AWS re:Invent

DECEMBER 2 – 6, 2024 | LAS VEGAS, NV

CMP 333

Introducing Amazon Trn2 Instances, featuring Trainium2

Joe Senerchia
Sr Product Manager
Amazon

James Bradbury
Distinguished Engineer, Compute
Anthropic

Ron Diamant
Sr Principal Engineer
Amazon

Agenda

AWS AI Infrastructure and AI Chips

Building Trainium2

Scaling Frontier Models with Anthropic

“

Generative AI may be the largest technology transformation since the cloud, and perhaps since the Internet.

Andy Jassy

Amazon CEO

AWS Generative AI Stack

APPLICATIONS TO BOOST PRODUCTIVITY

Amazon Q Business
INSIGHTS AND AUTOMATION

Amazon Q Developer
SOFTWARE DEVELOPMENT LIFECYCLE

MODELS AND TOOLS TO BUILD GENERATIVE AI APPS

Amazon Bedrock
AMAZON MODELS | PARTNER MODELS

INFRASTRUCTURE TO BUILD AND TRAIN AI MODELS

Amazon SageMaker
MANAGED INFRASTRUCTURE

AWS Trainium
AWS Inferentia
GPUs
HIGH PERFORMANCE COMPUTE

AWS Infrastructure for AI

ML Frameworks & Libraries

PyTorch

JAX

OpenXLA

Hugging Face

Orchestration & Management

Amazon EKS

Amazon ECS

AWS Parallel Cluster

DL Containers and AMIs

Amazon SageMaker

Storage

Amazon S3

Amazon FSx for Lustre

Amazon EFS

Amazon EBS

Networking

Elastic Fabric Adapter

EC2 UltraCluster

COMPUTE

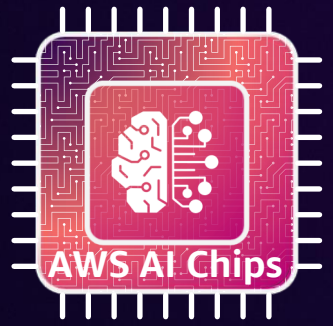
Trainium

Inferentia

GPUs

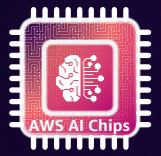
Innovating at the silicon level

AWS Trainium



AWS Inferentia

AWS Trainium



AWS Inferentia

Inf1 Instances

Lowest cost per inference in the cloud for running deep learning models

Inf2 Instances

High performance at the lowest cost per inference for LLMs and diffusion models

Trn1 Instances

Cost-efficient, high-performance training of LLMs and diffusion models

Trn2 Instances

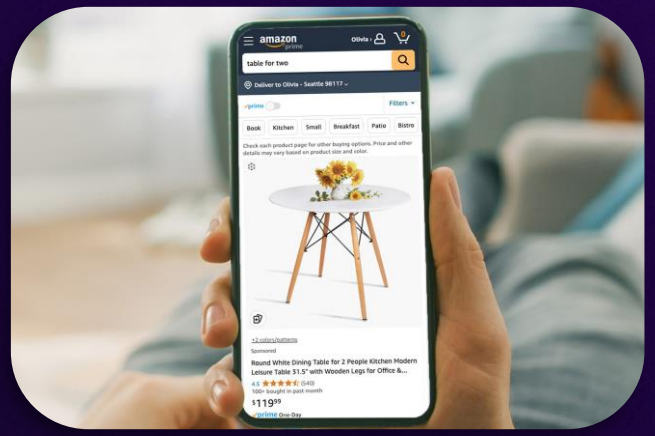
NEW

Highest performing EC2 instances for deep learning and generative AI

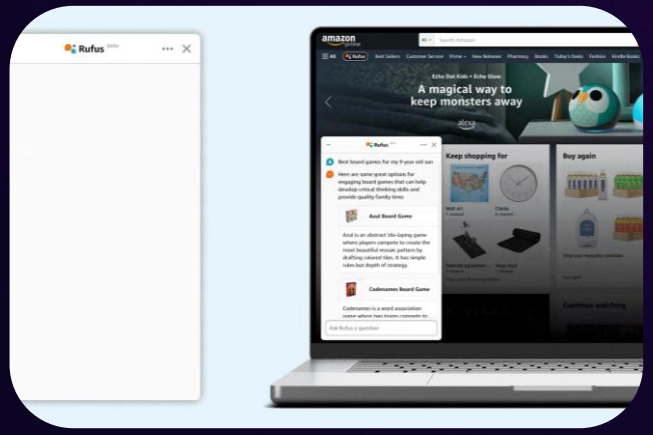
AWS AI Chips

Powering AI Innovation at Amazon

Alexa
conversational AI



Search
on Amazon



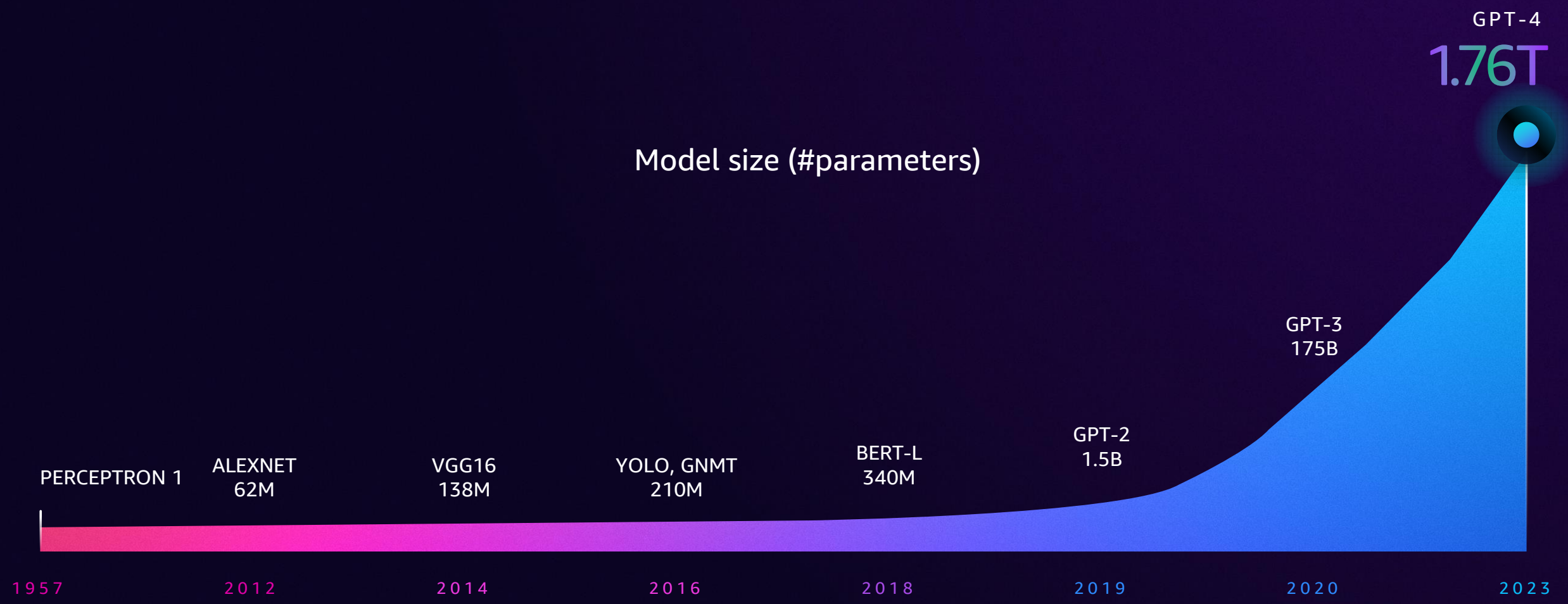
Rufus
AI shopping assistant

AWS Trainium and Inferentia customers

ANTHROP\C
databricks
poolside
Luma
NinjaTech AI
arcee.ai
IBM
RICOH imagine. change.
Stockmark
brave
ELYZA
Itaú
mimecast
8090
refact.ai
amazon
KARAKURI

PyTorch
OpenXLA
Hugging Face
Lightning AI
Creators of PyTorch Lightning
Weights & Biases
Domino
Anyscale
DATADOG
scale
Outerbounds

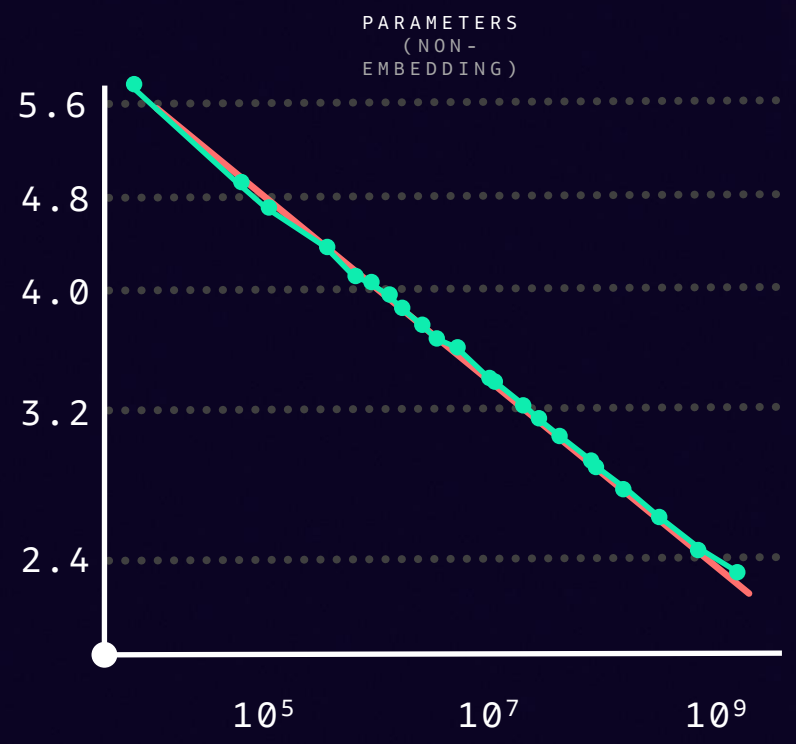
Model scaling . . . what's next?

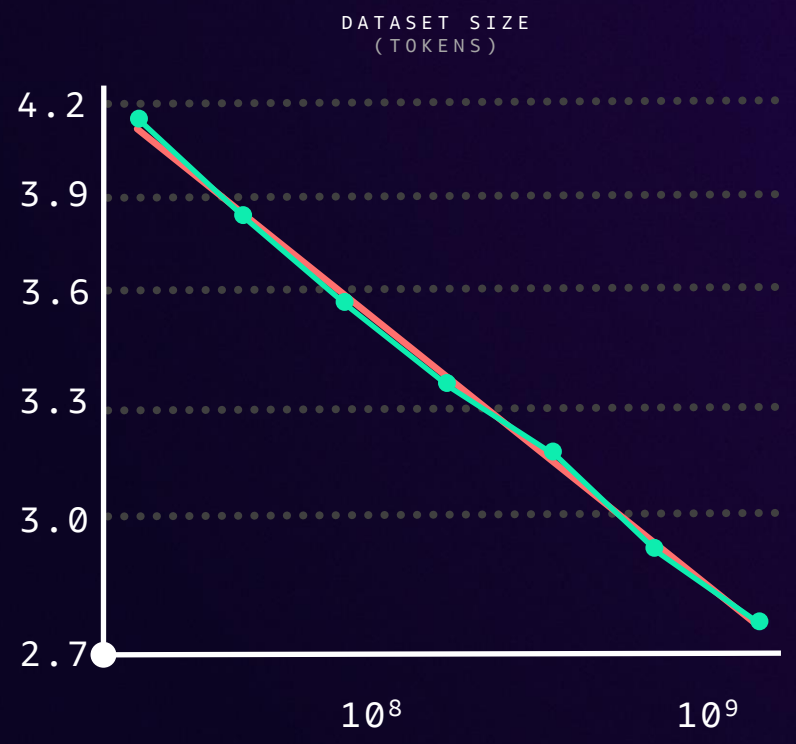


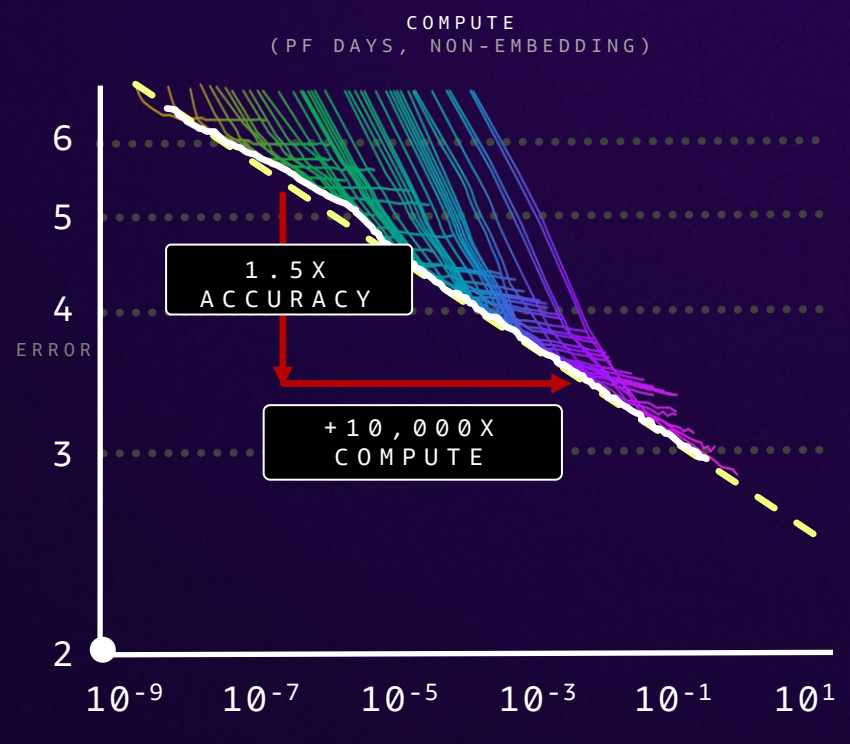
Why keep scaling?

- Scale improves overall intelligence
- Scale unlocks new capabilities
- Predictable improvement in loss through scaling

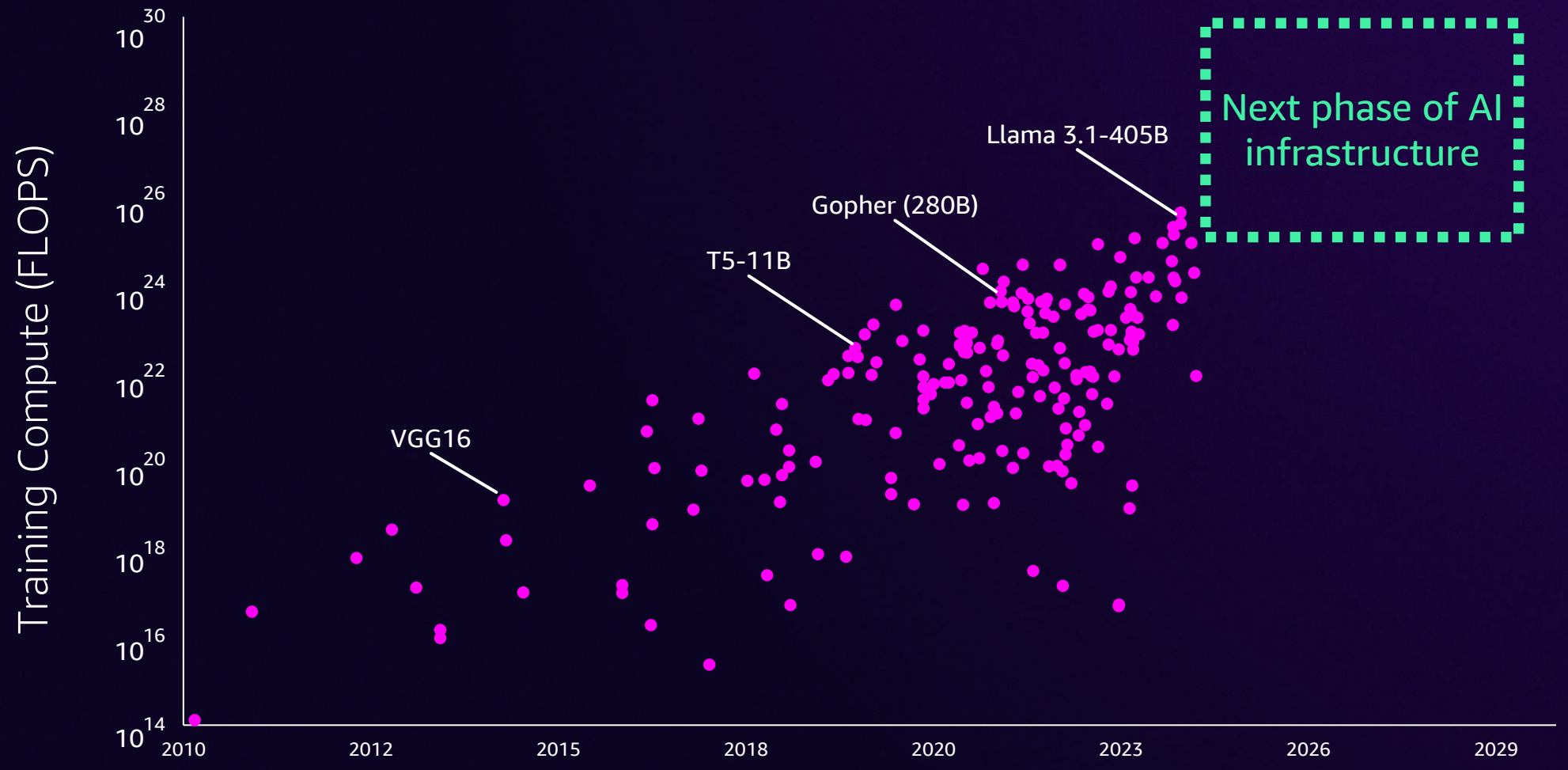
Why keep scaling?







