



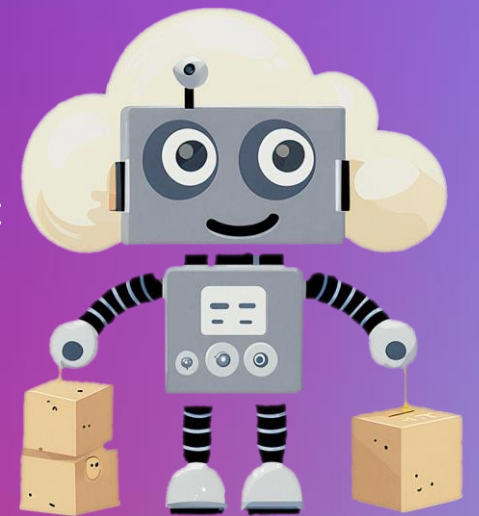
WOR101

# Efficiently train and deploy LLMs on AWS Trainium and Inferentia2 accelerators

Luca Perrozzi  
Solutions Architect  
AWS

Dmitri Laptev  
Solutions Architect  
AWS

Roy Allela  
AIML Specialist Solutions Architect  
AWS



# Agenda

Trends fueling generative AI innovation

Common challenges and trade-offs

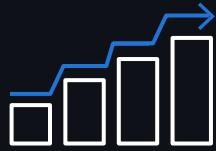
AWS Trainium

AWS Inferentia2

Hands-on Lab

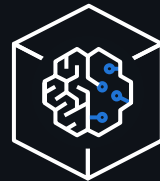
Q&A

# Trends fueling artificial intelligence (AI) innovation



## Large language models

Explosive growth of large language models based on transformer architectures



## Self-supervised learning

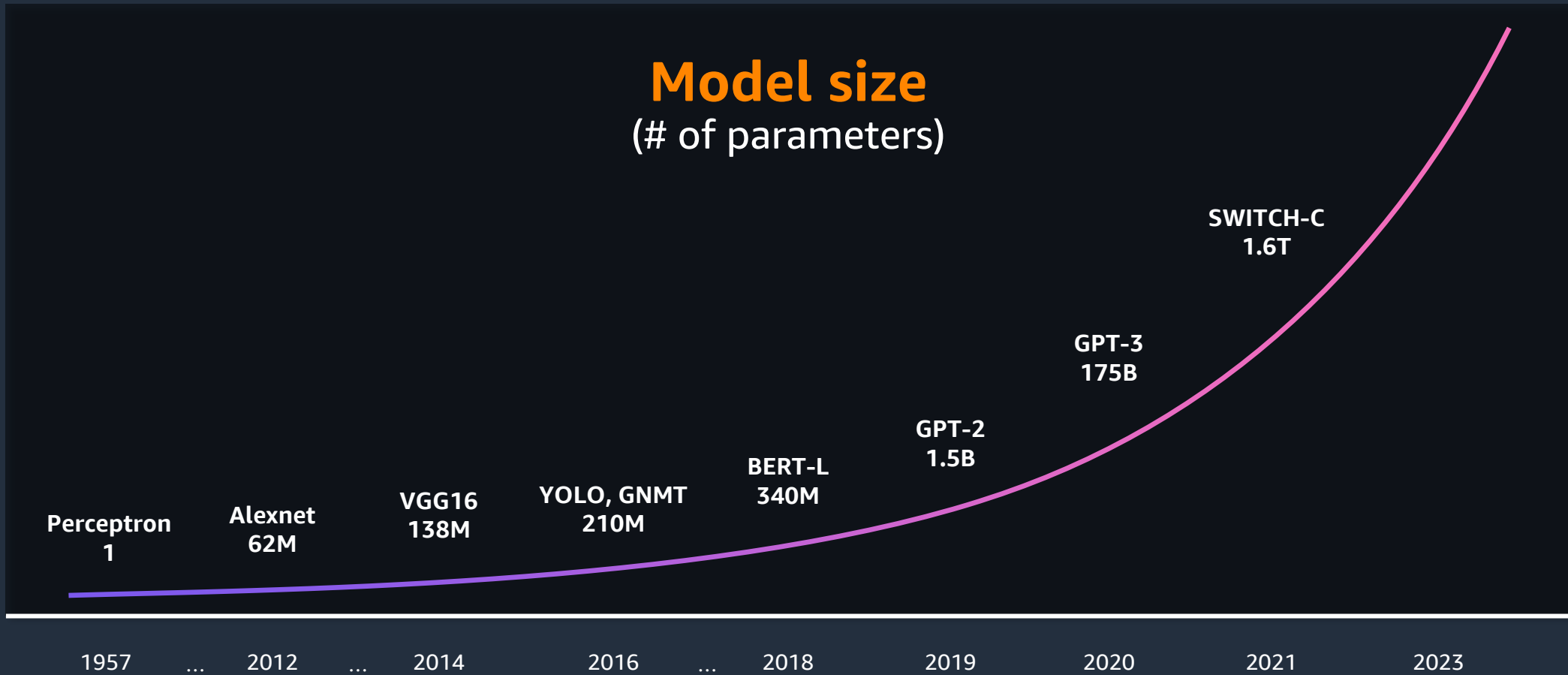
Data scientists are no longer limited to human-labeled data for training models



