GENERATIVE AI IMMERSION DAY

Implementing Generative AI in Organizations

Challenges and Opportunities

Aris Tsakpinis

AI/ML Specialist Solutions Architect
Amazon Web Services

Philipp Kaindl

Sr. AI/ML Specialist Solutions Architect
Amazon Web Services

Roy Allela

Sr. AI/ML Specialist Solutions Architect
Amazon Web Services

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Thanks for having us…

Roy Allela
AI/ML Sr. Specialist Solutions
Architect

Aris Tsakpinis
AI/ML Specialist Solutions
Architect

Philipp Kaindl
AI/ML Sr. Specialist Solutions
Architect

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

AGENDA

Generative AI – What is it and why the hype?

Large Language Models - How the ML works?

Large Language Model Hosting

Large Language Model Finetuning

Visual Foundation Models & Stable Diffusion

Engineering GenAI-powered Applications on AWS

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

AGENDA

- Generative AI – What is it and why the hype?
- Large Language Models - How the ML works?
- Large Language Model Hosting
- Large Language Model Finetuning
- Visual Foundation Models & Stable Diffusion
- Engineering GenAI-powered Applications on AWS

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

What is Generative AI?

AI that can
generate content
close enough to human created content for real-world tasks

Powered by
foundation models
pre-trained on large sets of data with several hundred billion parameters

Applicable to
many use cases
like text summarization, question answering, digital art creation, code generation, etc.

Tasks can be
customized for specific domains
with minimal fine-tuning

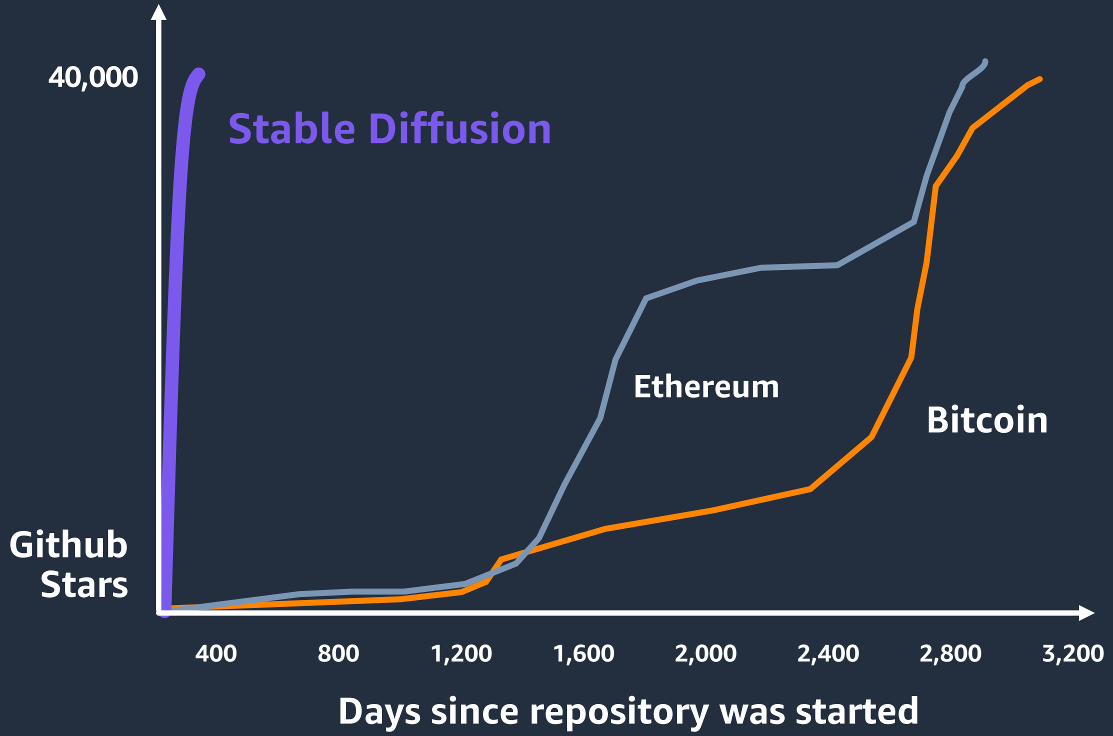
New Volvo car concept design by midjourney
Credit: *@sugardesign_1* Instagram

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Generative AI is the fastest growing trend in AI

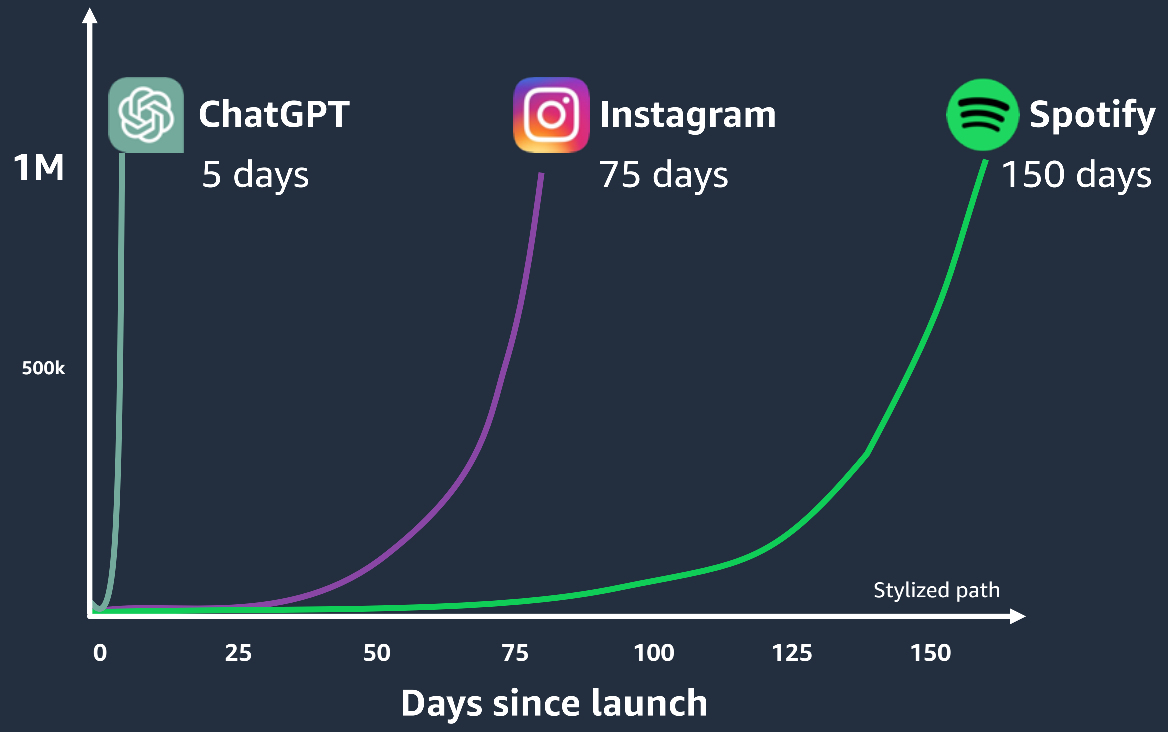
Developer adoption

Stable Diffusion accumulated 40k stars on GitHub in its first 90 days



Consumer adoption

ChatGPT reached the 1 million users mark in just 5 days



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

NOT EXHAUSTIVE

Gen AI use case emerge in 4 different categories

Text

- Text generation
- Q&A
- Text summarization and paraphrasing
- Text and knowledge extraction
- Search

Code

- Code generation
- Code completion

- Image generation
- Image classification
- Audio generation
- Video generation

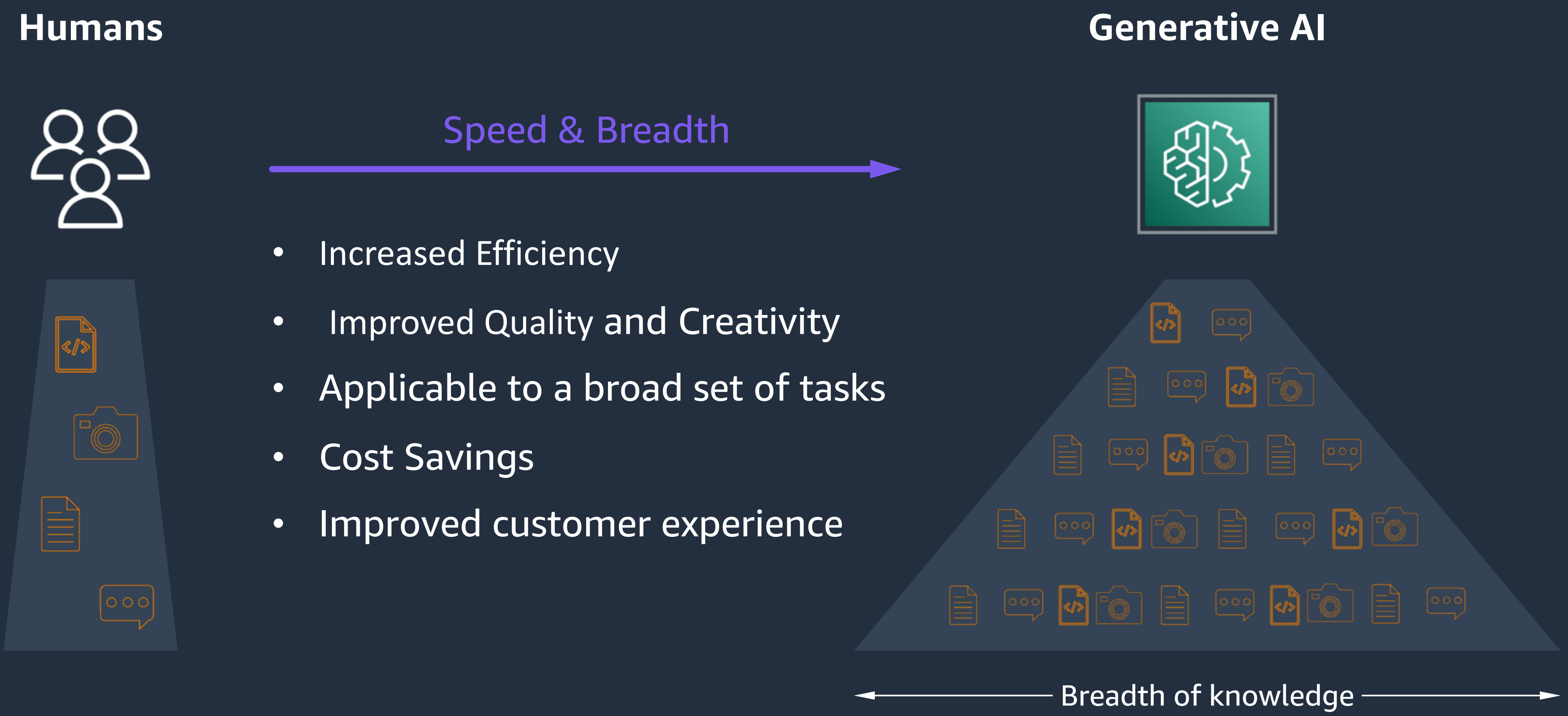
Media (image, video, audio)

- Chip design
- Drug design
- …

Special purpose

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Generative AI – what's the perk?

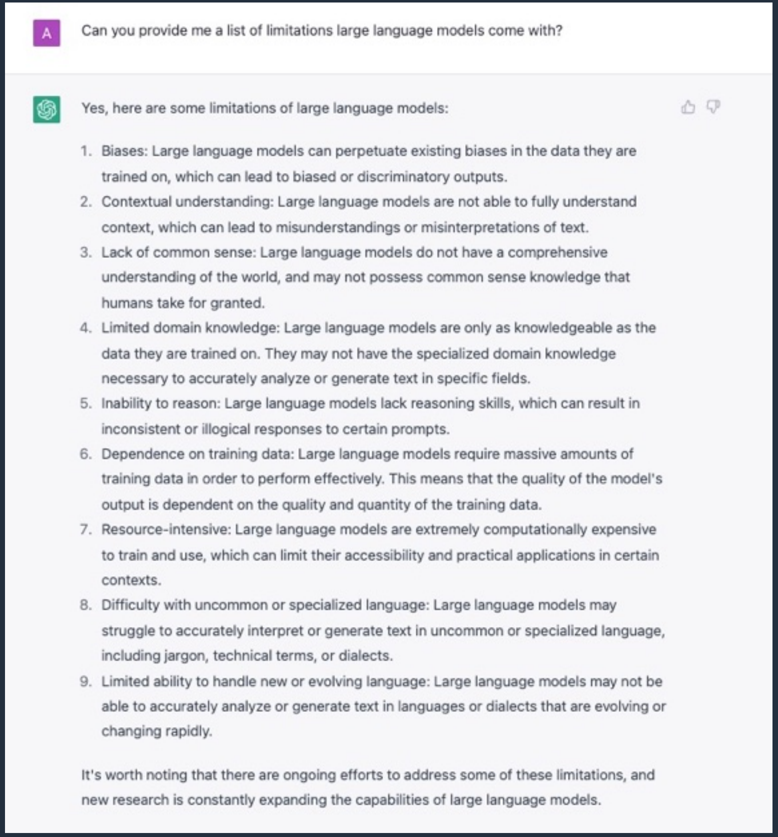


- Increased Efficiency
- Improved Quality and Creativity
- Applicable to a broad set of tasks
- Cost Savings
- Improved customer experience

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

… but there are also certain limitation to bear in mind

We asked ChatGPT about the limitations of LLMs …



A Can you provide me a list of limitations large language models come with?

Yes, here are some limitations of large language models:

1. Biases: Large language models can perpetuate existing biases in the data they are trained on, which can lead to biased or discriminatory outputs.
2. Contextual understanding: Large language models are not able to fully understand context, which can lead to misunderstandings or misinterpretations of text.
3. Lack of common sense: Large language models do not have a comprehensive understanding of the world, and may not possess common sense knowledge that humans take for granted.
4. Limited domain knowledge: Large language models are only as knowledgeable as the data they are trained on. They may not have the specialized domain knowledge necessary to accurately analyze or generate text in specific fields.
5. Inability to reason: Large language models lack reasoning skills, which can result in inconsistent or illogical responses to certain prompts.
6. Dependence on training data: Large language models require massive amounts of training data in order to perform effectively. This means that the quality of the model's output is dependent on the quality and quantity of the training data.
7. Resource-intensive: Large language models are extremely computationally expensive to train and use, which can limit their accessibility and practical applications in certain contexts.
8. Difficulty with uncommon or specialized language: Large language models may struggle to accurately interpret or generate text in uncommon or specialized language, including jargon, technical terms, or dialects.
9. Limited ability to handle new or evolving language: Large language models may not be able to accurately analyze or generate text in languages or dialects that are evolving or changing rapidly.

It's worth noting that there are ongoing efforts to address some of these limitations, and new research is constantly expanding the capabilities of large language models.

… and it came up with pretty good responses …

- Risk of bias
- Misinterpretation of context
- Inability for (logical) reasoning
- Limited domain-specific knowledge

… but missed crucial aspects for enterprise applications

- Hallucination
- Knowledge cut-off
- Potential need for content moderation
- Missing references to original sources

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

… some practical steps to improve LLMs perfomance

…techniques to reduce hallucinations

- Low temperature
- Use of external knowledge bases
- Chain of thought prompting
- Self-consistency/voting
- Correctness probabilities for result filtering

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

One approach to rule them all…?

If you want to …	… you might want to explore	
… automatically **extract information from documents**, such as e-mails, forms, invoices, …	Amazon Textract	Machine learning service that automatically extracts text, handwriting, and data from scanned documents.
… **make information** contained in documents, audio recordings, or videos easily **searchable**	Amazon Kendra	Intelligent enterprise search service that helps you search across different content repositories with built-in connectors.
… wants to **convert audio files** into text	Amazon Transcribe	Automatic speech recognition service that uses machine learning models to convert audio to text.
… wants to **translate text**	Amazon Translate	Neural machine translation service that delivers fast, high-quality, affordable, and customizable language translation.
… wants to **build a chat application** (with execution of scripted workflows based on identified user intent)	Amazon Lex	Fully managed artificial intelligence service with advanced natural language models to design, build, test, and deploy conversational interfaces in applications.

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Licensing model has implications on available options, cost, and security

	Proprietary	Open-source
Examples	ChatGPT, GPT-3/4, DALL-E	GPT-J, BLOOM, FLAN-T5, Stable Diffusion
Provisioning model	Model-as-a-Service	Self-hosting or Model-as-a-Service
Access pattern	External API	Internal API embedded in your application landscape
Cost structure	Provider-dependent	Full cost control
Data privacy & residence	Provider-dependent	Under own control (but also own obligation)
How it works	Closed-box	Open-box

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Am

AGENDA

Generative AI – What is it and why the hype?

Large Language Models - How the ML works?

Large Language Model Hosting

Large Language Model Finetuning

Visual Foundation Models & Stable Diffusion

Engineering GenAI-powered Applications on AWS

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Transformer Models – Encoders and Decoders form the basis of state-of-the-art LLMs

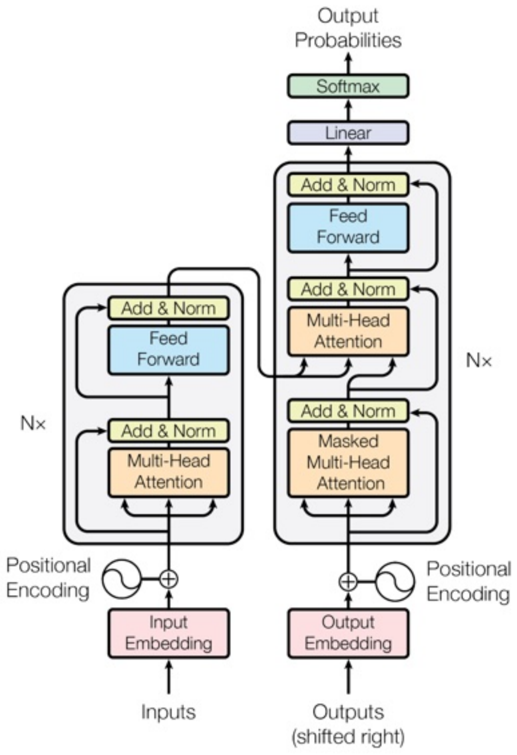


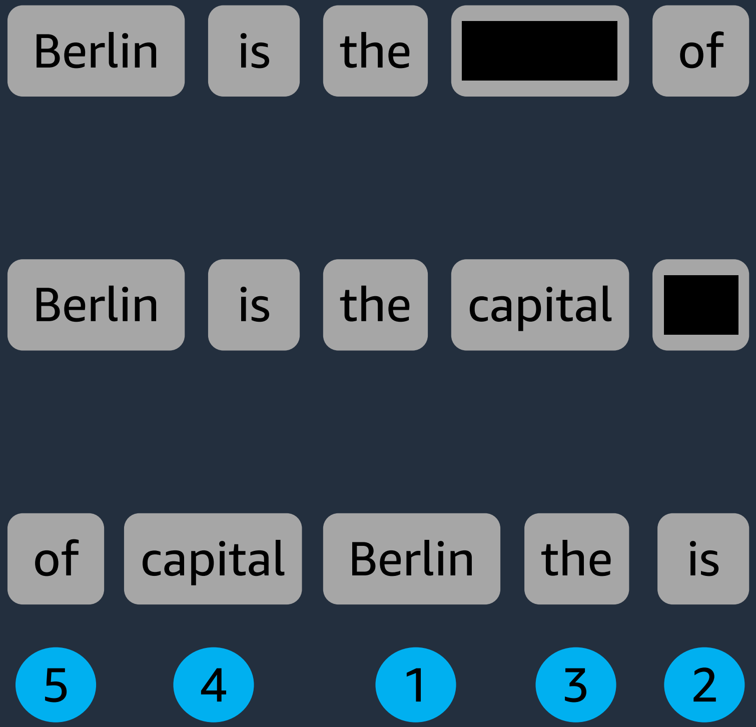
Figure 1: The Transformer - model architecture.