Scaling compute requires next-generation AI infrastructure



Source: Epoch AI, 'Data on Notable AI Models'. 26 Nov 2024

AVAILABLE NOW

Amazon EC2 Trn2 Instances

The highest performing EC2 instances for deep learning and generative AI

30% more compute and 25% more HBM than the next most powerful EC2 instance, at a lower price

60K Trainium2 chips in a non-blocking, petabit scale UltraClusters

1T+ parameter generative AI model training and inference

Amazon EC2 Trn2 instances powered by AWS Trainium2

THE HIGHEST PERFORMING EC2 INSTANCES FOR DEEP LEARNING AND GENERATIVE AI

HIGH PERFORMANCE

training and inference of trillion+ parameter Generative AI models

BEST PRICE-PERF

for generative AI and deep learning on AWS

UP TO 46 TB/s

of HBM Bandwidth, ideal for memory intensive token generation

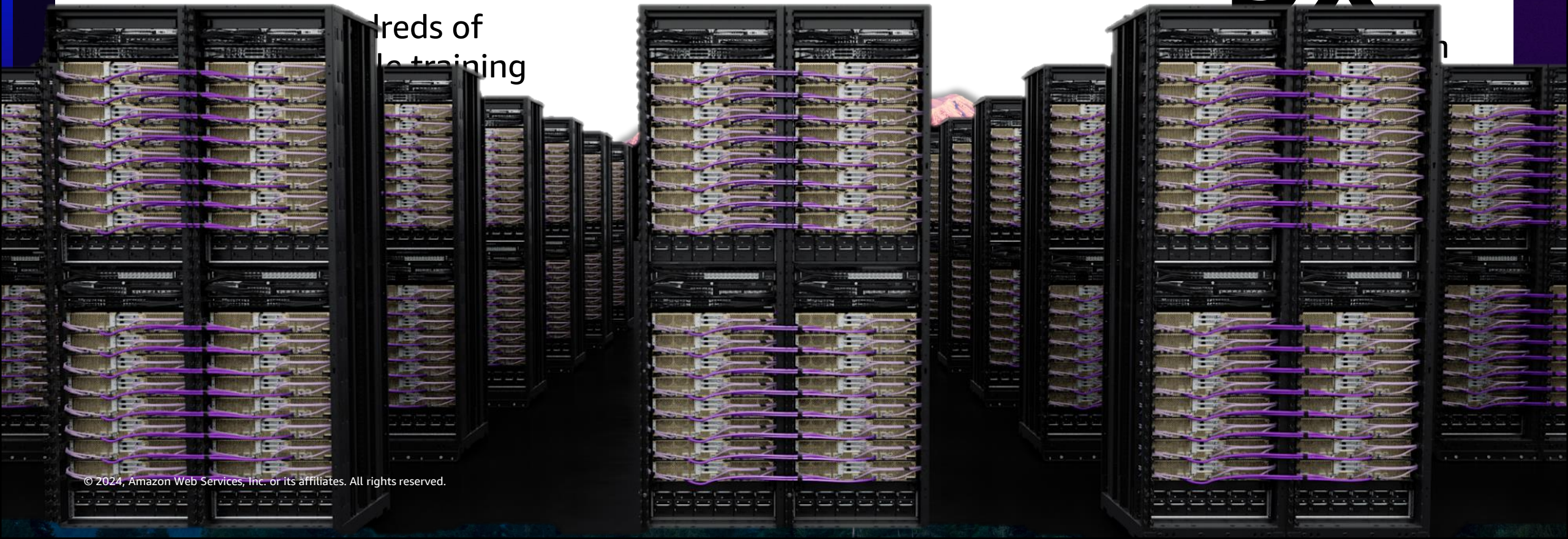
Instance size	Trainium2 chips	Chip memory	Chip Memory Bandwidth	vCPUs	Instance Memory	Storage	NeuronLink	EFAv3	Capacity Block Price	3Yr RI Price
trn2.48xlarge	16	1.5TB	46 TB/s	192	2TB	4x 1.92TB NVMe	1 TB/s	3.2 Tb/s	$44.70/hr	$34.39/hr

Project Rainier

amazon | ANTHROP\C

Project Rainier **is hundreds of thousands** of AWS Trainium2
dreds of
le training

5x



Building Trainium2

Our most powerful server for machine learning training

High performance

Cost Efficiency

Scale

Ease of use

Innovation

High performance

Cost efficiency

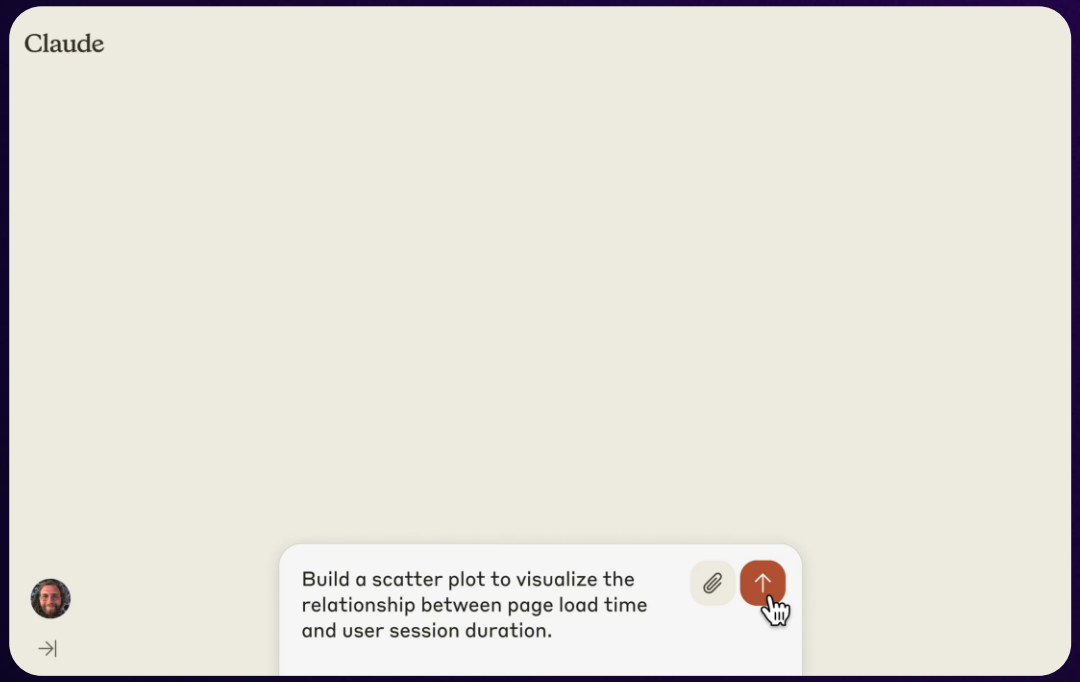
Scale

Ease of use

Innovation

What drives performance?

TFLOPS



What drives performance?

TFLOPS

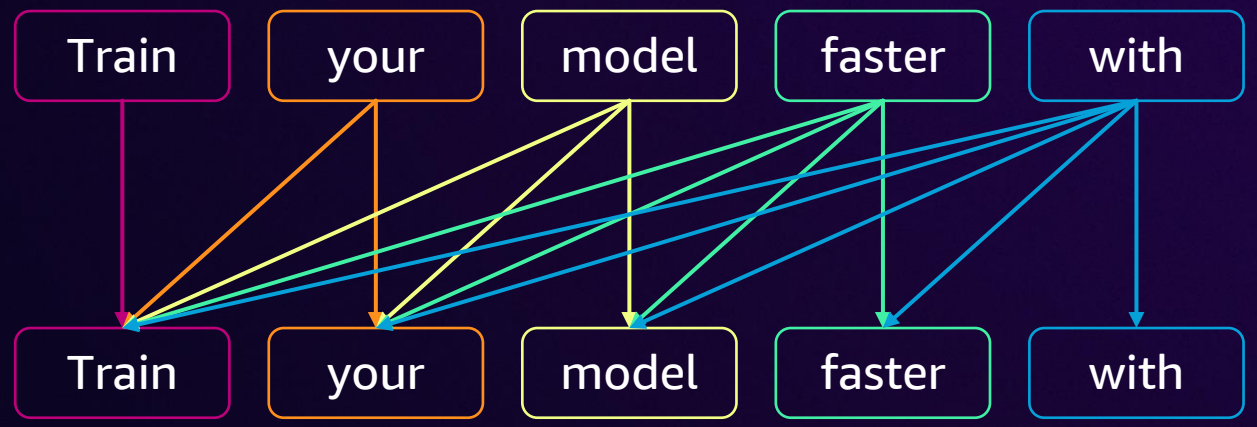
Input processing

Train | your | model | faster | with

What drives performance?

TFLOPS ✓

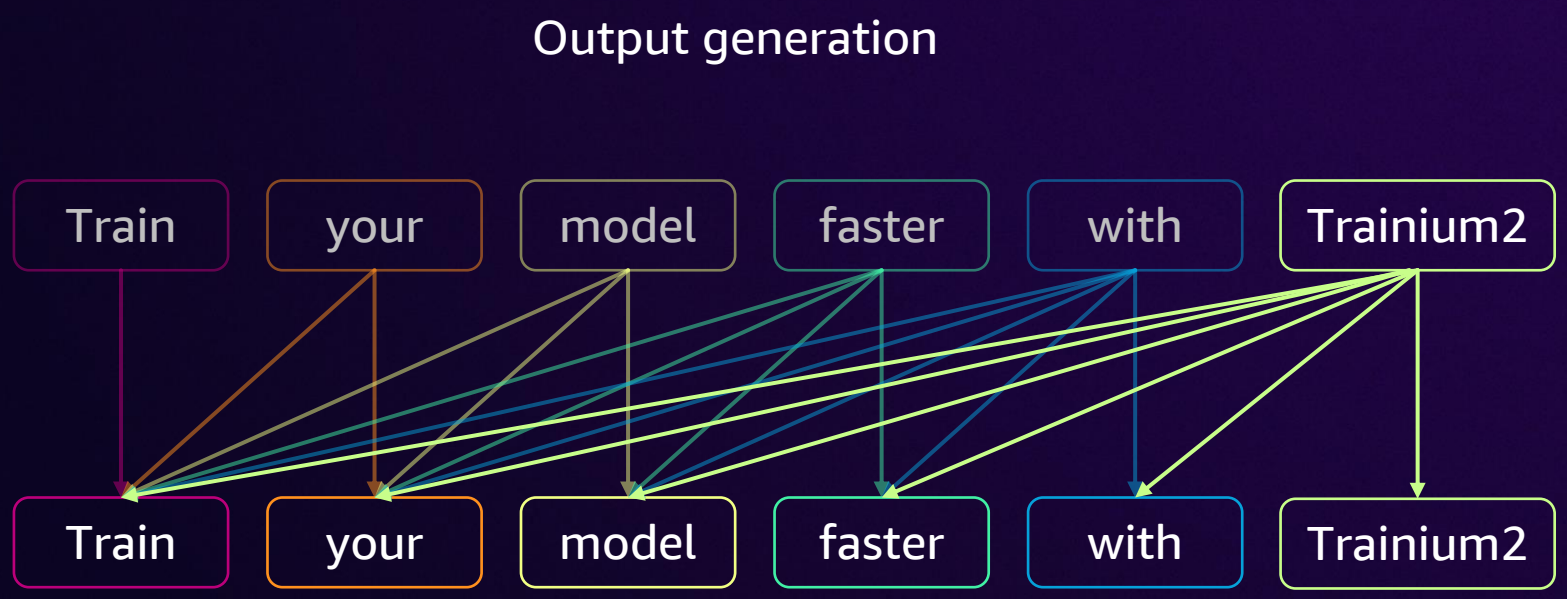
Input processing



What drives performance?

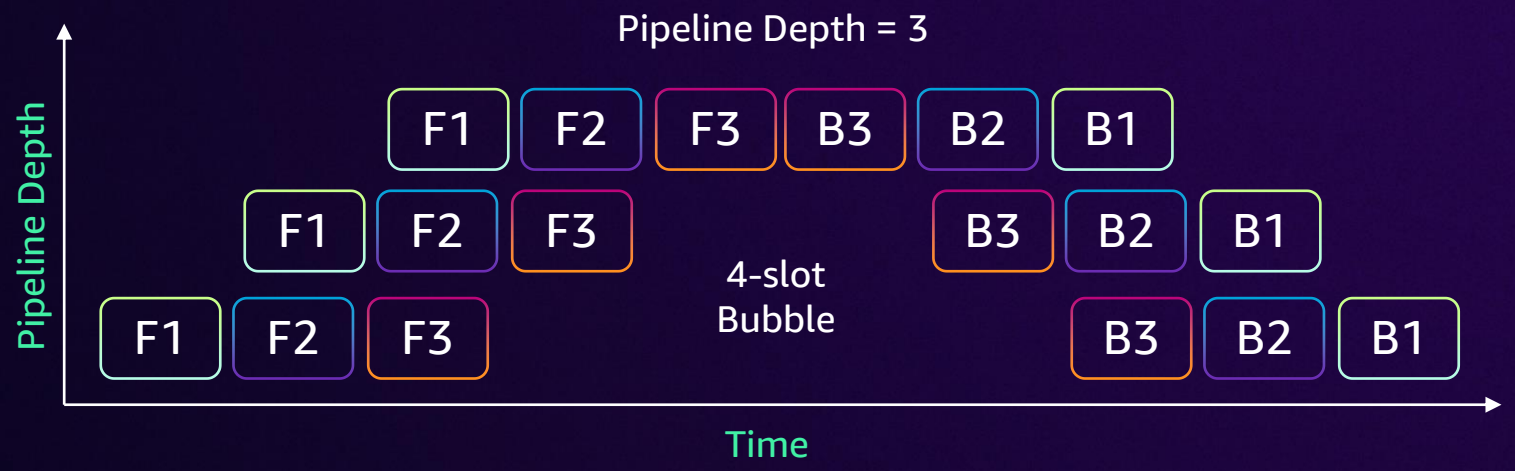
TFLOPS

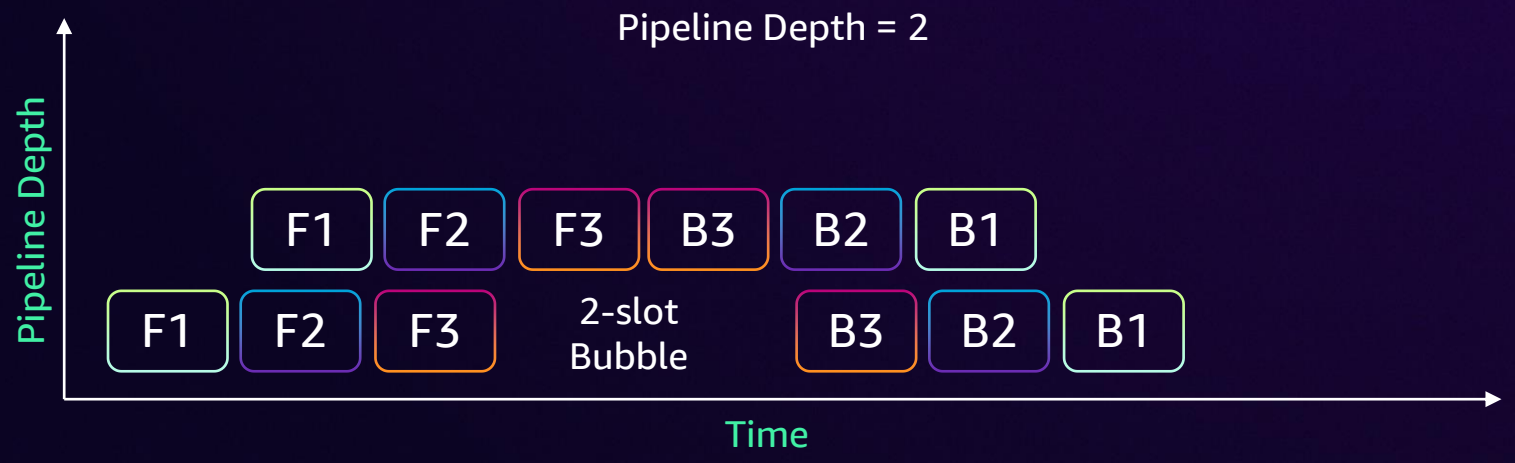
Memory bandwidth ✓



What drives performance?

