
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

AIM404

Train and host foundation models with PyTorch on AWS

Aditya Bindal
Principal Product Manager – Tech
Amazon Web Services

Youngjoon Choi
AI/ML Expert SA
Amazon Web Services

Gwangmo Song
Research Scientist, Ph.D.
LG AI Research

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How did we get here?



Source: youtube.com/corridordigital

Studio-quality movie made with Stable Diffusion

Fine-tuning with a small sample of examples

Created primarily with text prompts as inputs

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Source: https://xkcd.com/1425/

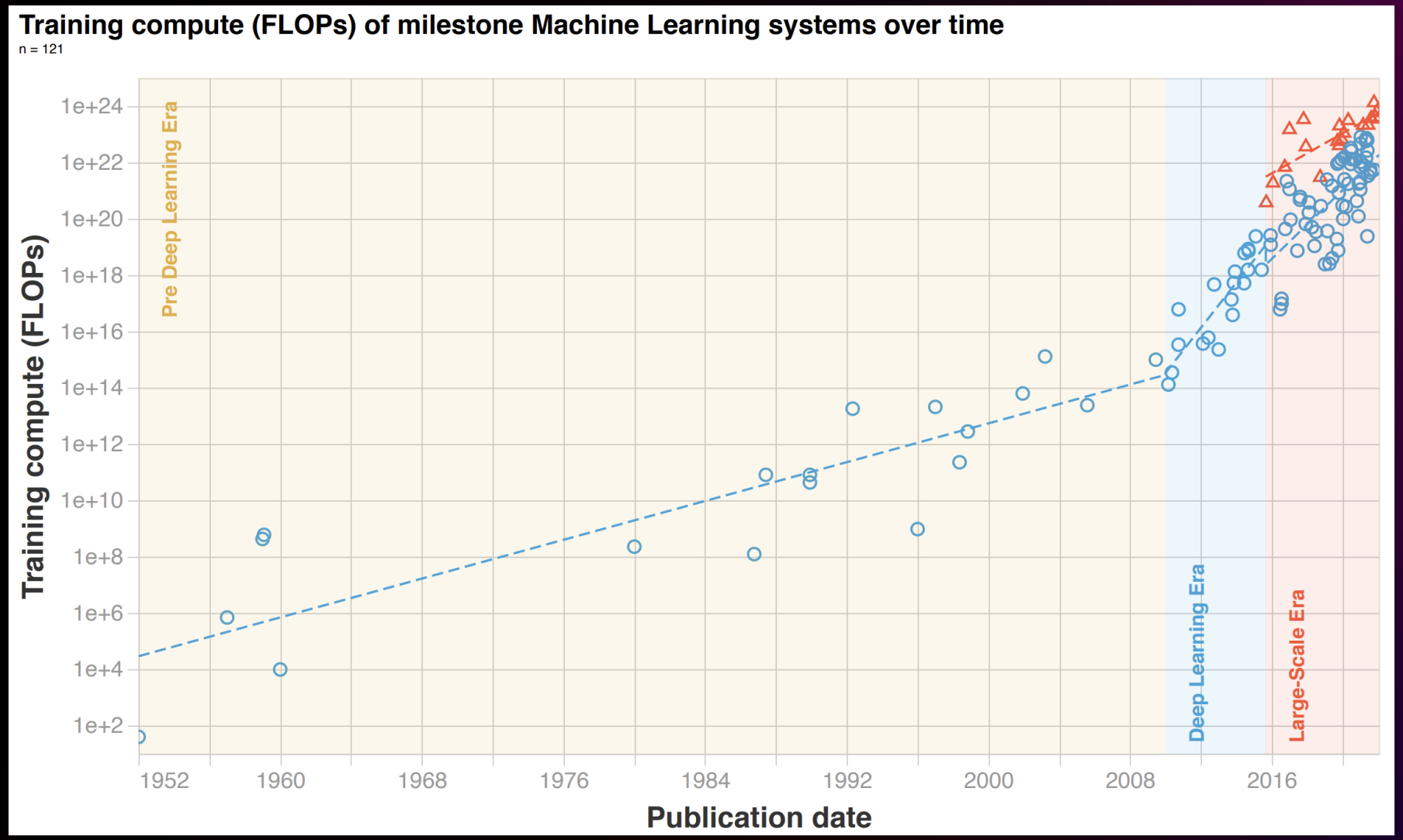
2014 → 2022



Source: "Dynabench: Rethinking Benchmarking in NLP," https://arxiv.org/abs/2104.14337

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

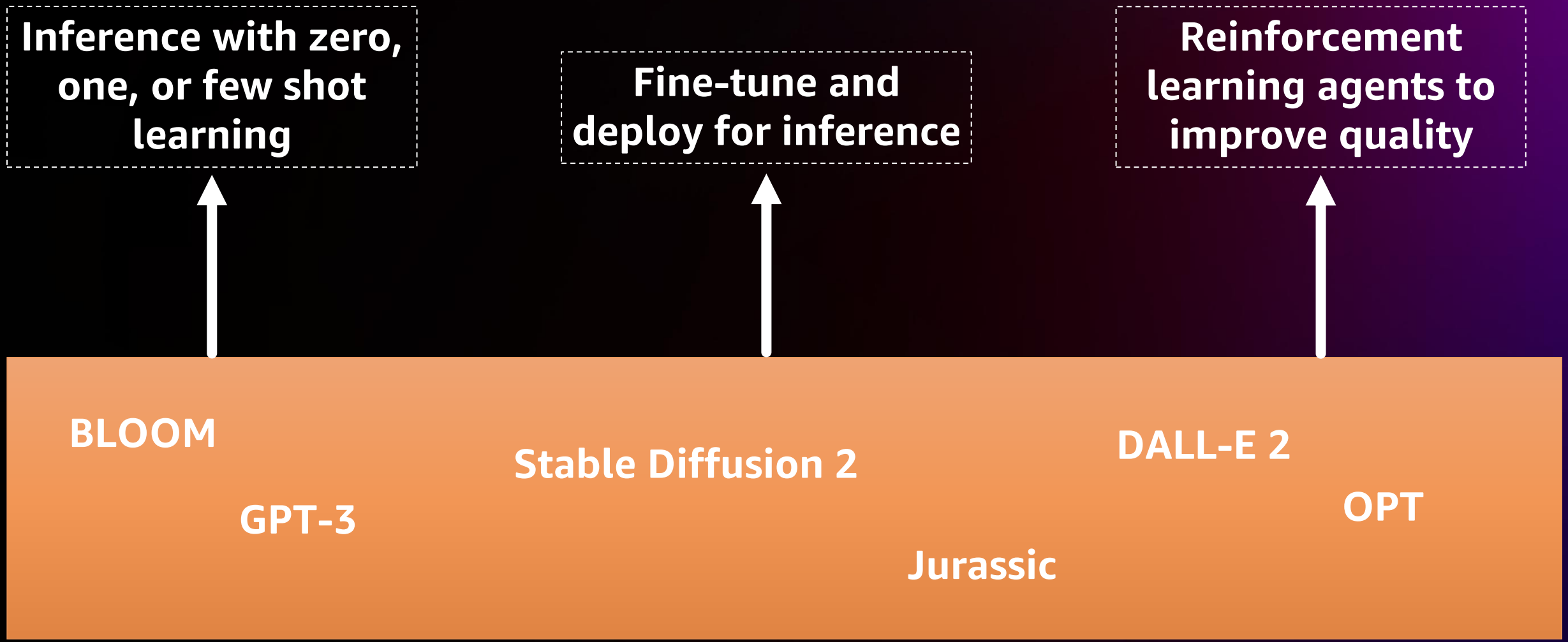
Scaling laws at work



Source: “Compute Trends Across Three Eras of Machine Learning,” https://arxiv.org/abs/2202.05924