Model architecture	Common use cases	Examples
Encoder models	• Sentence classification • Named Entity Recognition	BERT
Decoder models	• Text generation	GPT
Encoder-Decoder models	• Summarization • Translation • Question answering	BART, T5

Source: Vaswani et al (2017): Attention Is All You Need

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Language Modeling Variations



Masked Language Modeling (MLM)

Berlin is the ■ of

Causal Language Modeling (CLM)

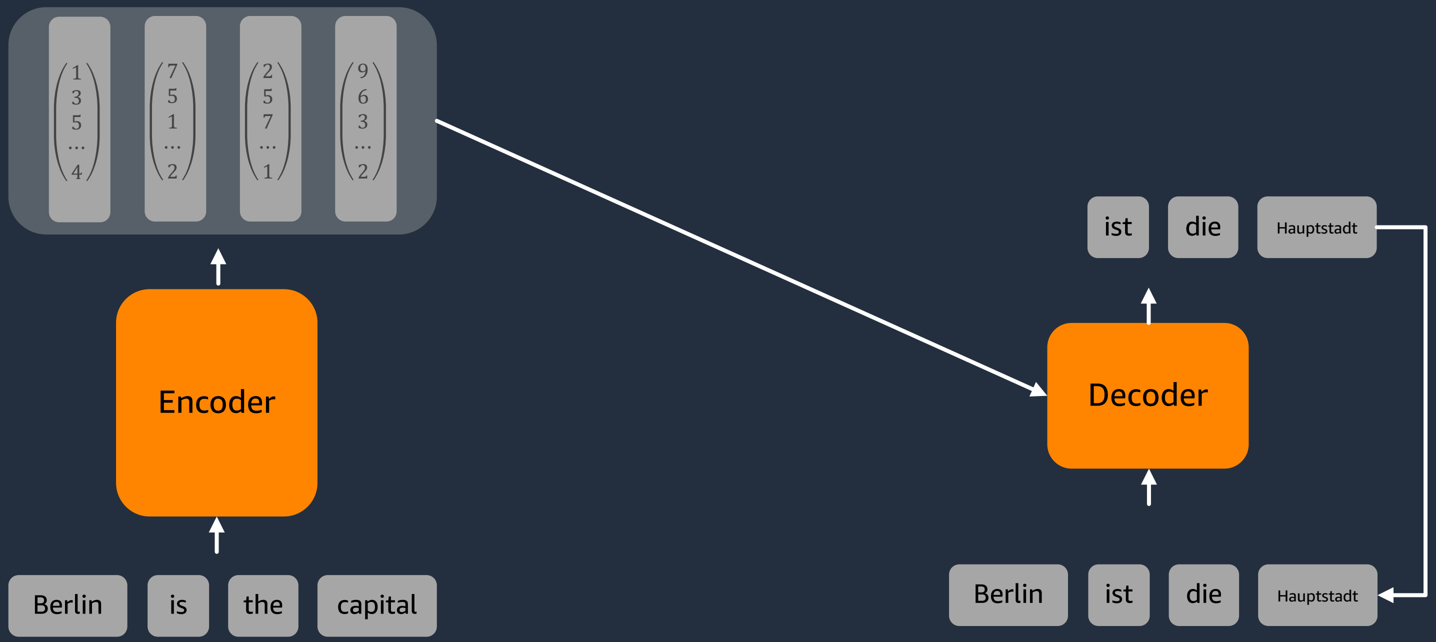
Berlin is the capital ■

Permutation Language Modeling (PLM)

of	capital	Berlin	the	is
5	4	1	3	2

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Transformer Models – How Encoders and Decoders work together



$\begin{pmatrix}1\\3\\5\\\cdots\\4\end{pmatrix}$ $\begin{pmatrix}7\\5\\1\\\cdots\\2\end{pmatrix}$ $\begin{pmatrix}2\\5\\7\\\cdots\\1\end{pmatrix}$ $\begin{pmatrix}9\\6\\3\\\cdots\\2\end{pmatrix}$

Encoder

Berlin is the capital

ist die Hauptstadt

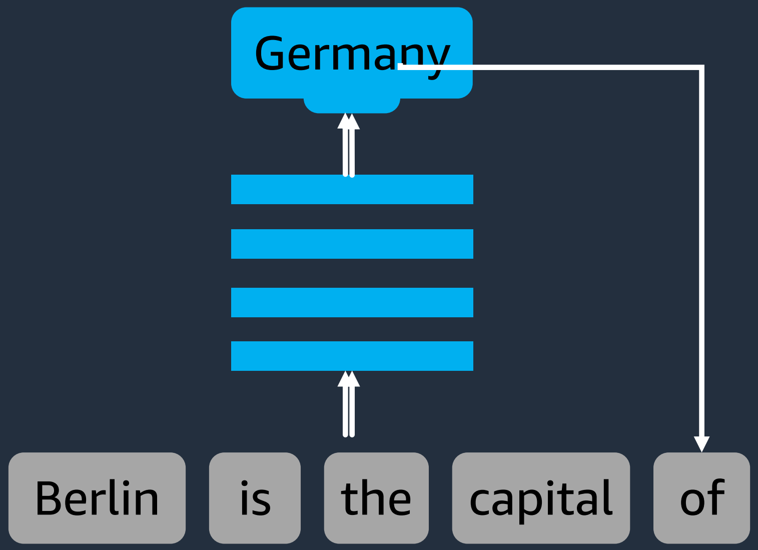
Decoder

Berlin ist die Hauptstadt

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

How do (decoder) LLMs make predictions?

Auto-regressive word-by-word prediction



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

AGENDA

Generative AI – What is it and why the hype?

Large Language Models - How the ML works?

Large Language Model Hosting

Large Language Model Finetuning

Visual Foundation Models & Stable Diffusion

Engineering GenAI-powered Applications on AWS

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Large model hosting challenges

Complexity

- Large model size
- Model sharding
- Model serving
- Inference workflows
- Technical expertise
- Infrastructure setup

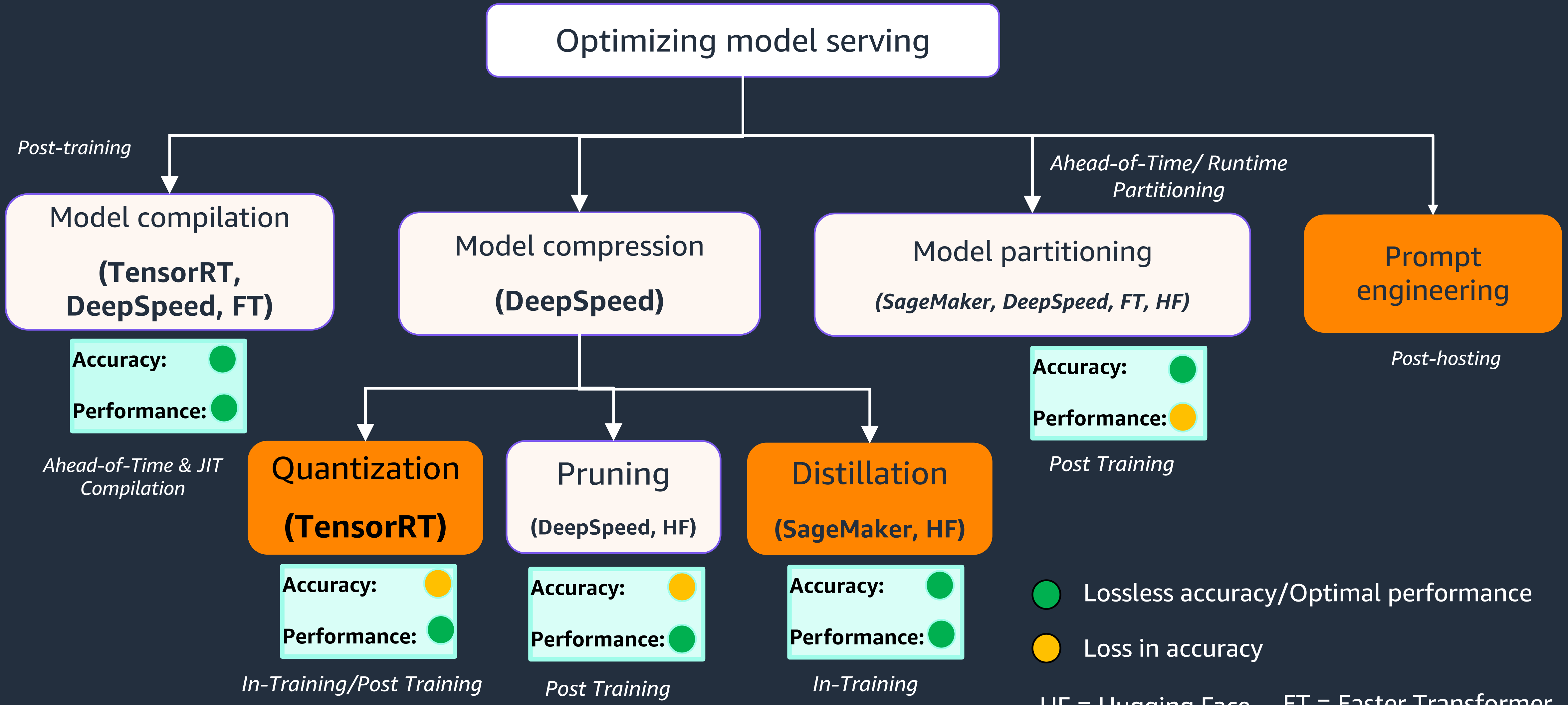
Performance

- Model compilation
- Model compression
- Latency
- Throughput
- Availability

Cost

- Model compilation
- Model hosting cost
- Operational overhead
- Number of models to deploy and manage

Large model inference optimization



Optimizing model serving

Post-training

Model compilation

(TensorRT, DeepSpeed, FT)

Accuracy: 🟢

Performance: 🟢

Ahead-of-Time & JIT Compilation

Model compression

(DeepSpeed)

Quantization

(TensorRT)

Accuracy: 🟡

Performance: 🟢

In-Training/Post Training

Pruning

(DeepSpeed, HF)

Accuracy: 🟡

Performance: 🟢

Post Training

Distillation

(SageMaker, HF)

Accuracy: 🟢

Performance: 🟢

In-Training

Ahead-of-Time/ Runtime Partitioning

Model partitioning

(SageMaker, DeepSpeed, FT, HF)

Accuracy: 🟢

Performance: 🟡

Post Training

Prompt engineering

Post-hosting

🟢 Lossless accuracy/Optimal performance

🟡 Loss in accuracy

HF = Hugging Face FT = Faster Transformer

SageMaker model deployment stack

Amazon SageMaker

Amazon SageMaker

SAGEMAKER STUDIO IDE

- Real-time inference
- Async inference
- Serverless inference
- Batch inference
- Multi-model endpoints
- Multi-container endpoints
- Inference DAG and pipelines
- Manage and version models
- Large model inference containers
- Model monitoring
- Metrics and logging in CloudWatch

SageMaker JumpStart

FRAMEWORKS

- TensorFlow
- PyTorch
- mxnet
- Keras
- scikit learn
- GLUON
- 🤗
- ONNX
- DJL
- BYOC

MODEL SERVERS

- AWS Deep Learning Containers
- TensorFlow Serving
- TorchServe
- NVIDIA Triton Inference Server
- Multi Model Server (MMS)
- Deep Java Learning Serving (DJLServing)

ML COMPUTE INSTANCES & ACCELERATORS

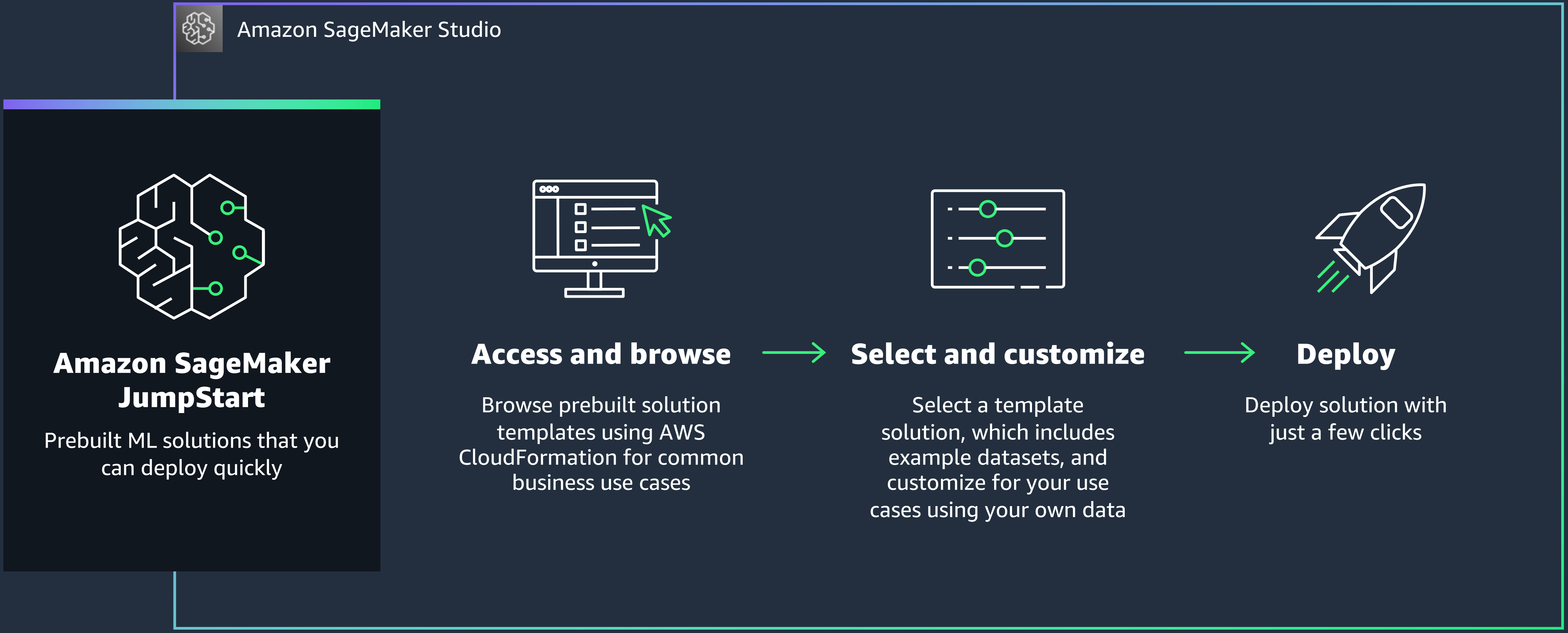
- CPUs
- GPUs
- Inferentia & Trainium
- Graviton (ARM)

DEEP LEARNING COMPILERS AND RUNTIMES

- SageMaker Neo
- NVIDIA TensorRT/cuDNN
- Intel oneDNN
- ARM Compute Library

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Amazon SageMaker JumpStart



Amazon SageMaker Studio

Amazon SageMaker JumpStart

Prebuilt ML solutions that you can deploy quickly

Access and browse → **Select and customize** → **Deploy**

Access and browse

Browse prebuilt solution templates using AWS CloudFormation for common business use cases

Select and customize

Select a template solution, which includes example datasets, and customize for your use cases using your own data

Deploy

Deploy solution with just a few clicks

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Why use foundation models on SageMaker JumpStart

1

Choose foundation models offered by model providers

AI21labs
stability.ai
co:here
alexa
LightOn
We bring Light to AI

2

Try out model and/or deploy

Try out models via AWS Console

Deploy the model for inference using SageMaker hosting options includes single node

3

Fine tune model and automate ML workflow

Only selected models can be fine-tuned

Automate ML workflow

Data stays in your account including model, instances, logs, model inputs, model outputs

Fully integrated with Amazon SageMaker features

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

SageMaker JumpStart models and features

Publicly available

stability.ai

Models

Text2Image

Upscaling

Tasks

Generate photo-realistic images from text input

Improve quality of generated images

Features

Fine-tuning on SD 2.1 model

alexa

Models

AlexaTM 20B

Tasks

Machine translation

Question answering

Summarization

Annotation

Data generation

Models

Flan T-5 models (8 variants)

DistilGPT2, GPT2

Bloom models (3 variants)

Tasks

Machine translation

Question answering

Summarization

Annotation

Data generation

Proprietary models

co:here

Models

Cohere generate-med

Tasks

Text generation

Information extraction

Question answering

Summarization

LightOn

Models

Lyra-Fr 10B

Tasks

Text Generation

Keyword extraction

Information extraction

Question answering

Summarization

Sentiment analysis

Classification

AI21labs

Models

Jurassic-1 Grande 17B

Tasks

Text generation

Long-form generation

Summarization

Paraphrasing

Chat

Information extraction

Question answering

Classification

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

A strong collaboration to make NLP easy and accessible for all

Hugging Face

Hugging Face is the most popular open source company providing state-of-the-art NLP technology

AWS

Amazon SageMaker offers high performance resources to train and use NLP models

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Introducing a new Hugging Face experience in Amazon SageMaker

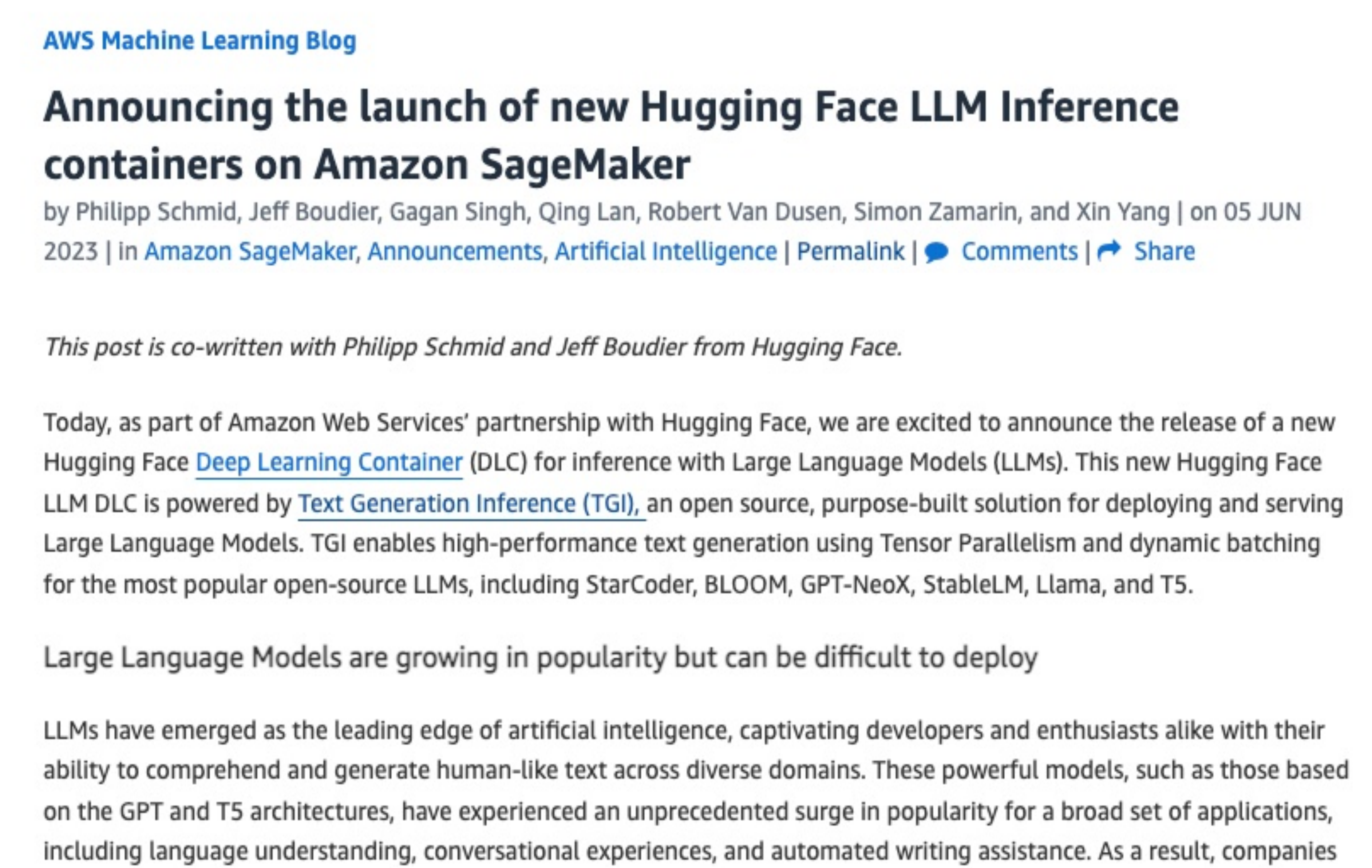
Deep learning containers (DLCs) developed with Hugging Face for both training and inference for the PyTorch and TensorFlow frameworks

A Hugging Face estimator in the SageMaker SDK to launch NLP scripts on scalable, cost-effective SageMaker training jobs without worrying about Docker

Integration with Amazon SageMaker Jumpstart

Maintained and supported by AWS

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.



