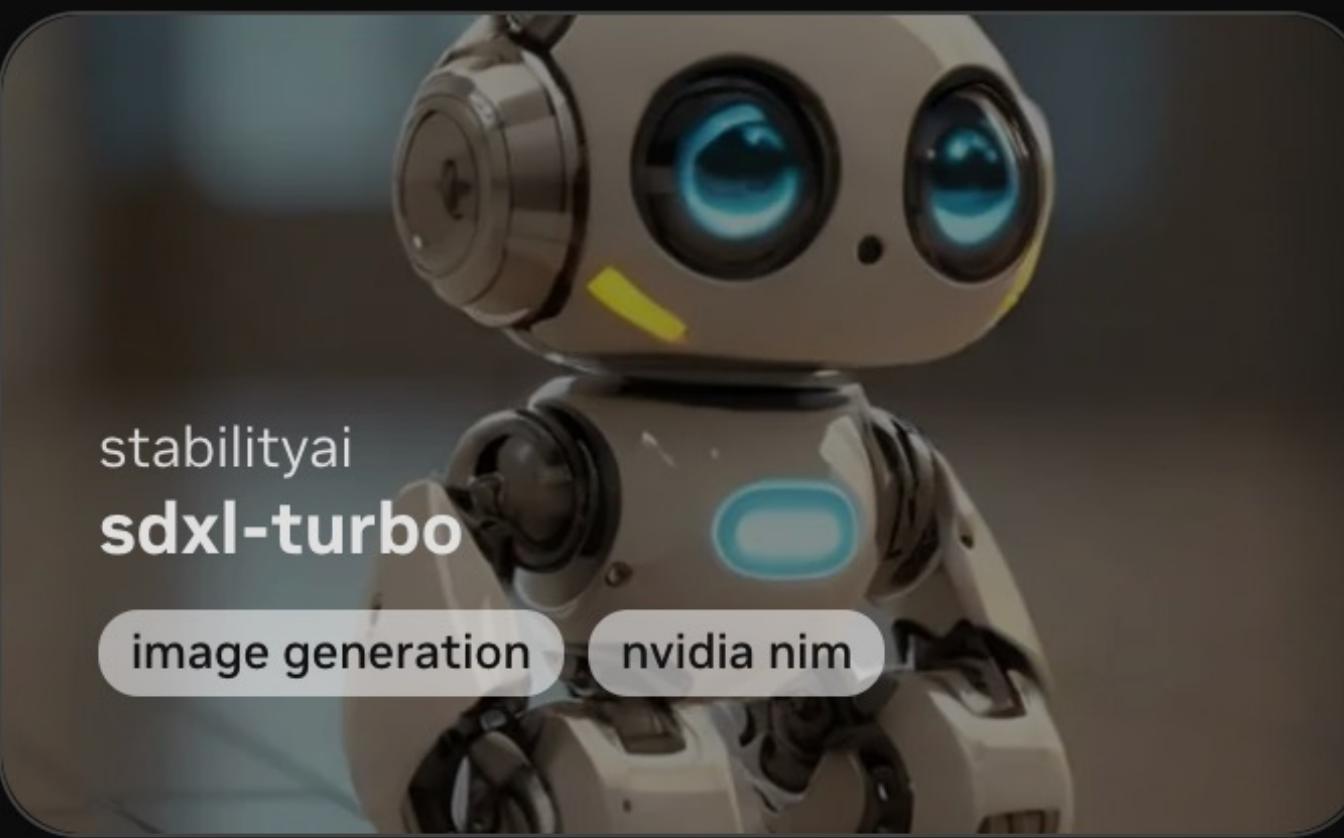
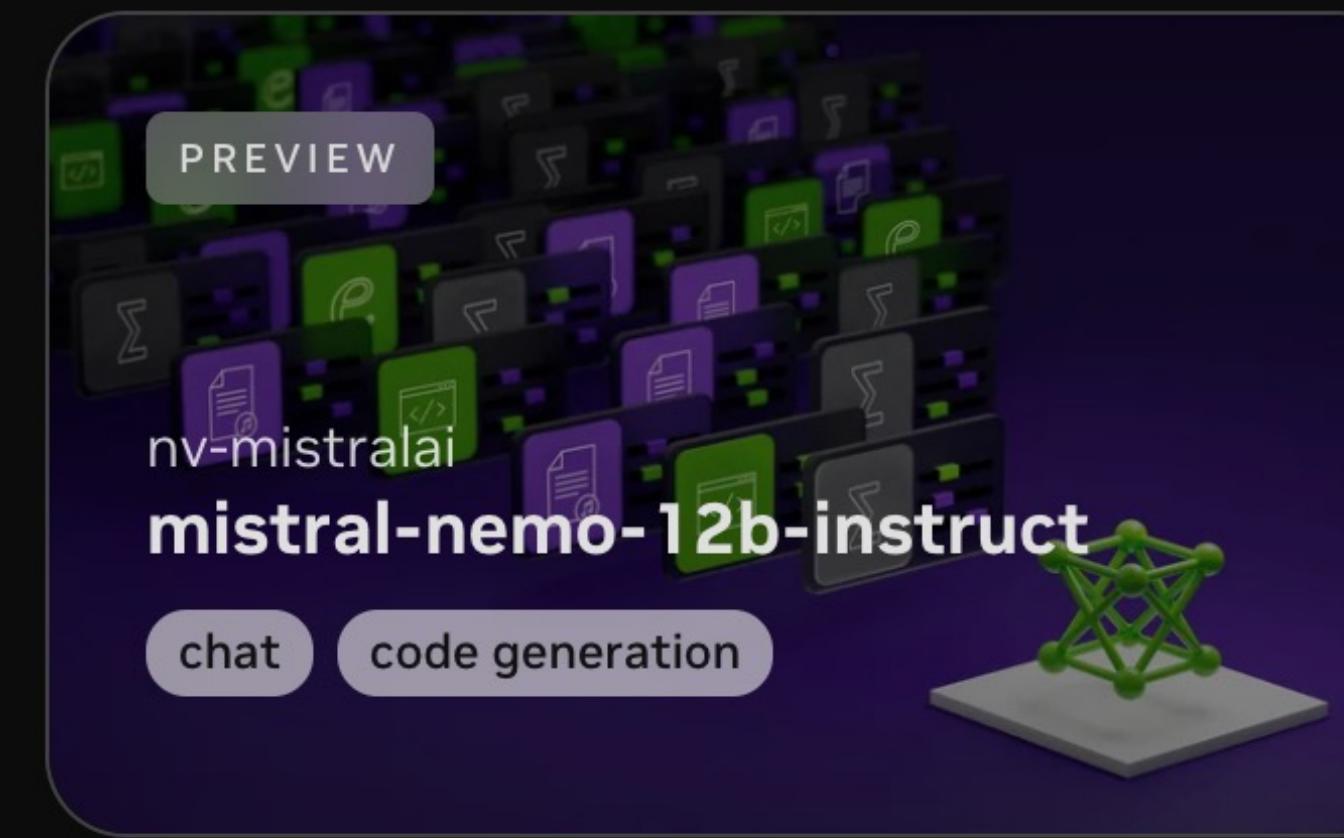
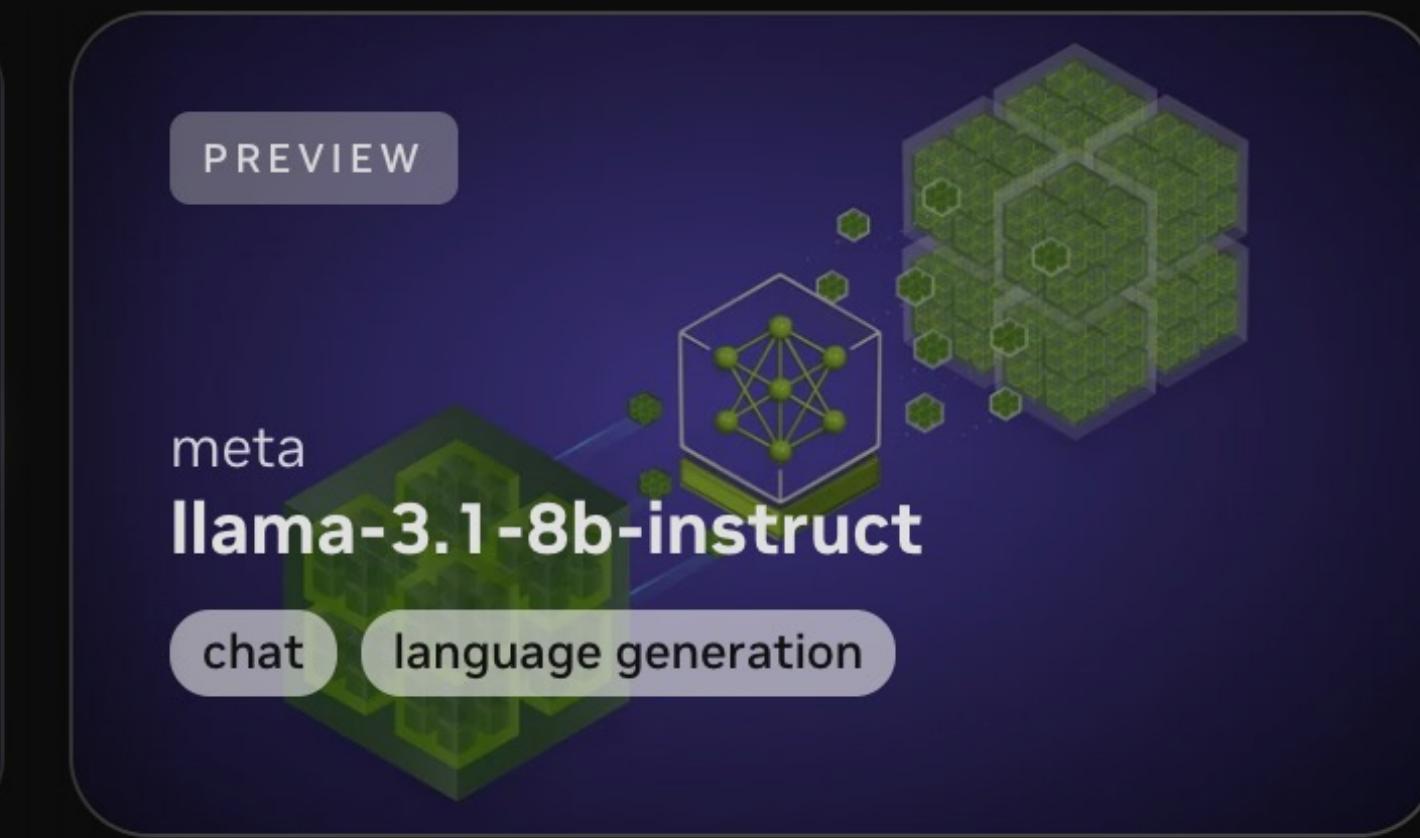
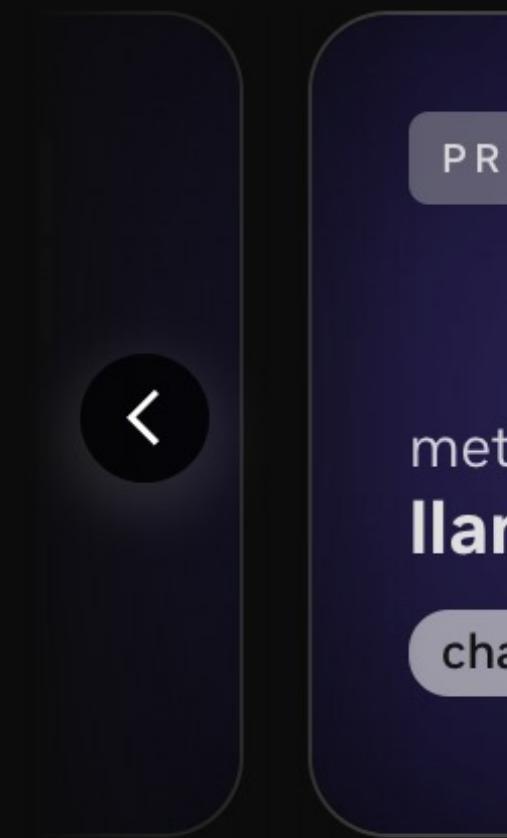
INDUSTRIES