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Foundation models



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Foundation models

Inference with zero, one, or few shot learning

+ Single API call, no infra

+ No data needed for fine-tuning

- Harder to customize, depending on use-case

- Harder to have memory, context

Fine-tune and deploy for inference

+ Easier to customize

+ Distill and lower costs, improve latency

- Requires infra, data, and specialized skills

- Model drift

Reinforcement learning agents to improve quality

+ Improve quality, customize without fine-tuning

+ Human-in-the-loop helps with QA and validation

+ Scale via UI to end-users

- Relatively new, less mature (this may change in 6 months)

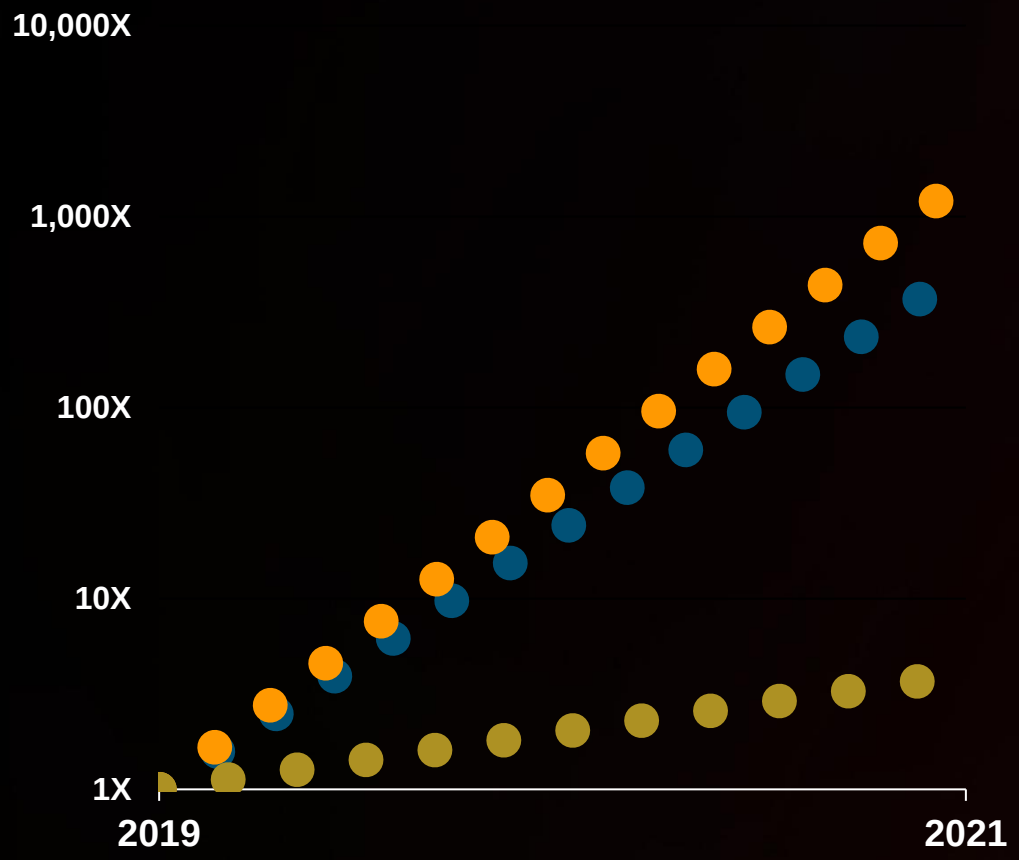
- Needs more UI and product integration

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Need for software and framework optimization

ML model complexity continues to grow exponentially, requiring more hardware and software optimizations



GPU compute requirements grew by 1,538x

State-of-the-art models went from requiring 22 to 35,000 GPU-days

Model complexity grew by 530x

State-of-the-art models went from having 330 Million to 175 Billion parameters

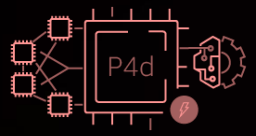
Hardware performance grew by 4x

NVIDIA A100 GPUs improved performance by 4x over V100 GPUs.

aws

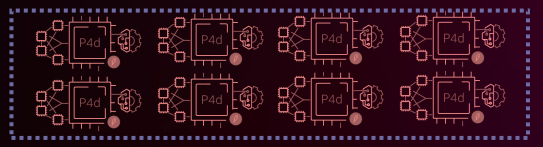
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Optimizing model performance at scale



Single-node

- **50% faster training** for TensorFlow language models (e.g., GPT-2)
- **54% faster training** for PyTorch language models
- **55% faster inference** for most popular TensorFlow and PyTorch Computer Vision and Language models



Distributed

- **Up to 58% faster GPT-NeoX performance on 64 instances**
- **Up to 23% faster training** for TensorFlow language models scaling up to 256 instances
- **Up to 60% faster training** for PyTorch computer vision and language models scaling up to 256 instances

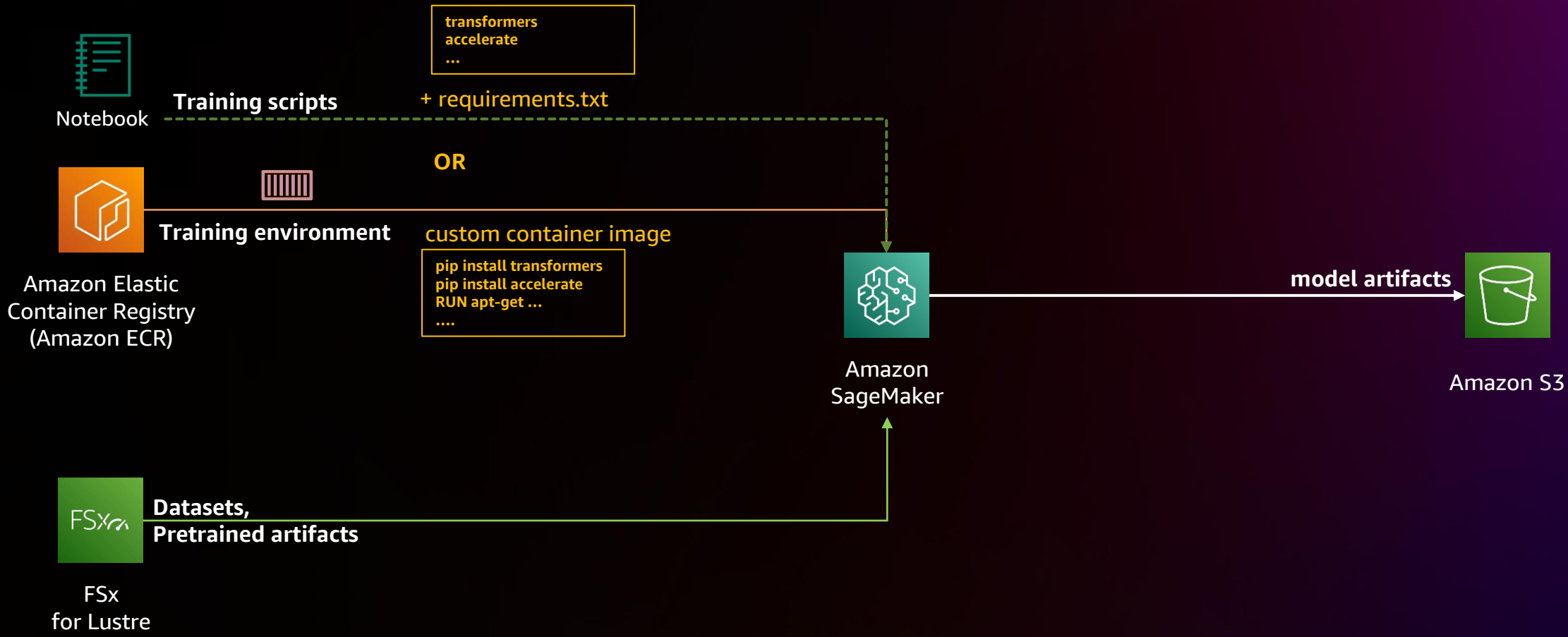
THOMSON REUTERS deepset stability.ai LG HYUNDAI