## Open-source momentum

Increased momentum from industry offering open-source, pretrained models

# AI models are getting bigger...

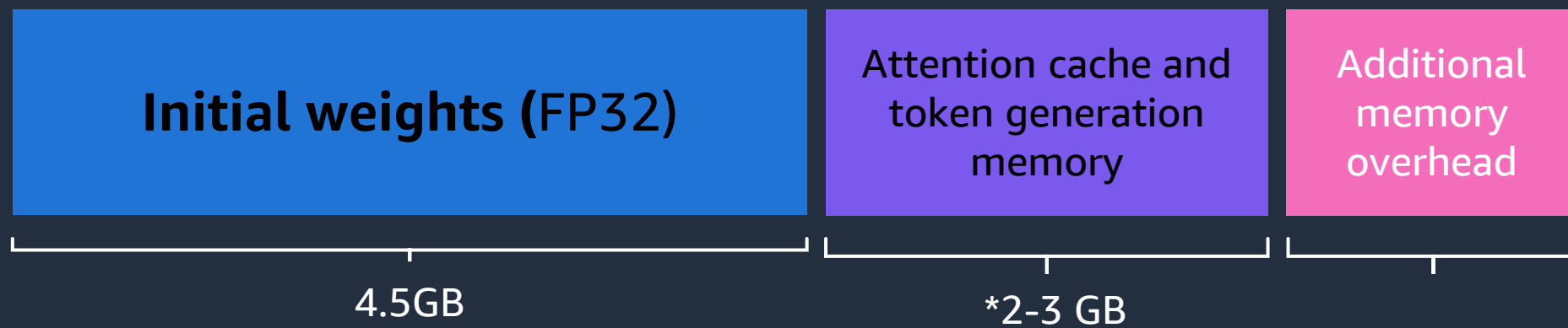


# Challenges of LLM

EXAMPLE: TINYLLAMA-1.1B MODEL FOR INFERENCE

**Model size = Parameters (1-billion) x 4 bytes (FP32) = 4.5 GB**

TinyLlama-1.1B



Size comparison:  
FP 16 = 1/2 FP 32  
BF 16 = 1/2 FP 32  
INT 8 = 1/4 FP 32

\* The required memory depends on input/output sequence length, decoding strategy, model architecture, and batch size

# Common challenges and trade-offs



**Performance**



**Cost**



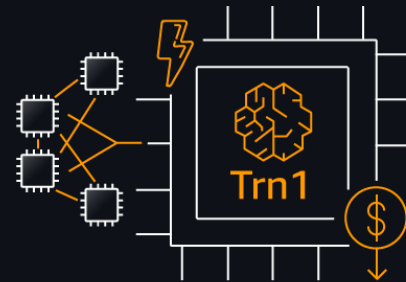
**Ease of use**



**Energy efficiency**

# AWS purpose-built accelerators for generative AI

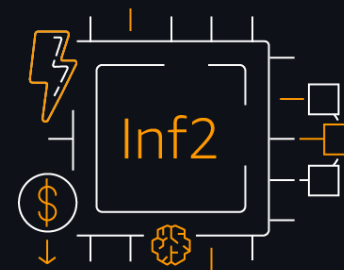
## AWS Trainium



The most cost efficient for high-performance training of LLMs and diffusion models

Up to 50% cost-to-train savings over comparable GPU-based instances

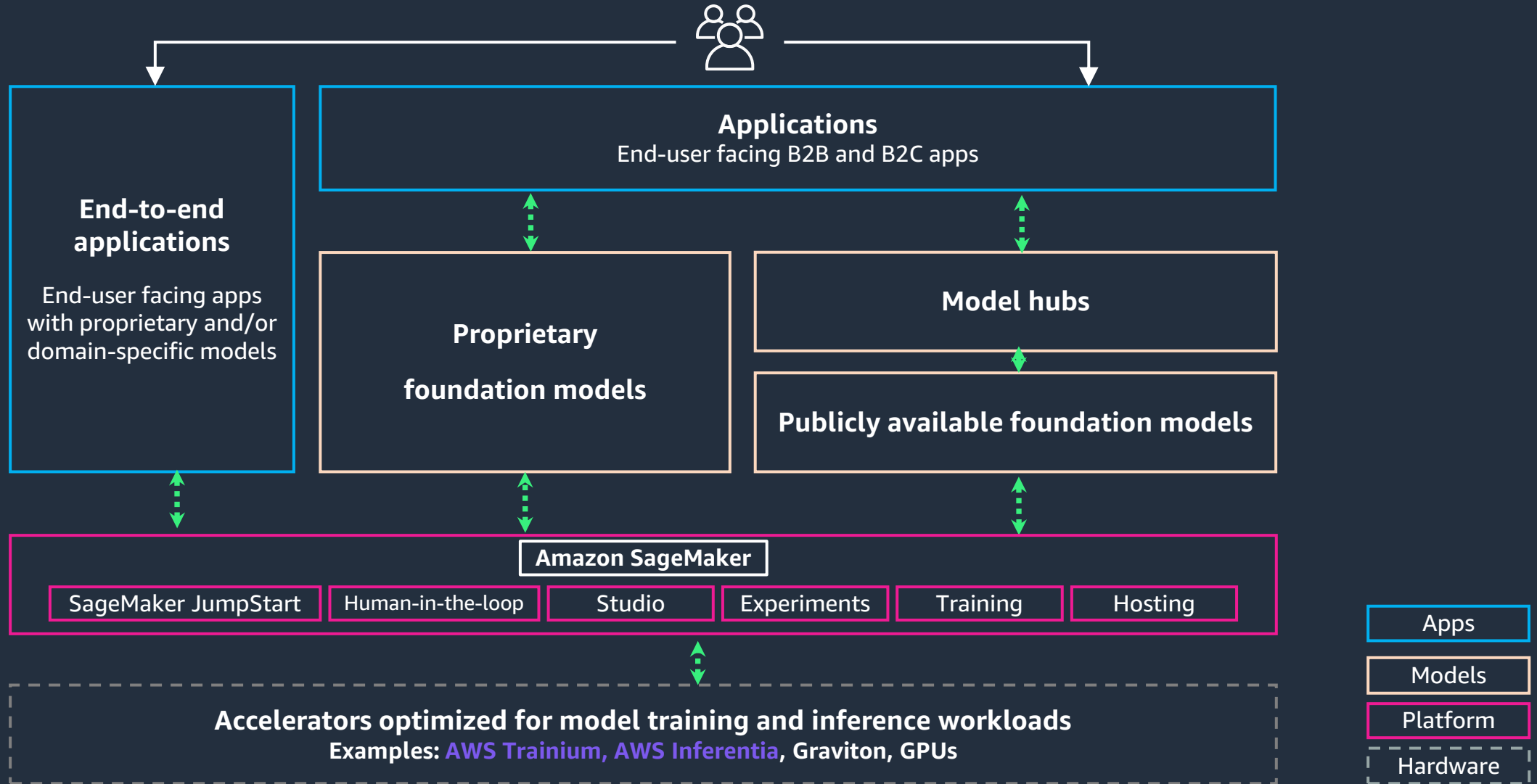
## AWS Inferentia2



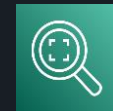
High performance at the lowest cost per inference for LLMs and diffusion models