Gaming

Healthcare

Trending Now

The leading open models built by the community, optimized and accelerated by NVIDIA's enterprise-ready inference runtime



Open Full Page

Build with this NIM

AI models generate responses and outputs based on complex algorithms and machine learning techniques, and those responses or outputs may be inaccurate, harmful, biased or indecent. By testing this model, you assume the risk of any harm caused by any response or output of the model. Please do not upload any confidential information or personal data unless expressly permitted. Your use is logged for security purposes.

Preview JSON

Python Langchain Node Shell Docker

Reset Chat

Get API Key Copy Code

```
from openai import OpenAI  
  
client = OpenAI(  
    base_url = "https://integrate.api.nvidia.com/v1",  
    api_key = "$API_KEY_REQUIRED_IF_EXECUTING_OUTSIDE_NGC"
```



Get API Key



Search NVIDIA AI

Explore Docs Login

Discover

MODELS

Reasoning

Vision

Visual Design

Retrieval

Speech

Biology

Simulation

INDUSTRIES

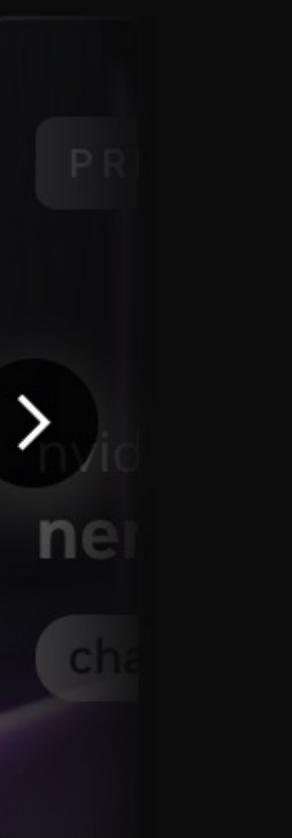
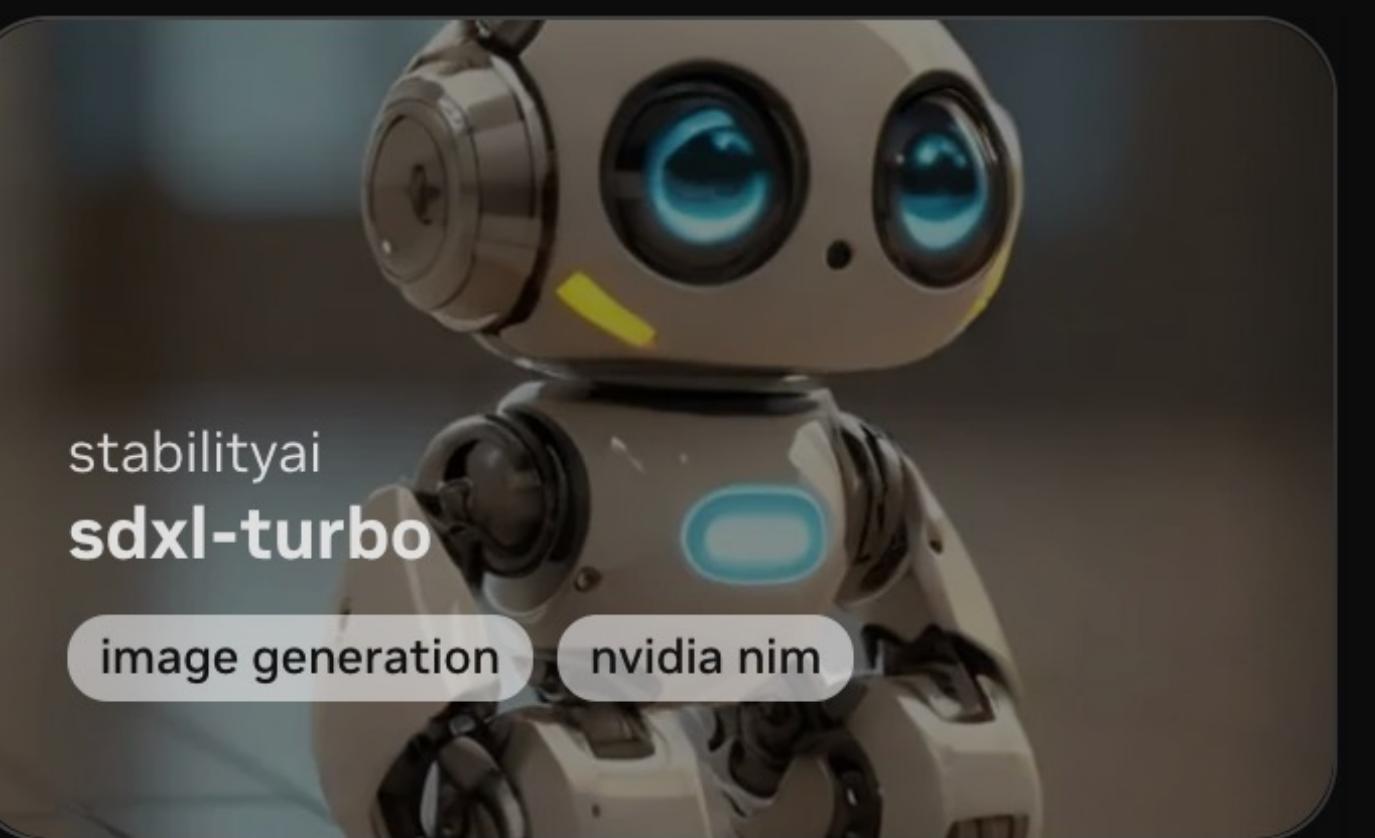
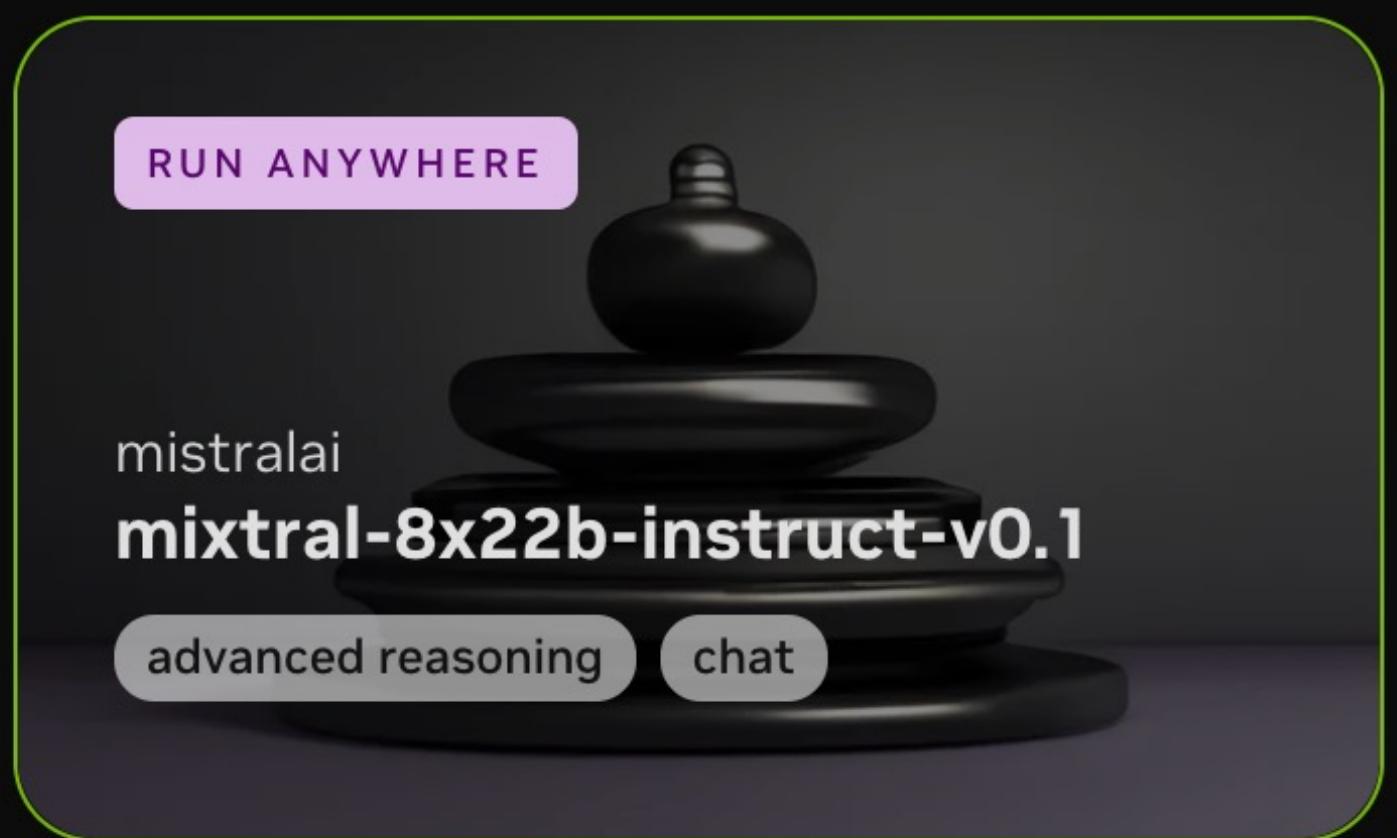
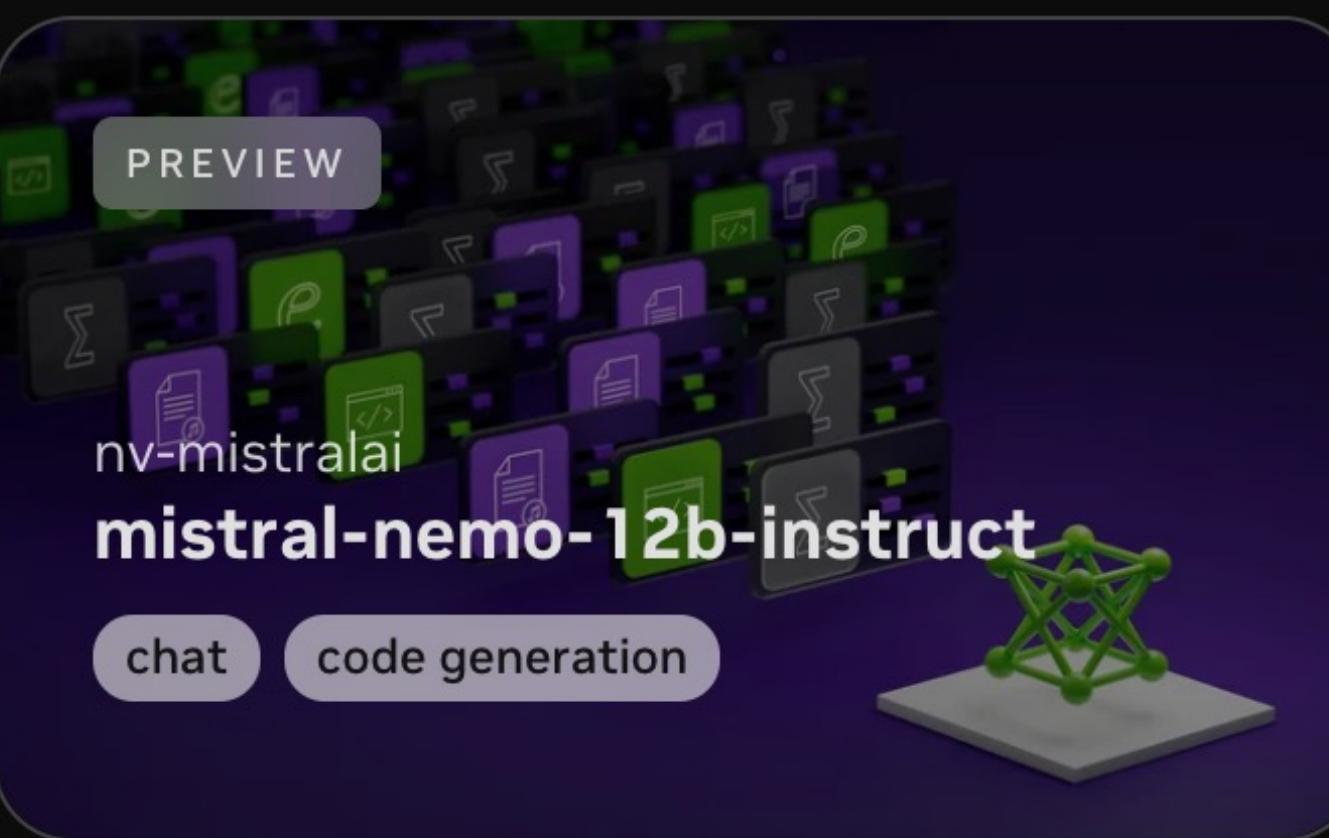
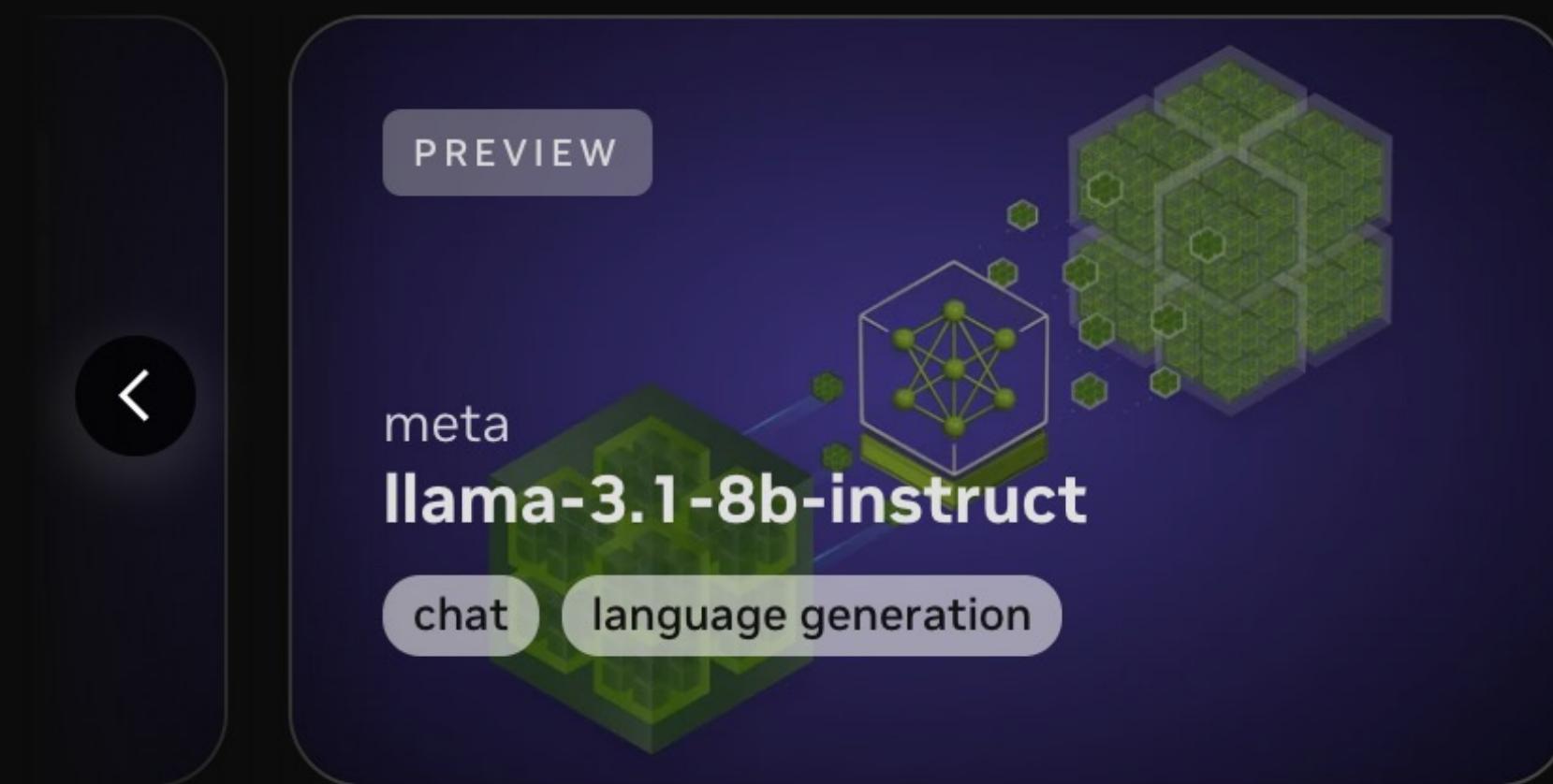
Gaming