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Training environment of foundation models

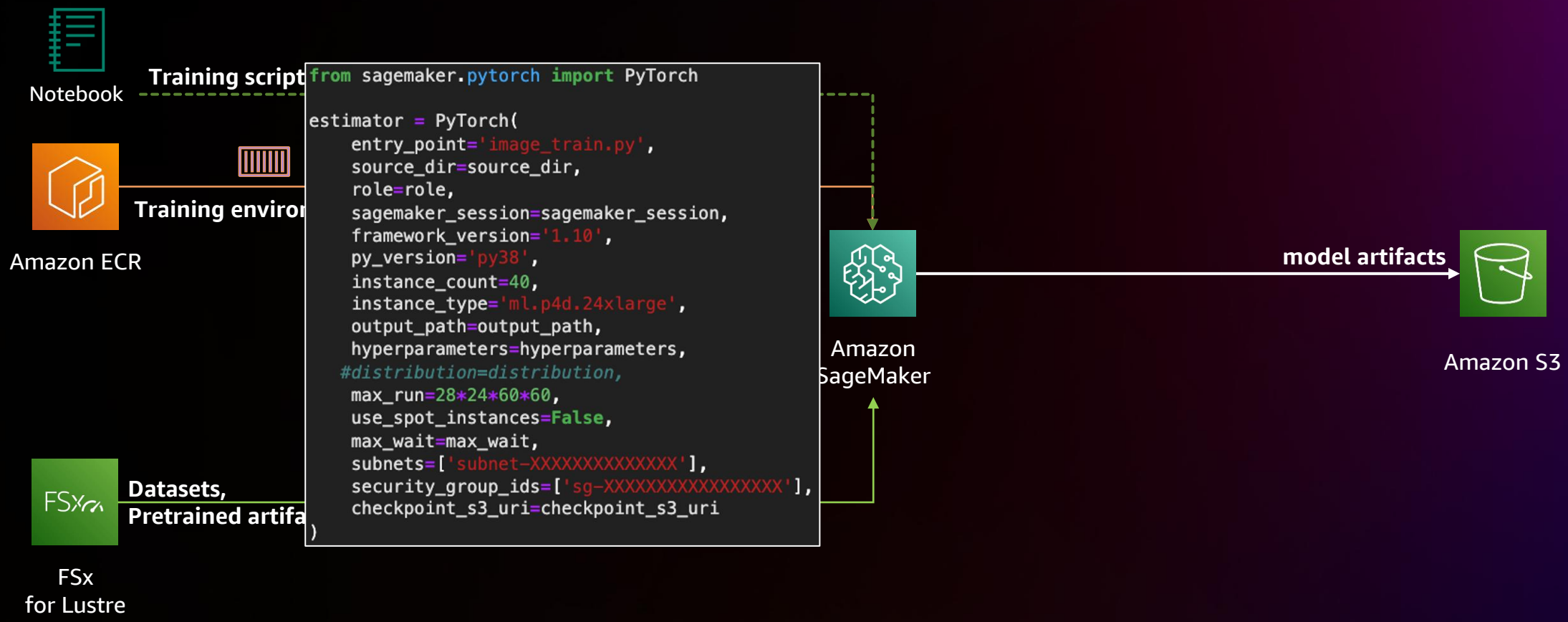
Data scientist-friendly training environment for foundation models



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Training environment of foundation models

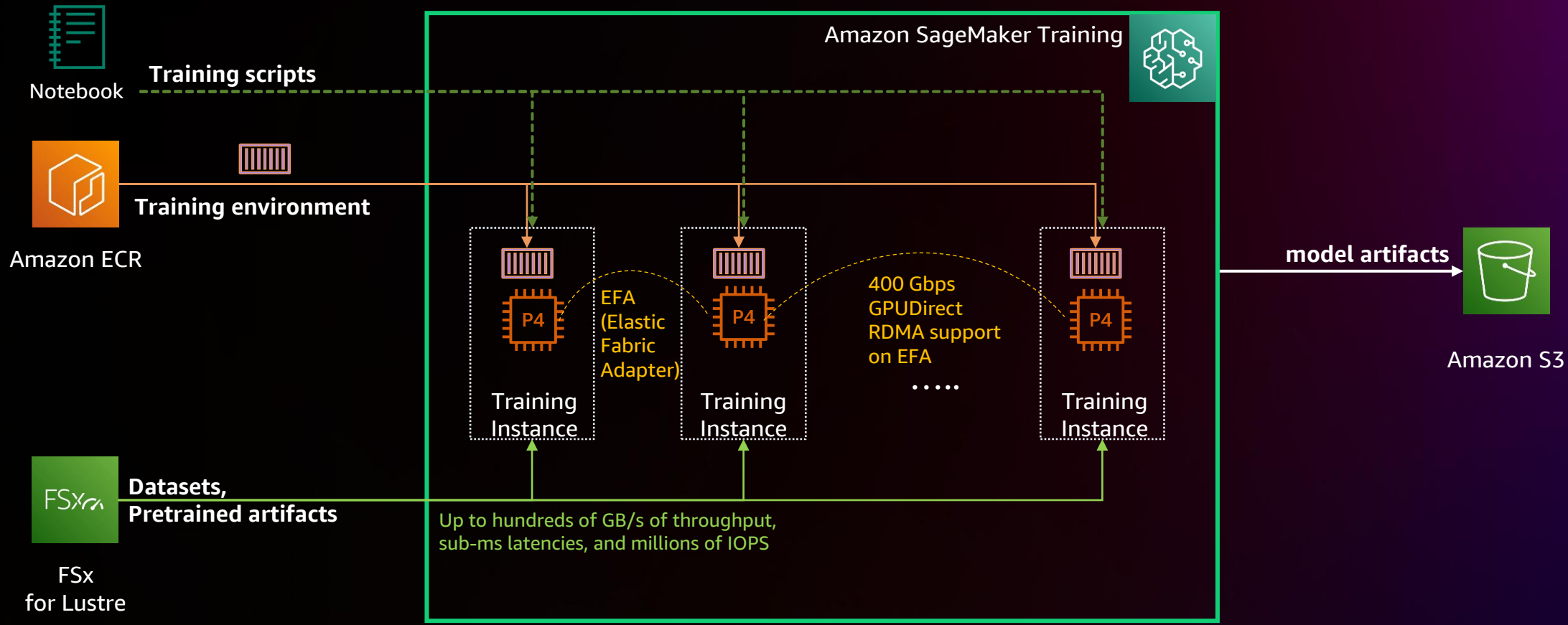
Data scientists can make a training environment only using python



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Training environment of foundation models