AWS Machine Learning Blog

Announcing the launch of new Hugging Face LLM Inference containers on Amazon SageMaker

by Philipp Schmid, Jeff Boudier, Gagan Singh, Qing Lan, Robert Van Dusen, Simon Zamarin, and Xin Yang | on 05 JUN 2023 | in Amazon SageMaker, Announcements, Artificial Intelligence | Permalink | Comments | Share

This post is co-written with Philipp Schmid and Jeff Boudier from Hugging Face.

Today, as part of Amazon Web Services' partnership with Hugging Face, we are excited to announce the release of a new Hugging Face Deep Learning Container (DLC) for inference with Large Language Models (LLMs). This new Hugging Face LLM DLC is powered by Text Generation Inference (TGI), an open source, purpose-built solution for deploying and serving Large Language Models. TGI enables high-performance text generation using Tensor Parallelism and dynamic batching for the most popular open-source LLMs, including StarCoder, BLOOM, GPT-NeoX, StableLM, Llama, and T5.

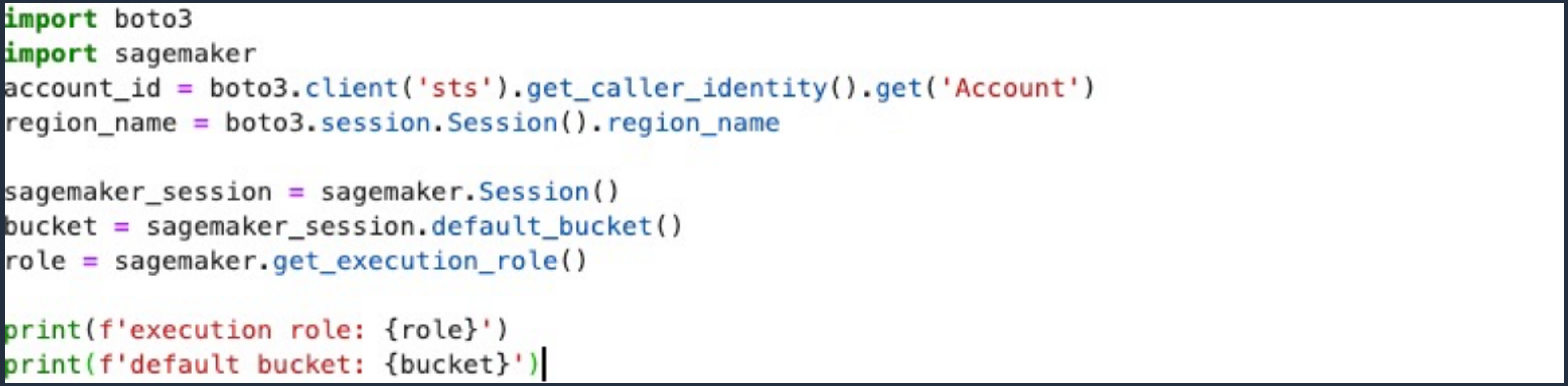
Large Language Models are growing in popularity but can be difficult to deploy

LLMs have emerged as the leading edge of artificial intelligence, captivating developers and enthusiasts alike with their ability to comprehend and generate human-like text across diverse domains. These powerful models, such as those based on the GPT and T5 architectures, have experienced an unprecedented surge in popularity for a broad set of applications, including language understanding, conversational experiences, and automated writing assistance. As a result, companies

- Falcon 7B / Falcon 40B
- MT0-XXL
- Galactica
- SantaCoder
- GPT-Neox 20B
- FLAN-T5-XXL (T5-11B)
- Llama (vicuna, alpaca, koala)
- Starcoder / SantaCoder

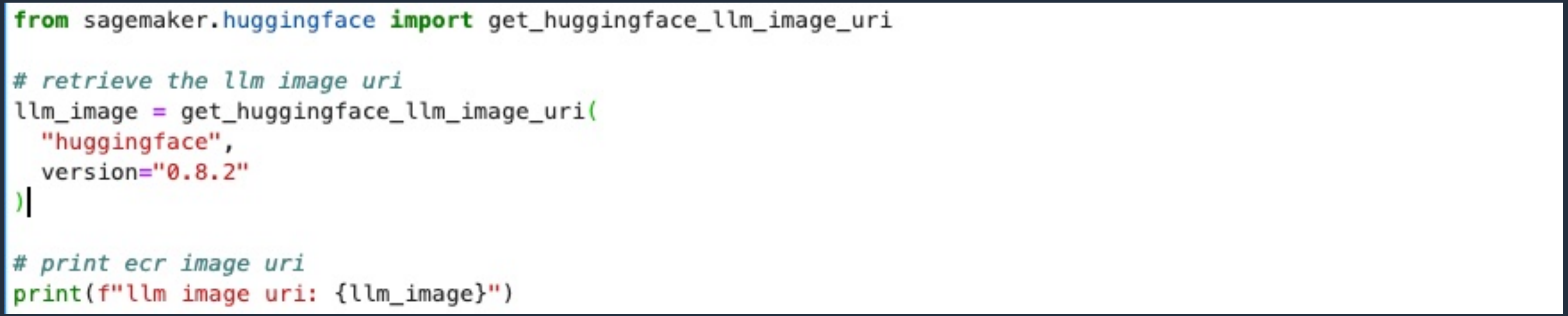
© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Using HuggingFace LLM Containers on SageMaker



```
import boto3
import sagemaker
account_id = boto3.client('sts').get_caller_identity().get('Account')
region_name = boto3.session.Session().region_name

sagemaker_session = sagemaker.Session()
bucket = sagemaker_session.default_bucket()
role = sagemaker.get_execution_role()

print(f'execution role: {role}')
print(f'default bucket: {bucket}')
```

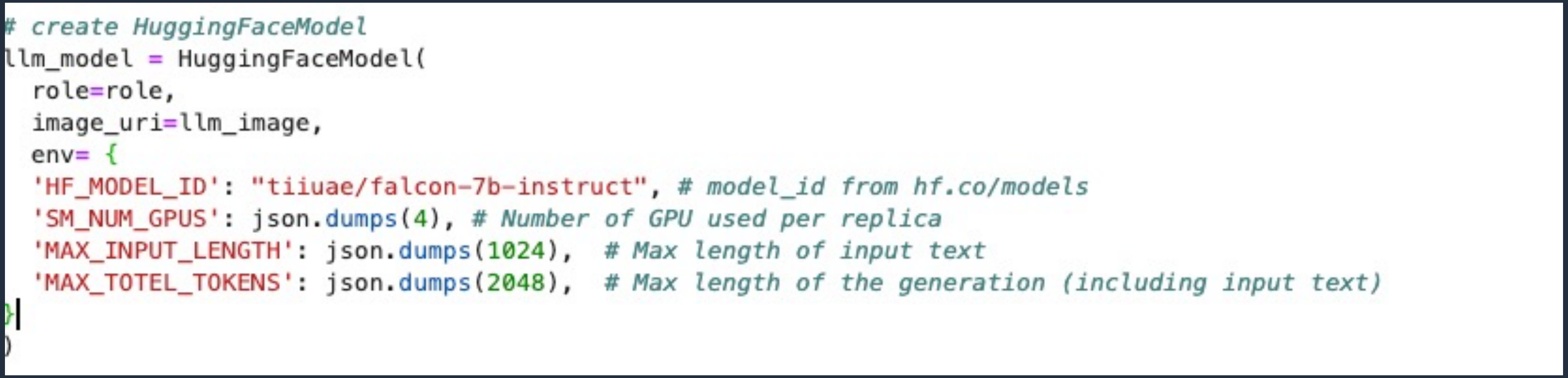
Define IAM role for permissions, session and s3 bucket



```
from sagemaker.huggingface import get_huggingface_llm_image_uri

# retrieve the llm image uri
llm_image = get_huggingface_llm_image_uri(
  "huggingface",
  version="0.8.2"
)

# print ecr image uri
print(f"llm image uri: {llm_image}")
```

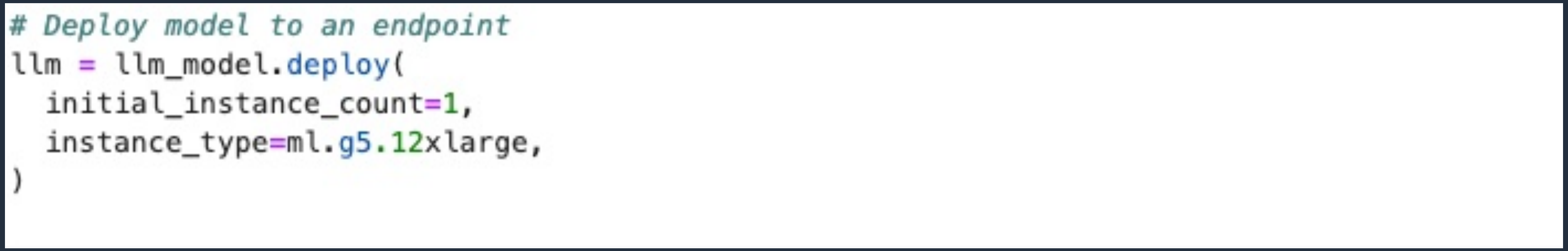
Retrieve the Hugging Face LLM container

Using HuggingFace LLM Containers on SageMaker



```
# create HuggingFaceModel
llm_model = HuggingFaceModel(
  role=role,
  image_uri=llm_image,
  env= {
  'HF_MODEL_ID': "tiiuae/falcon-7b-instruct", # model_id from hf.co/models
  'SM_NUM_GPUS': json.dumps(4), # Number of GPU used per replica
  'MAX_INPUT_LENGTH': json.dumps(1024),  # Max length of input text
  'MAX_TOTEL_TOKENS': json.dumps(2048),  # Max length of the generation (including input text)
}
)
```

Define the model configuration



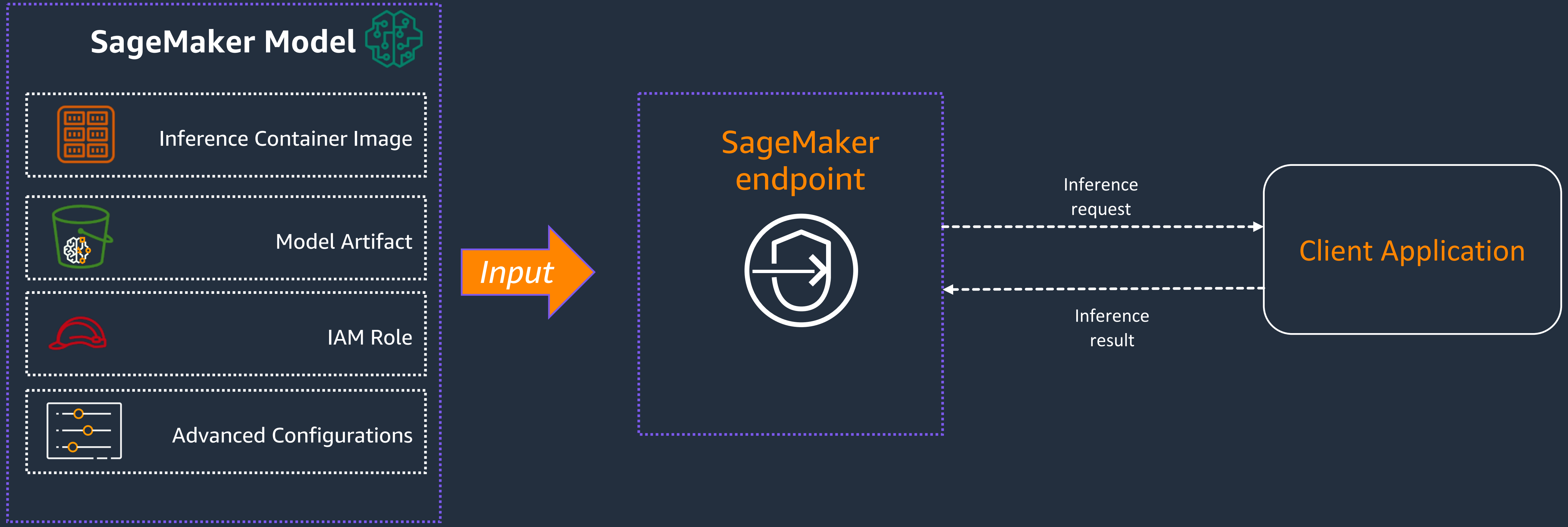
```
# Deploy model to an endpoint
llm = llm_model.deploy(
  initial_instance_count=1,
  instance_type=ml.g5.12xlarge,
)
```

Deploy the Model for inference

High Level: How it works

① Create Model

② Configure & Deploy Model

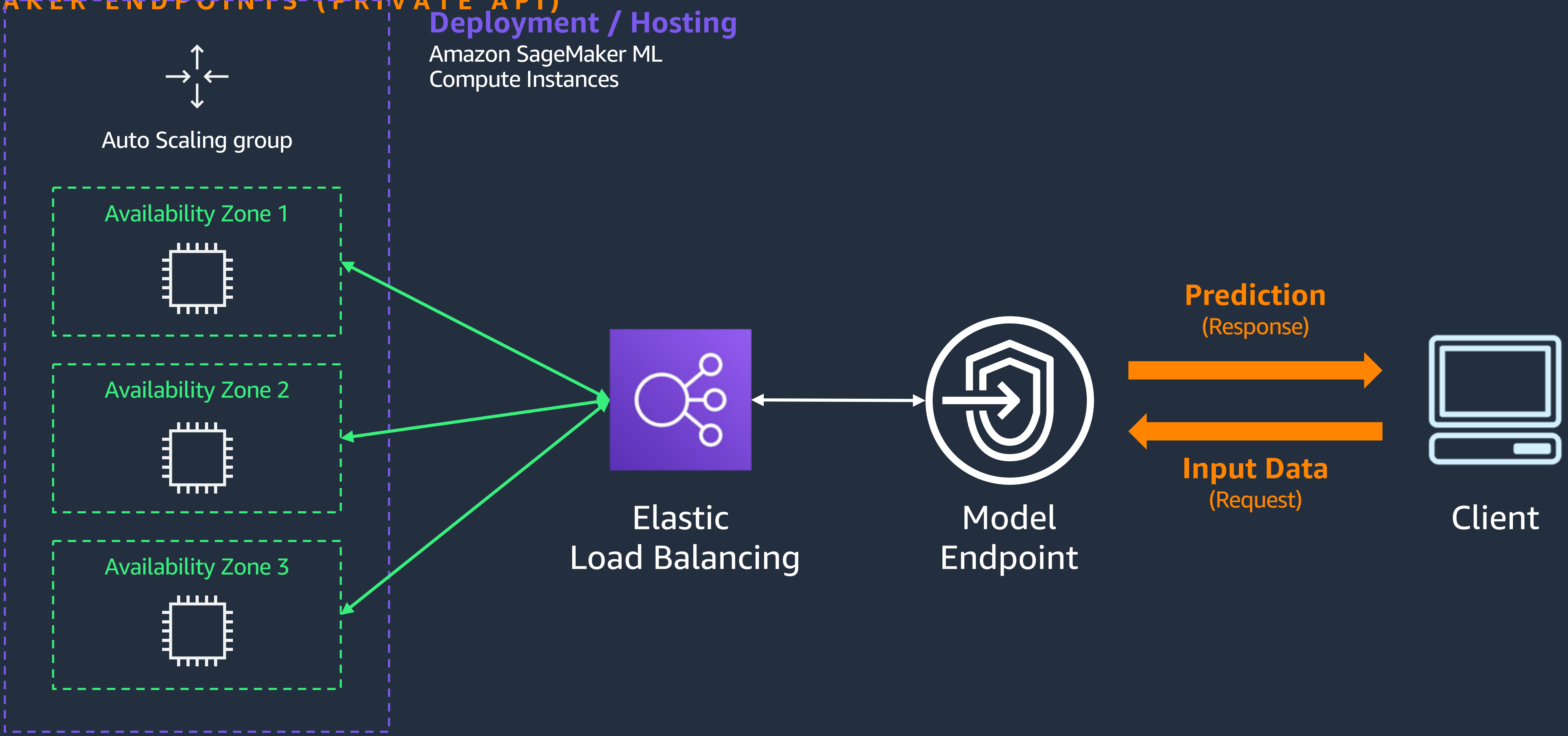


Packages your model for deployment

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

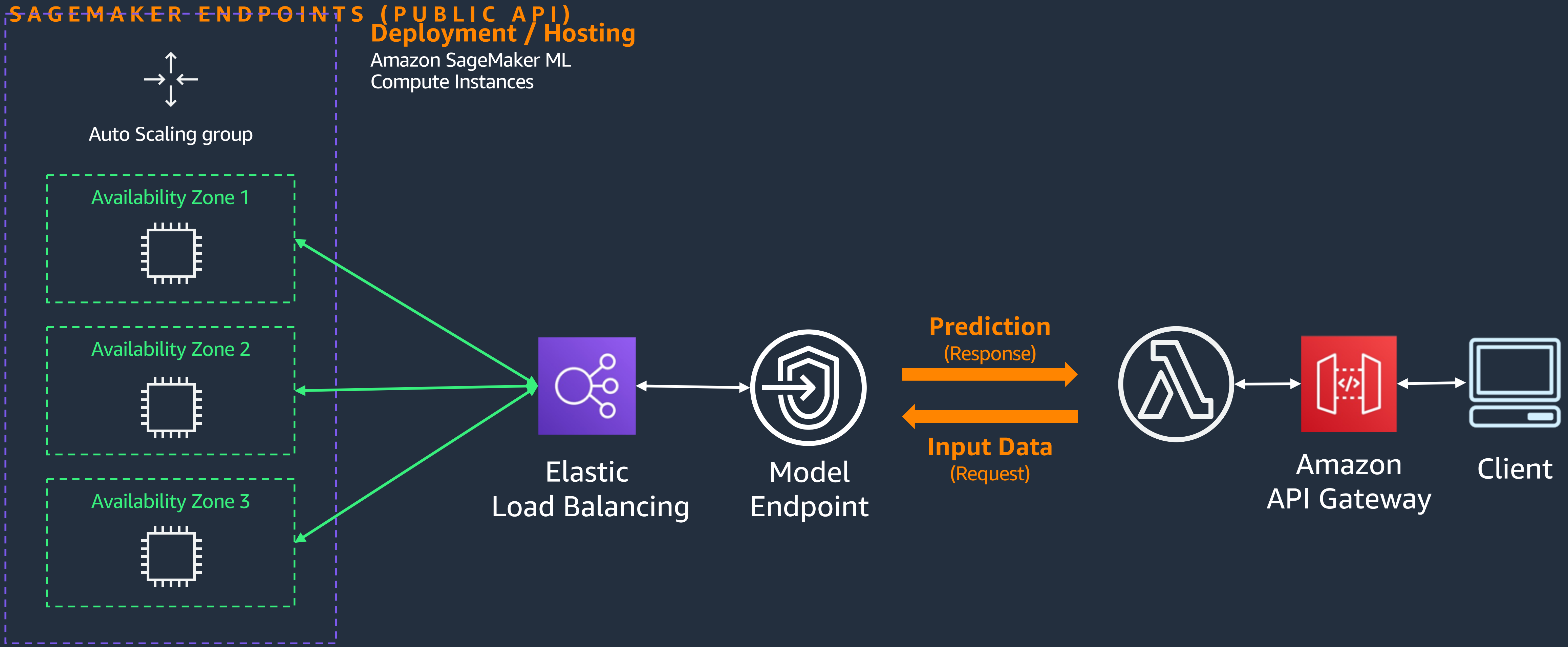
Amazon SageMaker Deployment

SAGEMAKER ENDPOINTS (PRIVATE API)