Healthcare

Industrial

Trending Now

The leading open models built by the community, optimized and accelerated by NVIDIA's enterprise-ready inference runtime



Open Full Page

Build with this NIM

AI models generate responses and outputs based on complex algorithms and machine learning techniques, and those responses or outputs may be inaccurate, harmful, biased or indecent. By testing this model, you assume the risk of any harm caused by any response or output of the model. Please do not upload any confidential information or personal data unless expressly permitted. Your use is logged for security purposes.

Preview JSON

Python Langchain Node Shell Docker

Reset Chat

Get API Key Copy Code

```
from openai import OpenAI  
  
client = OpenAI(  
    base_url = "https://integrate.api.nvidia.com/v1",  
    api_key = "$API_KEY_REQUIRED_IF_EXECUTING_OUTSIDE_NGC"
```



Enter your NGC account



Search NVIDIA AI

Explore Docs Login

Discover

MODELS

Reasoning

Vision

Visual Design

Retrieval

Speech

Biology

Simulation

INDUSTRIES

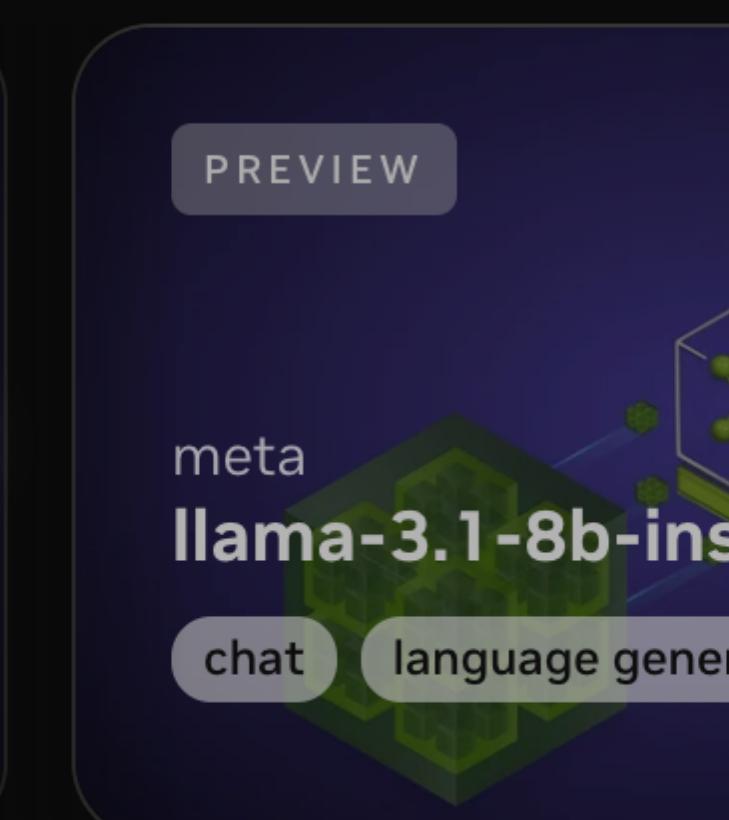
Gaming

Healthcare

Industrial

Trending Now

The leading open models built by NVIDIA



AI models generate responses at no cost. You are responsible for the risk of any harm caused by any response generated by an AI model.

Preview JSON

Login to start building with NVIDIA NIM

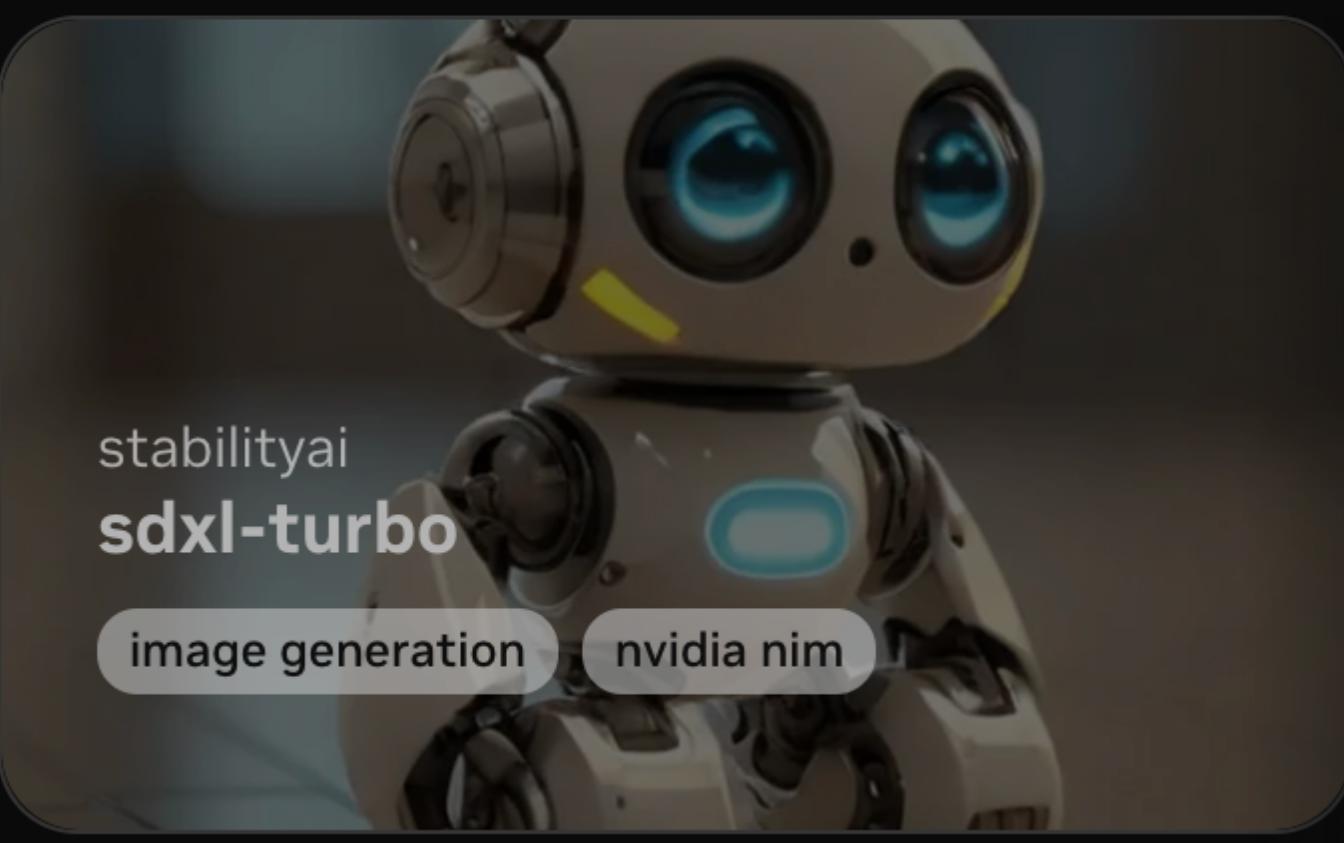
Get Started with your free API credits

Use a Business Email

- ✓ 5000 API credits to use any NIM
- ✓ Self-host NIMs on your infrastructure
- ✓ Enterprise-grade support for 90 days
- ✓ Security and vulnerability reports