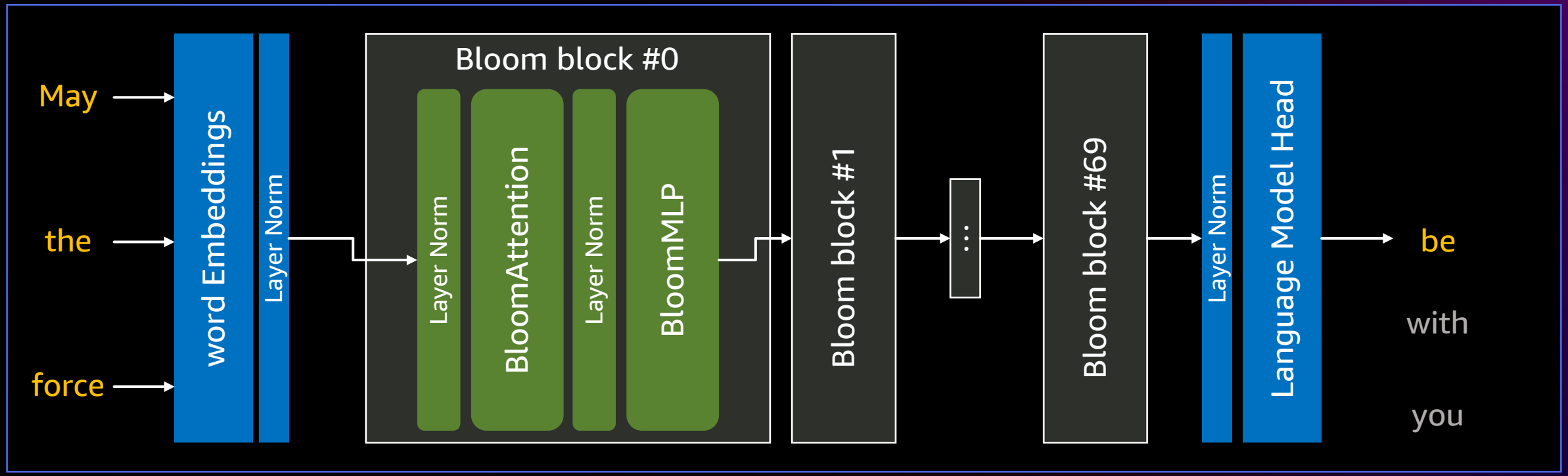
No manual infrastructure configuration required for training foundation models



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introduction to the Bloom model

- An autoregressive Large Language Model (LLM) with up to 176B parameters
- Similar to GPT3, but has been trained in 46 different languages and 13 programming languages



reference : https://huggingface.co/docs/transformers/model_doc/bloom,
https://towardsdatascience.com/run-bloom-the-largest-open-access-ai-model-on-your-desktop-computer-f48e1e2a9a32

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Application of the Bloom model

Language Generation

It is a truth universally acknowledged, that a single man in possession of a good fortune, must be in want of a wife."

It is a truth universally acknowledged, that a single man in possession of a good fortune, must be in want of a wife."
As you get a little older, you realize what is truly important in life and where your priorities lie. Finding your true love is a priority, and in some cases, it can be more important than anything else. It doesn't matter if you are rich, poor, skinny, fat,

Language translation (En – Code)

Write a code snippet with O(log(n)) computational complexity.

Write a code snippet with O(log(n)) computational complexity.

```
def binary_search(a, x):
  low = 0
  high = len(a) - 1
  while low <= high:
    mid = (low + high) // 2
```

Question-answering

Which book contains this sentence that it is a truth universally acknowledged, that a single man in possession of a good fortune, must be in want of a wife?

Which book contains this sentence that it is a truth universally acknowledged, that a single man in possession of a good fortune, must be in want of a wife? Pride and prejudice

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Training the Bloom model with HuggingFace

Hugging Face

```
raw_datasets = load_dataset(...)
tokenizer = BloomTokenizerFast.from_pretrained(...)
tokenized_datasets = raw_datasets.map(...)
lm_datasets = tokenized_datasets.map(...)
lm_datasets.save_to_disk(data_output)
```

```
BloomForCausalLM.from_pretrained('bigscience/bloom')
```

1. Preprocessing Step
   Downloading/making dataset with SageMaker Notebook or Processing Job, and moving it to FSx for Lustre
2. Training Step
   Running training job



FSx
for Lustre



Amazon
SageMaker

```
lm_datasets = load_from_disk(model_args.cache_dir)
```

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Setting data/model paths with FSx for Lustre

Set the FSx/L as a data source and cache directory

```
trainingdata = FileSystemInput(
                    file_system_id=file_system_id,
                              file_system_type=file_system_type,
                              directory_path=fsx_data_path,
                              file_system_access_mode="ro")

model = FileSystemInput(
                    file_system_id=file_system_id,
                    file_system_type=file_system_type,
                                        directory_path=fsx_model_path,
                                        file_system_access_mode="ro")

data_channels = {”training_data": trainingdata, ”model_weight": model}
```

Set data source and cache directory in training scripts

```
hyperparameters = {
          'model_name_or_path' : '/opt/ml/input/data/model_weight/’,
          'train_file' : '/opt/ml/input/data/trainingdata/XXXXXX.json’,
          'cache_dir' : '/opt/ml/input/data/training_data/cache_dir’,
          …}
```

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

HuggingFace with SageMaker Model Parallel

Set the distribution parameter of SageMaker Model Parallel (SMP)

- pipeline_parallel_degree : The No. of partitions to split the model
- tensor_parallel_degree : The No. of GPUs over which the TP[1)] modules will be distributed
- microbatches : The No. of microbatches to perform pipelining over.

1) TP : Tensor Parallel

```
distribution = {}

# SageMaker Model Parallel 설정 (SageMaker MP)
distribution["smdistributed"]={
    "modelparallel": {
        "enabled": True,
        "parameters": {
            "tensor_parallel_degree": 4,
            "pipeline_parallel_degree": 1,
            "microbatches": 12,
            "ddp": True,
        }
    }
}
```

```
estimator = PyTorch(
        entry_point="run_clm.py",

source_dir='./transformers/examples/pytorch/language-modeling',
        role=role,
        instance_type="ml.p4d.24xlarge",
        instance_count=32,
        sagemaker_session=sagemaker_session,
        distribution=distribution,
        framework_version='1.12',
        py_version='py38',
        hyperparameters=hyperparameters,
        security_group_ids=["sg-xxxxxxxxxxxx"],
        subnets=["subnet-xxxxxxxxxxxx"]
  )
```