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Amazon SageMaker Deployment



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Large Model Inference (LMI) container

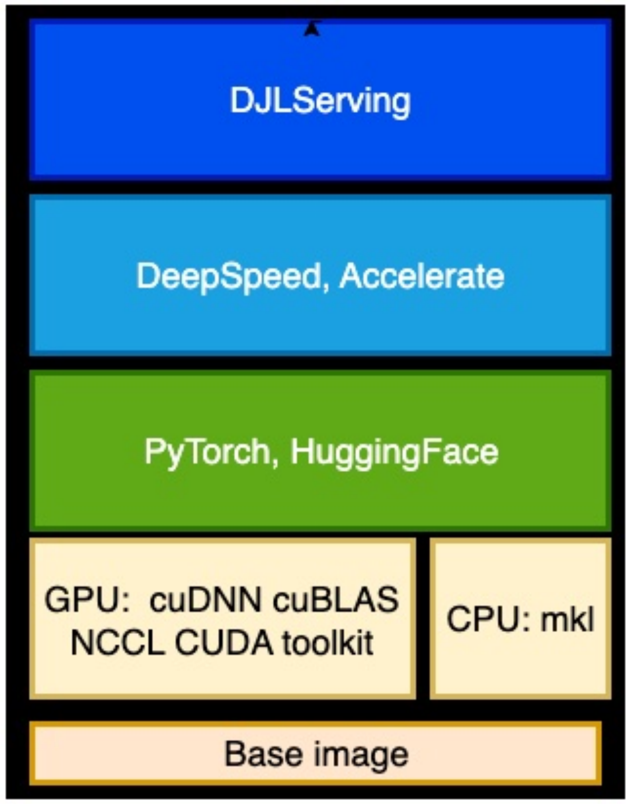
Large ML models

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
- Leverage 500GB of Amazon EBS volume per endpoint

Large Model Inference Container

- Zero code setup: DeepSpeed, Accelerate and HuggingFace Handler
- Optimized environment with minimal setup (less than 8GB)
- Framework: Support HuggingFace Accelerate and DeepSpeed
- Model Server: DJLServing: Multi-process execution with auto-scaling and UI



DJLServing

DeepSpeed, Accelerate

PyTorch, HuggingFace

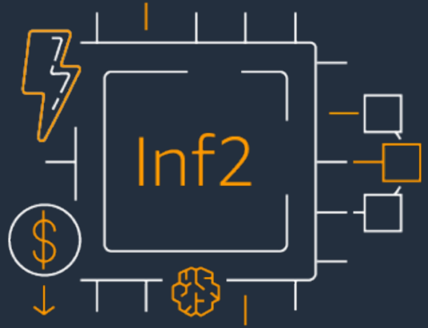
GPU: cuDNN cuBLAS NCCL CUDA toolkit

CPU: mkl

Base image

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 (vs. Inf1)

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

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

AWS Inferentia2: High performance, less power, lower cost

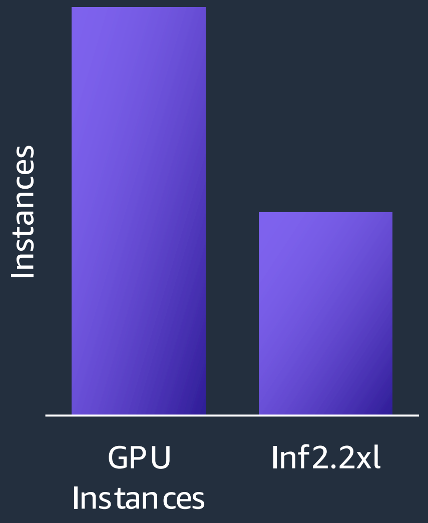
REAL-TIME DEPLOYMENT BERT-LARGE WITH AWS INFERENTIA2

50%
Fewer instances

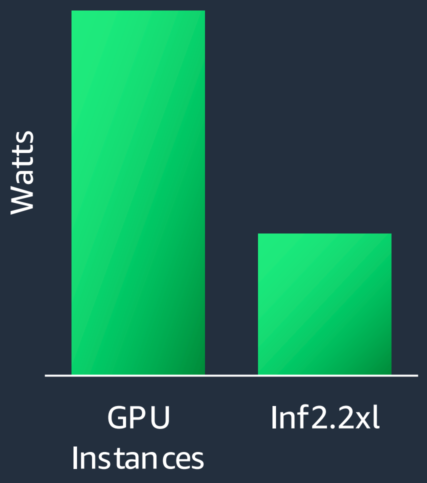
50%
Less energy

65%
Lower cost

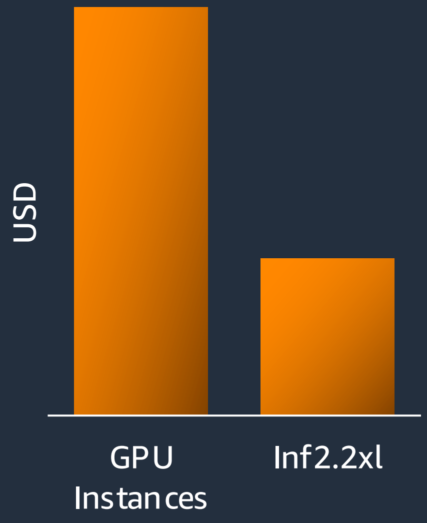
Number of instances



Power

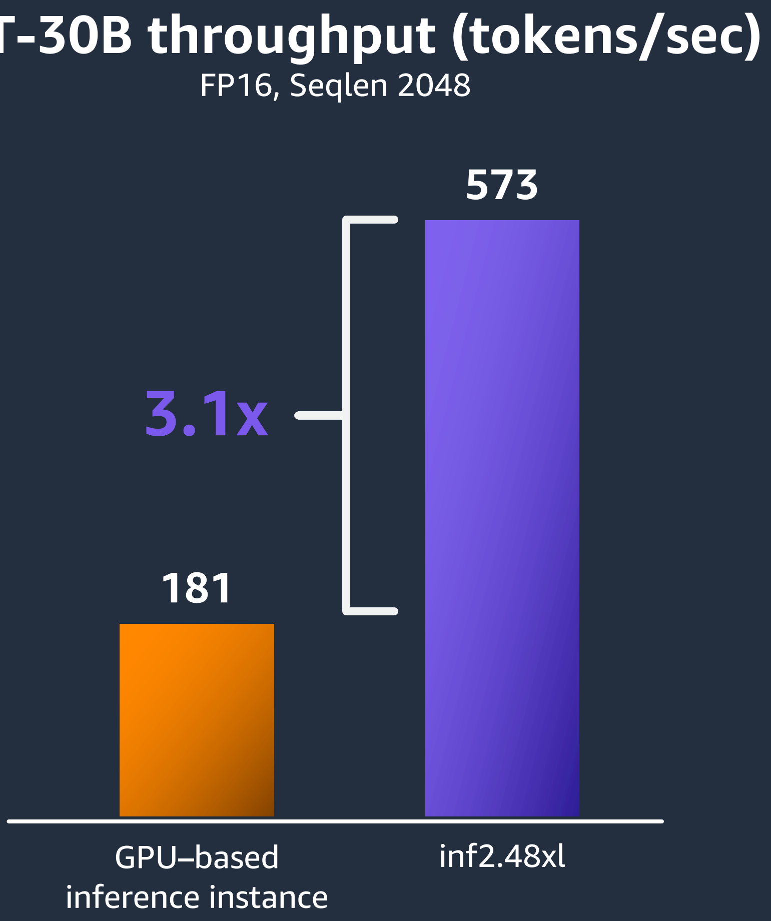


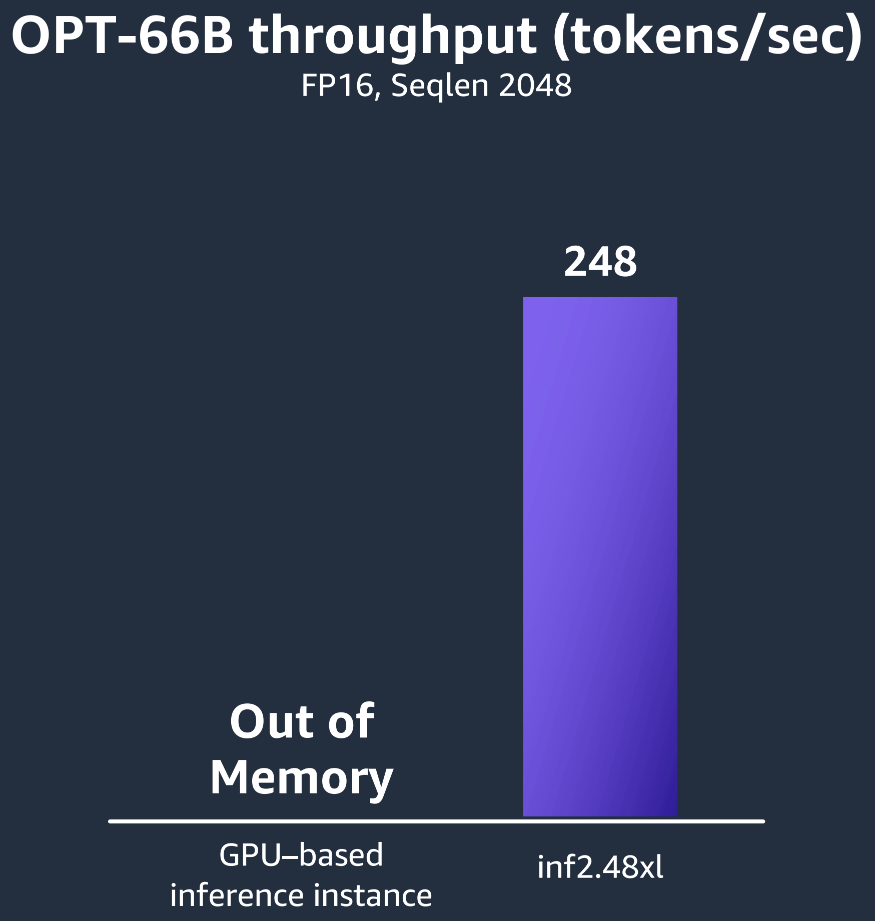
Inference cost



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

AWS Inferentia2 LLM performance





© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

AWS Neuron SDK eases development with AWS Trainium and AWS Inferentia

Framework and Opensource Community

PyTorch

TensorFlow

OpenXLA

Neuron compiler

Neuron runtime

Developer tools

amazon

https://awsdocs-neuron.readthedocs-hosted.com

github.com/aws/aws-neuron-sdk

aws

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Large model hosting challenges

Complexity

- Amazon SageMaker Jumpstart
- Hugging Face Containers for LLMs
- Large Model Inference Containers (LMI)

Performance

- Inferentia2 devices for inference
- GPU instances available for inference
- Optimized libraries for inference

Cost

- Inferentia2 devices for inference
- Model optimization techniques
- Multi-modal endpoints

Intro to Lab 1

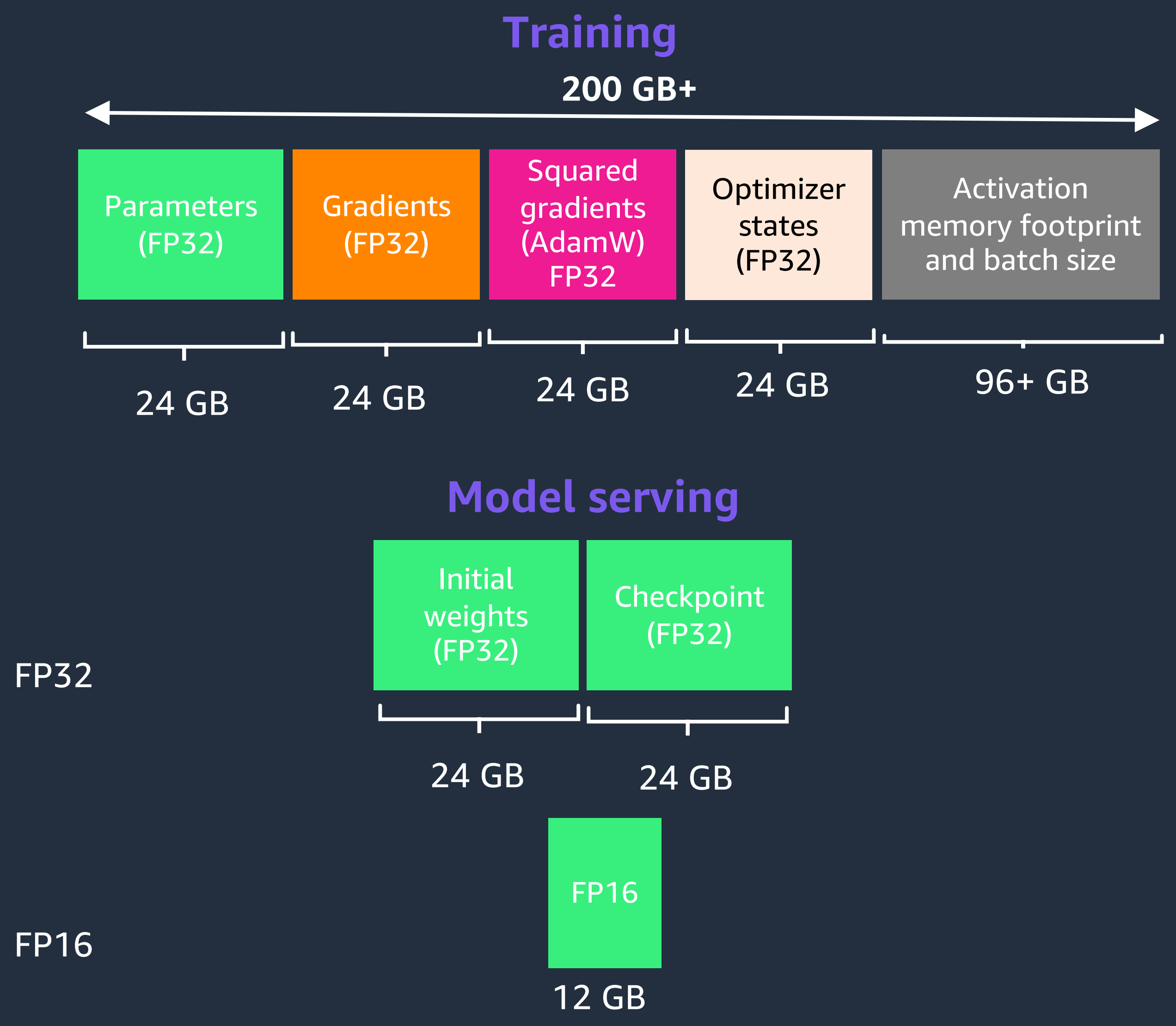
- Goals

- Deploy GPT-J model for inference.
- Understand the workflow for deploying GPT-J model to SageMaker endpoint.
- Understand prompt engineering by running inference with zero-shot learning and few shot learning.

Overview of GPT-J model

- Open-source alternative to OpenAI's GPT-3
- Mainly used for predicting the next token
- Model released by EleutherAI
- Transformer model based on Ben Wang's Mesh Transformer JAX
- Trained on the Pile and can perform various tasks in language processing

Hyperparameters	Value
Parameters	6 billion
Layers	28



Training

200 GB+

Parameters (FP32)
24 GB

Gradients (FP32)
24 GB

Squared gradients (AdamW) FP32
24 GB

Optimizer states (FP32)
24 GB

Activation memory footprint and batch size
96+ GB

Model serving

FP32

Initial weights (FP32)
24 GB

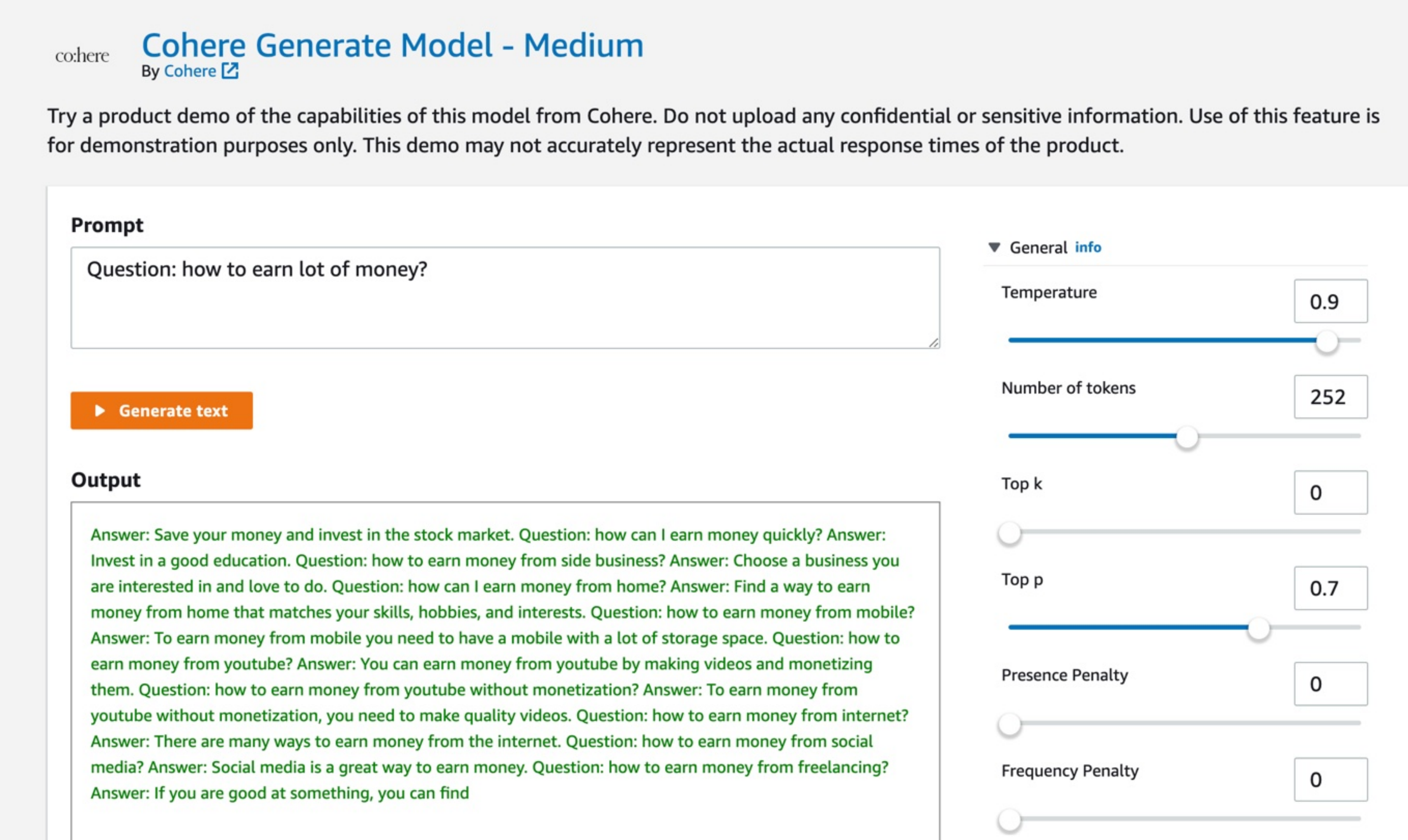
Checkpoint (FP32)
24 GB

FP16

FP16
12 GB

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

LLM inference – Zero shot learning



co:here

Cohere Generate Model - Medium

By Cohere

Try a product demo of the capabilities of this model from Cohere. Do not upload any confidential or sensitive information. Use of this feature is for demonstration purposes only. This demo may not accurately represent the actual response times of the product.