Use a Personal Email

- ✓ 1000 API credits to use any NIM
- ✓ Self-host NIMs on your infrastructure
- ✓ Community support via developer forums
- ✗ Security and vulnerability reports



Open Full Page

Build with this NIM

Enter your email ID

Next

By proceeding, I agree to the [NVIDIA Technology Access Terms of Use](#)

Reset Chat

Get API Key Copy Code

```
from openai import OpenAI  
  
client = OpenAI(  
    base_url = "https://integrate.api.nvidia.com/v1",  
    api_key = "$API_KEY_REQUIRED_IF_EXECUTING_OUTSIDE_NGC"
```



Generate Key



Search NVIDIA AI

Explore

Docs



Discover

MODELS

Reasoning

Vision

Visual Design

Retrieval

Speech

Biology

Simulation

INDUSTRIES

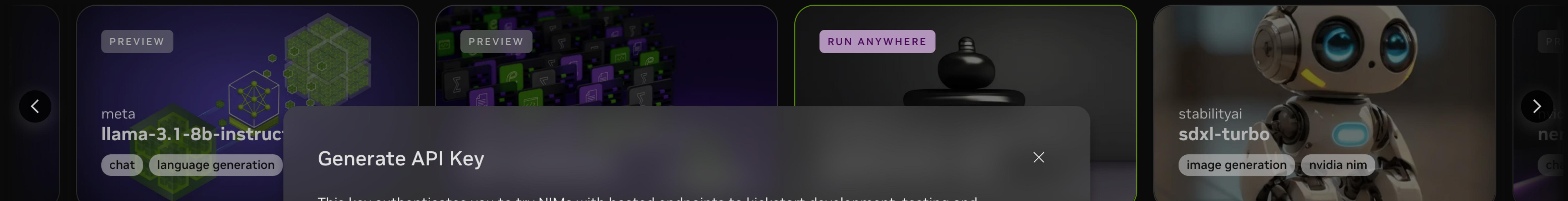
Gaming

Healthcare

Industrial

Trending Now

The leading open models built by the community, optimized and accelerated by NVIDIA's enterprise-ready inference runtime



Generate API Key

This key authenticates you to try NIMs with hosted endpoints to kickstart development, testing and integration.

Cancel

Generate Key

Open Full Page

Build with this NIM

AI models generate responses and outputs that may contain harmful, biased or indecent language. You assume the risk of any harm caused by any response or output of the model. Please do not upload any confidential information or personal data unless expressly permitted. Your use is logged for security purposes.

Preview JSON

Python Langchain Node Shell Docker

Reset Chat

Get API Key Copy Code

```
from openai import OpenAI  
  
client = OpenAI(  
    base_url = "https://integrate.api.nvidia.com/v1",  
    api_key = "$API KEY REQUIRED IF EXECUTING OUTSIDE NGC"
```



Copy API Key



Search NVIDIA AI

Explore Docs

Discover

MODELS

Reasoning

Vision

Visual Design

Retrieval

Speech

Biology

Simulation

INDUSTRIES

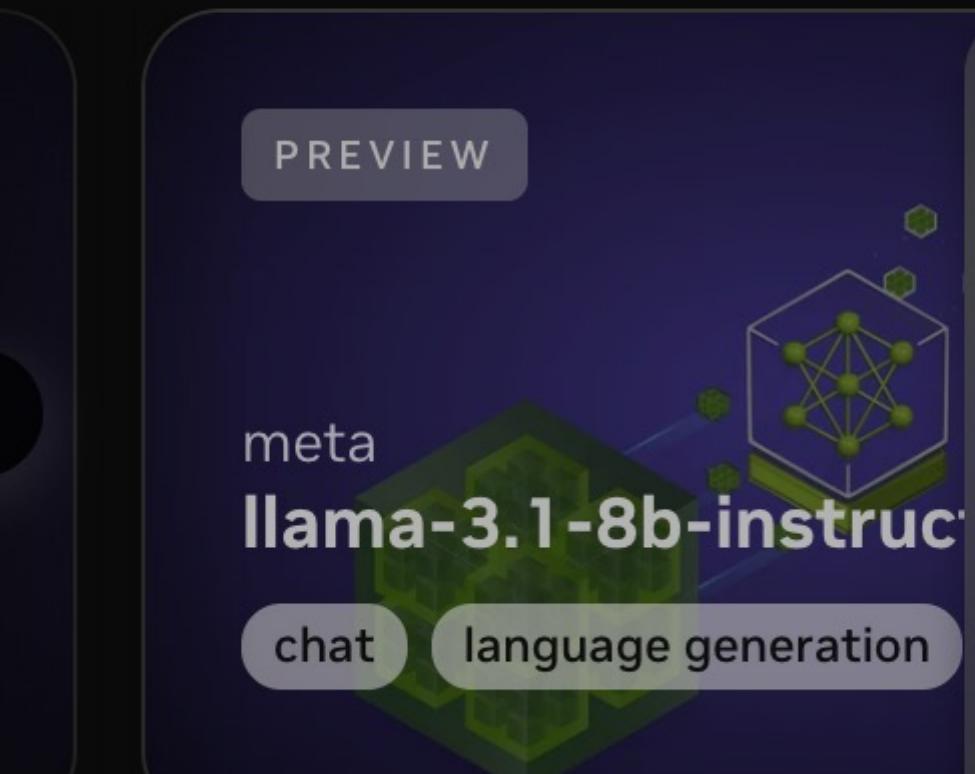
Gaming

Healthcare

Industrial

Trending Now

The leading open models built by the community, optimized and accelerated by NVIDIA's enterprise-ready inference runtime



Key Generated

The following key has been successfully generated. This is the only time your key will be displayed. This key is for API testing use only and is valid for 1 year.

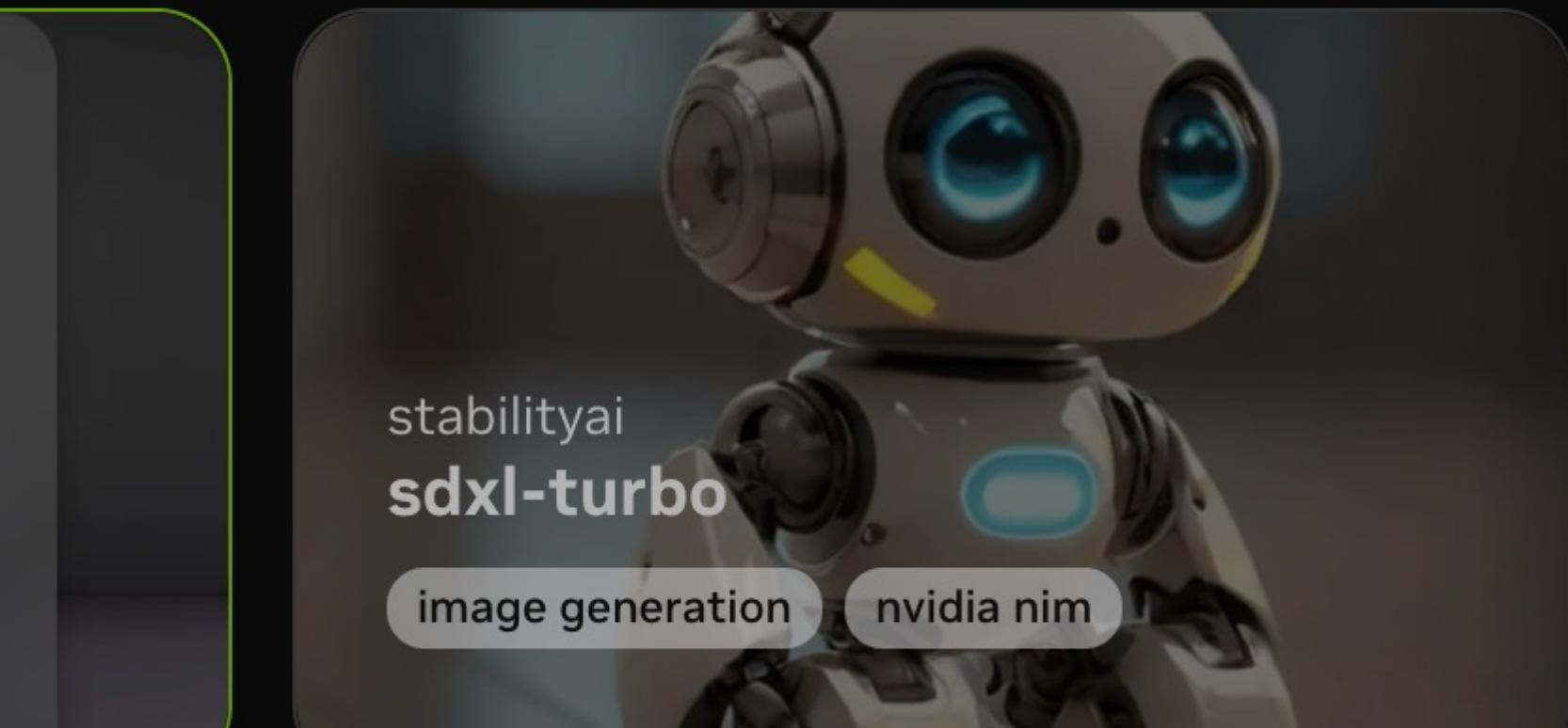
ID
7ba884bb-616c-4795-ba33-97bd0959b54f

Name
__ autogenerated_playgrounds_0505300891762545

Expiration
08/07/2025

nvapi-mT6N

Keep your key a secret. Do not share it or store it in a place where others can see or copy it.