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

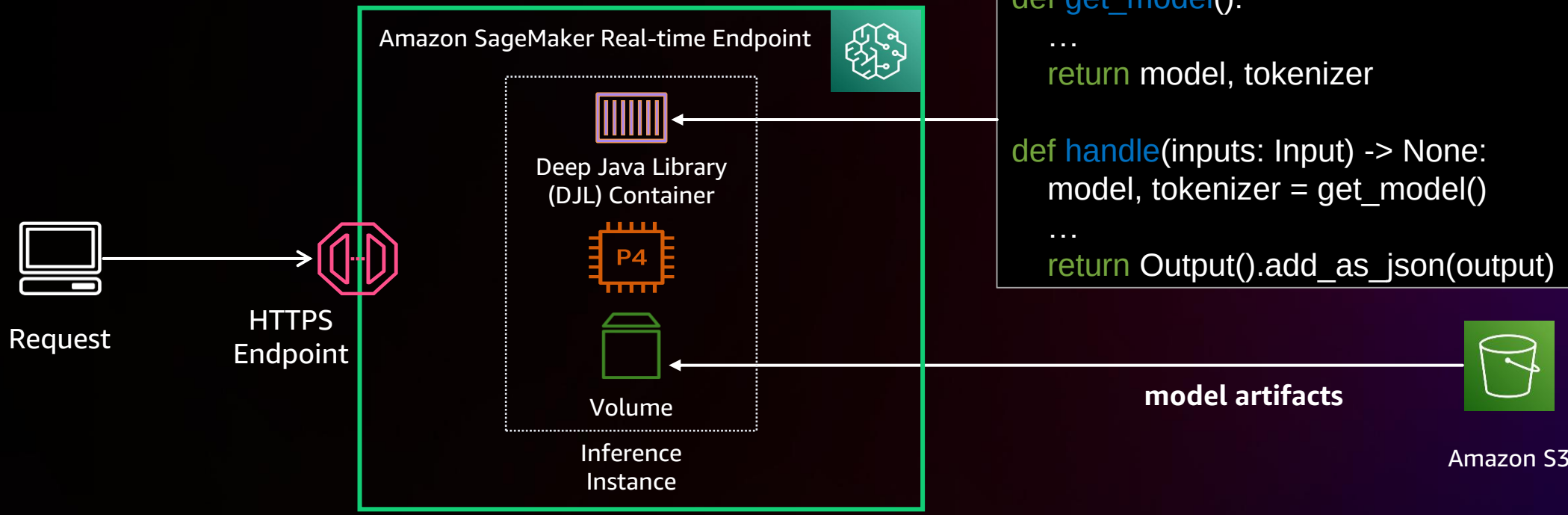
Hosting foundation models on SageMaker

- Serving SageMaker endpoints with EBS volumes up to 500 GB
- Configurable up to 60 minutes for container health check and download timeout

Custom inference.py (optional)

```
from djl_python import Input, Output
def get_model():
    …
    return model, tokenizer

def handle(inputs: Input) -> None:
    model, tokenizer = get_model()
    …
    return Output().add_as_json(output)
```



ref: https://huggingface.co/blog/deploy-hugging-face-models-easily-with-amazon-sagemaker
https://aws.amazon.com/ko/blogs/machine-learning/deploy-bloom-176b-and-opt-30b-on-amazon-sagemaker-with-large-model-inference-deep-learning-containers-and-deepspeed/

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Benchmarking performance of SMP

- SageMaker Model Parallel (SMP) achieved 39.7% speed up compared to DeepSpeed ZeRO-3
- The benchmark results of the 30B GPT-2

Model Hyperparameters	Num. of P4d instances (8 GPUs/instance)	Speed (samples/sec)		Speed up Of SMP (%)
		DeepSpeed	SMP	
• Seq. length : 512 • Global batch size : 3072 • FP16	16 (128 GPUs)	142	181.05	27.5
• Seq. length:2048 • Global batch size 1536 • FP16	32 (256 GPUs)	52.6	73.52	39.77

The distribution configuration for Model Parallel libraries

Model Parallel libraries	Seq. length	Parameters
SMP v1.11	512	sharded_data_parallel_degree : 64, Activation checkpointing, gradient_accumulation : 1
	2048	sharded_data_parallel_degree : 128, Activation checkpointing, gradient_accumulation : 1
DeepSpeed v0.7.2	Activation checkpointing, gradient_accumulation_steps:2	

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

ref: https://aws.amazon.com/blogs/machine-learning/train-gigantic-models-with-near-linear-scaling-using-sharded-data-parallelism-on-amazon-sagemaker/

Models Generating Images From Text

Image Generation Models

CLIP, ICML 2021 (PyTorch)

ViLT, ICML 2021 (PyTorch Lightning)

ALBEF, NeurIPS 2021 (PyTorch)

⋮

BLIP, ICML 2022 (PyTorch)

Stable Diffusion, CVPR 2022 (PyTorch Lightning)

L-Verse, CVPR2022 (PyTorch Lightning)

VQ-Diffusion, CVPR 2022 (PyTorch)

For faster training,

Distributed Data Parallelism is used

- SageMaker Distributed Data Parallelism (SMDDP)
- PyTorch DDP
- FairScale

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Pytorch lightning with SageMaker DDP

Training Scripts

```
from pytorch_lightning.strategies import DDPStrategy
import smdistributed.dataparallel.torch.torch_smddp

env = LightningEnvironment()
ddp = DDPStrategy(
                    cluster_environment=env,
                    process_group_backend="smddp",
                    accelerator="gpu”
    )

num_gpus = int(os.environ["SM_NUM_GPUS"])
num_nodes = int(int(os.environ["WORLD_SIZE"])/num_gpus)

trainer = pl.Trainer(
       devices=num_gpus,
       num_nodes=num_nodes,
       strategy=ddp,
…)
```

Training Job Launch

```
distribution = {}

# SageMaker Data Parallel 설정 (SageMaker DDP)
distribution["smdistributed"]={
    "dataparallel": {"enabled": True}
}
```

```
from sagemaker.pytorch import PyTorch

estimator = PyTorch(
    entry_point='image_train.py',
    source_dir=source_dir,
    role=role,
    sagemaker_session=sagemaker_session,
    framework_version='1.10',
    py_version='py38',
    instance_count=40,
    instance_type='ml.p4d.24xlarge',
    output_path=output_path,
    hyperparameters=hyperparameters,
    distribution=distribution,
    max_run=28*24*60*60,
    use_spot_instances=False,
    max_wait=max_wait,
    subnets=['subnet-XXXXXXXXXXXXXX'],
    security_group_ids=['sg-XXXXXXXXXXXXXXXX'],
    checkpoint_s3_uri=checkpoint_s3_bucket
)
```

aws

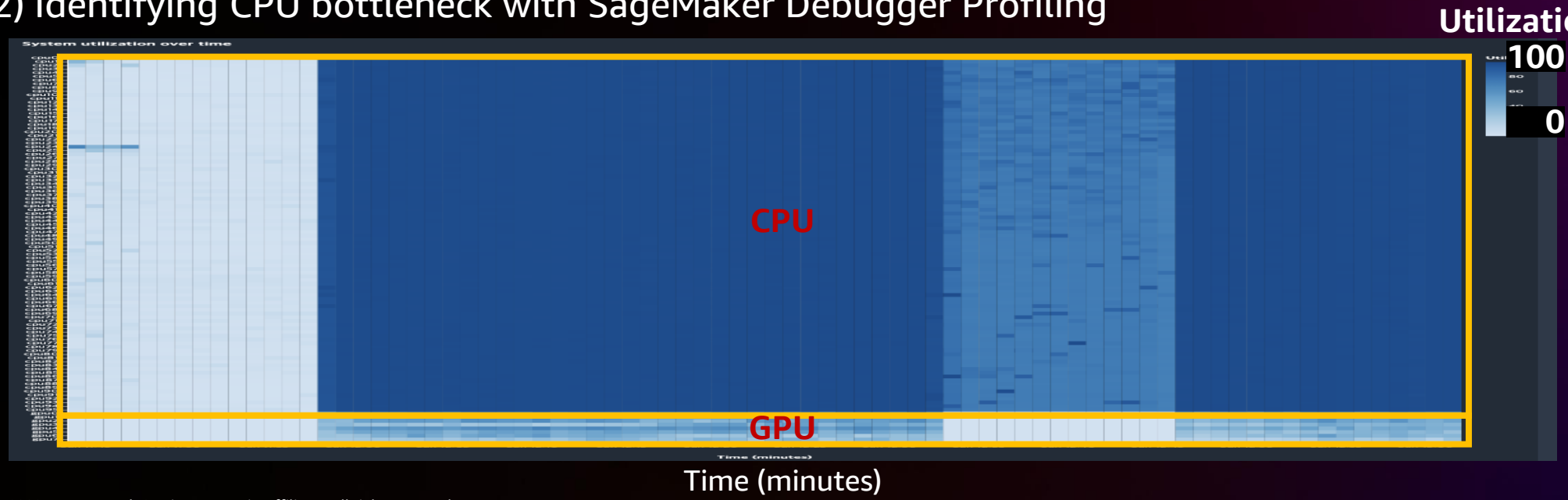
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