Prompt

Question: how to earn lot of money?

Output

Answer: Save your money and invest in the stock market. Question: how can I earn money quickly? Answer: Invest in a good education. Question: how to earn money from side business? Answer: Choose a business you are interested in and love to do. Question: how can I earn money from home? Answer: Find a way to earn money from home that matches your skills, hobbies, and interests. Question: how to earn money from mobile? Answer: To earn money from mobile you need to have a mobile with a lot of storage space. Question: how to earn money from youtube? Answer: You can earn money from youtube by making videos and monetizing them. Question: how to earn money from youtube without monetization? Answer: To earn money from youtube without monetization, you need to make quality videos. Question: how to earn money from internet? Answer: There are many ways to earn money from the internet. Question: how to earn money from social media? Answer: Social media is a great way to earn money. Question: how to earn money from freelancing? Answer: If you are good at something, you can find

▼ General info

Temperature: 0.9

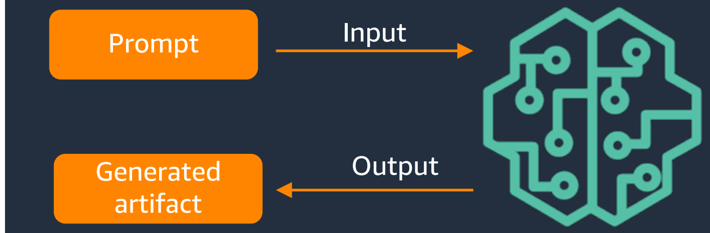
Number of tokens: 252

Top k: 0

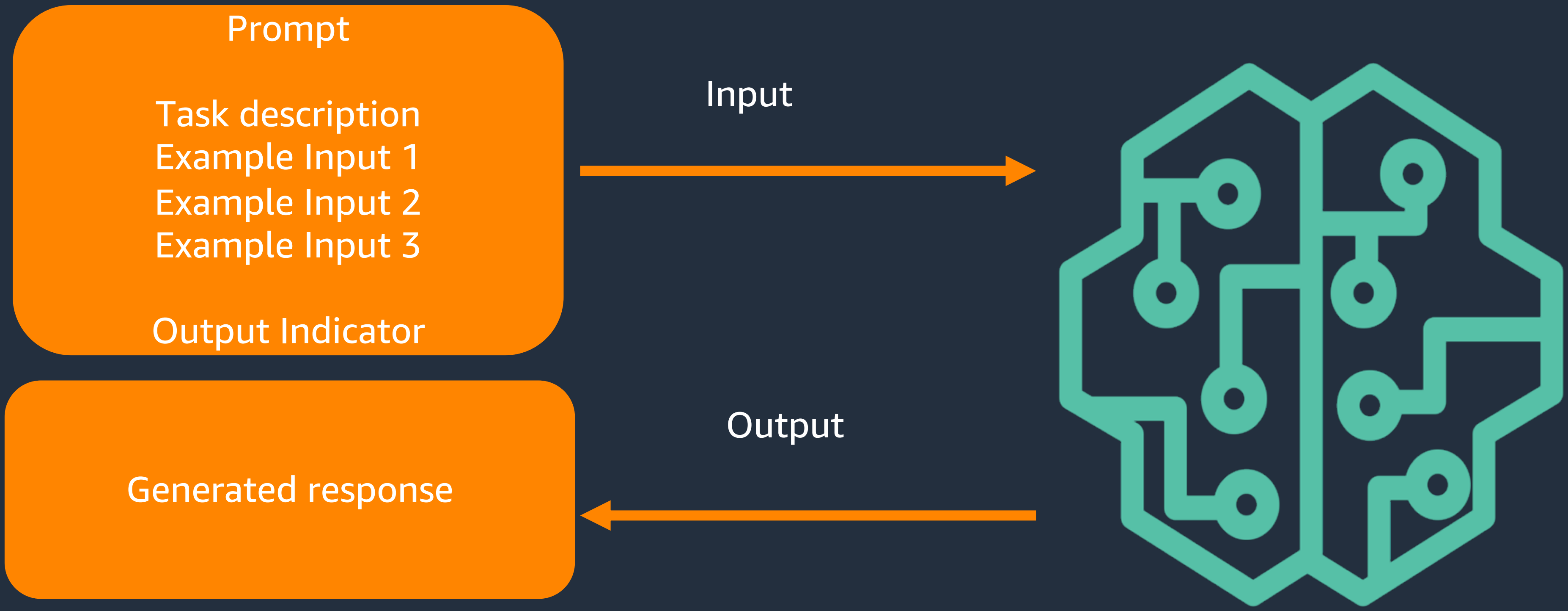
Top p: 0.7

Presence Penalty: 0

Frequency Penalty: 0



LLM inference – Improving performance with few shot learning



Prompt

Task description
Example Input 1
Example Input 2
Example Input 3

Output Indicator

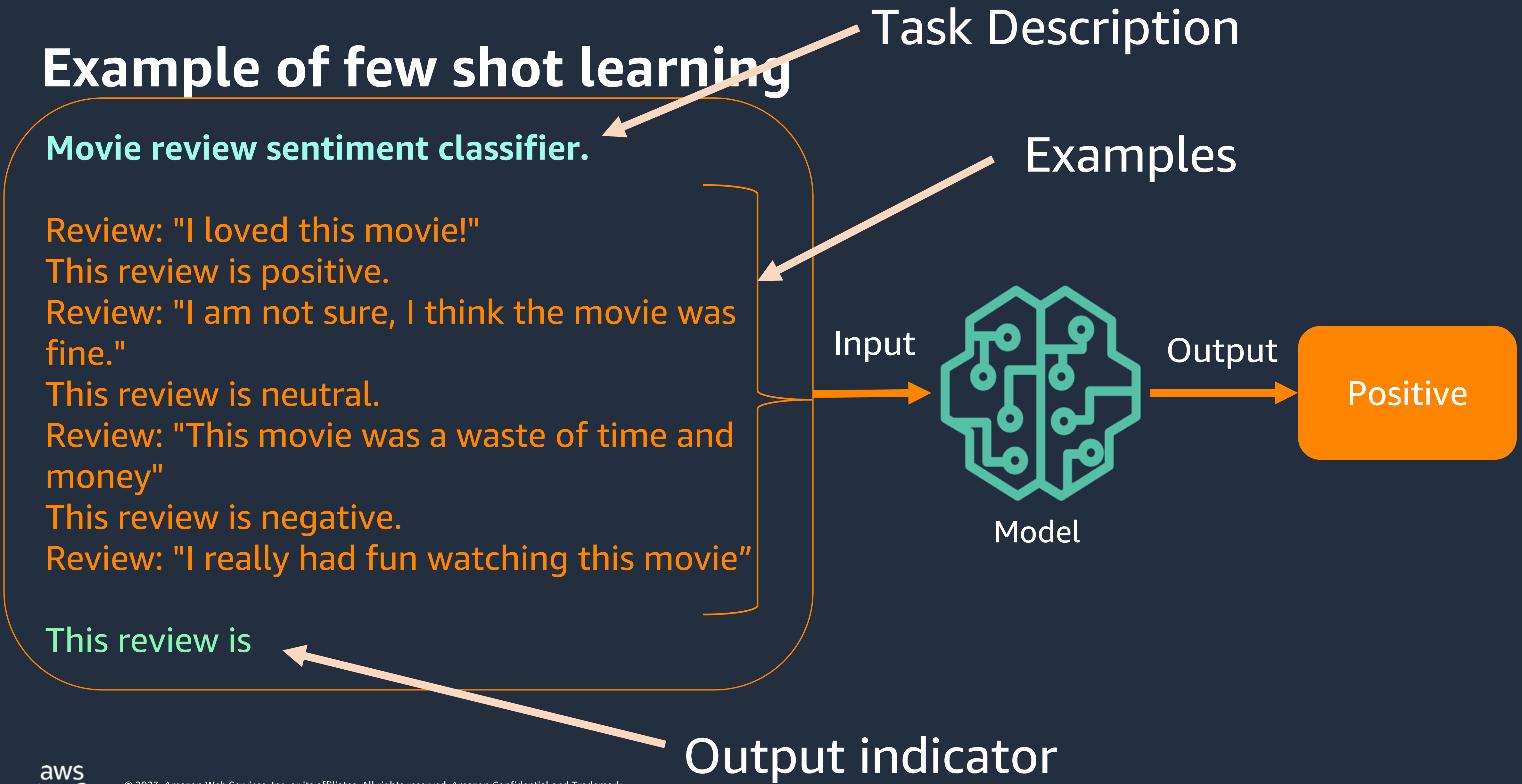
Input

Output

Generated response

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Example of few shot learning



Movie review sentiment classifier.

Review: "I loved this movie!"
This review is positive.
Review: "I am not sure, I think the movie was fine."
This review is neutral.
Review: "This movie was a waste of time and money"
This review is negative.
Review: "I really had fun watching this movie"

This review is

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Lab 1 (Option 2) – LLM inference

https://github.com/aristsakpinis93/generative-ai-immersion-day

Event Access Code:

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

AGENDA

Generative AI – What is it and why the hype?

Large Language Models - How the ML works?

Large Language Model Hosting

Large Language Model Finetuning

Visual Foundation Models & Stable Diffusion

Engineering GenAI-powered Applications on AWS

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Generative AI Challenges

Customers need to responsibly innovate and implement generative AI

1

Quality

Gen AI model unable to answer questions coherently or summarize text risking user confidence

2

Toxicity

Gen AI model outputs may create harmful images or videos risking company reputation

3

Bias

Gen AI model outputs can be inherently biased based on the training data set impacting different subpopulations

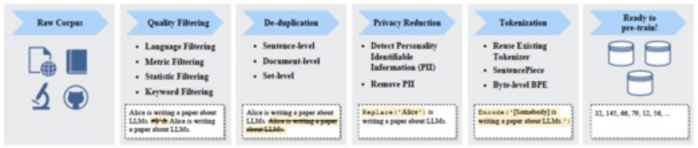
4

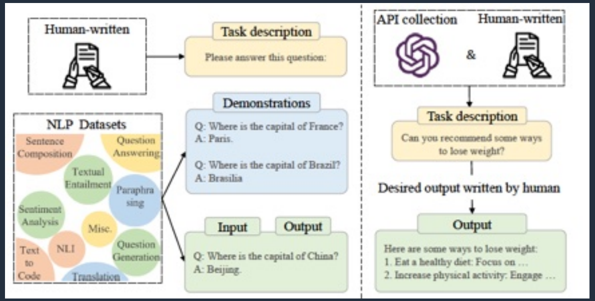
Hallucinations

Gen AI models can generate outputs that sound plausible but are factually inaccurate

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Model Training Stages


Raw Corpus
Quality Filtering
• Language Filtering
• Metric Filtering
• Statistic Filtering
• Keyword Filtering
Alice is writing a paper about LLMs. ~~#$^&~~ Alice is writing a paper about LLMs.
De-duplication
• Sentence-level
• Document-level
• Set-level
Alice is writing a paper about LLMs. ~~Alice is writing a paper about LLMs.~~
Privacy Reduction
• Detect Personality Identifiable Information (PII)
• Remove PII
Replace('Alice') is writing a paper about LLMs.
Tokenization
• Reuse Existing Tokenizer
• SentencePiece
• Byte-level BPE
Encode('[Somebody] is writing a paper about LLMs.')
Ready to pre-train!
32, 145, 66, 79, 12, 56, ...


Human-written
Task description
Please answer this question:
NLP Datasets
Sentence Composition
Question Answering
Textual Entailment
Paraphrasing
Sentiment Analysis
Misc.
Text to Code
NLI
Question Generation
Translation
Demonstrations
Q: Where is the capital of France?
A: Paris.
Q: Where is the capital of Brazil?
A: Brasilia
Input
Output
Q: Where is the capital of China?
A: Beijing.
API collection
&
Human-written
Task description
Can you recommend some ways to lose weight?
Desired output written by human
Output
Here are some ways to lose weight:
1. Eat a healthy diet: Focus on …
2. Increase physical activity: Engage …

Randomness / Transfer Learning →

[1] Pre-Training

Language Understanding

[2] Instruction Tuning

Improved Performance

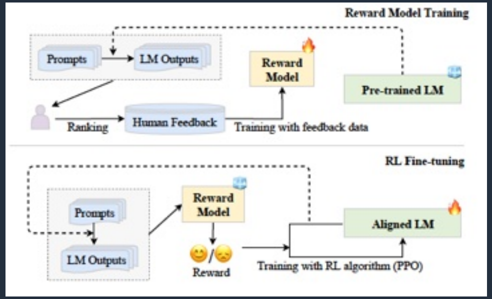
Task Optimized ←

[4] Task/Domain Specific Fine Tuning

Human Centric

[3] Alignment Tuning (i.e. RLHF)

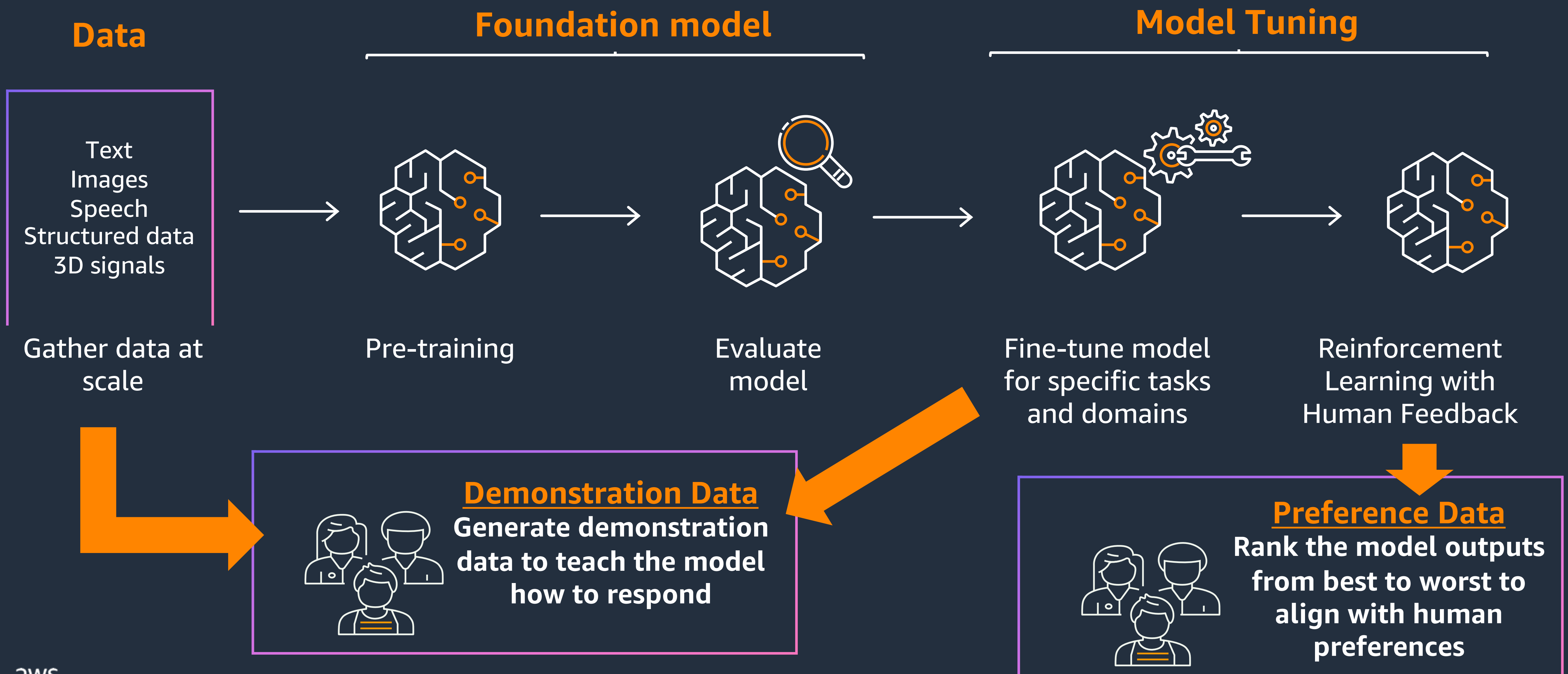
Task/Domain Specific Prompt and Responses



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Data labeling needs for Generative AI models

Helping customers to generate, automate, and personalize the customer experience



Data

Text
Images
Speech
Structured data
3D signals

Gather data at scale

Foundation model

Pre-training

Evaluate model

Model Tuning

Fine-tune model for specific tasks and domains

Reinforcement Learning with Human Feedback

Demonstration Data

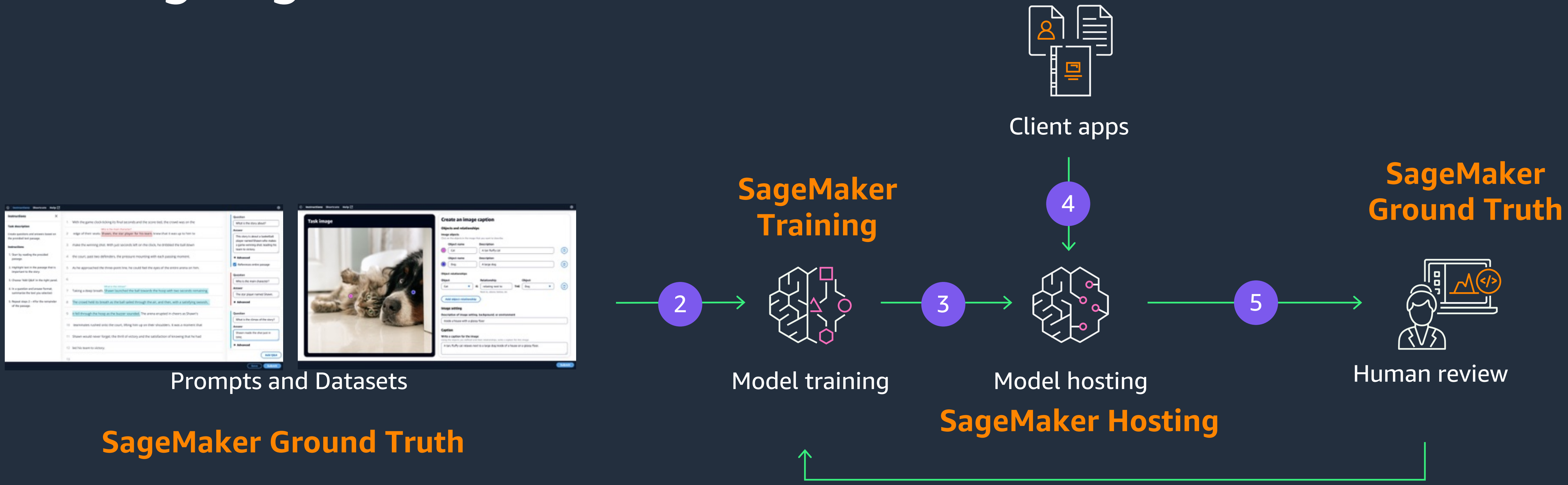
Generate demonstration data to teach the model how to respond