- TFLOPS
- Memory bandwidth
- Memory capacity ✓





What drives performance?

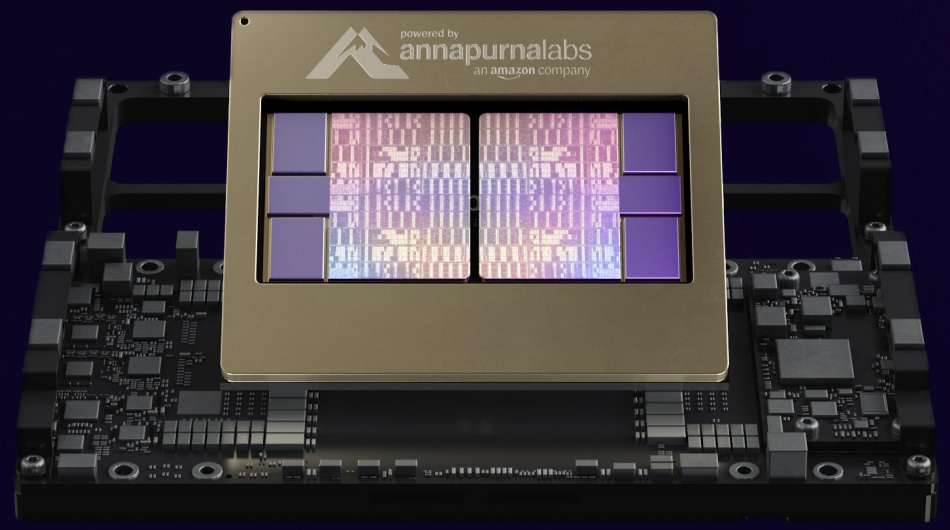
- TFLOPS
- Memory bandwidth
- Memory capacity
- Interconnect ✓

10p10u

AWS designed network fabric delivering 10s of petabits under 10 microseconds

AWS
Trainium2

Third-generation chip purpose built for generative AI and ML training



1.3 PFLOPS
Dense compute

5.2 PFLOPS
Sparse compute

96 GB
HBM capacity

2.9 TB/s
HBM bandwidth

Trainium2 Server

Our most powerful server for ML training

20.8
PFLOPS
DENSE COMPUTE

83.2
PFLOPS
SPARSE COMPUTE

46
TB/s
HBM BANDWIDTH

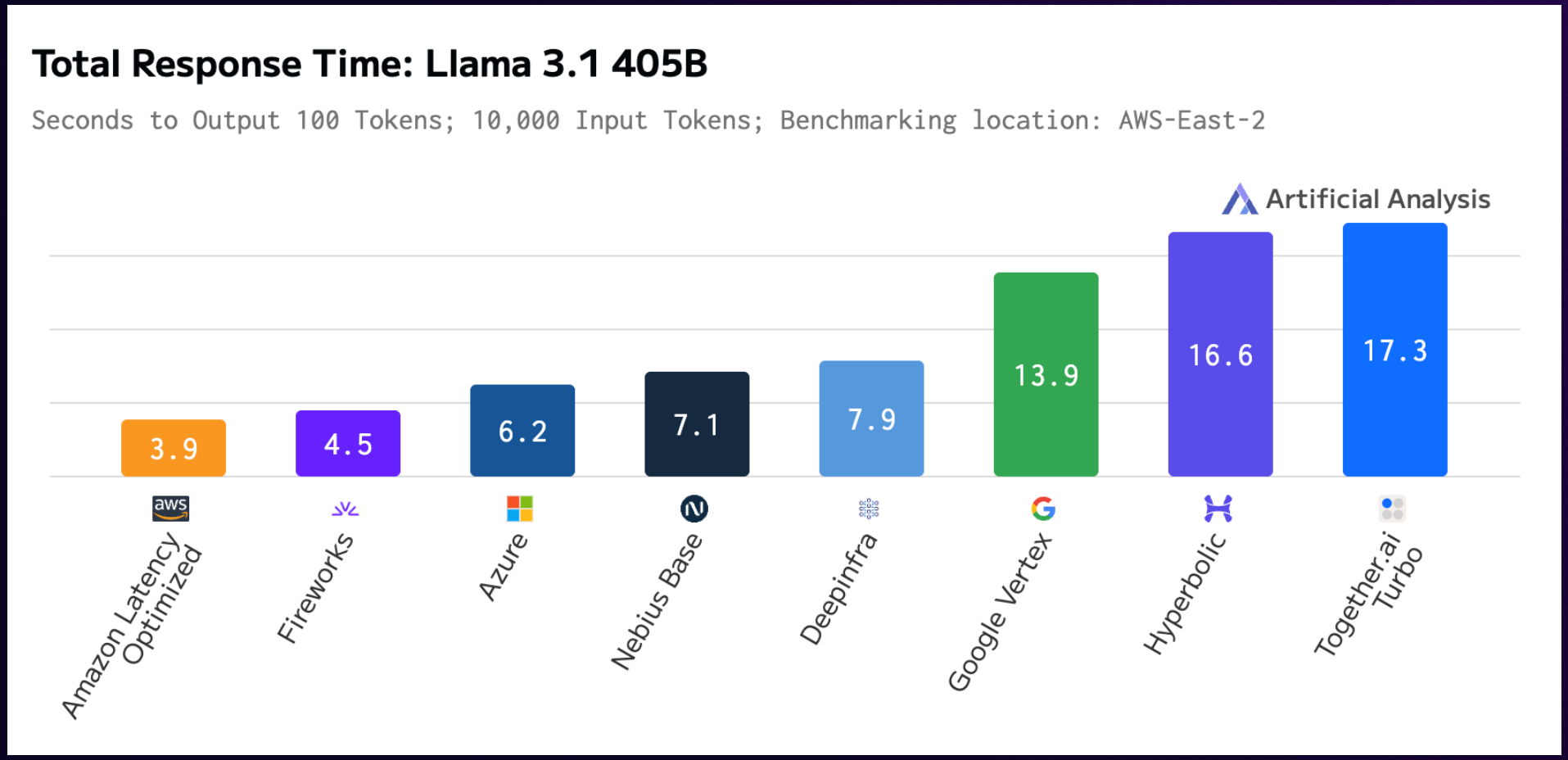
1.5
TB
HBM CAPACITY

1
TB/s
NeuronLink Bandwidth

3.2
Tb/s
EFAv3

LLM inference performance

LLAMA 3.1 405B ON TRN2

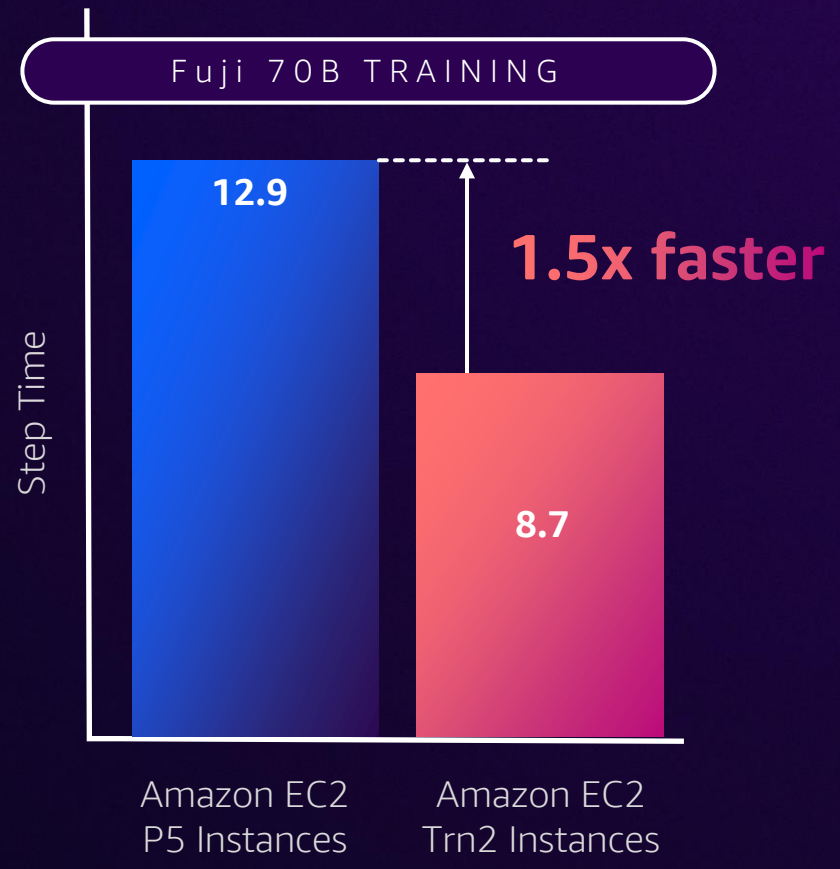


LLM training performance

TRAIN FASTER WITH TRAINIUM2

1.7x

Lower cost-to-train



JAX/AXLearn framework, 64 node cluster

AWS Trainium2 **UltraServer**

2 racks, 4 servers, connected together via NeuronLink-v3
providing the highest performance ML server in EC2