Open Full Page

Build with this NIM

Keep your key a secret. Do not share it or store it in a place where others can see or copy it. Harmful, biased or indecent. By testing this model, you assume the risk of any harm caused by any response. Our use is logged for security purposes.

Close

Copy Key

Python Langchain Node Shell Docker

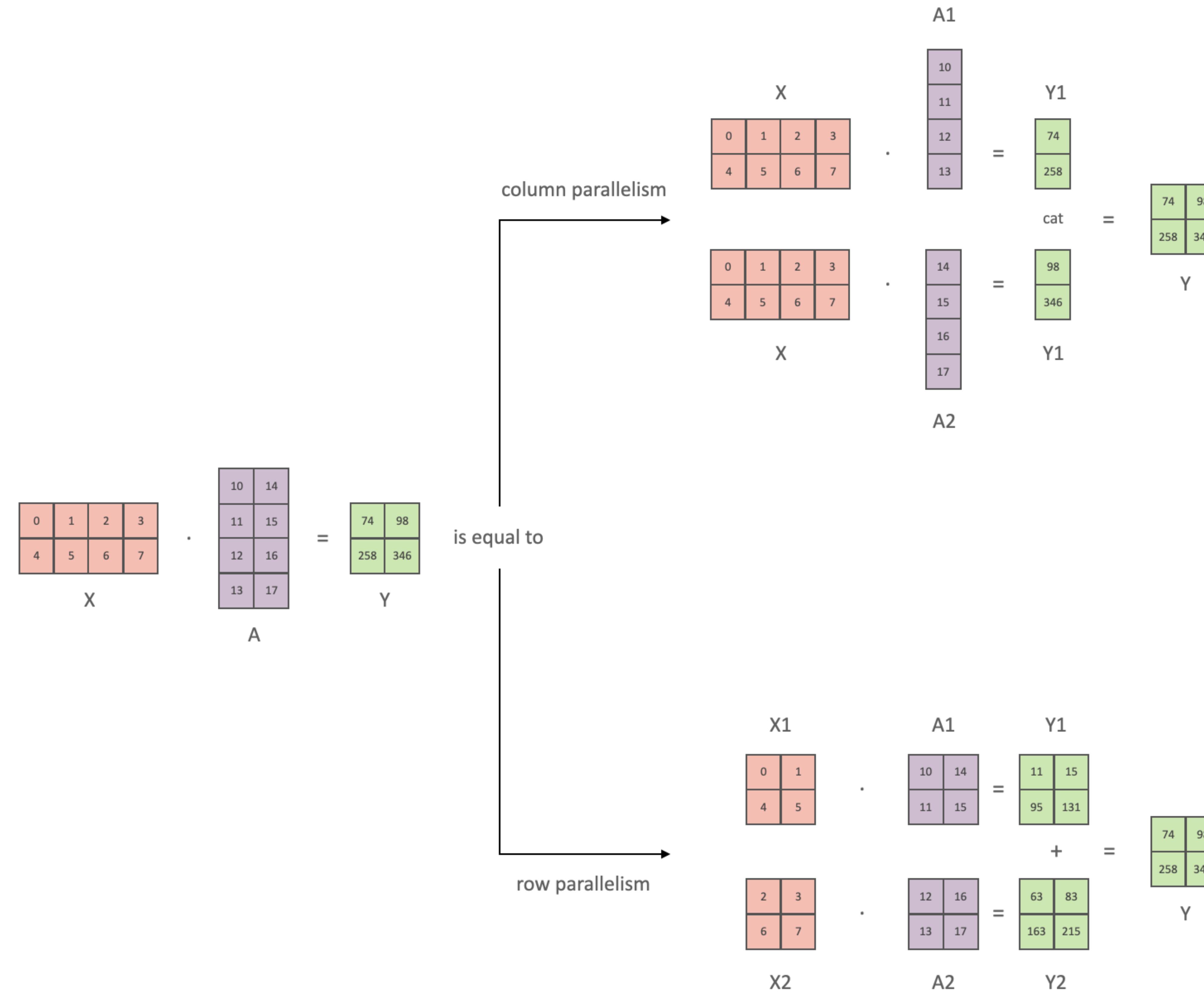
```
from openai import OpenAI
client = OpenAI(
    base_url = "https://integrate.api.nvidia.com/v1",
    api_key = "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"
```