Preference Data

Rank the model outputs from best to worst to align with human preferences

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Using SageMaker for Generative AI



Client apps

Prompts and Datasets

SageMaker Ground Truth

SageMaker Training

Model training

Model hosting

SageMaker Hosting

SageMaker Ground Truth

Human review

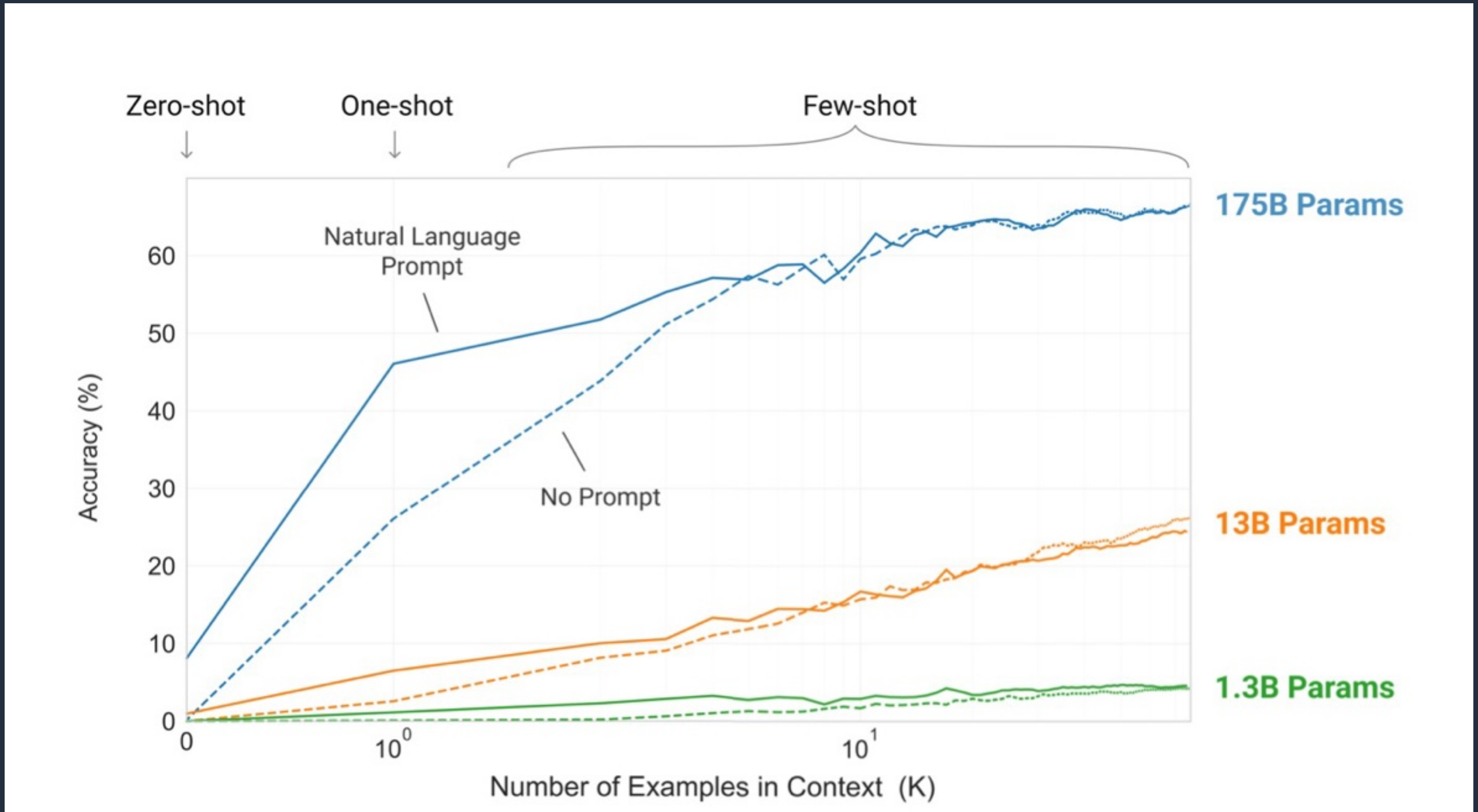
Human Feedback and Preference Data

SageMaker Ground Truth

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Before Fine-Tuning, try Prompt Engineering

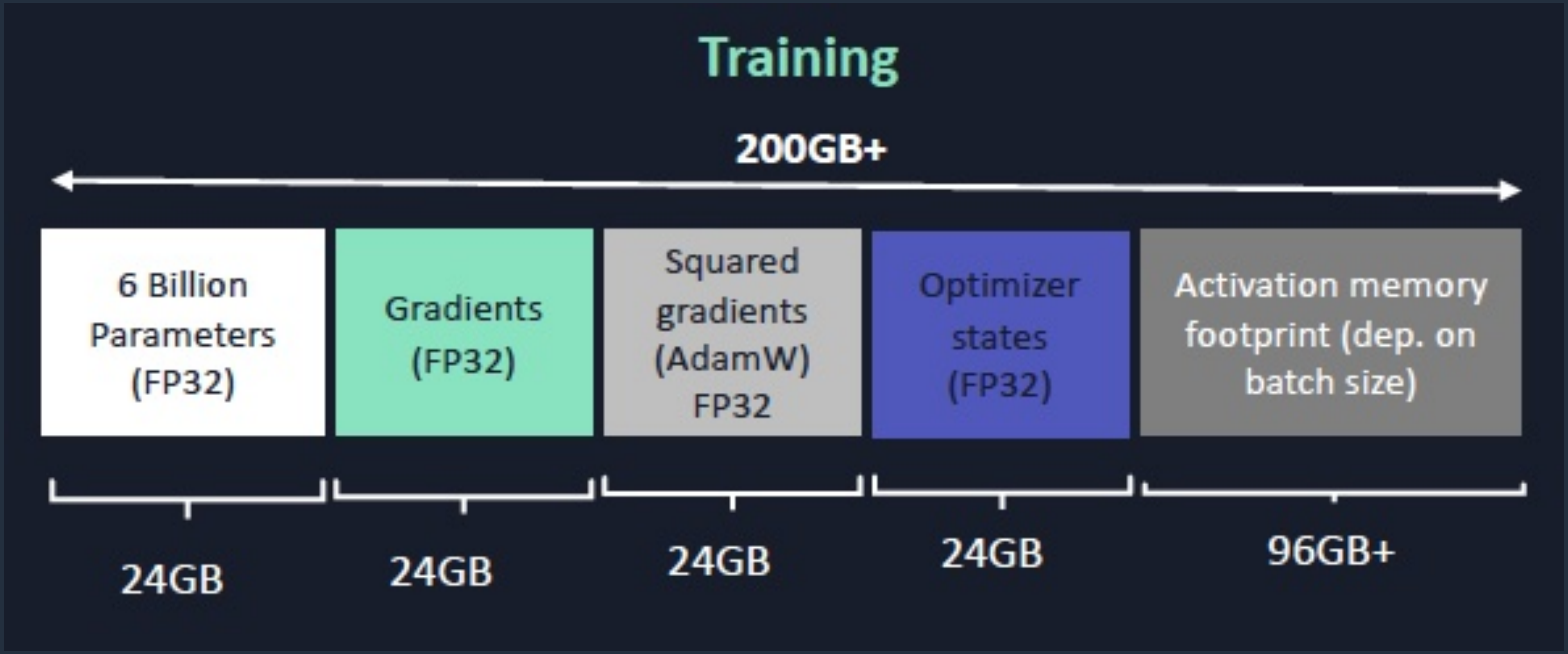
Impact of model size on prompt accuracy due to increase in pattern recognition abilities and ‘learn’ from in-context learning for Zero-, One- and Few-shot prompts.



Source: Brown et. al. https://arxiv.org/abs/2005.14165

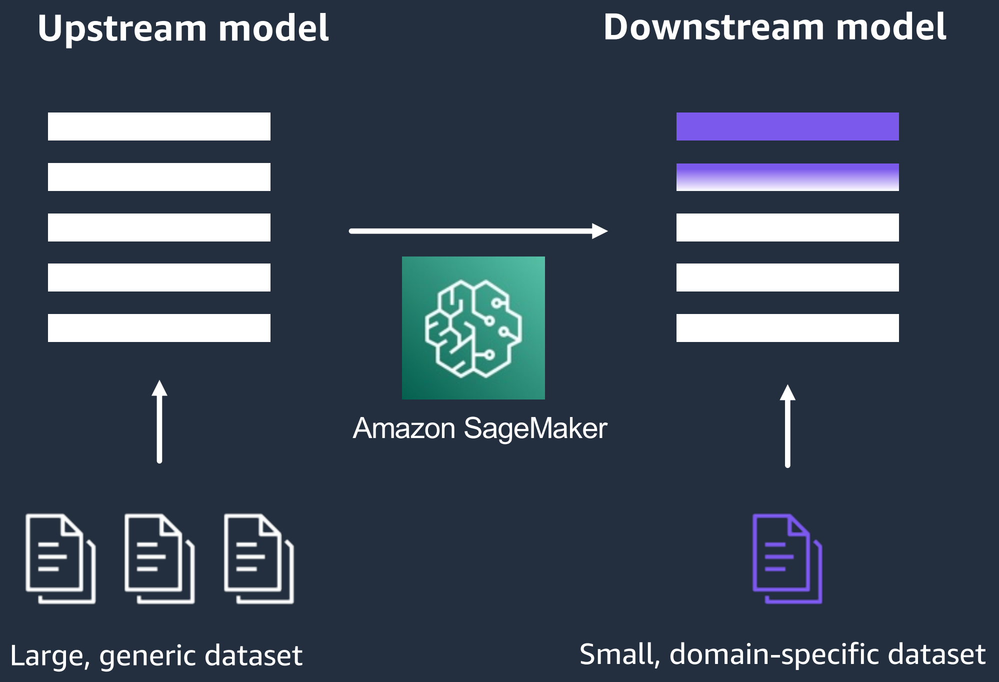
© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Fine Tuning a GPT-J 6B model


Training
200GB+
6 Billion Parameters (FP32)
Gradients (FP32)
Squared gradients (AdamW) FP32
Optimizer states (FP32)
Activation memory footprint (dep. on batch size)
24GB
24GB
24GB
24GB
96GB+

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Improving LLM performance by fine-tuning

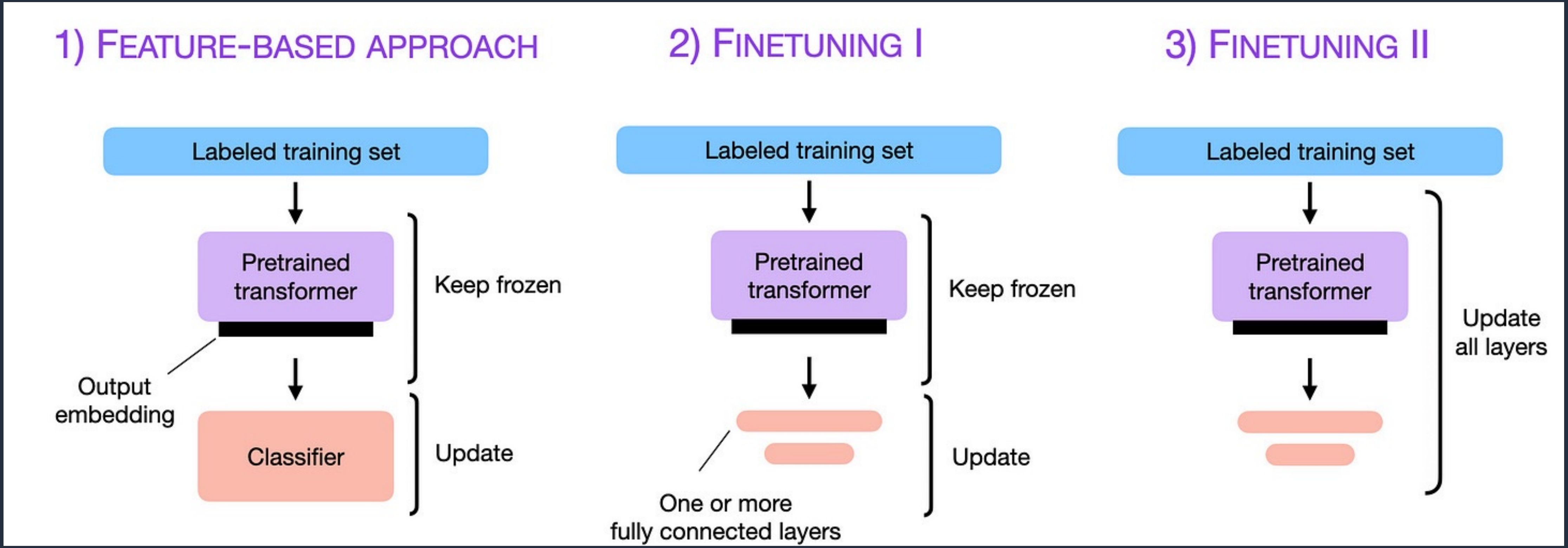


- Transfer learning of domain-specific knowledge into a foundation model at reasonable cost
- Update of weights in the network, while architecture is kept
- Fine-tuning is task-specific, either semi-supervised (e.g., MLM, CLM, PLM, …) or supervised (e.g. translation, classification, …)

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Domain task specific fine-tuning approaches

Classification example with an encoder model

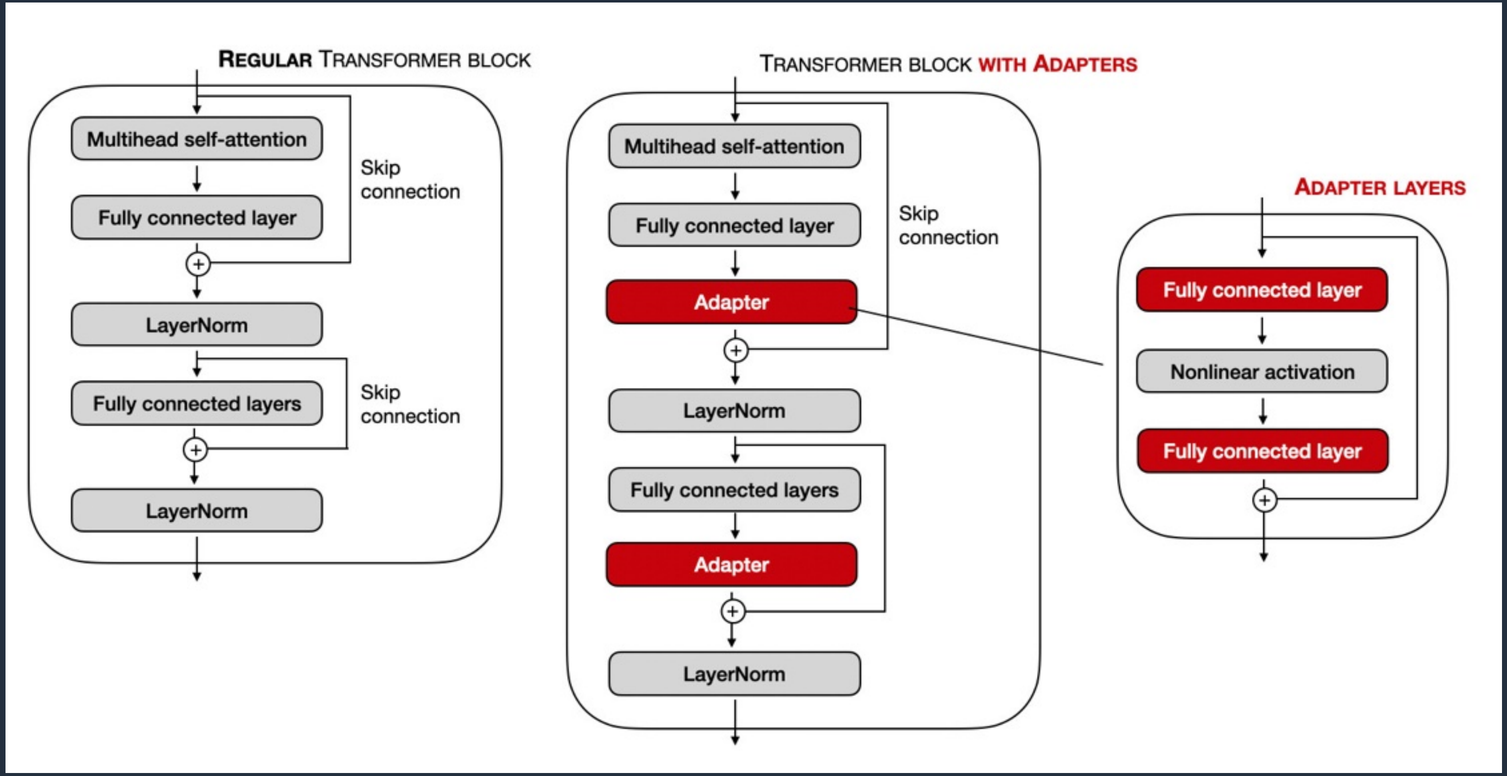


* https://magazine.sebastianraschka.com/p/finetuning-large-language-models

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Parameter efficient Fine Tuning Techniques (1/2)
Adding adapter layers

We add adapter layers in our transformer architectur and only fine tune those.