83.2
PFLOPS
DENSE COMPUTE

332.8
PFLOPS
SPARSE COMPUTE

2
TB/s
NEURONLINK BANDWIDTH

6
TB
HBM CAPACITY

185
TB/s
HBM BANDWIDTH

12.8
Tb/s
EFAv3 BANDWIDTH

High performance

Cost efficiency

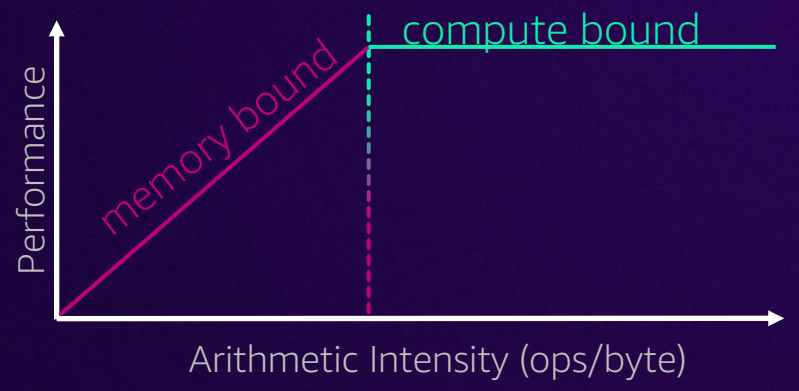
Scale

Ease of use

Innovation

Energy efficient compute

SYSTOLIC ARRAYS FOR OPTIMIZED ARITHMETIC INTENSITY

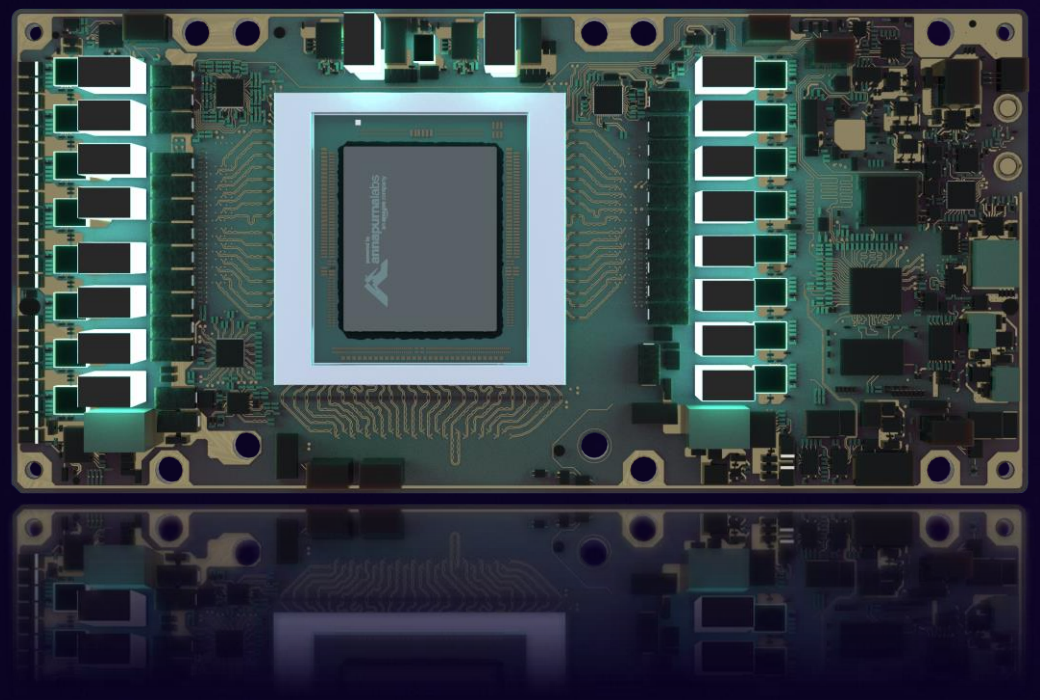


- Parallel processing
- Local communication
- Data reuse

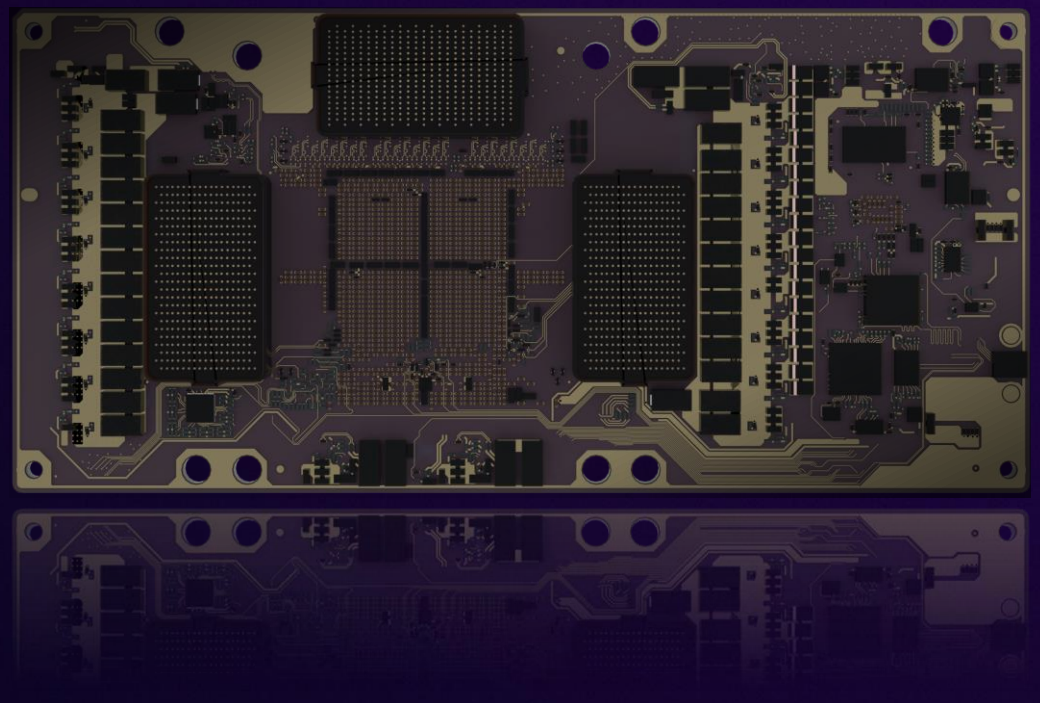
Optimized power delivery

TRAINIUM1 – LATERAL POWER DELIVERY

TOP



BOTTOM

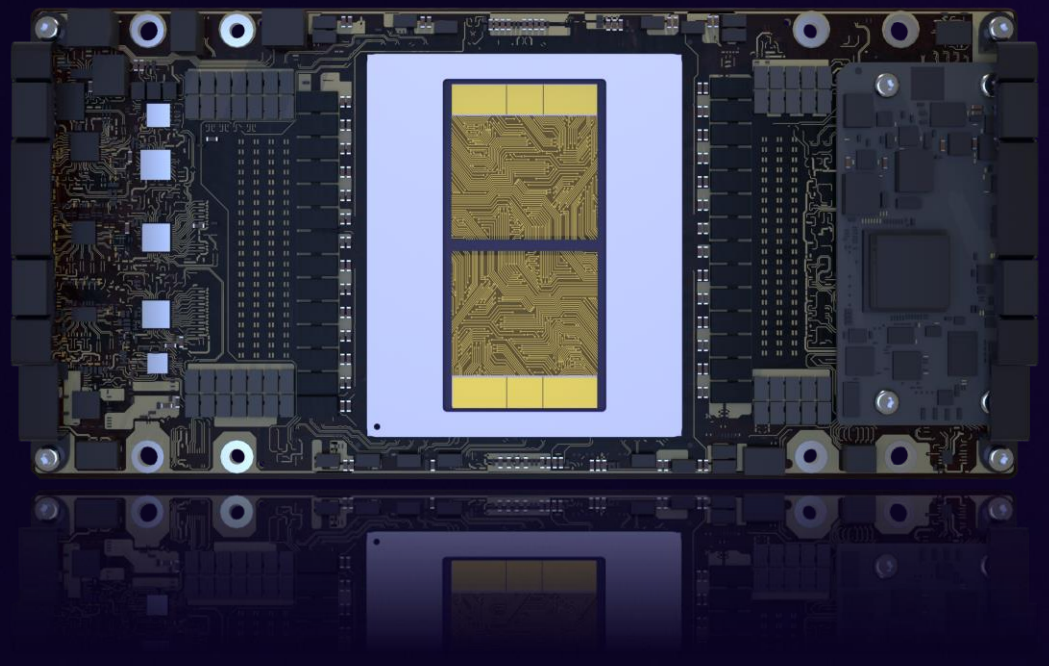


VOLTAGE REGULATORS

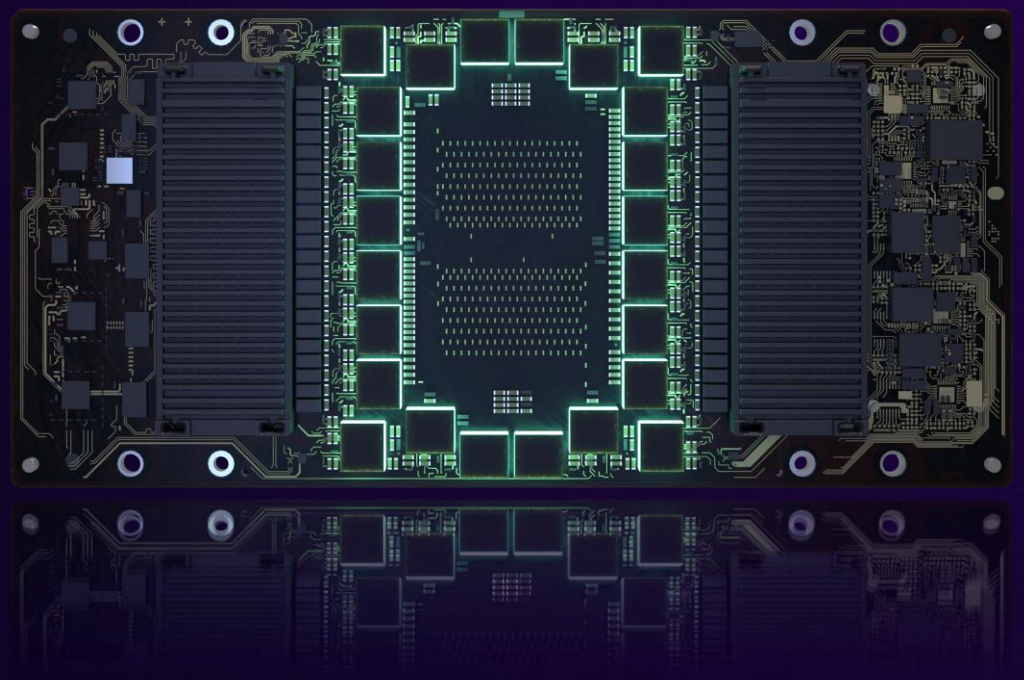
Optimized power delivery

TRAINIUM2 – VERTICAL POWER DELIVERY

TOP

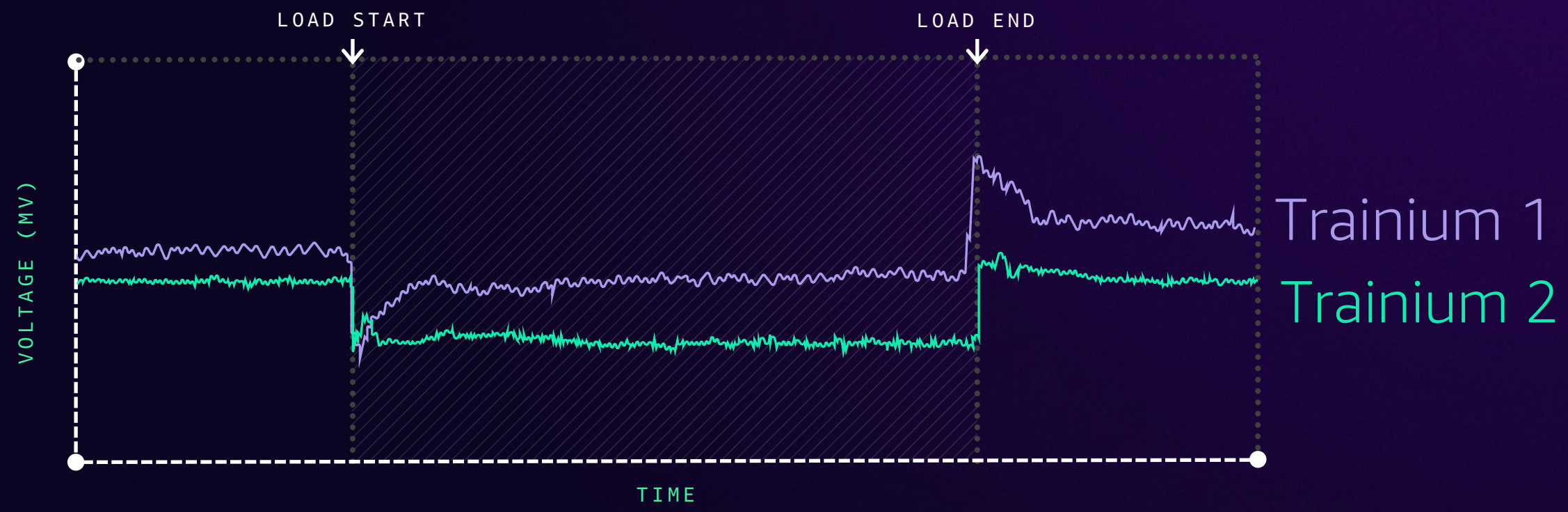


BOTTOM



VOLTAGE REGULATORS

Optimized power delivery



High performance

Cost efficiency

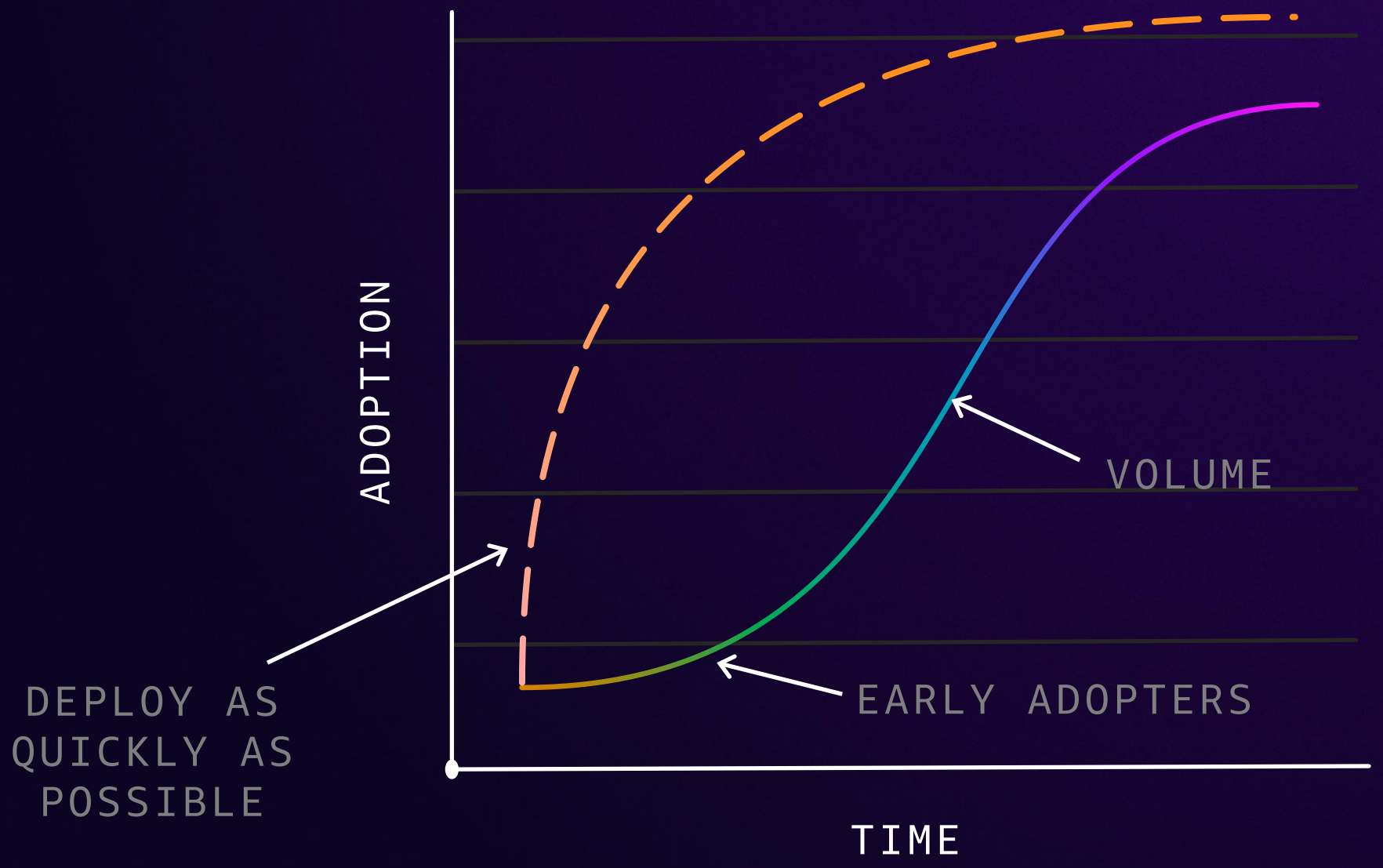
Scale

Ease of use

Innovation

Adoption curves in ML

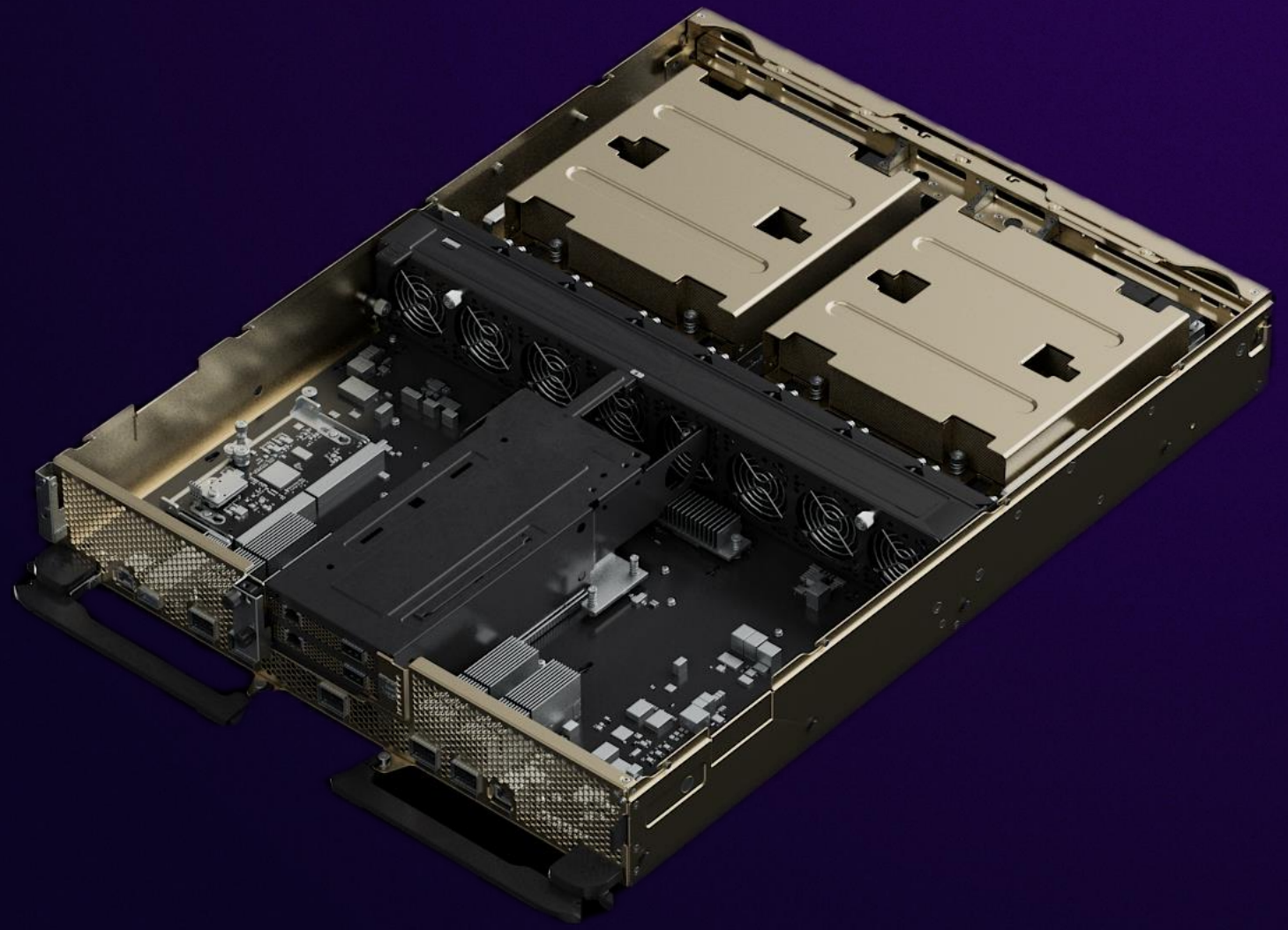