Get API Key Copy Code

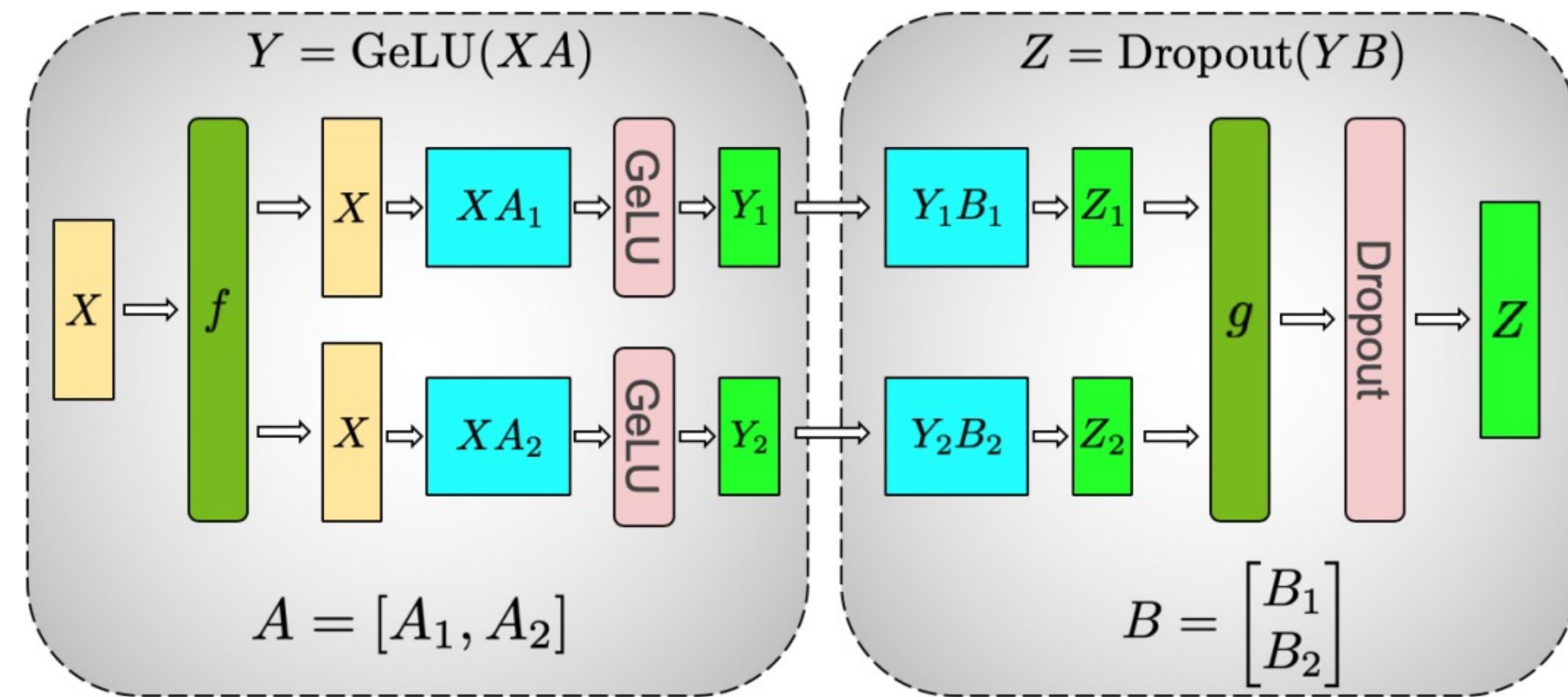


Appendix

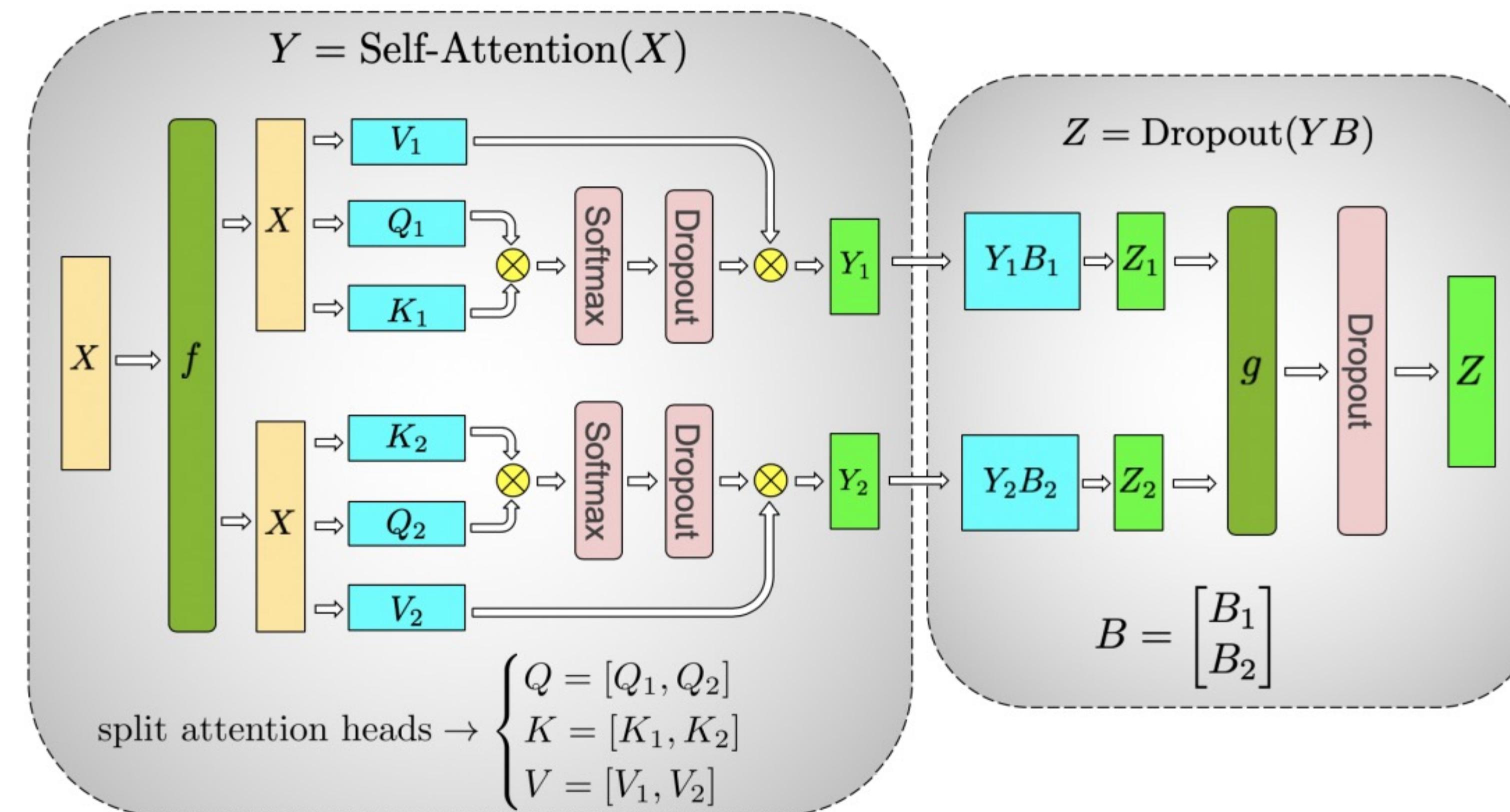
Tensor Parallelism



Tensor Parallelism



(a) MLP



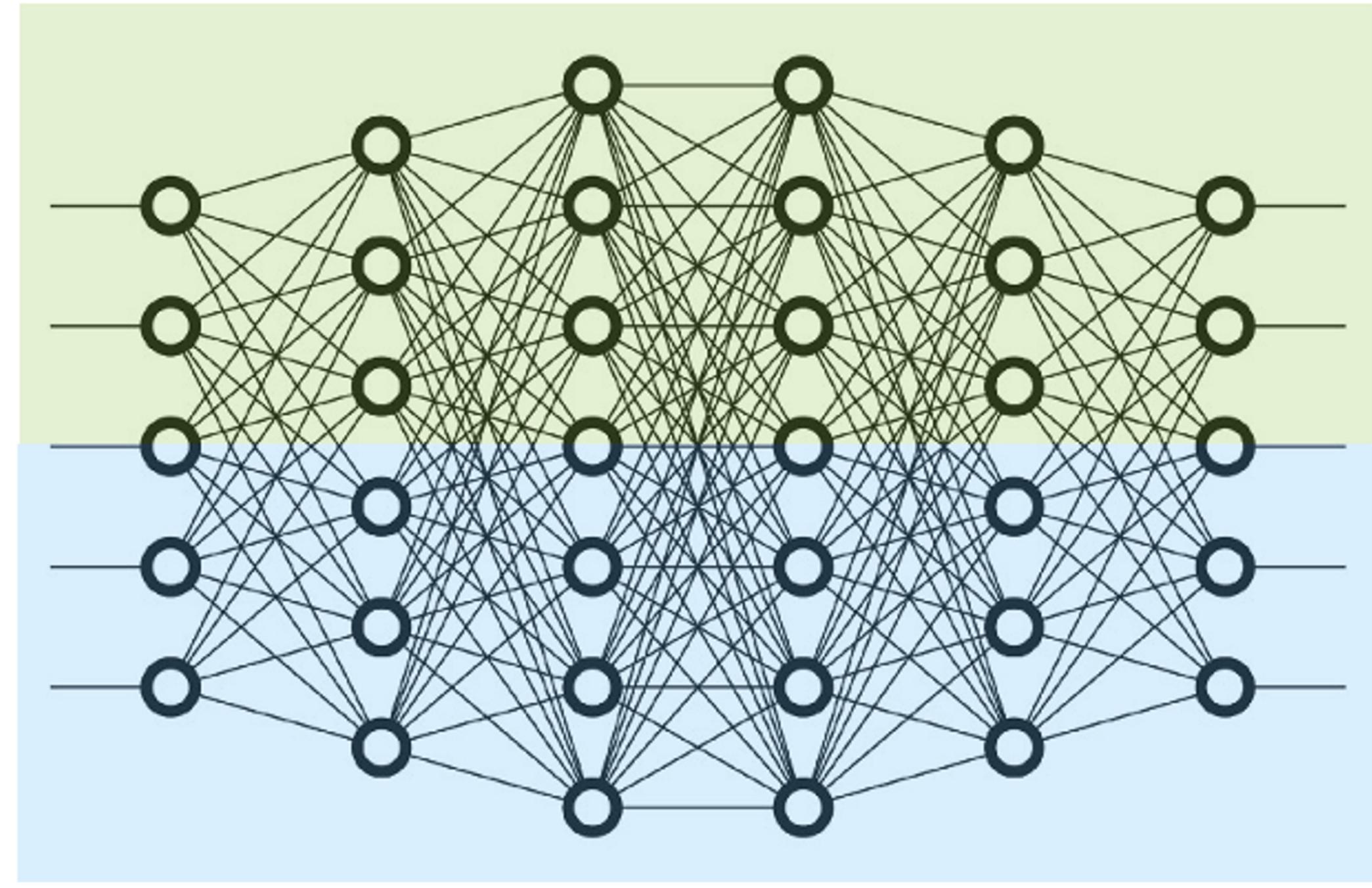
(b) Self-Attention

Model parallelism

Tensor and Pipeline parallelism

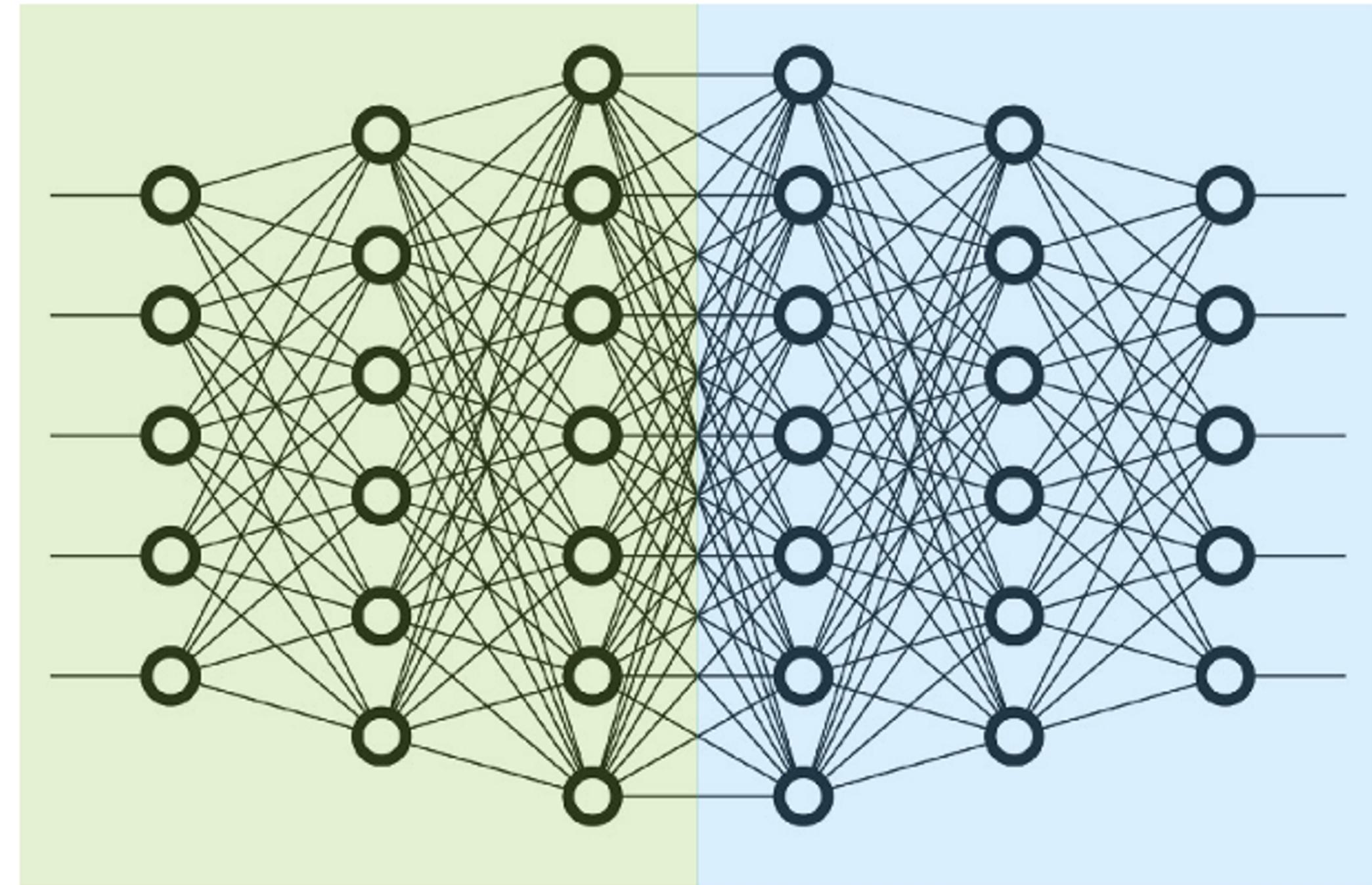
- **Tensor (Intra-Layer) Parallelism**

- Split individual layers across multiple GPUs
- Both devices compute different parts of Layers 0,1,2,3,4,5