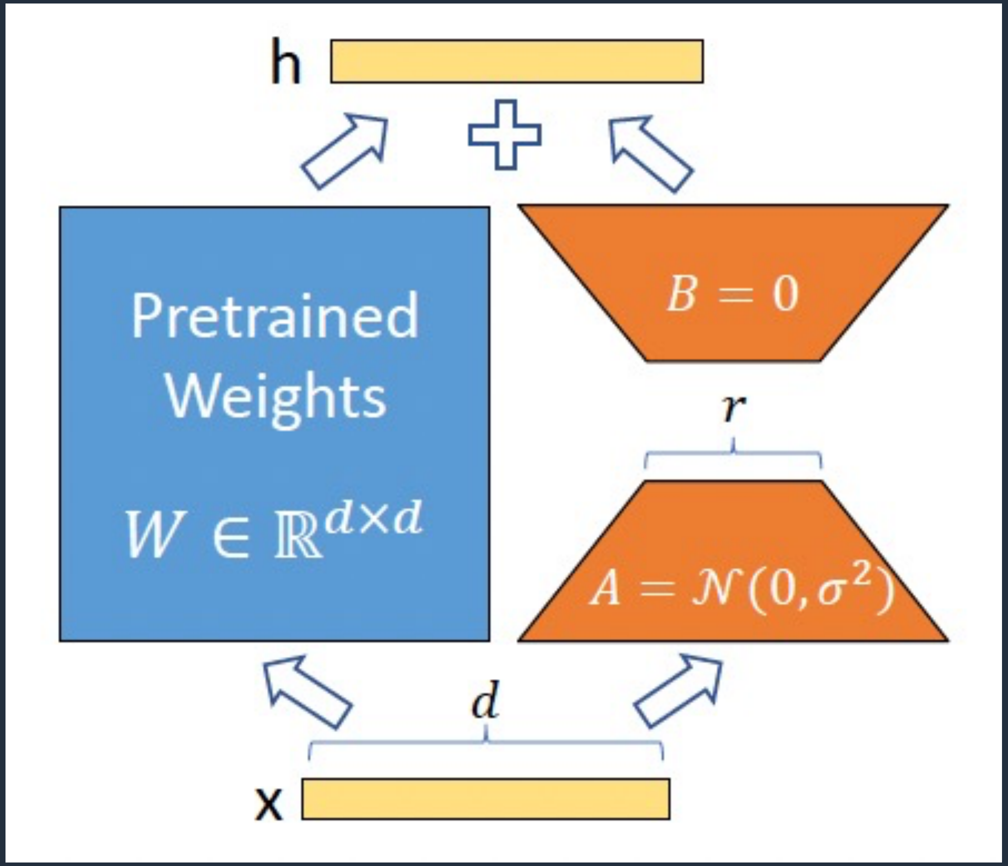


* https://magazine.sebastianraschka.com/p/finetuning-large-language-models

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Parameter efficient Fine Tuning Techniques (2/2)
Low Rank Adaptation

- Hypothesis: Weights of Fine-tuned (FT) LLMs have a low rank.
- Using matrix decomposition to exploit this characteristic to get new matrixes **B** and **A**.
- We only have to update **B** and **A**. Can finally be merged with W for 0 latency increase.

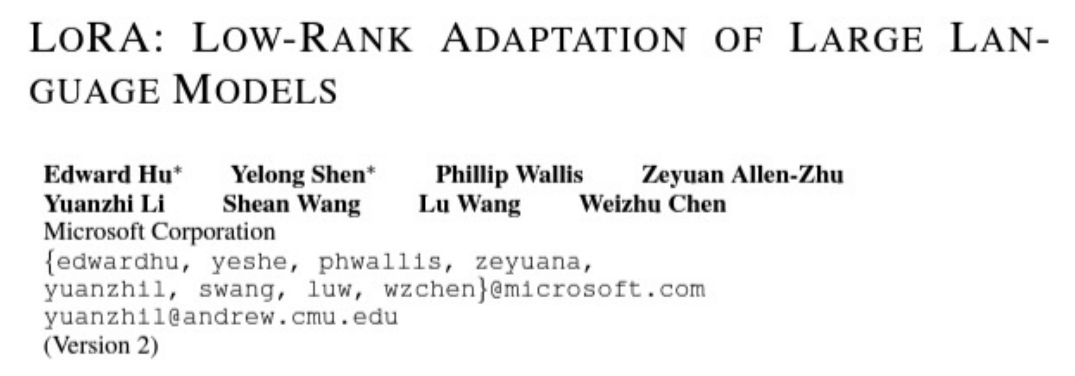

h
Pretrained Weights
$W \in \mathbb{R}^{d \times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

* Hu et al. 2021

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

LoRA

- Pick a rank for that is right for your use case (experimentation might be needed)
- Adapt the weight matrices for Query and Value in the attention blocks



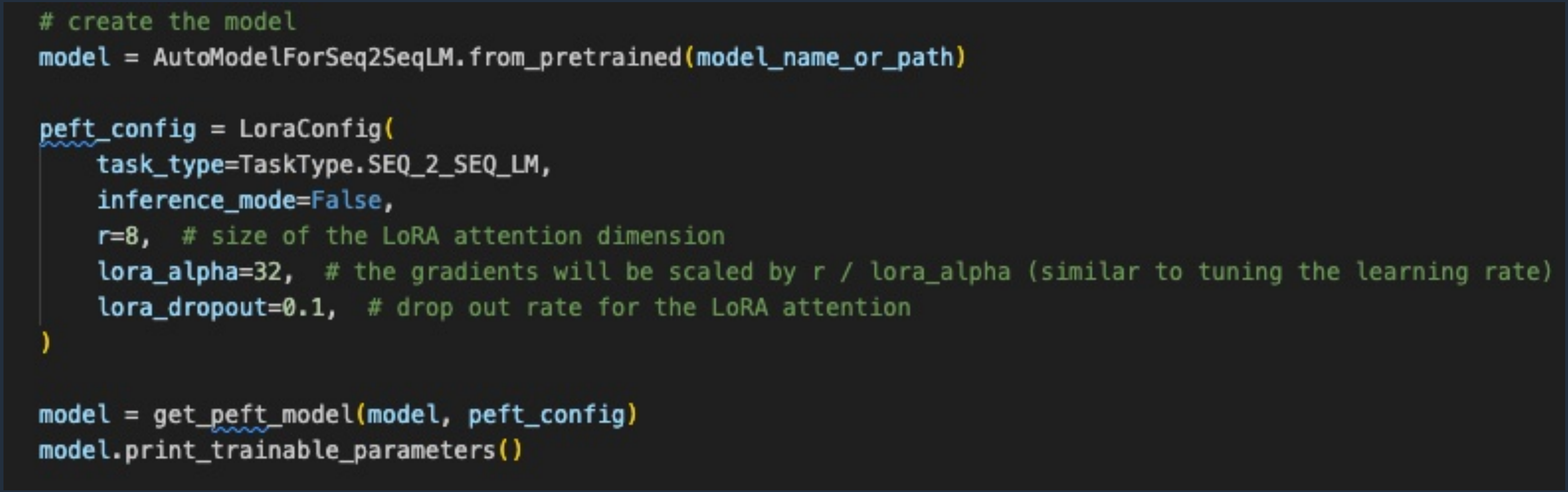
LoRA: Low-Rank Adaptation of Large Language Models

Edward Hu* **Yelong Shen*** **Phillip Wallis** **Zeyuan Allen-Zhu**
Yuanzhi Li **Shean Wang** **Lu Wang** **Weizhu Chen**
Microsoft Corporation
`{edwardhu, yeshe, phwallis, zeyuana,`
`yuanzhil, swang, luw, wzchen}@microsoft.com`
`yuanzhil@andrew.cmu.edu`
(Version 2)

* Hu et al. 2021

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

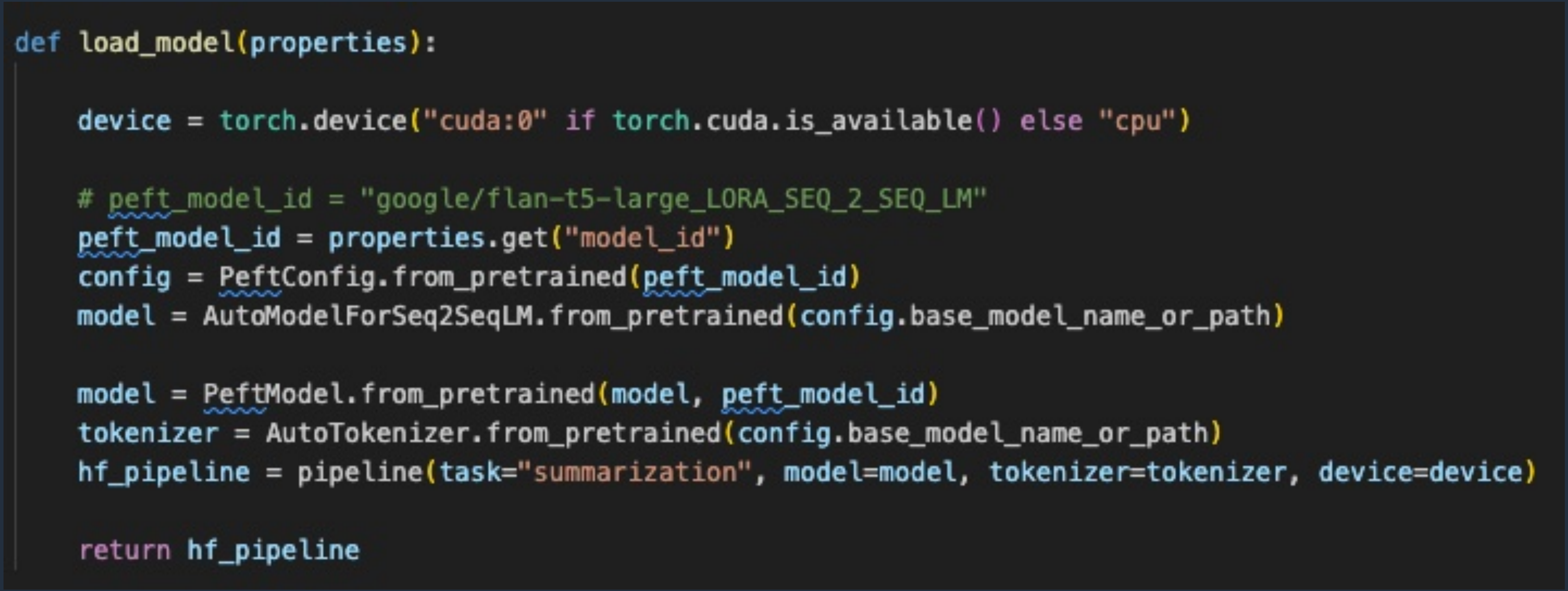
LoRA in code – implement it in the training loop



```
# create the model
model = AutoModelForSeq2SeqLM.from_pretrained(model_name_or_path)

peft_config = LoraConfig(
    task_type=TaskType.SEQ_2_SEQ_LM,
    inference_mode=False,
    r=8,  # size of the LoRA attention dimension
    lora_alpha=32,  # the gradients will be scaled by r / lora_alpha (similar to tuning the learning rate)
    lora_dropout=0.1,  # drop out rate for the LoRA attention
)

model = get_peft_model(model, peft_config)
model.print_trainable_parameters()
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

LoRA in code – implement it in the training loop



```
def load_model(properties):

    device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")

    # peft_model_id = "google/flan-t5-large_LORA_SEQ_2_SEQ_LM"
    peft_model_id = properties.get("model_id")
    config = PeftConfig.from_pretrained(peft_model_id)
    model = AutoModelForSeq2SeqLM.from_pretrained(config.base_model_name_or_path)

    model = PeftModel.from_pretrained(model, peft_model_id)
    tokenizer = AutoTokenizer.from_pretrained(config.base_model_name_or_path)
    hf_pipeline = pipeline(task="summarization", model=model, tokenizer=tokenizer, device=device)