Up to 40% better price performance than comparable GPU-based instances

# Generative AI: Lay of the land using SageMaker



# Customer momentum for AWS Trainium and AWS Inferentia



Amazon Rekognition

# Better together: Support for AWS Trainium and AWS Inferentia



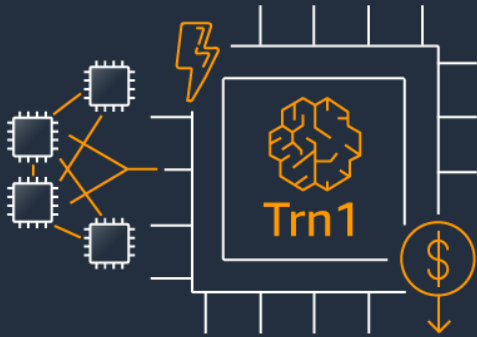


# AWS Trainium



# Amazon EC2 Trn1 instances powered by AWS Trainium

COST-EFFICIENT, HIGH-PERFORMANCE DL TRAINING INSTANCES



High performance on training of popular NLP models on AWS

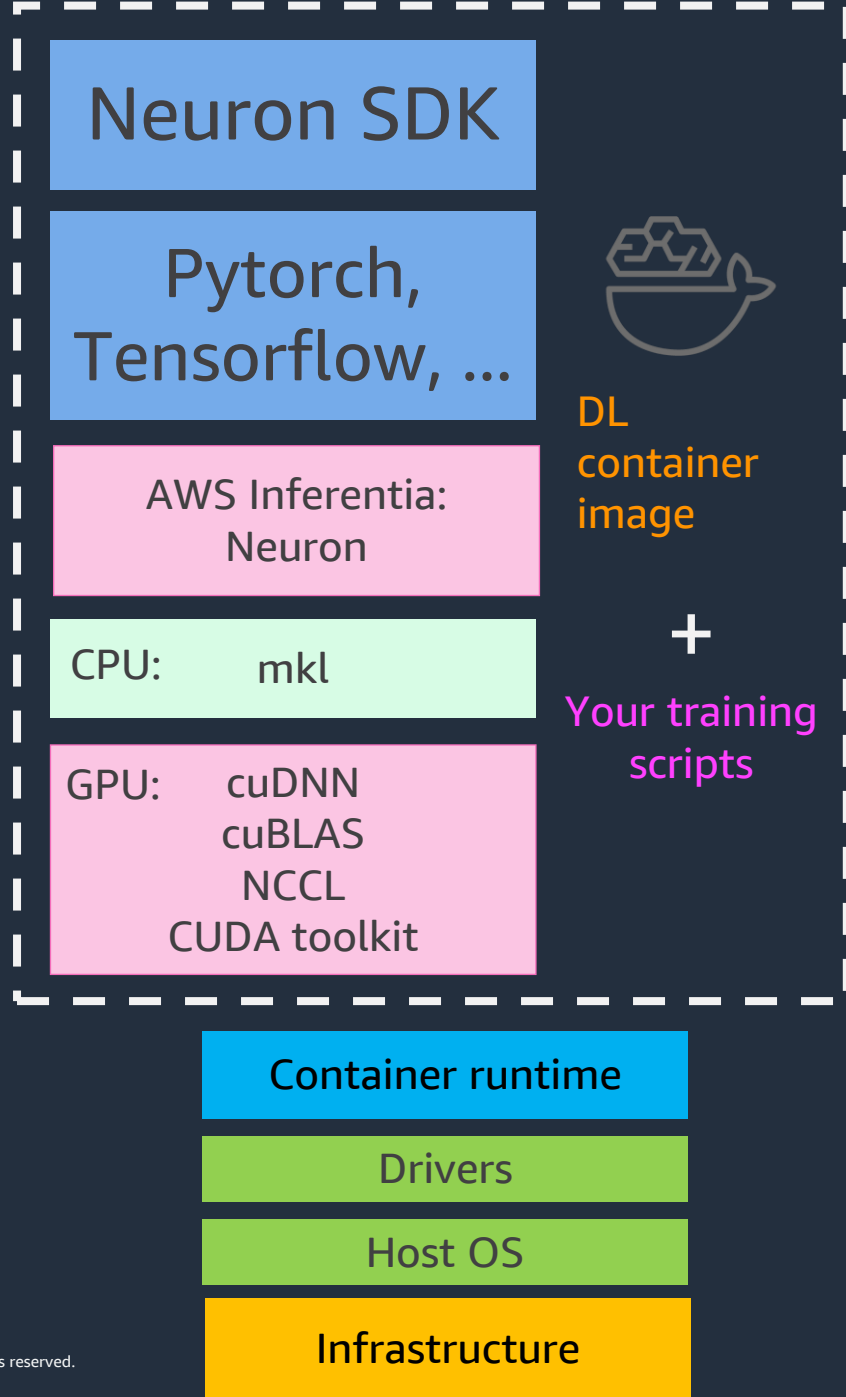
Up to 50% cost-to-train savings

Up to 4x network bandwidth

Instance size	vCPUs	Instance memory	Trainium chips	Accelerator memory	NeuronLink	Instance networking	On-demand price
Trn1.2xlarge	8	32 GB	1	32 GB	N/A	Up to 10 Gbps	\$1.34/hr
Trn1.32xlarge	128	512 GB	16	512 GB	Yes	800 Gbps	\$21.5/hr
Trn1n.32xlarge	128	512 GB	16	512 GB	Yes	1600 Gbps	\$24.78/hr

Trn1 available now in US-East-1 (N. Virginia) and US-West-2 (Oregon)