- Need high communication bandwidth



- **Pipeline (Inter-Layer) Parallelism**

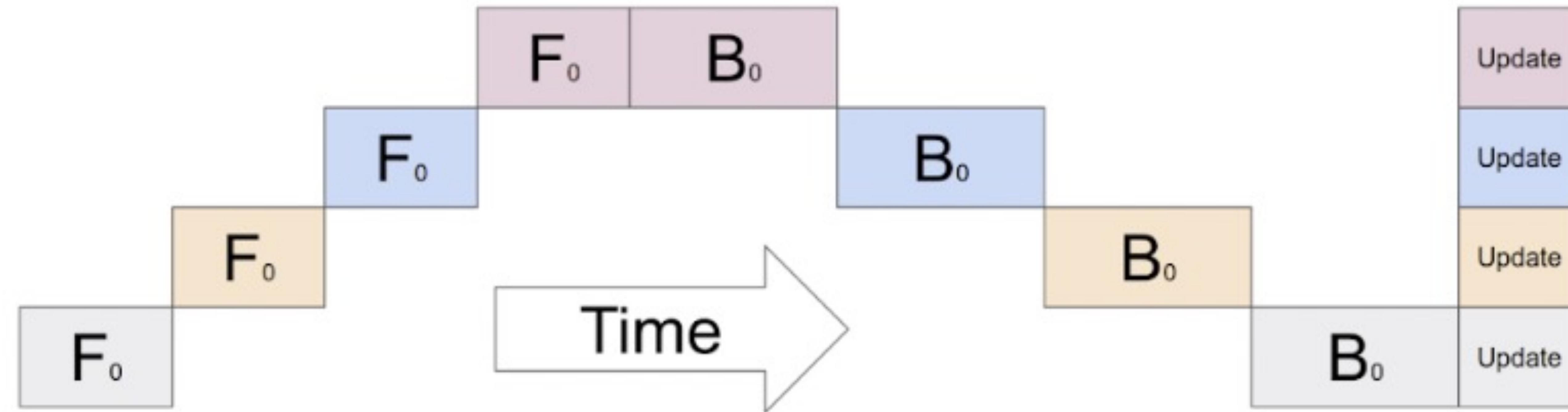
- Split contiguous sets of layers across multiple GPUs
- Layers 0,1,2 and layers 3,4,5 are on different GPUs



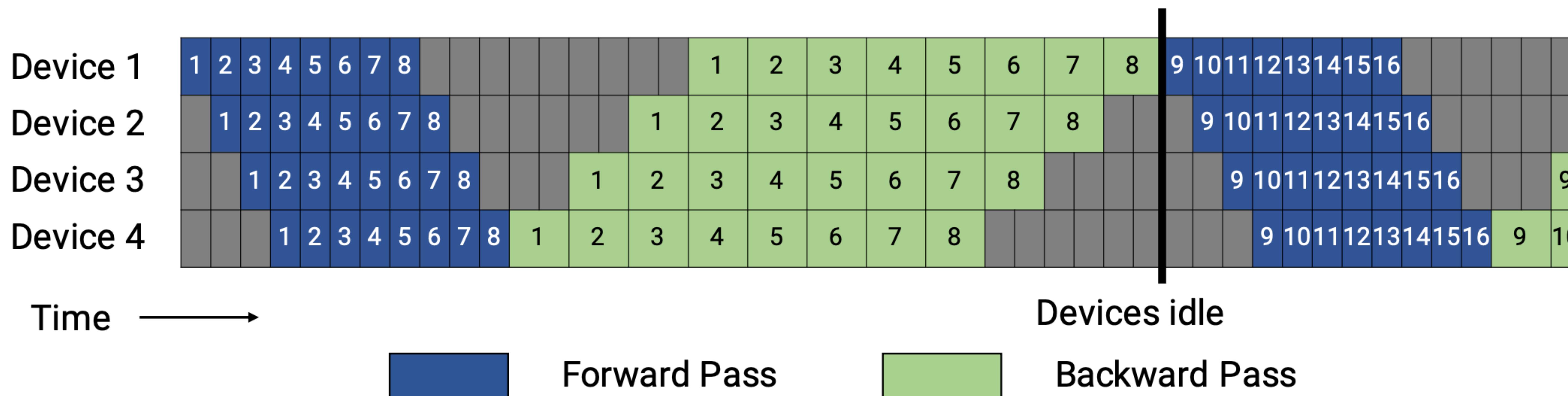
Pipeline Parallelism

Native and Gpipe pipeline schedules

Native



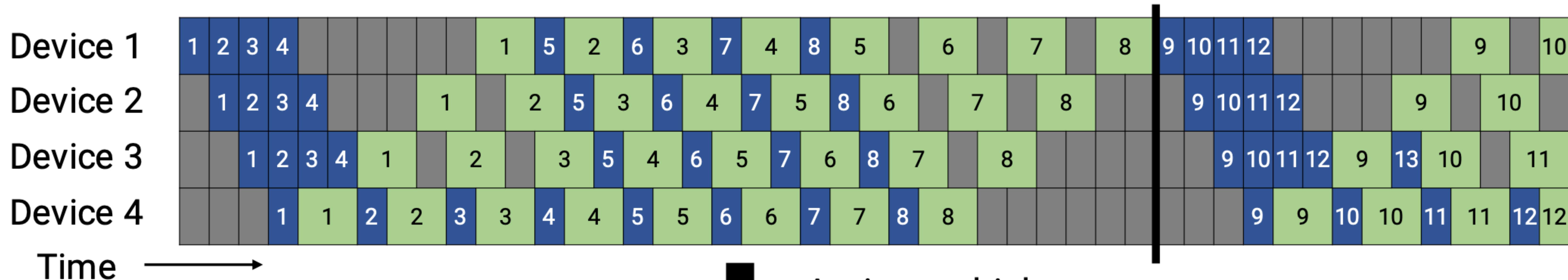
GPipe



Pipeline Parallelism

Default and interleaved 1F1B pipeline schedules

Default 1F1B



Interleaved