- Conventional adoption
- ML adoption



Manufacturing at scale

Modular and robust design – minimize cables/components

Simple cooling solution



Manufacturing at scale

Modular and robust design – minimize cables/components

Simple cooling solution

Production-line automation

In-fleet scan and pre-flight checks

High performance

Cost efficiency

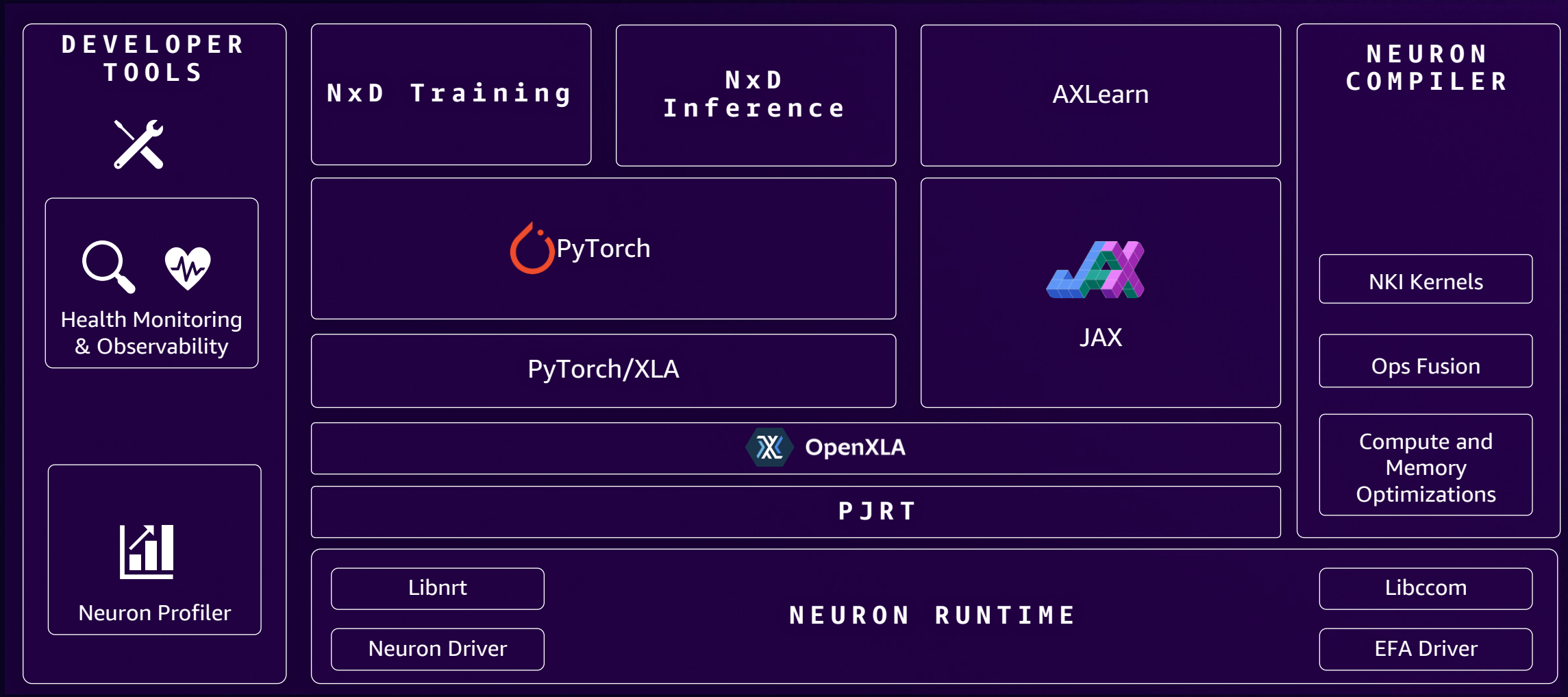
Scale

Ease of use

Innovation

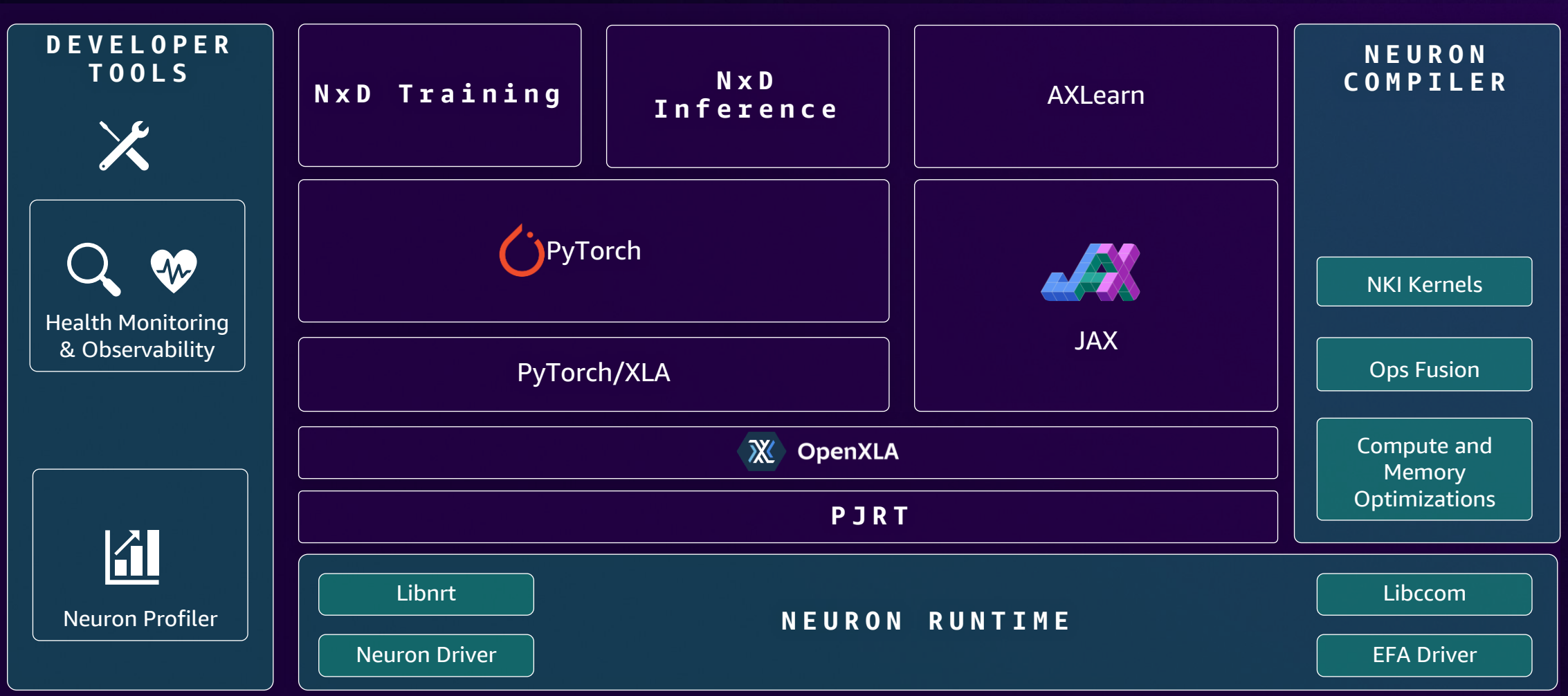
Neuron SDK software stack

AWS Neuron



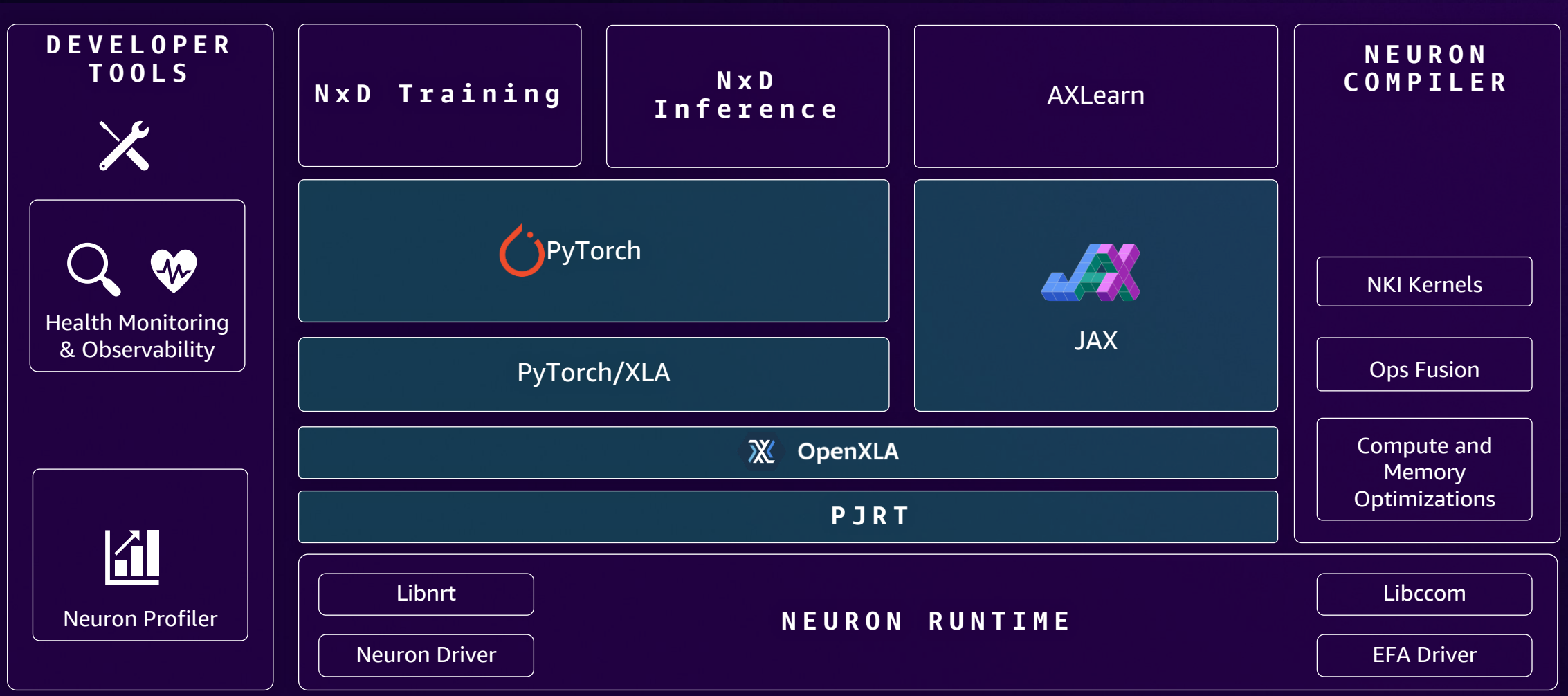
Neuron SDK software stack

AWS Neuron



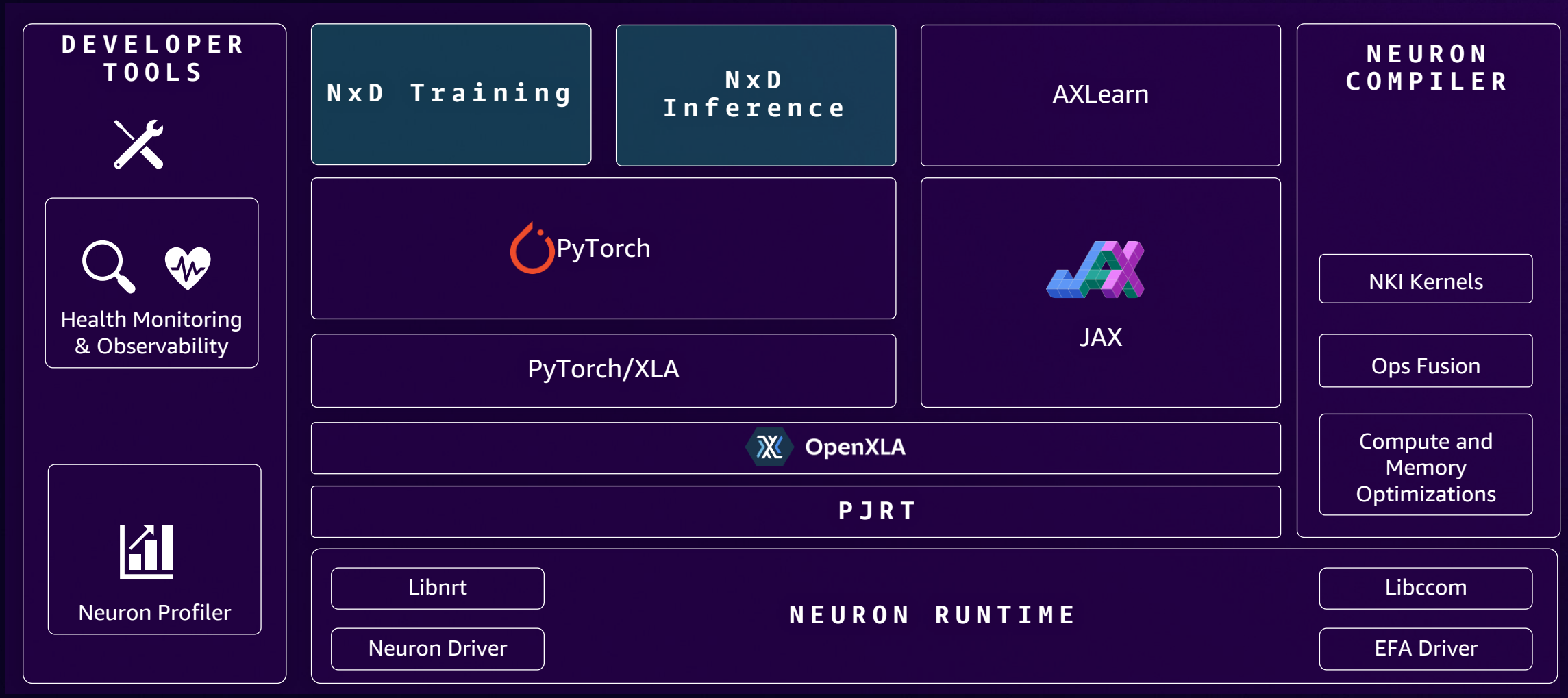
Neuron SDK software stack

AWS Neuron



Neuron SDK software stack

AWS Neuron



Neuron SDK integration

AWS SERVICES AND THIRD-PARTY SOFTWARE

AWS Neuron

ML frameworks and libraries

PyTorch Lightning
RAY
Hugging Face
vLLM
Weights & Biases
DOMINO
Outerbounds

AWS services

AWS Parallel Cluster
Amazon SageMaker
AWS Batch
Amazon ECS
Amazon EKS
Neuron DL Containers
Neuron DL AMIs