# AWS Deep Learning Containers



## ML environments that are:

- Lightweight
- Portable
- Scalable
- Consistent

## Packages:

- Training code
- Dependencies
- Configurations

# High Level Workflow

```
import argparse
import os

if __name__ == '__main__':

    parser = argparse.ArgumentParser()

    # hyperparameters sent by the client are passed as command-line arguments to the script.
    parser.add_argument('--epochs', type=int, default=50)
    parser.add_argument('--batch-size', type=int, default=64)
    parser.add_argument('--learning-rate', type=float, default=0.05)
    parser.add_argument('--use-cuda', type=bool, default=False)

    # Data, model, and output directories
    parser.add_argument('--output-data-dir', type=str, default=os.environ['SM_OUTPUT_DATA_DIR'])
    parser.add_argument('--model-dir', type=str, default=os.environ['SM_MODEL_DIR'])
    parser.add_argument('--train', type=str, default=os.environ['SM_CHANNEL_TRAIN'])
    parser.add_argument('--test', type=str, default=os.environ['SM_CHANNEL_TEST'])

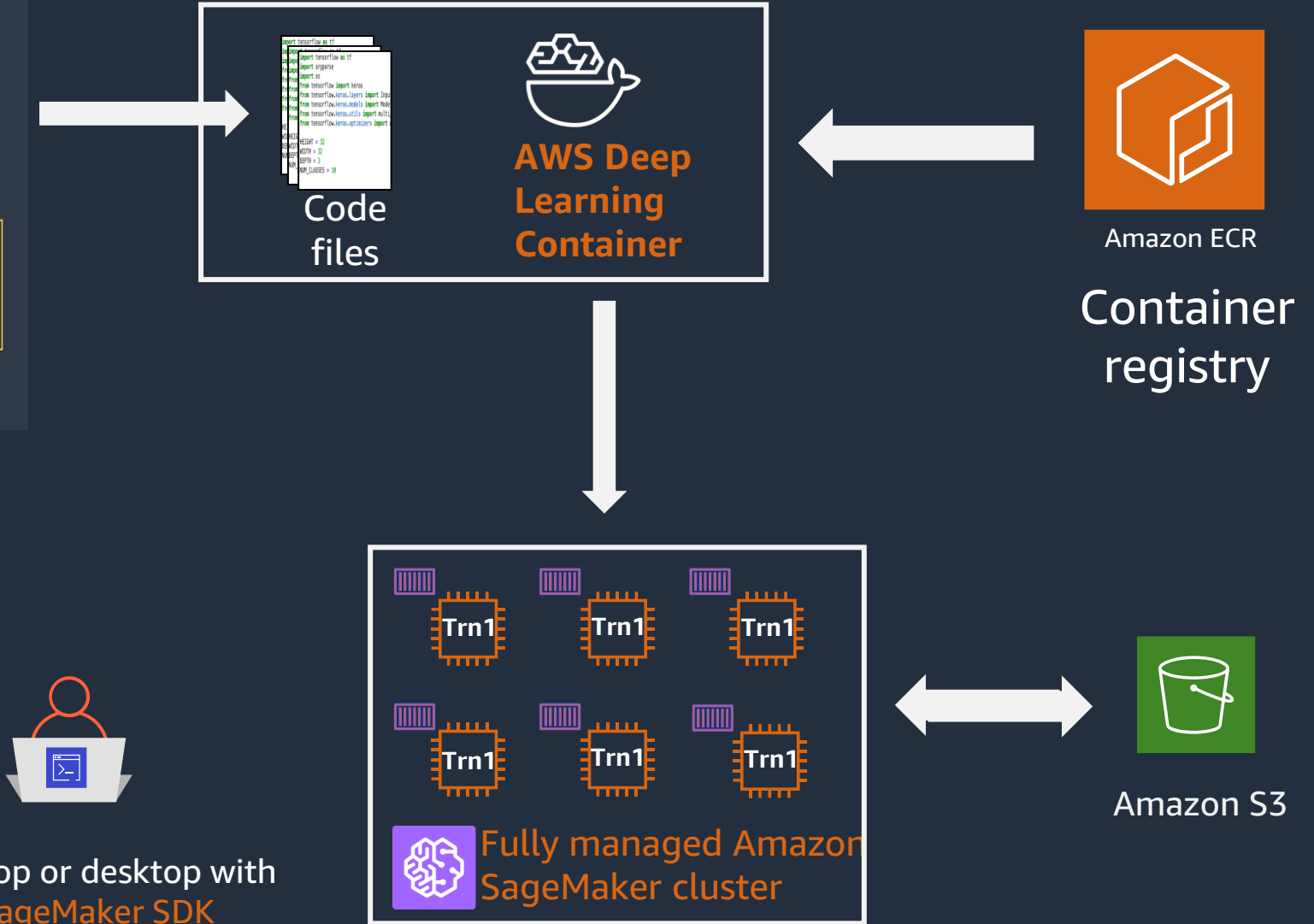
    args, _ = parser.parse_known_args()

    # ... load from args.train and args.test, train a model, write model to args.model_dir.

    from sagemaker.pytorch import PyTorch

    pt_estimator = PyTorch(entry_point="pytorch-train.py",
                           role=role,
                           instance_type="ml.trn1.2xlarge",
                           instance_count=1,
                           framework_version="1.6.0",
                           py_version="py36",
                           hyperparameters=hyperparameters)

    pt_estimator.fit({'training': inputs})
```



# Ease of use

Import model from Hugging Face

Use [DL Neuron containers](#)

<https://github.com/huggingface/optimum-neuron>



```

1 from optimum.neuron import NeuronTrainer as Trainer
2 from optimum.neuron.distributed import lazy_load_for_parallelism
3
4 # [...]
5
6 if model_args.model_name_or_path:
7     torch_dtype = (
8         model_args.torch_dtype
9         if model_args.torch_dtype in ["auto", None]
10        else getattr(torch, model_args.torch_dtype)
11    )
12    with lazy_load_for_parallelism(tensor_parallel_size=training_args.tensor_parallel_size):
13        model = AutoModelForCausalLM.from_pretrained(
14            model_args.model_name_or_path,
15            from_tf=bool(".ckpt" in model_args.model_name_or_path),
16            config=config,
17            cache_dir=model_args.cache_dir,
18            revision=model_args.model_revision,
19            use_auth_token=True if model_args.use_auth_token else None,
20            torch_dtype=torch_dtype,
21            low_cpu_mem_usage=model_args.low_cpu_mem_usage,
22        )
23
24 # [...]
25
26 # Initialize our Trainer
27 trainer = Trainer(
28     model=model,
29     args=training_args,
30     train_dataset=train_dataset if training_args.do_train else None,
31     eval_dataset=eval_dataset if training_args.do_eval else None,
32     tokenizer=tokenizer,
33     # Data collator will default to DataCollatorWithPadding, so we change it.
34     data_collator=default_data_collator,
35     compute_metrics=compute_metrics if training_args.do_eval and not is_torch_tpu_available() else None,
36     preprocess_logits_for_metrics=preprocess_logits_for_metrics if training_args.do_eval and not is_torch_tpu_available() else None,
37 )
38
39 # Training
40 if training_args.do_train:
41     checkpoint = None
42     if training_args.resume_from_checkpoint is not None:
43         checkpoint = training_args.resume_from_checkpoint
44     elif last_checkpoint is not None:
45         checkpoint = last_checkpoint
46     train_result = trainer.train(resume_from_checkpoint=checkpoint)
47     trainer.save_model() # Saves the tokenizer too for easy upload
48
49     metrics = train_result.metrics
50
51     max_train_samples = (
52         data_args.max_train_samples if data_args.max_train_samples is not None else len(train_dataset)
53     )
54     metrics["train_samples"] = min(max_train_samples, len(train_dataset))
55
56     trainer.log_metrics("train", metrics)
57     trainer.save_metrics("train", metrics)
58     trainer.save_state()