CPU resource bottleneck

1) Training Speed

```
Step: [10209][8] Train_Time=1.348, Train_Speed=1898.902 samples/sec,
Step: [10210][8] Train_Time=0.620, Train_Speed=4126.326 samples/sec,
Step: [10211][8] Train_Time=0.619, Train_Speed=4132.699 samples/sec,
Step: [10212][8] Train_Time=14.027, Train_Speed=182.510 samples/sec,
Step: [10213][8] Train_Time=0.618, Train_Speed=4143.007 samples/sec,
Step: [10214][8] Train_Time=10.006, Train_Speed=255.856 samples/sec,
```

2) Identifying CPU bottleneck with SageMaker Debugger Profiling



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Mitigation technique : Nvidia DALI

- NVIDIA Data Loading Library (DALI)

Data Loading Process	Original Pipeline	DALI Pipeline
Open, (seek), read and (close)	CPU	CPU
Decode, resize, gray scale normalize, etc.	CPU	GPU
Random resize and crop, color jitter, add noise, etc.	CPU	GPU
Convert to tensor and put the data on GPU	GPU	GPU

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How to use the Nvidia DALI

- How to convert to DALI from Pytorch dataloader

Native PyTorch dataloader

```
train_loader =
torch.utils.data.DataLoader(…)
prefetcher =
data_prefetcher(train_loader)
input, target = prefetcher.next()

while input is not None:
    (…)
    input, target = prefetcher.next()
```



DALI iterator

```
dali_pipe = TrainPipe(…)
train_loader =
DALIClassificationIterator(dali_pipe)

for i, data in enumerate(train_loader):
    input = data[0]["data"]
    target = data[0]["label"],squeeze()
    (…)
```

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Results of mitigation techniques

1) Before using mitigation techniques

```
Step: [10209][8] Train_Time=1.348, Train_Speed=1898.902 samples/sec,
Step: [10210][8] Train_Time=0.620, Train_Speed=4126.326 samples/sec,
Step: [10211][8] Train_Time=0.619, Train_Speed=4132.699 samples/sec,
Step: [10212][8] Train_Time=14.027, Train_Speed=182.510 samples/sec,
Step: [10213][8] Train_Time=0.618, Train_Speed=4143.007 samples/sec,
Step: [10214][8] Train_Time=10.006, Train_Speed=255.856 samples/sec,
```

2) After using mitigation techniques

```
Step: [10495][8] Train_Time=0.617, Train_Speed=4152.188 samples/sec,
Step: [10496][8] Train_Time=0.614, Train_Speed=4170.514 samples/sec,
Step: [10497][8] Train_Time=0.622, Train_Speed=4116.202 samples/sec,
Step: [10498][8] Train_Time=0.622, Train_Speed=4116.785 samples/sec,
Step: [10499][8] Train_Time=0.618, Train_Speed=4141.851 samples/sec,
Step: [10500][8] Train_Time=0.613, Train_Speed=4173.294 samples/sec,
```

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

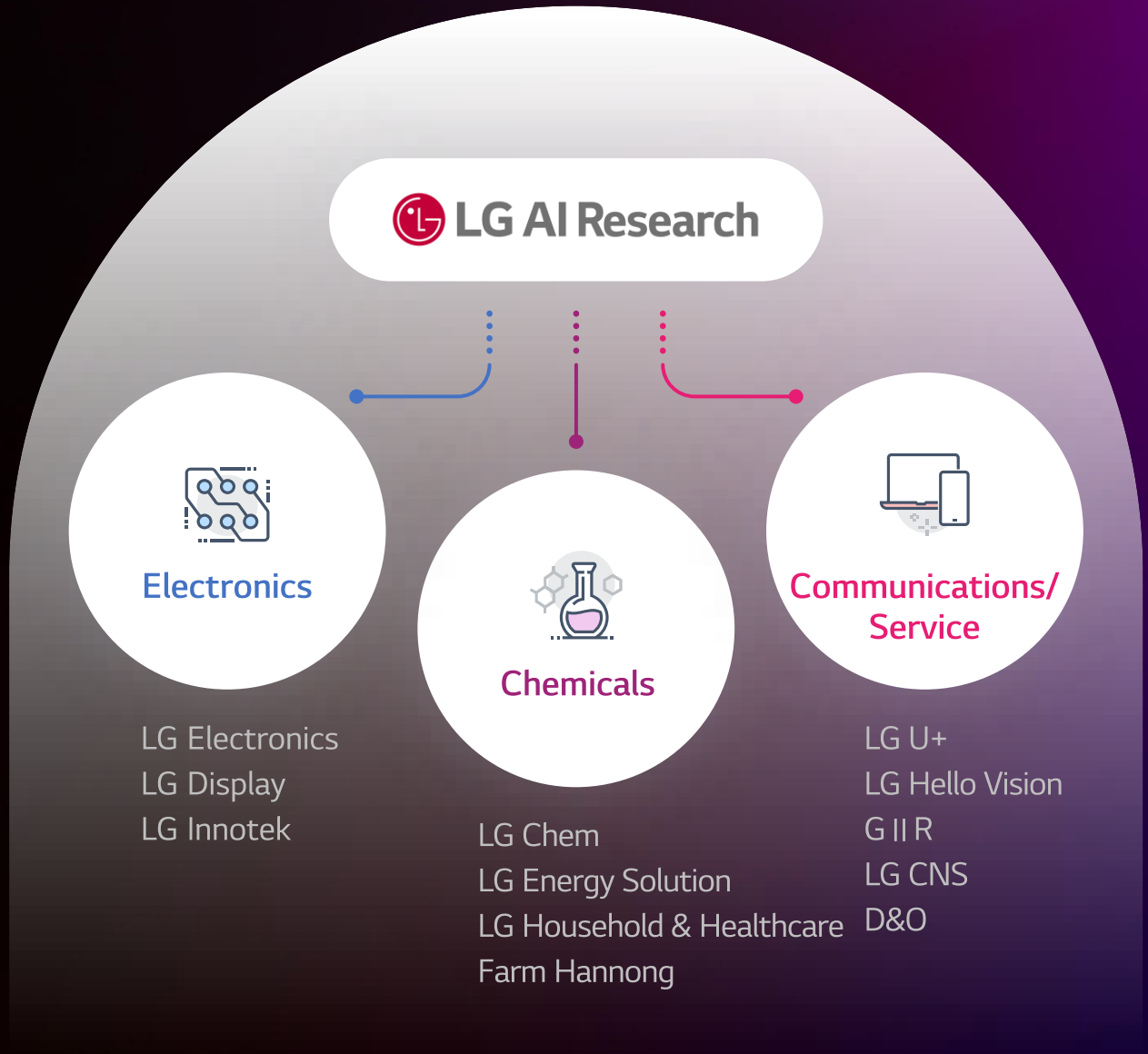
Customer Story: LG AI Research

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

LG AI Research

Advancing AI *for a* better Life

LG AI Research leads the strengthening of AI capabilities as an affiliate of the LG Group



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

LG AI Research

Advancing AI *for a* better Life

Using the latest AI, we solve issues
that can bring a big change in our lives
but are tricky to solve

Energy
Development of future electrical energy solutions


Electric Vehicle Battery



Energy Storage System

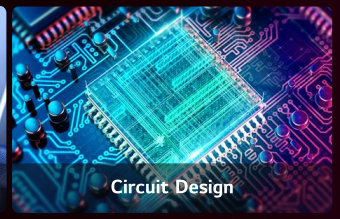
Materials
Discovery of biomaterials/new drugs/new materials