Neuron SDK

Neuron SDK Stack

High performance

Cost efficiency

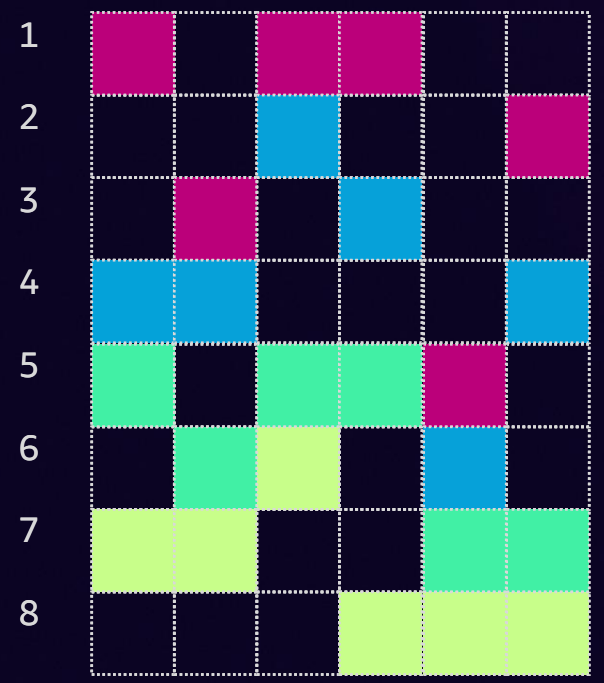
Scale

Ease of use

Innovation

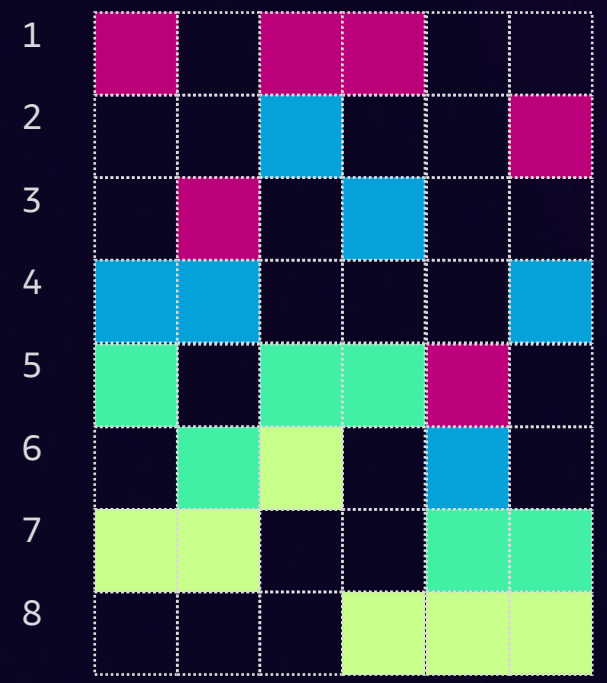
Sparsity in Trainium2

Sparse matrix

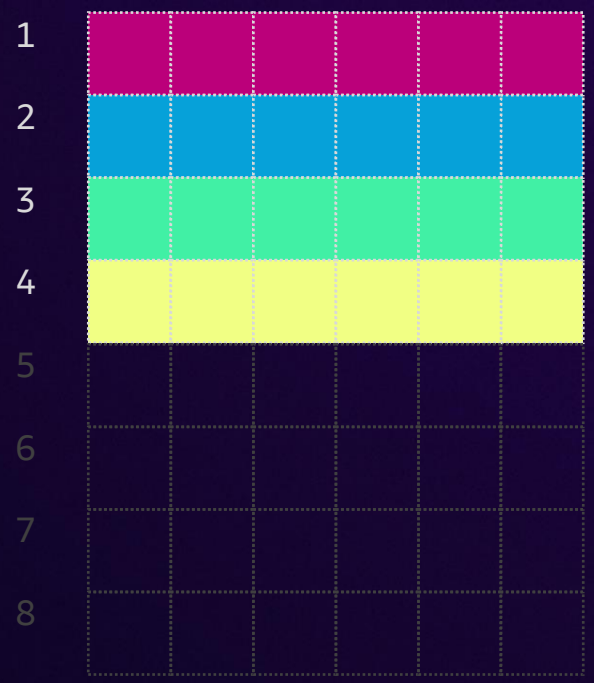


Sparsity in Trainium2

Sparse matrix



Dense matrix
(compression)



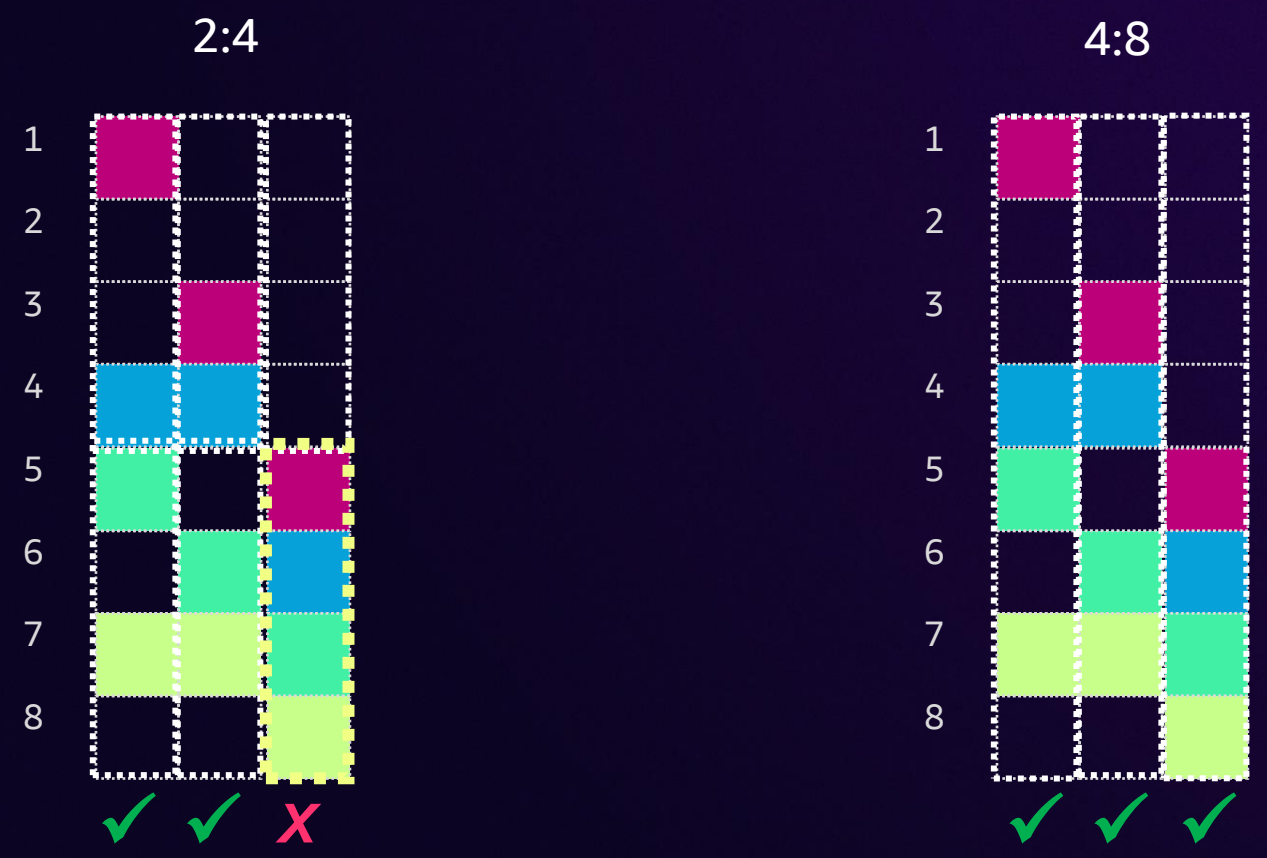
Sparsity in Trainium2

- Structured K:N sparsity (4x speedup!)
- Supported formats: 4:8, 4:12, 4:16
- Improves both compute and memory

Example: 4:8 sparsity

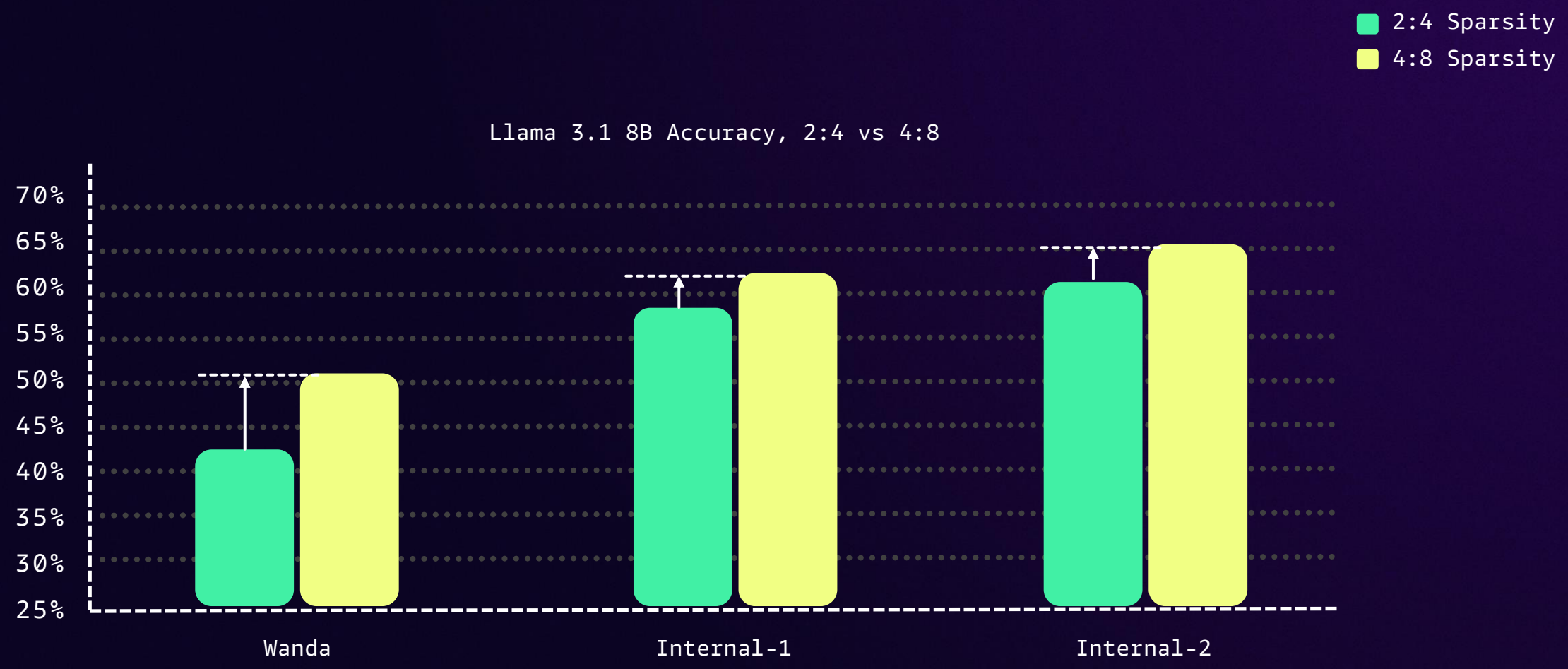
Sparsity in Trainium2

4:8 PERFORMS BETTER THAN 2:4



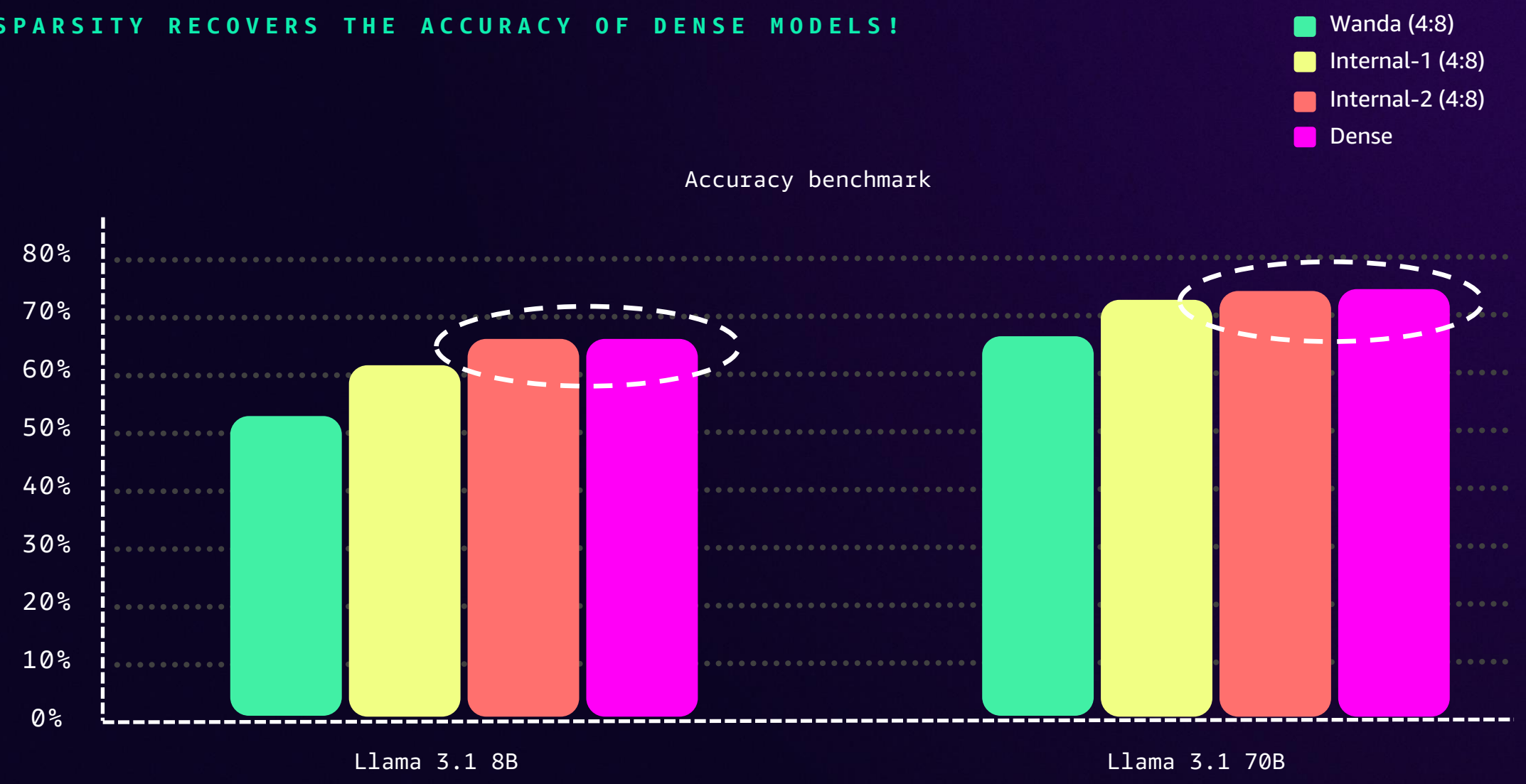
Sparsity in Trainium2

4:8 PERFORMS BETTER THAN 2:4



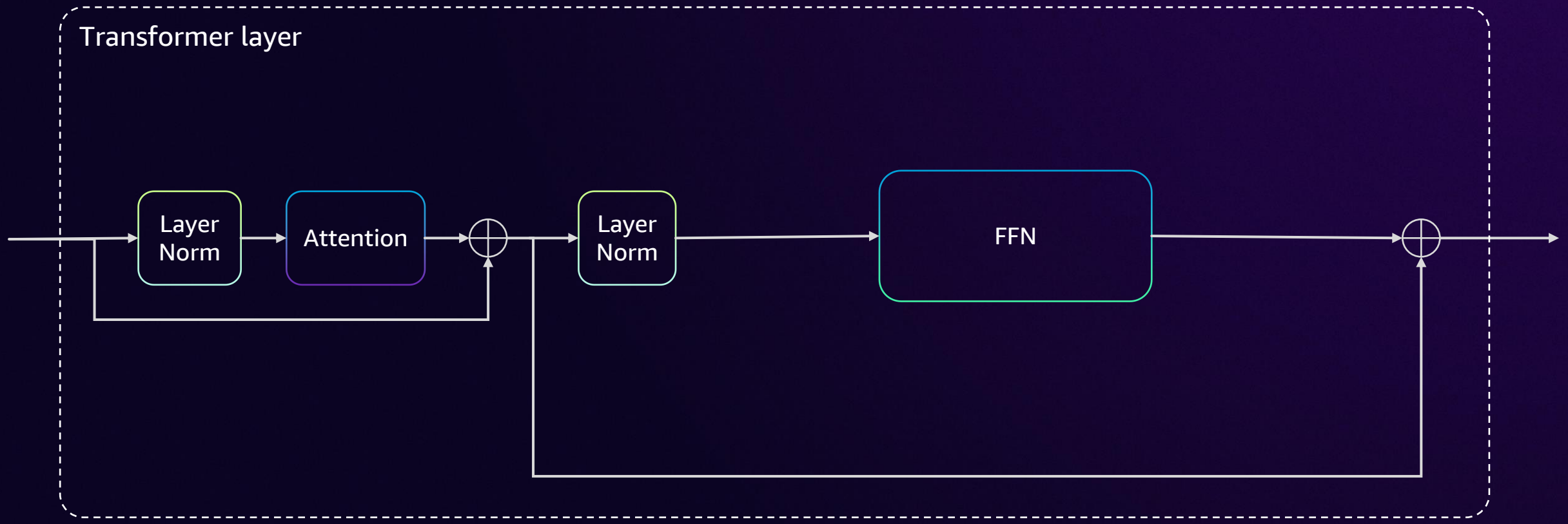
Sparsity in Trainium2

4:8 SPARSITY RECOVERS THE ACCURACY OF DENSE MODELS!