```

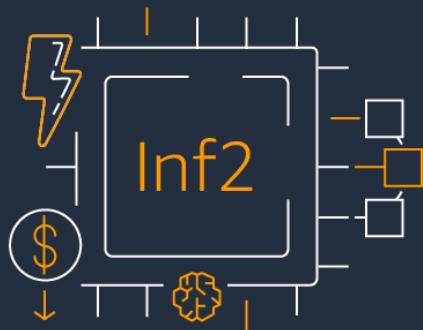


# AWS Inferentia2



# Amazon EC2 Inf2 instances powered by AWS Inferentia2

HIGH PERFORMANCE AT THE LOWEST COST FOR GENERATIVE AI MODELS



Up to 4x higher throughput and 10x lower latency

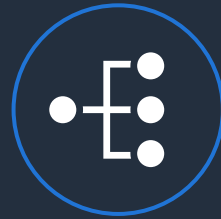
9.8 TB/s aggregated accelerator memory bandwidth

Support for ultra-large generative AI models

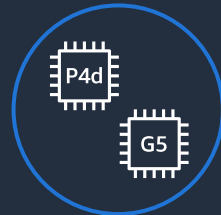
Instance size	vCPUs	Instance memory	Inferentia2 chips	Accelerator memory	NeuronLink	Instance networking	On-demand price
Inf2.xlarge	4	16 GB	1	32 GB	N/A	Up to 15 Gbps	\$0.76/hr
Inf2.8xlarge	32	128 GB	1	32 GB	N/A	Up to 25 Gbps	\$1.97/hr
Inf2.24xlarge	96	384 GB	6	192 GB	Yes	50 Gbps	\$6.49/hr
Inf2.48xlarge	192	768 GB	12	384 GB	Yes	100 Gbps	\$12.98/hr

# Large Model Inference (LMI) container

**Large ML models**  
with 100 billion + parameters



Easily parallelize models across multiple GPUs to fit models into the instance and achieve low latency

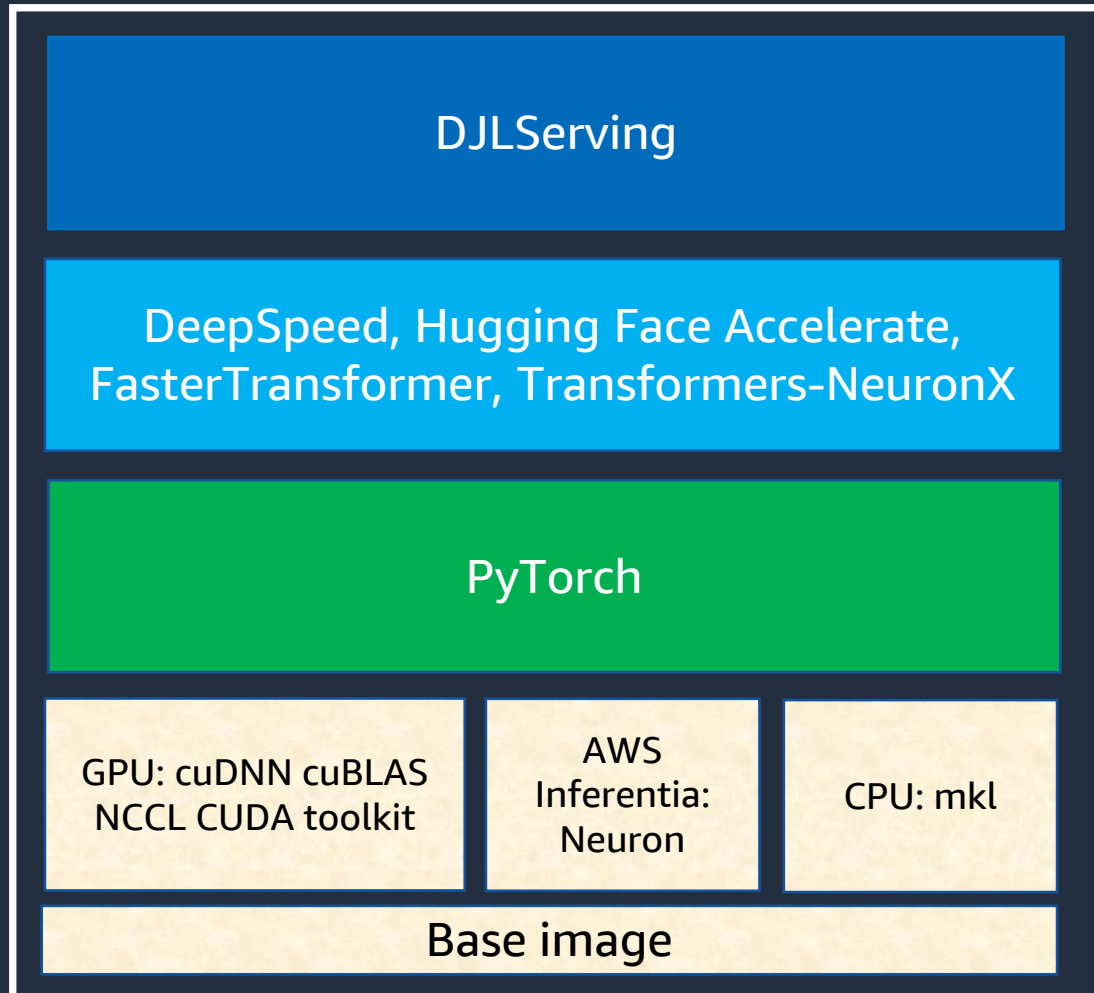


Deploy models on the most performant and cost-effective GPU-based instances or on AWS Inferentia