Anti-Cancer Drug


OLED Materials

Products
Innovation in product design and production process


Vision Inspection


Circuit Design

Optimal Decision
R&D strategy and Logistics optimization



Document Understanding


Logistics Optimization

Communication
Natural and more convenient communication


Machine Reading Comprehension


Text Analytics

Customer Service
Advancement of customer services


Recommendation


Customer Support

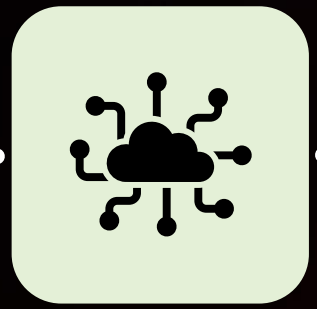
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

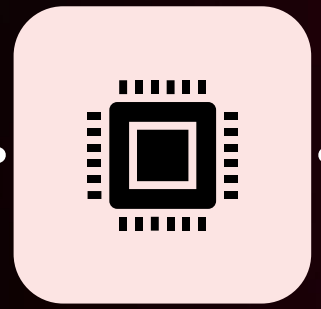
Foundation model



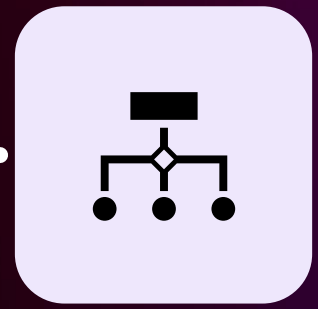
Big Data management



Large-capacity resources



Computing speed



Easy applicability

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Foundation model

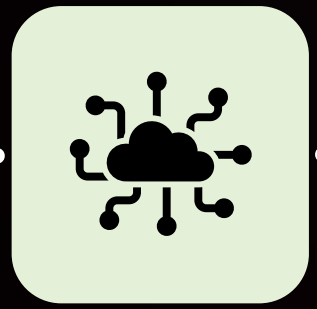


Big data management





Amazon FSx for Lustre

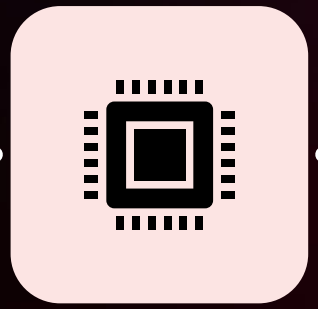


Large-capacity resources





Amazon SageMaker

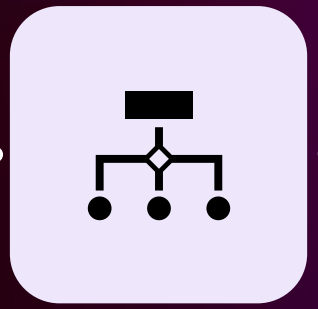


Computing speed



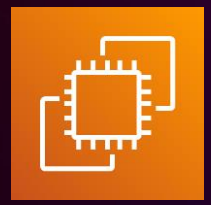


SageMaker data parallel



Easy applicability

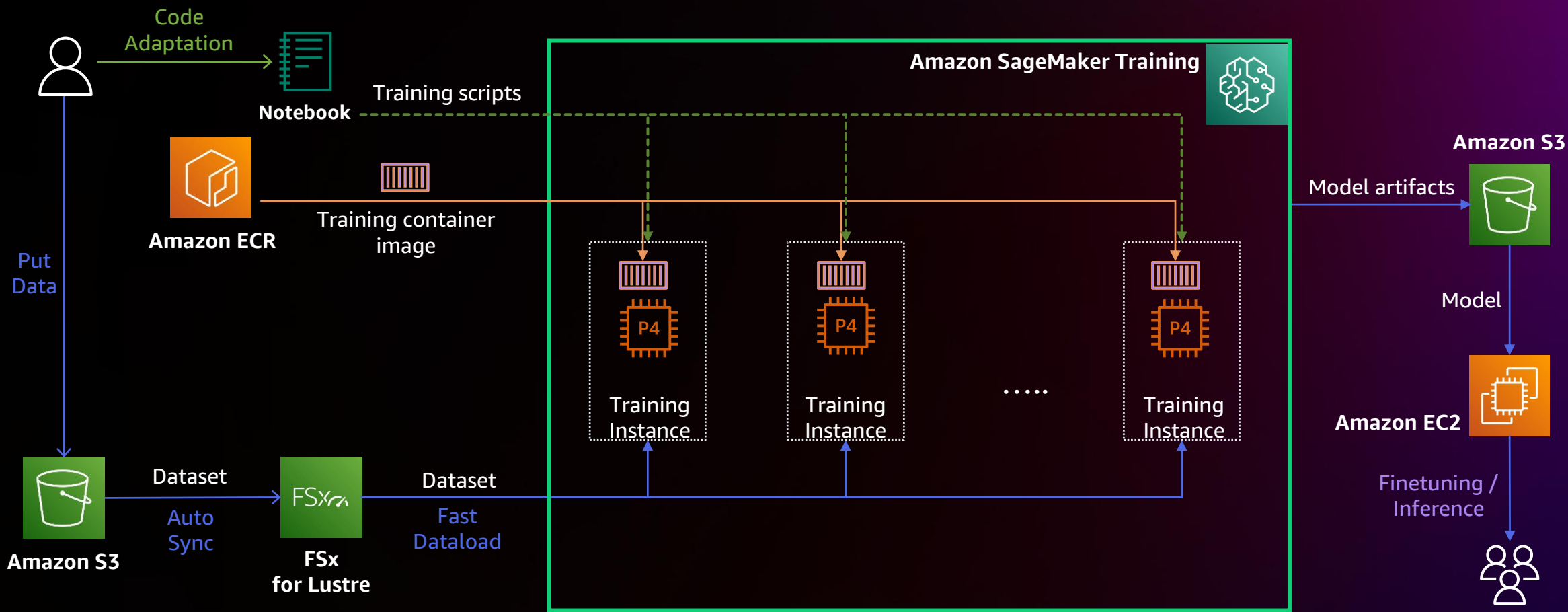




Amazon EC2

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Training Architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

SageMaker Distributed Data Parallel (SMDDP)

Notebook

```
distribution["smdistributed"]={
        "dataparallel": {
                "enabled": True
        }
}

estimator = PyTorch(
        …,
        distribution=distribution,
        …,
)
```

Training Script

```
import torch.distributed as dist
from torch.nn.parallel import DistributedDataParallel as DDP

import smdistributed.dataparallel.torch.torch_smddp
dist.init_process_group(backend='smddp')

rank = str(dist.get_rank())
world_size = str(dist.get_world_size())
local_rank = str(dist.get_local_rank())

os.environ["CUDA_VISIBLE_DEVICES"] = local_rank
os.environ["RANK"] = rank
os.environ["LOCAL_RANK"] = local_rank
os.environ["WORLD_SIZE"] = world_size
```