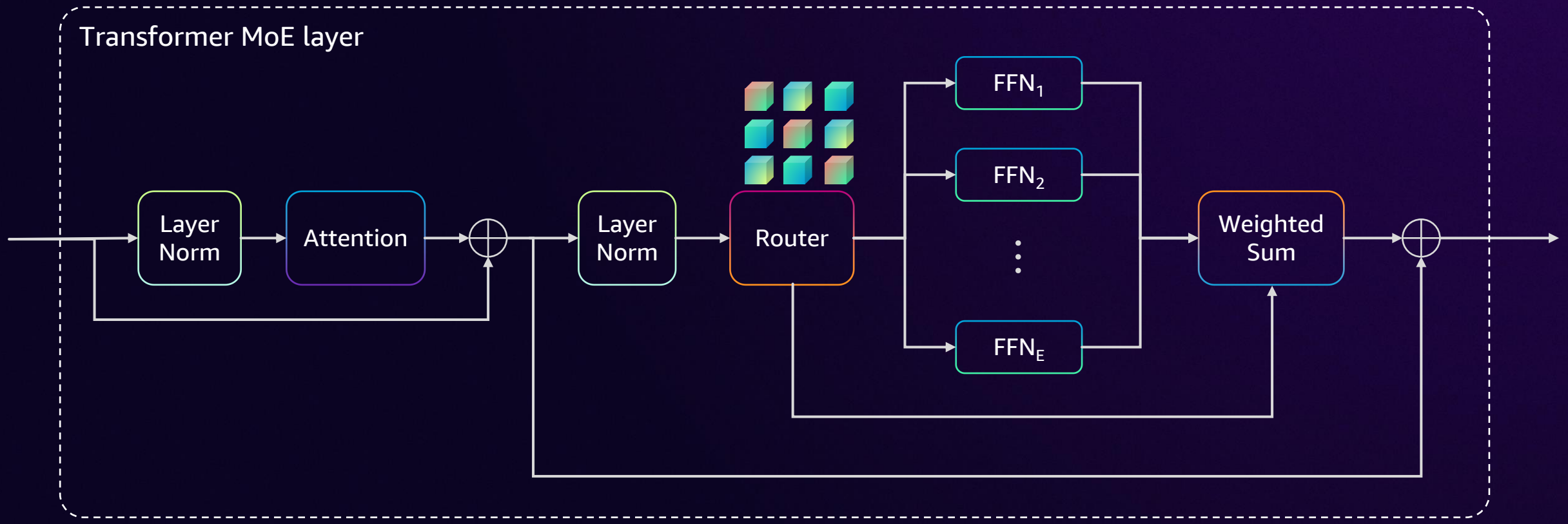
Mixture of Experts on Trainium

SCALING NUMBER OF PARAMETERS AT A FRACTION OF THE COMPUTATIONAL COST



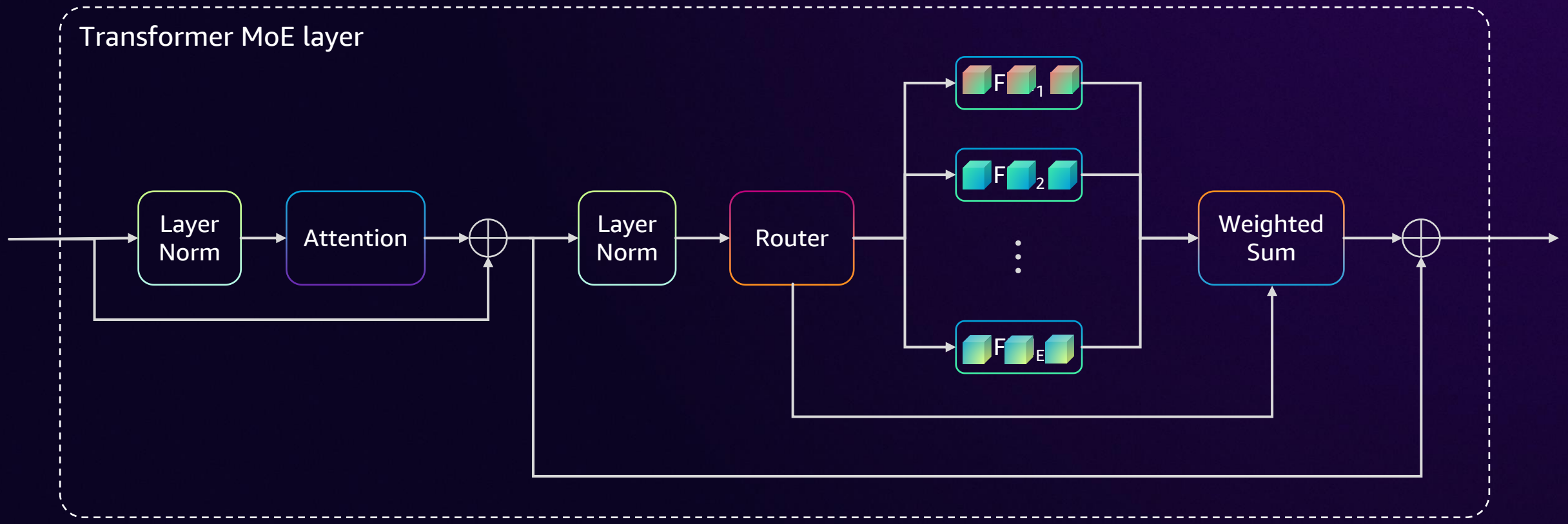
Mixture of Experts on Trainium

SCALING NUMBER OF PARAMETERS AT A FRACTION OF THE COMPUTATIONAL COST



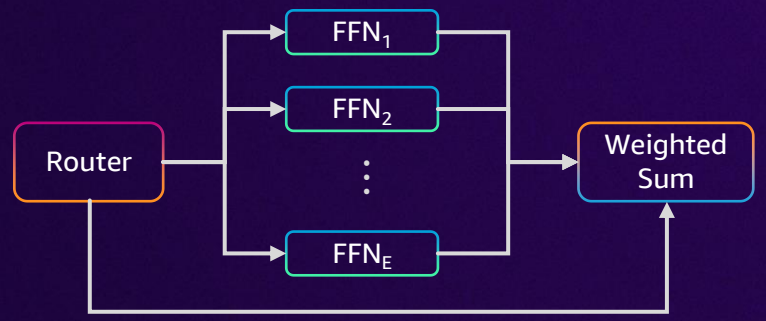
Mixture of Experts on Trainium

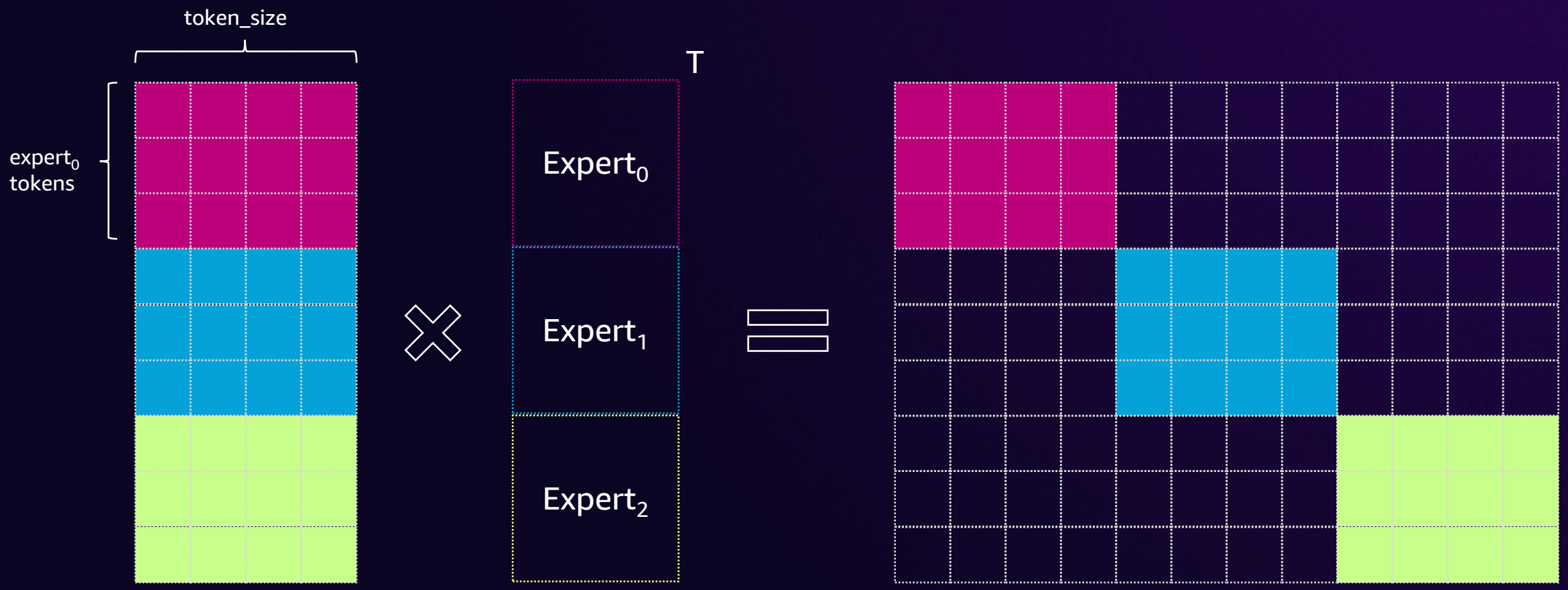
SCALING NUMBER OF PARAMETERS AT A FRACTION OF THE COMPUTATIONAL COST



Mixture of Experts on Trainium

DROPLESS MOE: BLOCK-SPARSE COMPUTATIONS FOR EFFICIENT MOE

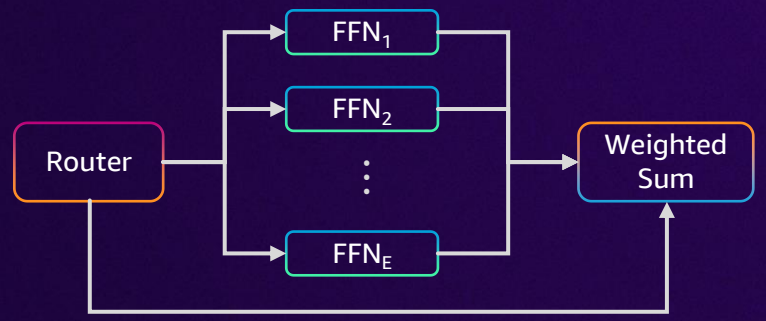


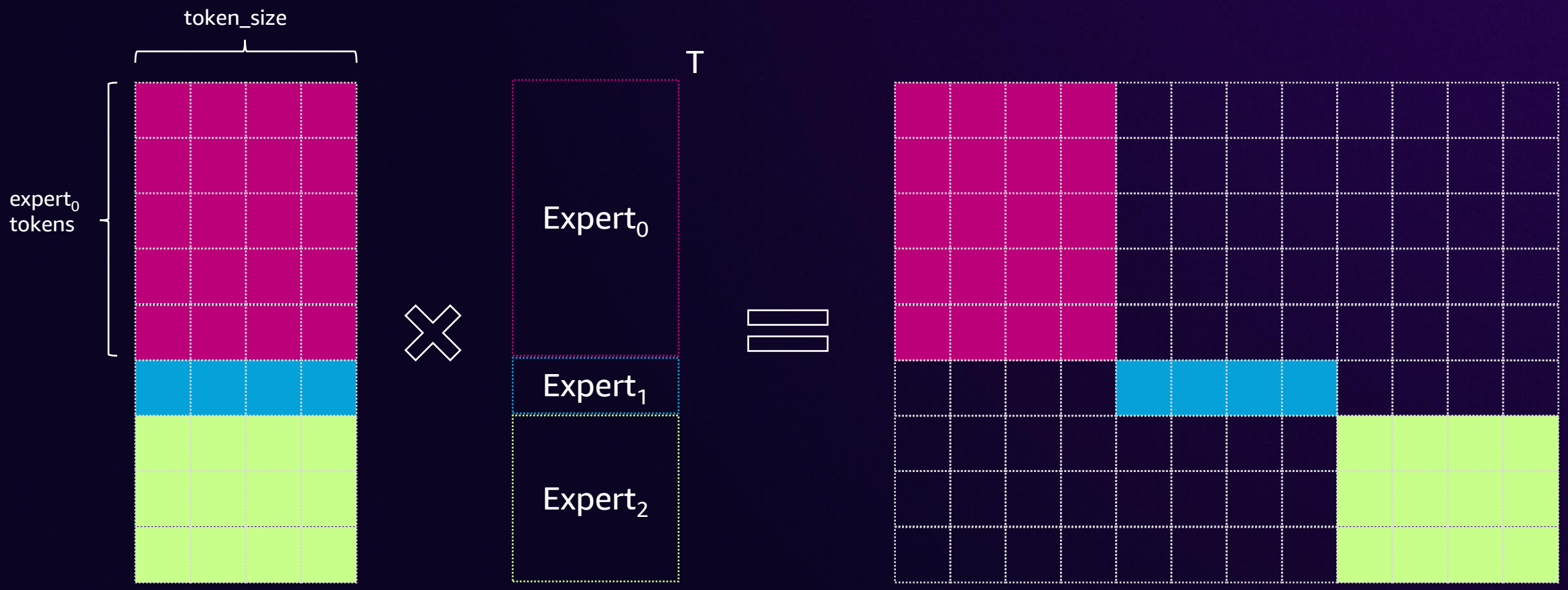


https://arxiv.org/pdf/2211.15841

Mixture of Experts on Trainium

DROPLESS MOE: BLOCK-SPARSE COMPUTATIONS FOR EFFICIENT MOE





https://arxiv.org/pdf/2211.15841

Mixture of Experts on Trainium

DROPLESS MOE: BLOCK-SPARSE COMPUTATIONS FOR EFFICIENT MOE

- Dynamic memory addressing (pick the relevant expert)
- NKI kernel for block-sparse computation

```
# Loop over all available experts in the model and perform the computation on each expert
for expert_idx in range(self.num_experts):
    expert_layer = self.experts[expert_idx]
    idx, top_x = torch.where(expert_mask[expert_idx])

    # Index the correct hidden states and compute the expert hidden state for
    # the current expert. We need to make sure to multiply the output hidden
    # states by `routing_weights` on the corresponding tokens (top-1 and top-2)
    current_state = hidden_states[None, top_x].reshape(-1, hidden_dim)
    current_hidden_states = expert_layer(current_state) * routing_weights[top_x, idx, None]

    # However `index_add_` only support torch tensors for indexing so we'll use
    # the `top_x` tensor here.
    final_hidden_states.index_add_(0, top_x, current_hidden_states.to(hidden_states.dtype))
```

Gather the indices of tokens assigned to expert: Non-static shape

If no tokens are assigned to expert, entire computation (and expert loading) is skipped

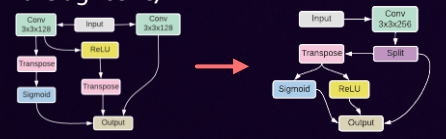
Objective: 'Selectively load' only the expert weights that are required.