Pre-built foundation software stack including PyTorch, NCCL, and MPI, Transformers-NeuronX, DeepSpeed, Hugging Face Accelerate, and FasterTransformer

# Large Model Inference (LMI) container on SageMaker



- Zero code setup: [DeepSpeed](#), [StableDiffusion](#) and [HuggingFace](#), FasterTransformer, Transformers-NeuronX Handlers
- Optimized environment with minimal setup (less than 8GB)
- Frameworks Support: HuggingFace Accelerate and DeepSpeed, FasterTransformer, Transformers-NeuronX
- Model Server: DJLServing - Multi-process execution with auto-scaling and UI

# Getting started with Hugging Face Optimum Neuron

## Hugging Face

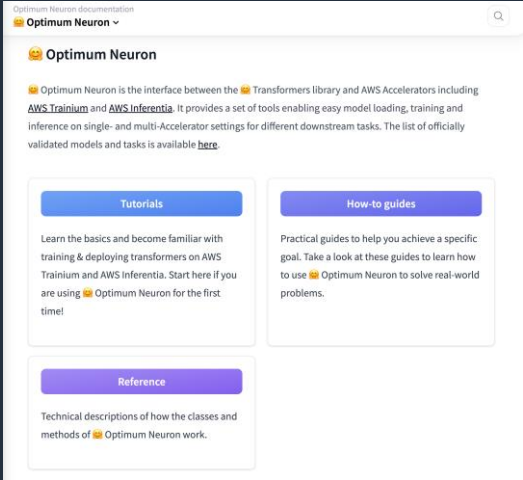


+




## Neuron SDK

## Documentation & tutorials



[Hugging Face documentation](#)

## Code examples



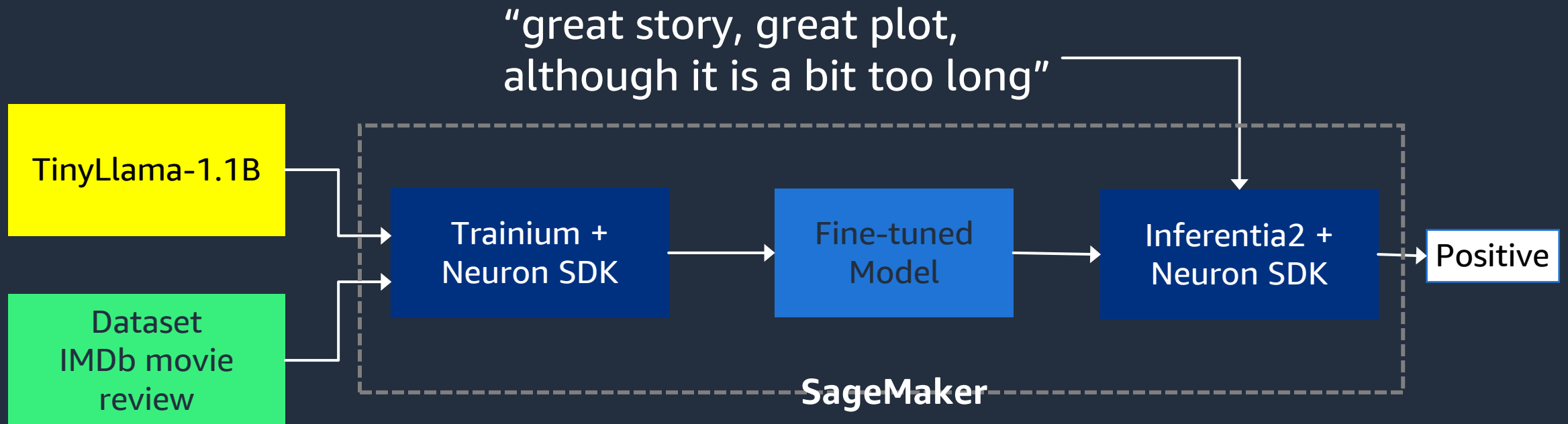
[Access GitHub](#)

# Hands-on Labs



# Getting started with this workshop

Fine-tune TinyLlama-1.1B model on ml.trn1.2xlarge and then deploy the fine-tuned model on ml.inf2.xlarge