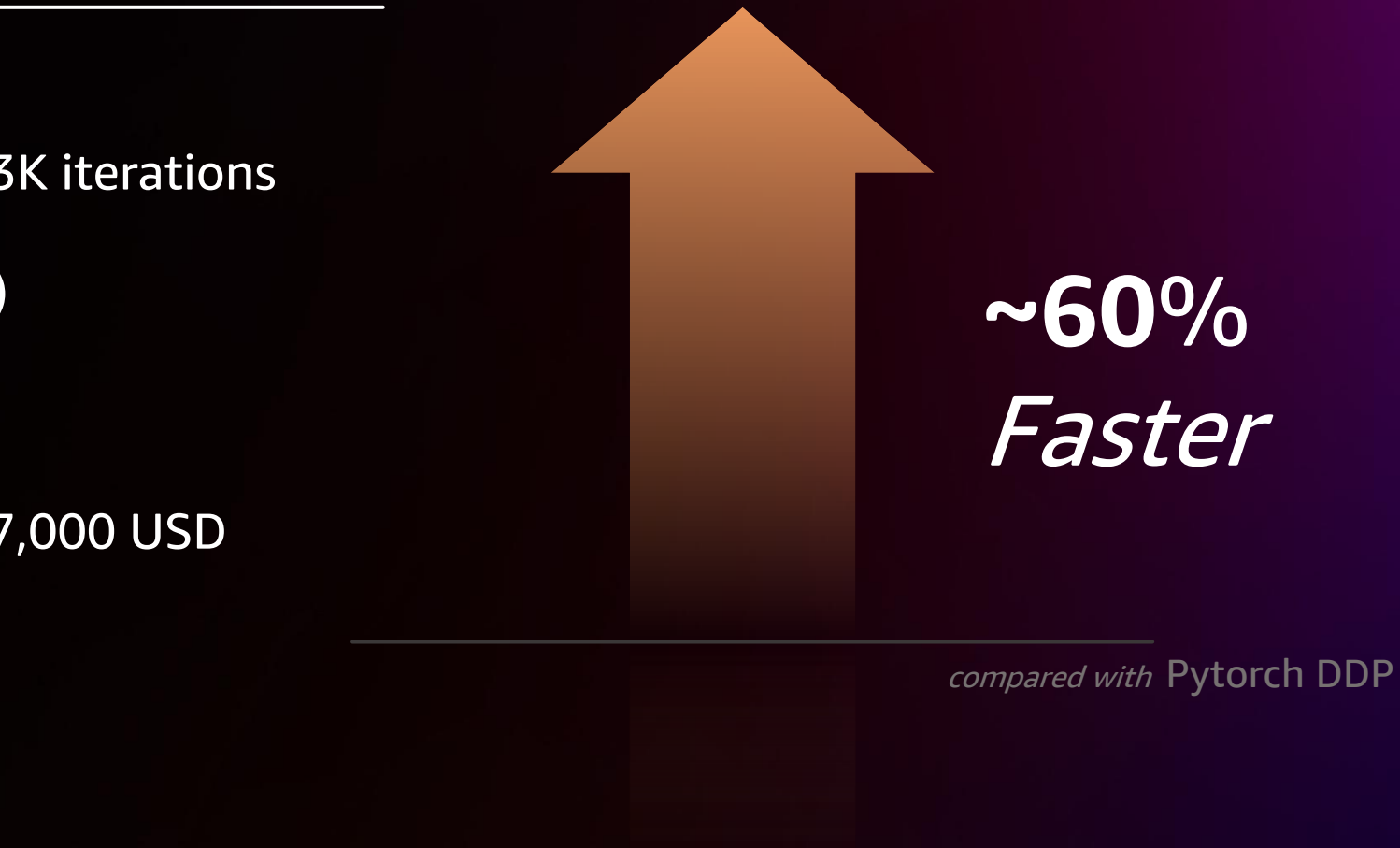
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Benchmarking performance of SMDDP

SMDDP vs. Pytorch DDP

- Training speed (1 day)
  ~10K iterations vs. ~6.3K iterations
- Training speed (1 epoch)
  ~4 days vs. ~6 days
- Training cost (1 epoch)
  ~144,000 USD vs. ~217,000 USD



~60%
Faster

compared with Pytorch DDP

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Our Results

Image Captioning



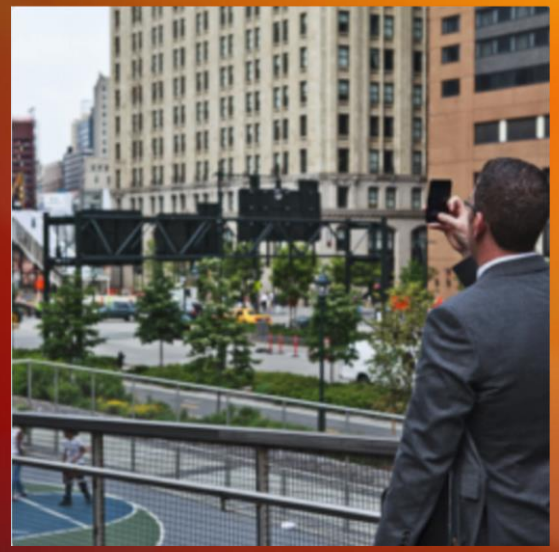
a young boy throwing a green frisbee in a lush green park



a bathroom with a shower curtain over the bathtub next to a toilet



A yellow bird sitting on a branch of a tree



A person walking through the city and taking a picture of buildings

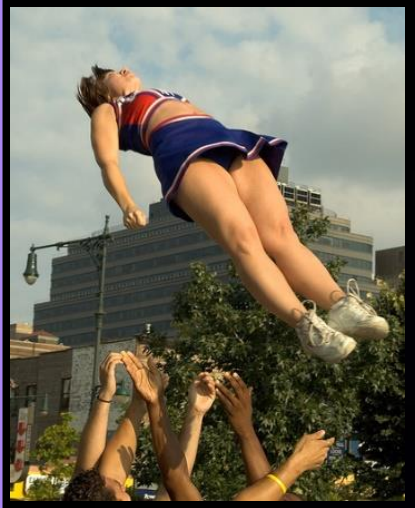
L-Verse: Bi-directional Generation Between Image and Text, CVPR2022 Oral

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Our Results

Image Captioning : Keywords



acrobatic, active, activity, adult, athlete, athletic, attractive, beautiful, body, cardio, competition, diet, energy, enjoy, exercise



background, ball, beautiful, blue, bright, clouds, color, colorful, evening, garden, golf, green, landscape, light, nature, outdoor



ancient, architecture, art, basilica, beguinage, bruges, brugge, building, canal, church, city, cityscape



locomotive, railway, locomotive, railroad, trains

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Our Results

Image Generation

A hat in the shape of the pumpkin



A woman wearing a shirt in pop art style



A close-up view of a tiger sitting in the jungle



Colorful tattoo image of a flying butterfly



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Our Results

Image Generation : Translation

Image Super-resolution



128X128

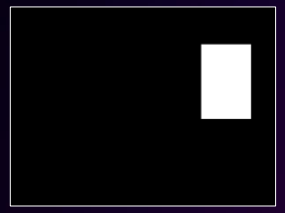


2048X2048

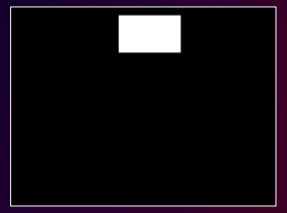
Image Control



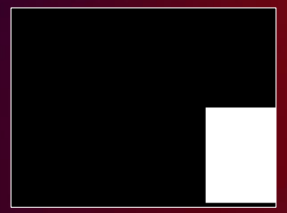
a vase with colorful flowers



a vase with colorful flowers



a blue couch

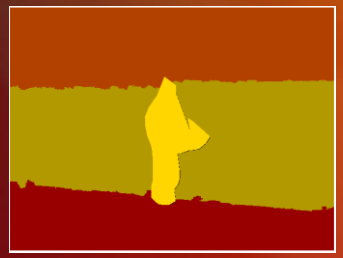








Segmentation Map Guided



a surfer standing on the beach



an elephant





© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Takeaways and next steps

- Convenience of using SageMaker
- Efficiency of SageMaker Distributed Data Parallel
- Close collaboration with the AWS teams
- Utilization Plan for SageMaker Model Parallel
- SageMaker Inference for our future business



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

Aditya Bindal
bindala@amazon.com

Youngjoon Choi
choijoon@amazon.com

Gwangmo Song
gwangmo.song@lgresearch.ai



Please complete the session survey in the **mobile app**

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.