Neuron Kernel Interface (NKI)

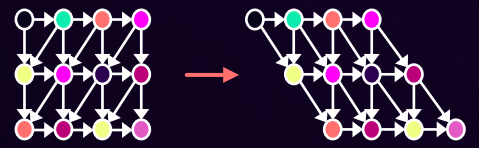
BARE METAL PROGRAMMING OF TRAINIUM DEVICES

```python
import neuronxcc.nki.language as nl


def nki_tensor_add_kernel_(a_input, b_input, c_output):
  # Generate tensor indices to index tensors a and b
  ix = nl.arange(128)[:, None]
  iy = nl.arange(512)[None, :]

  # Load input data from device memory (HBM) to on-chip memory (SBUF)
  # We refer to an indexed portion of a tensor as an intermediate tensor
  a_tile = nl.load(a_input[ix, iy])
  b_tile = nl.load(b_input[ix, iy])

  # compute a + b
  c_tile = a_tile + b_tile

  # store the addition results back to device memory (c_output)
  nl.store(c_output[ix, iy], value=c_tile)
```

Graph optimizations
(hardware agnostic)



Loop optimizations
(layout, tiling, vectorization, pipelining)

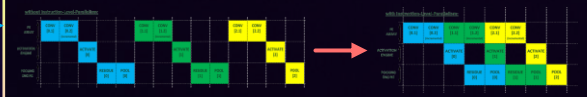


Hardware intrinsic mapping

```
for i=1,128
    for j=1,128
        for k=1,128
            z[i,j] = x[k,i]*y[k,j]
```

```
z =
matmul_128x128(
x,y)
```

Scheduling and allocation
(working set minimization, latency hiding)



Neuron Kernel Interface (NKI)

WHY SHOULD YOU USE NKI?

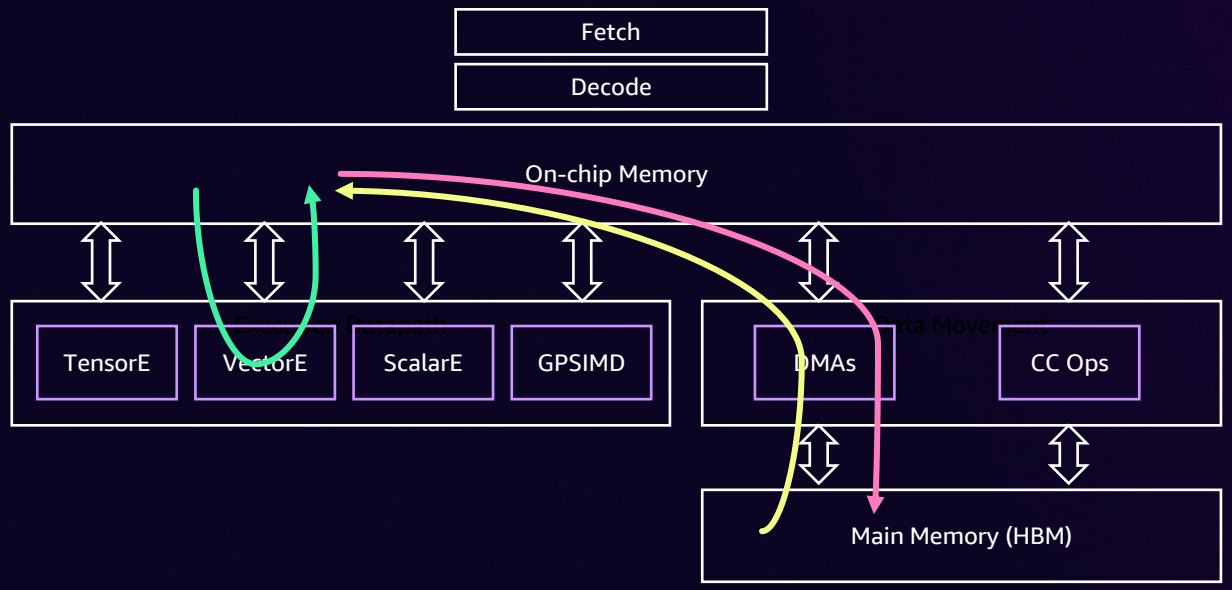
- Invent new operators
- Optimize performance
- Take full control of the hardware

```python
import neuronxcc.nki.language as nl


def nki_tensor_add_kernel_(a_input, b_input, c_output):
  # Generate tensor indices to index tensors a and b
  ix = nl.arange(128)[:, None]
  iy = nl.arange(512)[None, :]

  # Load input data from device memory (HBM) to on-chip memory (SBUF)
  # We refer to an indexed portion of a tensor as an intermediate tensor
  a_tile = nl.load(a_input[ix, iy])
  b_tile = nl.load(b_input[ix, iy])

  # compute a + b
  c_tile = a_tile + b_tile

  # store the addition results back to device memory (c_output)
  nl.store(c_output[ix, iy], value=c_tile)
```

Neuron Kernel Interface (NKI)

"HELLO WORLD" IN NKI

NKI Manual




```python
import neuronxcc.nki.language as nl

def nki_tensor_add_kernel_(a_input, b_input, c_output):
  # Generate tensor indices to index tensors a and b
  ix = nl.arange(128)[:, None]
  iy = nl.arange(512)[None, :]

  # Load input data from device memory (HBM) to on-chip memory (SBUF)
  # We refer to an indexed portion of a tensor as an intermediate tensor
  a_tile = nl.load(a_input[ix, iy])
  b_tile = nl.load(b_input[ix, iy])

  # compute a + b
  c_tile = a_tile + b_tile

  # store the addition results back to device memory (c_output)
  nl.store(c_output[ix, iy], value=c_tile)
```

```
for(ix in range(128)):
  for(iy in range(512)):
    c[ix,iy] = a[ix,iy] + b[ix,iy]
```

Mamba2 on NKI

ACCELERATING STATE SPACE MODELS

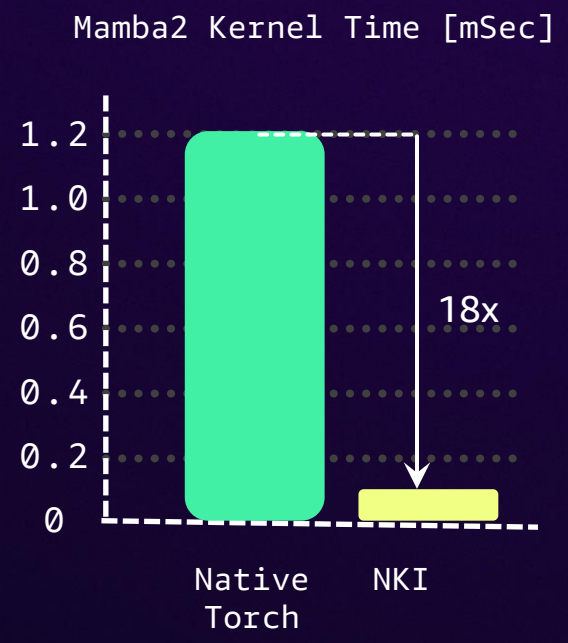
Mamba on Trainium tutorial

State Space Models (e.g., Mamba) provide an alternative to attention-based language models

Useful for handling very large contexts

Intrinsically sequential → requires careful optimization

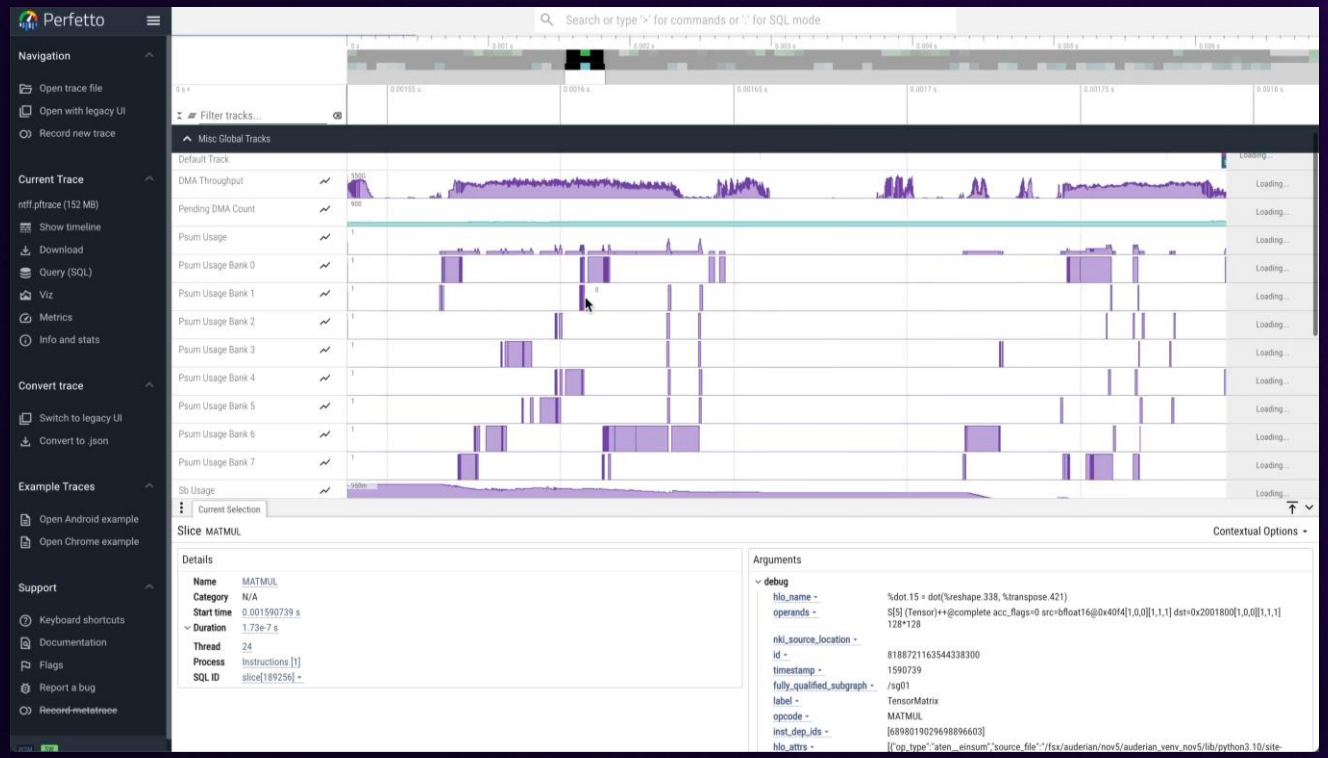
18x speedup with 77 lines of NKI code!



Neuron Profiler

Neuron Profiler

- A non-intrusive trace
  - Always on – no “Heisenbugs”
- Instruction-level timing
- Performance bottleneck warnings
- Multi-layer instruction source annotation



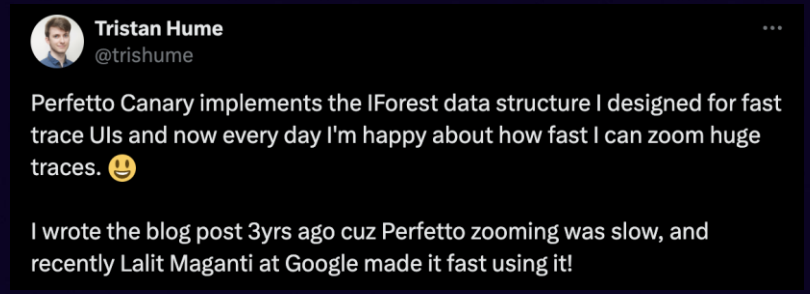


Image understanding on Trainium, LLaVA

The image features a painting of a dog wearing a hat, possibly a fez, and a cape. The dog is positioned in the center of the painting, with its head tilted to the side. The painting is quite detailed, capturing the dog's facial expression and the intricate design of the hat and cape. The background of the painting is a landscape, adding depth and context to the scene.

Pre-trained on LLaVA-Pretrain; Fine tuned on Llava-instruct-150k

Diffusion models on Trainium, PixArt

> A blue jay standing on a large basket of rainbow macarons

Pre-training on 10M SAM;
Fine tuning on 4.4M JourneyDB

Fine-tuning:

ANTHROP\C and Trainium

Powering the next generation of AI development with AWS

Nov 22, 2024 • 3 min read

amazon | ANTHROP\C

ANTHROP\C

Leading AI lab and model provider

- Founded four years ago by the team that built GPT-3
- Claude 3 Opus (in March) and Claude 3.5 Sonnet (in June)
- Focus on safety and responsible scaling
- Looking for the most cost-effective, scalable compute to power our training and inference

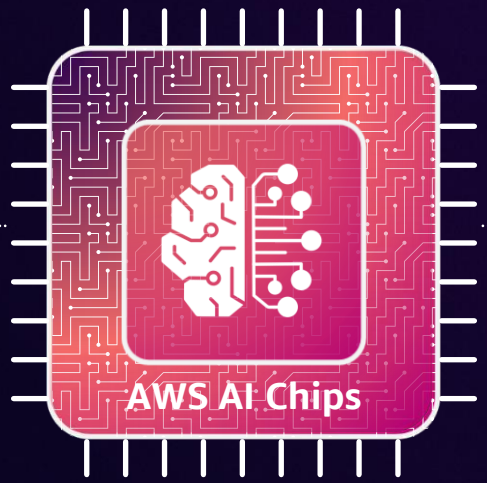
The Trainium bet

WHY ARE WE BETTING ON TRAINIUM?

Incredible price-performance, especially for HBM-intensive workloads

Delivery at scale with world-class datacenter infrastructure

Flexible and programmable chip architecture



Low-level access like the Neuron Kernel Interface

Trn2 UltraServers for scale-out training/inference of large models

Fantastic collaboration from Ron and the Annapurna team

Project Rainier

What will it take to train at this scale?

Training on Trn2 UltraServers

Careful attention to parallelism and scaling

Custom kernels with deep hardware optimization

Innovations in reliability and testing

…and lots of Anthropic secret sauce

Claude 3.5 Haiku on Trn2

OFFERING "LATENCY-OPTIMIZED" INFERENCE ON AMAZON BEDROCK

UP TO **60%**

FASTER THAN STANDARD CLAUDE 3.5 HAIKU SKU

UP TO **125**

OUTPUT TOKENS PER SECOND (OTPS)

Claude.app - ⌘⇧C

Claude ← All projects

Share

NKI Docs

Private

Created by James Bradbury

How can Claude help you today?

Claude Haiku 3.5 Speed SKU Choose style 1

NKI Docs

Activity Your chats

Your chats are private until shared

Add to project knowledge or share your chats to spark ideas, learn from teammates, and discover how your team uses Claude.

JB

Project knowledge

+ Add Content

+ Set project instructions Optional

48% of knowledge capacity used

- TEXT transpose2d.rst — 2 hours ago
- TEXT spmd_tensor_addition.rst — 2 hours ago
- TEXT spmd_multiple_nc_tensor_addi... — 2 hours ago
- TEXT rmsnorm.rst — 2 hours ago
- TEXT matrix_multiplication.rst — 2 hours ago
- TEXT layernorm.rst — 2 hours ago
- TEXT fused-self-attn.rst — 2 hours ago
- TEXT fused_mamba.rst — 2 hours ago · Large file
- TEXT average_pool2d.rst — 2 hours ago
- TEXT tutorials.rst

aws

Wrap up

Trn2 is available today!

Trn2 innovations deliver highly optimized performance, energy efficiency, and cost efficiency

We're laying the technological foundation, from chips to science, that will power the next generation of foundation models

Get started now!

AWS TRAINIUM AND INFERENTIA SESSIONS AT RE:INVENT 2024

STILL TIME TO LEARN MORE!!

Thursday

- CMP329-R1
  Beyond Text: Unlock multimodal AI with AWS AI chips
- CMP335-R1
  Drilling down into performance for distributed training
- CMP208
  Customer Stories: Optimizing AI performance and cost with AWS AI chips
- CMP307-R1
  Demystifying LLM deployment and optimization on AWS Inferentia (HandsON)

Friday

- CMP314-R2
  Keeping it small: Agentic workflows with SLMs on AWS Inferentia (HandsON)

Thank you!

Please complete the session survey in the mobile app

Joe Senerchia
jsenerch@amazon.com

James Bradbury
jek@anthropic.com

Ron Diamant
diamant@amazon.com