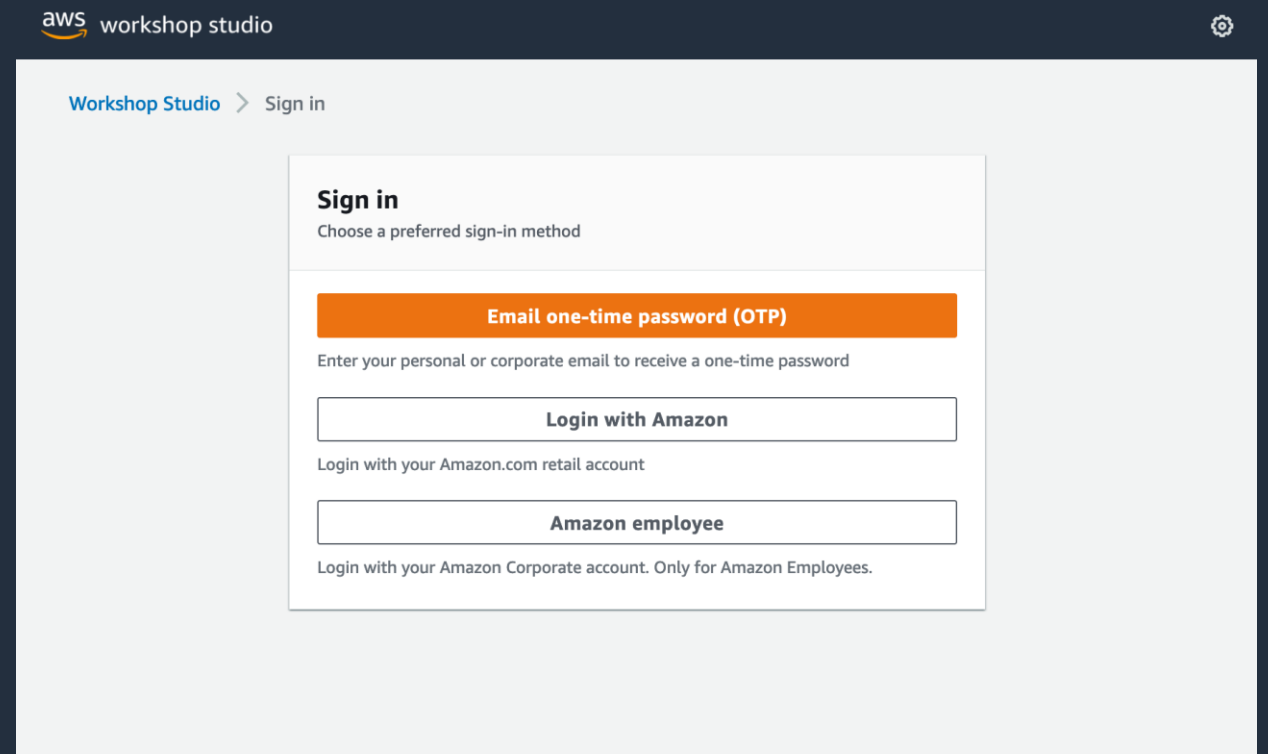


# Getting started with this workshop

- As a participant, you will have access to an AWS account with any optional pre-provisioned infrastructure and AWS Identity and Access Management (IAM) policies needed to complete this workshop
- The AWS account will only be available for the duration of this workshop – You will lose access to the account thereafter
- The optional pre-provisioned infrastructure will be deployed to a specific AWS Region; check your workshop content to determine whether other Regions will be used
- Be sure to review the terms and conditions of the event. Do not upload any personal or confidential information in the account

# Step 1: Sign-in via your preferred method

<http://tinyurl.com/bdd9kf5d>



# Step 2: Enter event access code

- Enter 12 digit event access code - **55fe-0d0eb7-c0**

The screenshot shows the AWS Workshop Studio interface. At the top left is the 'aws workshop studio' logo. On the right, there are settings and user profile icons. The main content area is titled 'Workshop Studio > Join event'. A progress indicator shows 'Step 1 Enter event access code' as the current step and 'Step 2 Review and join' as the next step. The main heading is 'Enter event access code'. Below this is a form box with the title 'Event access code'. Inside the form, it says 'Event access code' and 'A 12 digit code that was given to you for this event', followed by an empty input field. At the bottom right of the form are 'Cancel' and 'Next' buttons. The footer contains copyright information: '© 2008 - 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.' and links for 'Privacy policy' and 'Terms of use'.

# Step 3: Review terms and join event

The screenshot shows the AWS Workshop Studio interface for joining an event. The page is titled 'Review and join' and is part of a multi-step process. The left sidebar shows 'Step 1: Enter event access code' and 'Step 2: Review and join'. The main content area is divided into three sections: 'Event details', 'Description', and 'Terms and Conditions'. The 'Event details' section contains a table with the following information:

Name	Start time	Duration	Level
AWS General Immersion Day	9/23/2022 01:13 AM	12 hours	-

The 'Description' section contains the text: 'AWS General Immersion Day'. The 'Terms and Conditions' section contains the following text: 'Read and accept before joining the event'. Below this, there are four numbered terms and conditions:

1. By using AWS Workshop Studio for the relevant event, you agree to the AWS Event Terms and Conditions and the AWS Acceptable Use Policy. You acknowledge and agree that are using an AWS-owned account that you can only access for the duration of the relevant event. If you find residual resources or materials in the AWS-owned account, you will make us aware and cease use of the account. AWS reserves the right to terminate the account and delete the contents at any time.
2. You will not: (a) process or run any operation on any data other than test data sets or lab-approved materials by AWS, and (b) copy, import, export or otherwise create derivate works of materials provided by AWS, including but not limited to, data sets.
3. AWS is under no obligation to enable the transmission of your materials through Event Engine and may, in its discretion, edit, block, refuse to post, or remove your materials at any time.
4. Your use of AWS Workshop Studio will comply with these terms and all applicable laws, and your access to AWS Workshop Studio will immediately and automatically terminate if you do not comply with any of these terms or conditions.

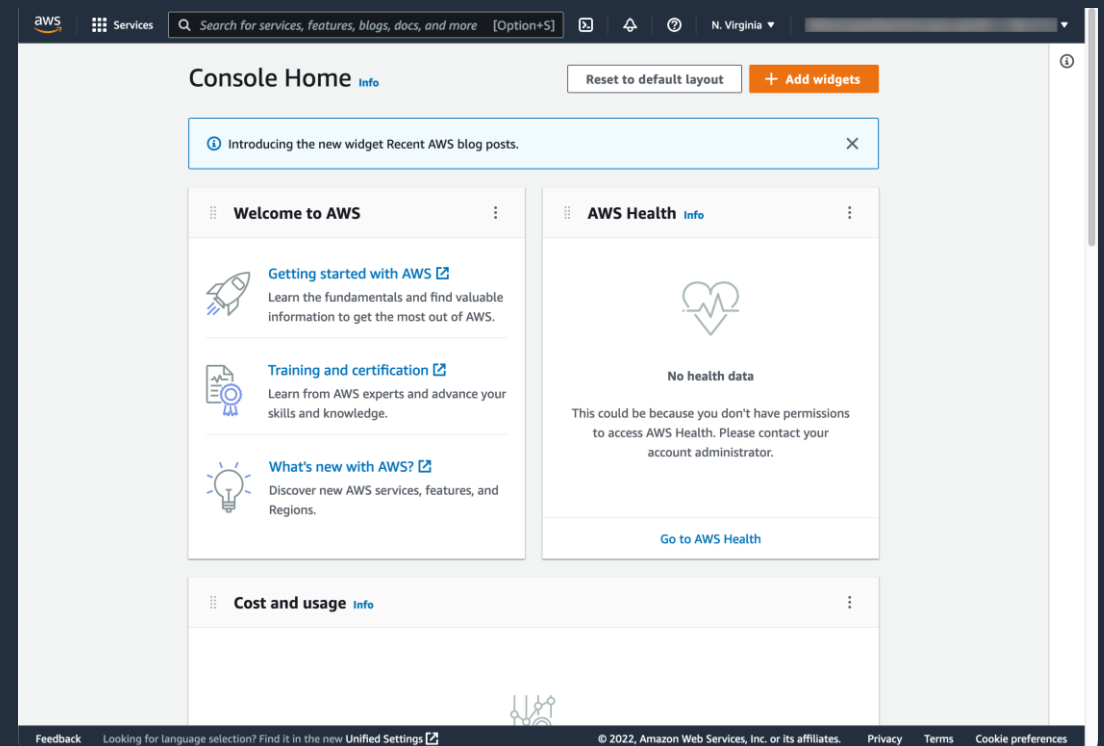
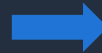
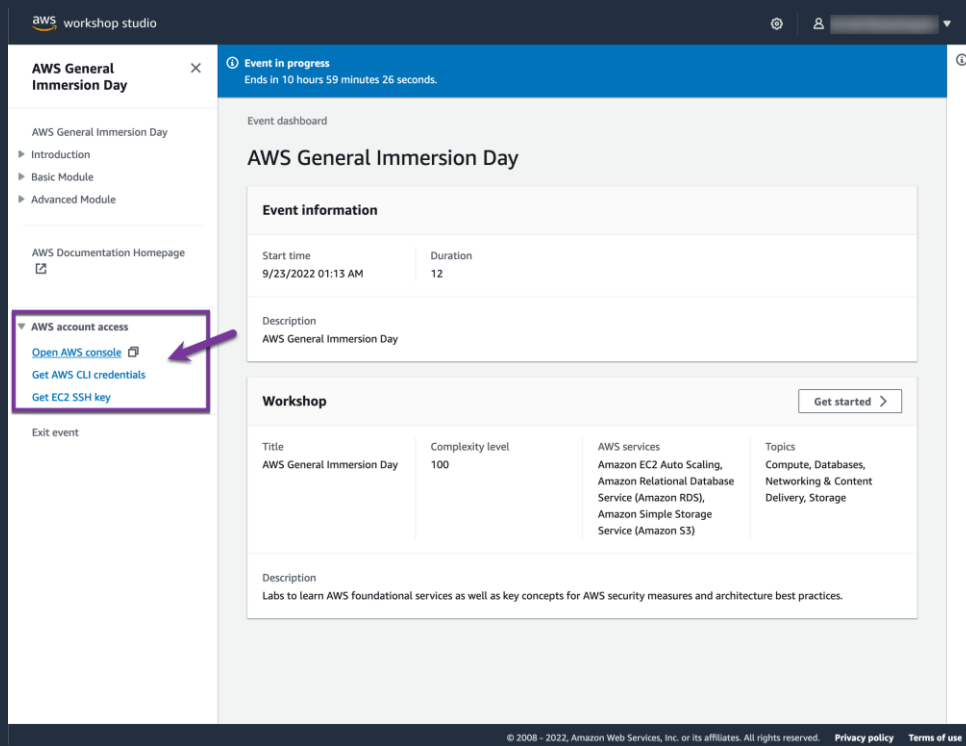
At the bottom of the 'Terms and Conditions' section, there is a checkbox labeled 'I agree with the Terms and Conditions' which is checked.

At the bottom right of the page, there are three buttons: 'Cancel', 'Previous', and 'Join event'.

© 2008 - 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy policy](#) [Terms of use](#)

# Step 4: Access AWS account

Access the AWS Console, or generate AWS CLI credentials as needed



# Step 5: Get started with the workshop

Event dashboard

## AWS General Immersion Day

**Event information**

Start time	Duration
9/23/2022 01:13 AM	12

Description  
AWS General Immersion Day

**Workshop**

Title	Complexity level	AWS services	Topics
AWS General Immersion Day	100	Amazon EC2 Auto Scaling, Amazon Relational Database Service (Amazon RDS), Amazon Simple Storage Service (Amazon S3)	Compute, Databases, Networking & Content Delivery, Storage

Description  
Labs to learn AWS foundational services as well as key concepts for AWS security measures and architecture best practices.

[Get started >](#)



Event dashboard > AWS General Immersion Day

## AWS General Immersion Day

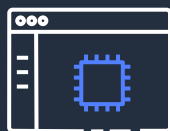
**IMMERSION DAYS**

In this General Immersion Day workshop, through a mix of service explanation and hands-on labs led by AWS, you will learn about AWS foundational services as well as key concepts for AWS security measures and architecture best practices. The hands-on labs are largely divided into **basic and advanced modules**.

In basic modules, you can learn various features of each AWS foundational service. In advanced modules, you can learn how to connect each service organically to create architecture like 3-tier web application.

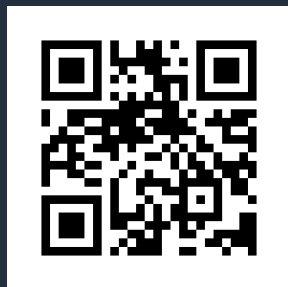
[Previous](#) [Next](#)

# Resources



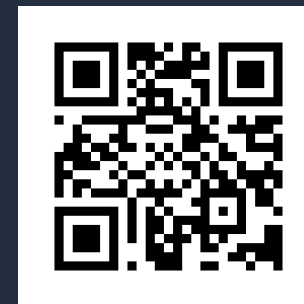
[AWS Neuron SDK Documentation](#)

---



[Neuron Containers Documentation](#)

---



[GitHub tutorials/project](#)

---



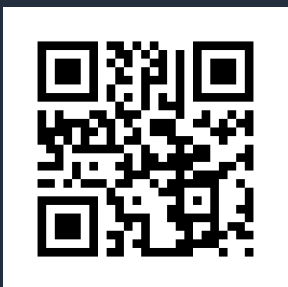
[Getting Started with PyTorch Neuron](#)

---



[AWS Neuron forum and blogs](#)

---



[Getting Started with TensorFlow Neuron](#)

---



# Q&A





# Thank you!



Luca Perrozzi

Solutions Architect

AWS

Dmitri Laptev

Solutions Architect

AWS

Roy Allela

AIML Specialist Solutions Architect  
AWS



Please complete the session survey.



# Backup



# Tensor parallel model

ALSO CALLED INTRA-LAYER MODEL PARALLELISM