    return hf_pipeline
```

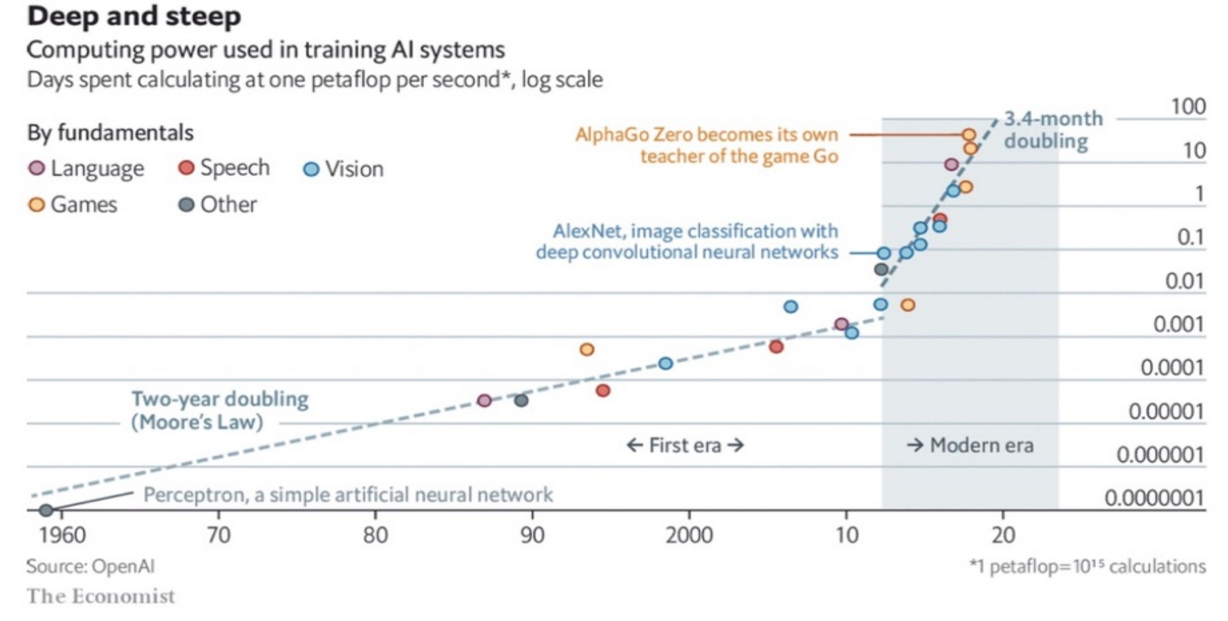
© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Challenges with large model training

MODELS GROW FASTER THAN HARDWARE, LEADING TO BOTTLENECKS

INCREASING COMPLEXITY

Compute power ~ 2x every 3.4 months



Deep and steep

Computing power used in training AI systems

Days spent calculating at one petaflop per second*, log scale

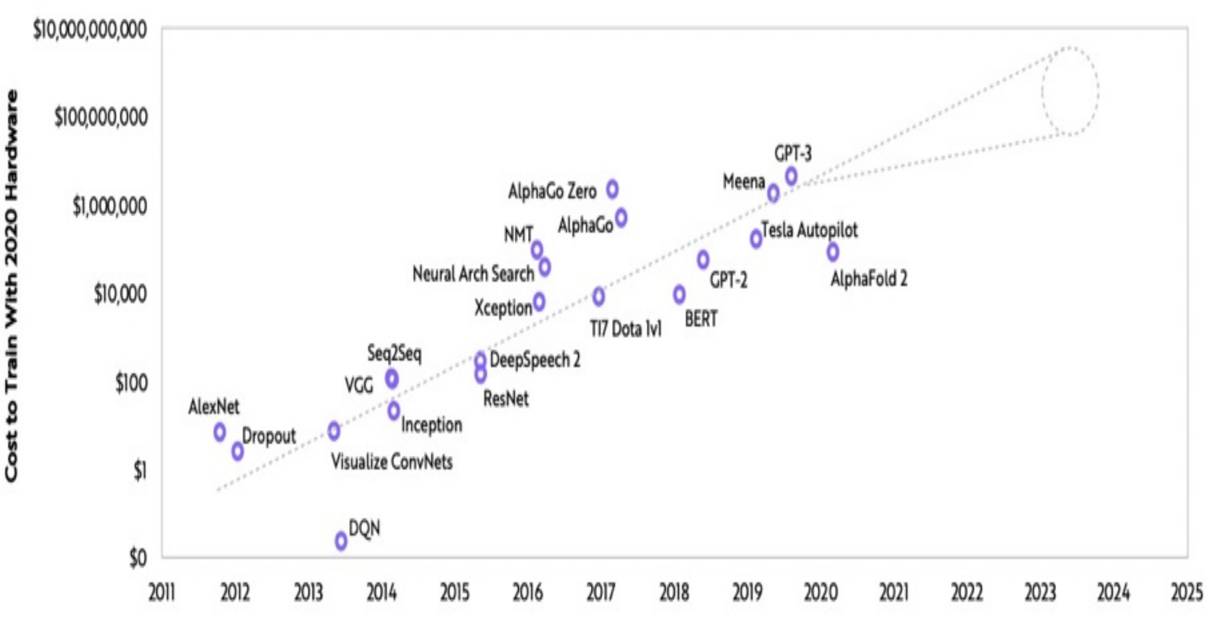
Source: OpenAI

*1 petaflop=10^{15} calculations

The Economist

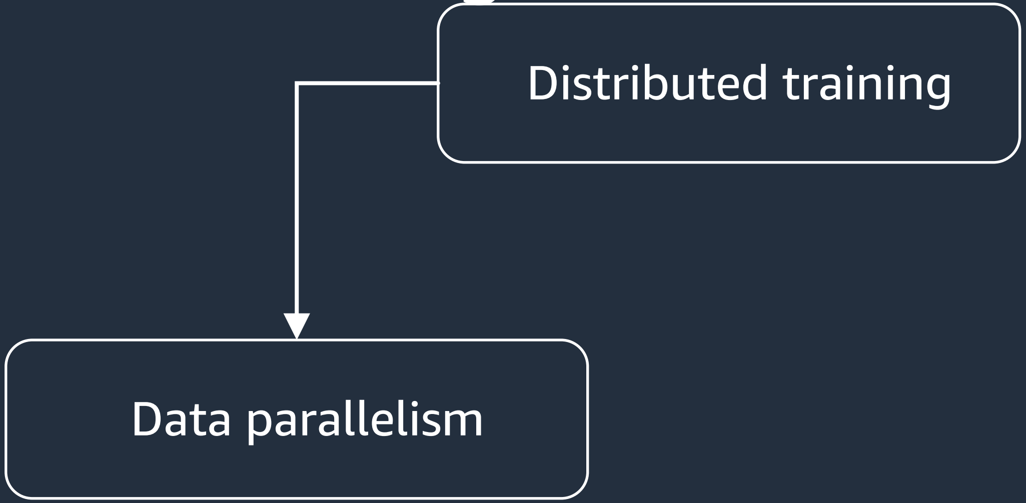
INCREASING COSTS

Model size increase ~ 10x/ year, Cost of Training increase ~ 100x by 2025



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Distributed training

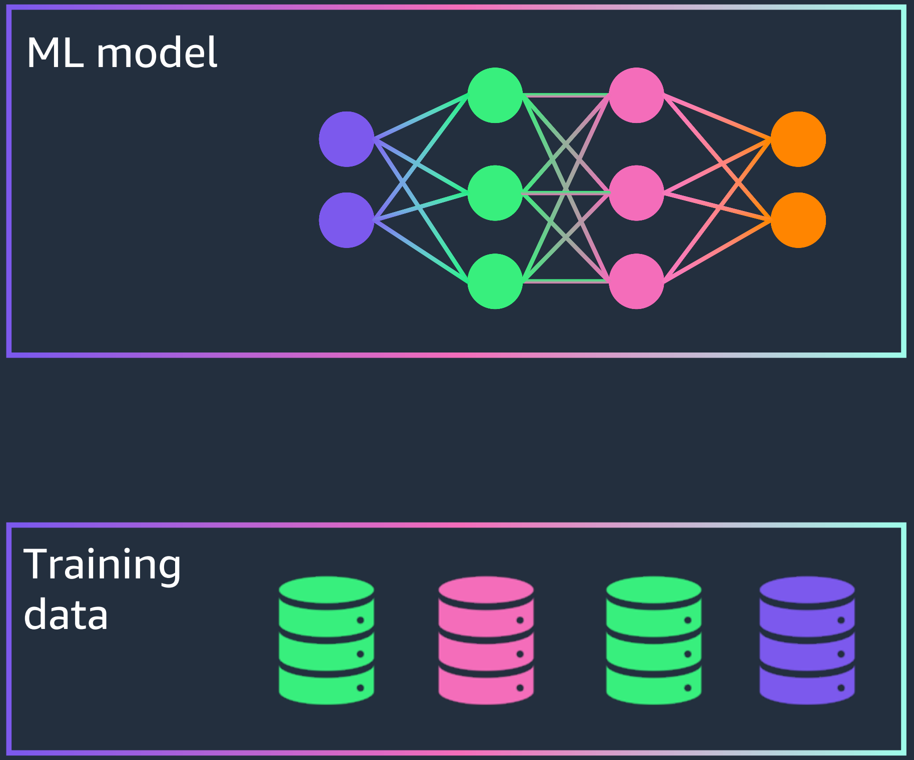


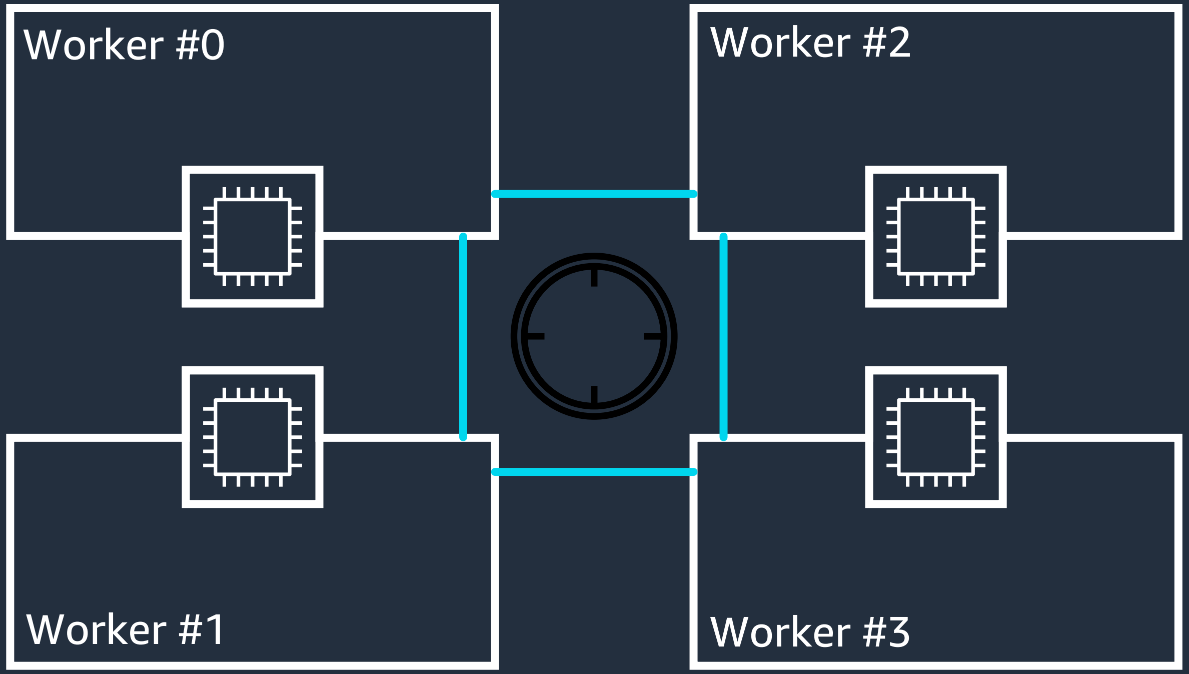
Distributed training

Data parallelism

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Data parallelism



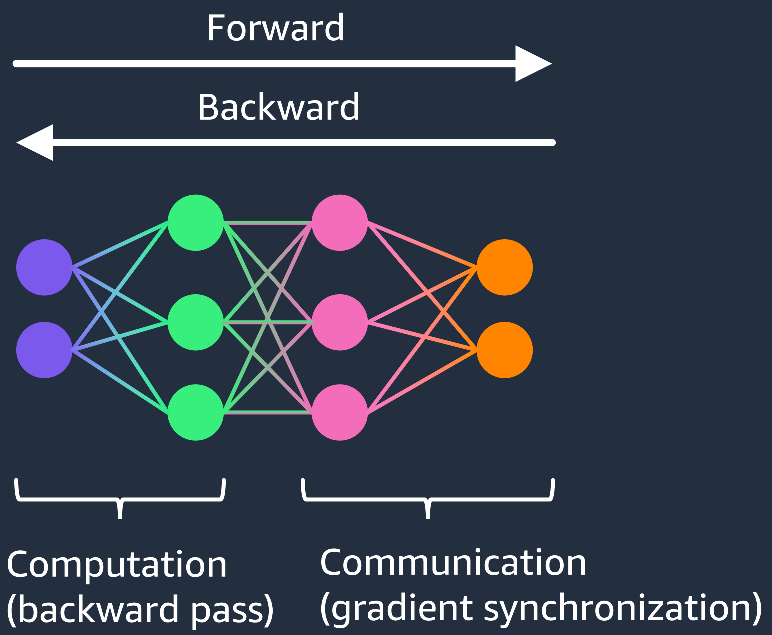


© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Data parallelism

Benefits

- Simple, scalable
- Simple computation/ communication overlap

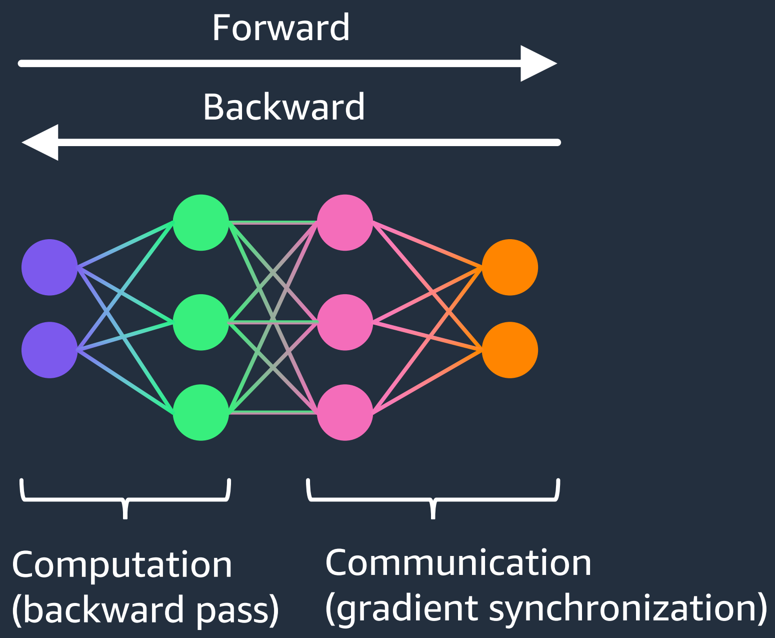

Forward
Backward
Computation (backward pass)
Communication (gradient synchronization)

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Data parallelism

Benefits

- Simple, scalable
- Simple computation/ communication overlap

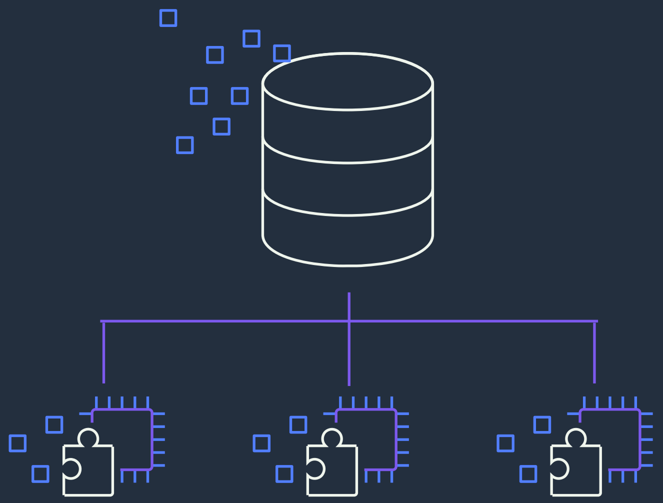

Forward
Backward
Computation (backward pass)
Communication (gradient synchronization)

Challenges

- What if the model doesn't fit?
  → Large memory

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

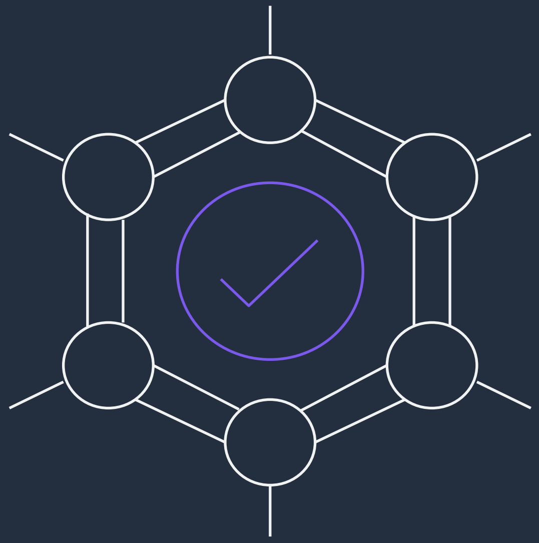
Distributed deep learning, historically



Parameter server

E.g., TensorFlow ParameterServerStrategy

Combiners worker & parameter servers to communicate and average gradients



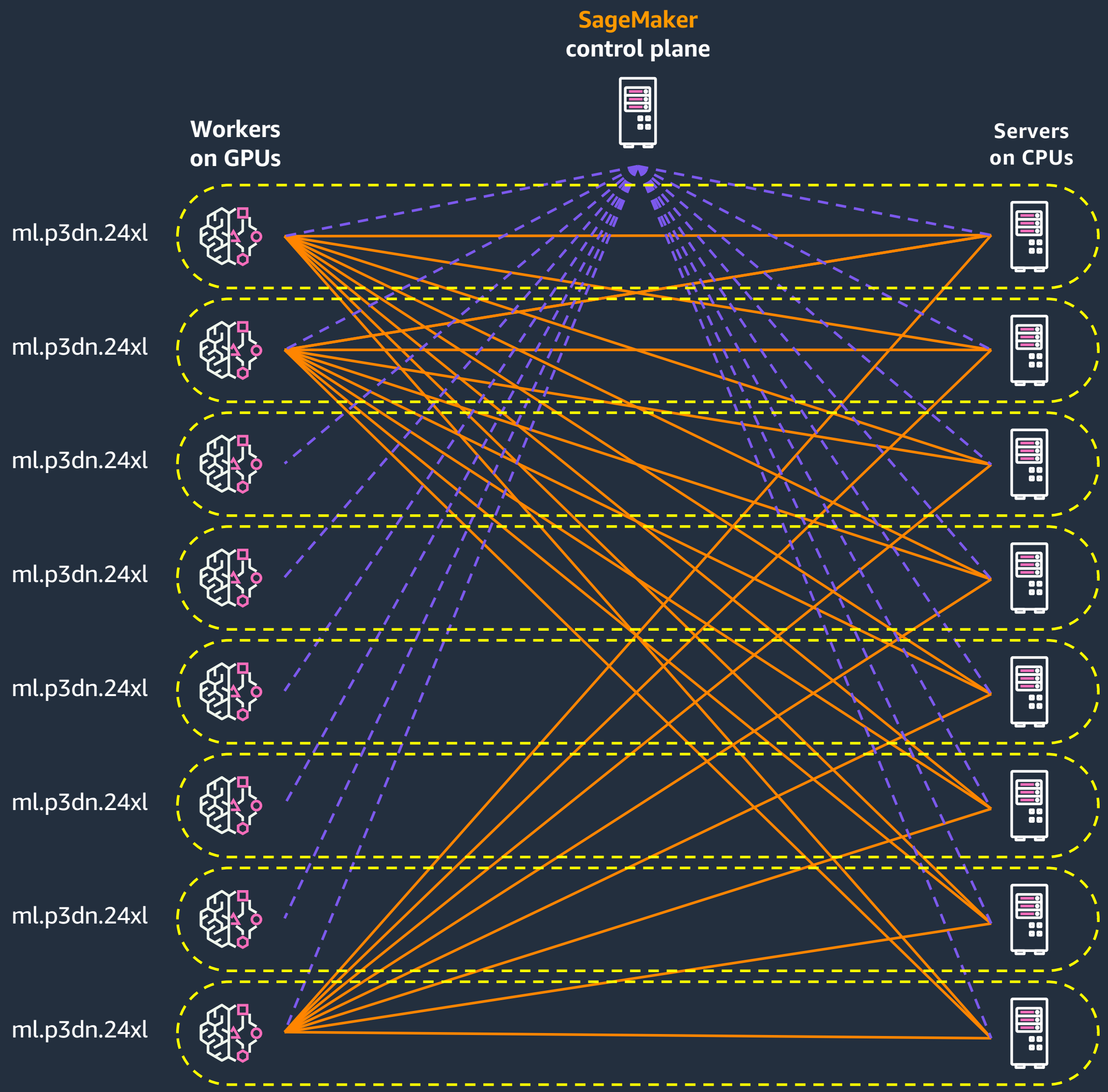
AllReduce

E.g., Horovod, PyTorch DistributedDataParallel

Uses MPI to allow GPU nodes to communicate directly with each other in a “ring” topology

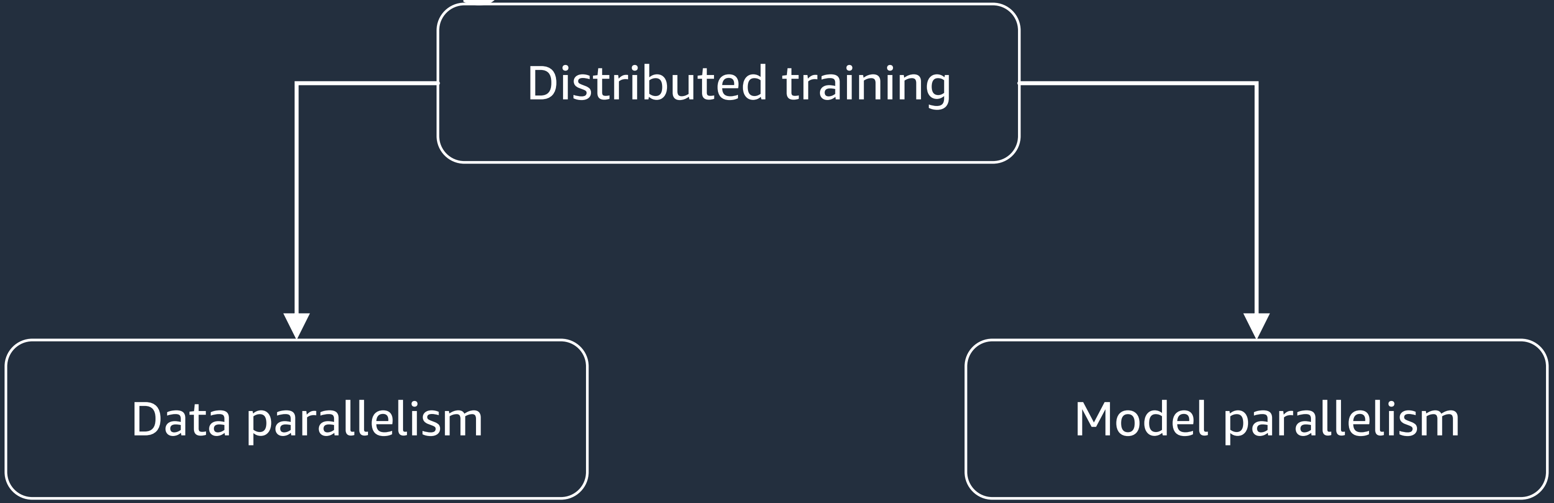
© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

SageMaker DataParallel under the hood



- With SageMaker DDP, the CPUs on your GPUs operate like parameter servers
- We introduce Balanced Fusion Buffers to optimize your network bandwidth, holding gradients until they hit a threshold size, then copy to CPU memory, shard into N parts for each node in cluster, send ith part to the ith server
- Now we can overlap backward pass and AllReduce

Distributed training



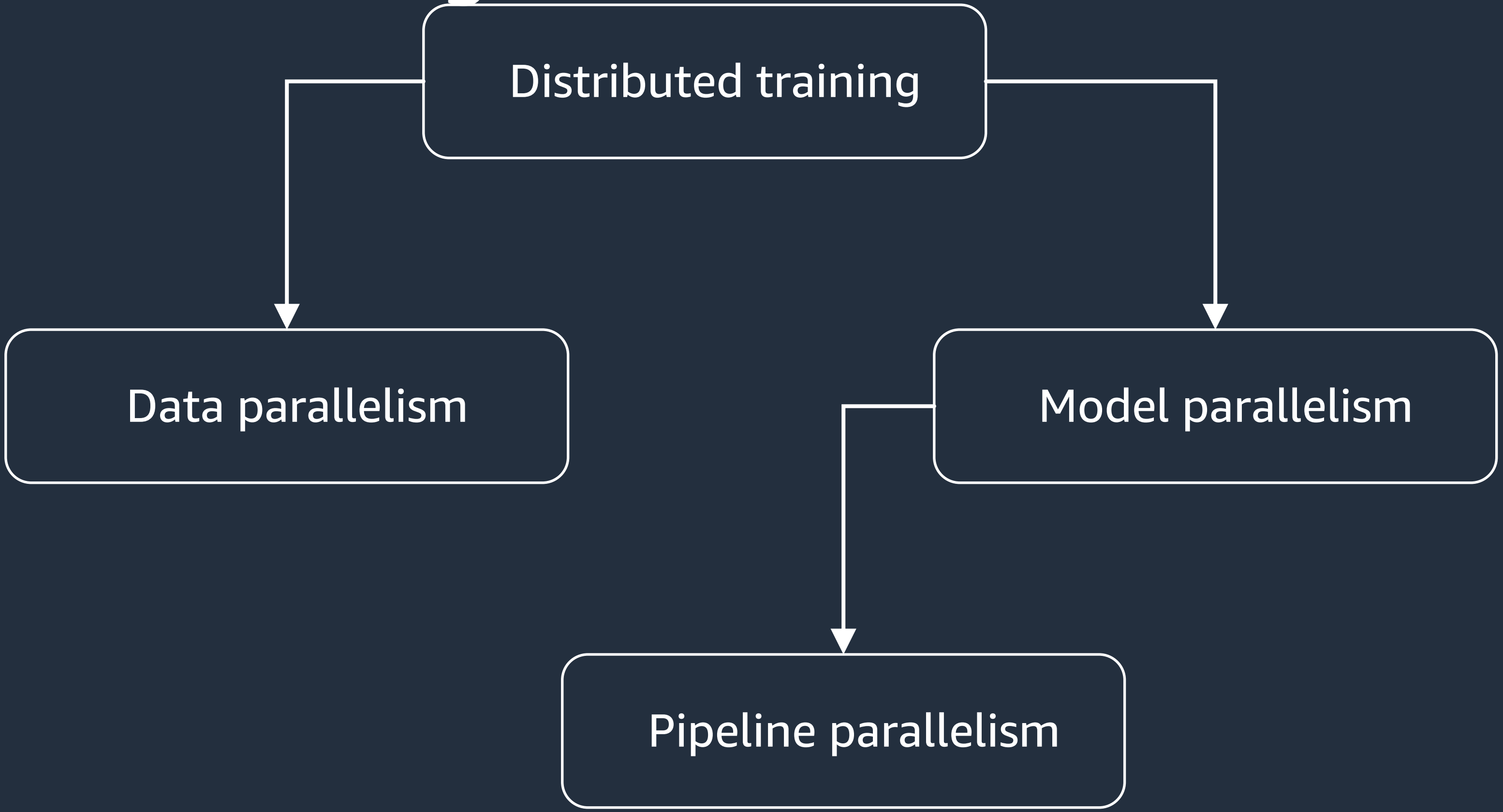
Distributed training

Data parallelism

Model parallelism

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Distributed training


Distributed training

Data parallelism

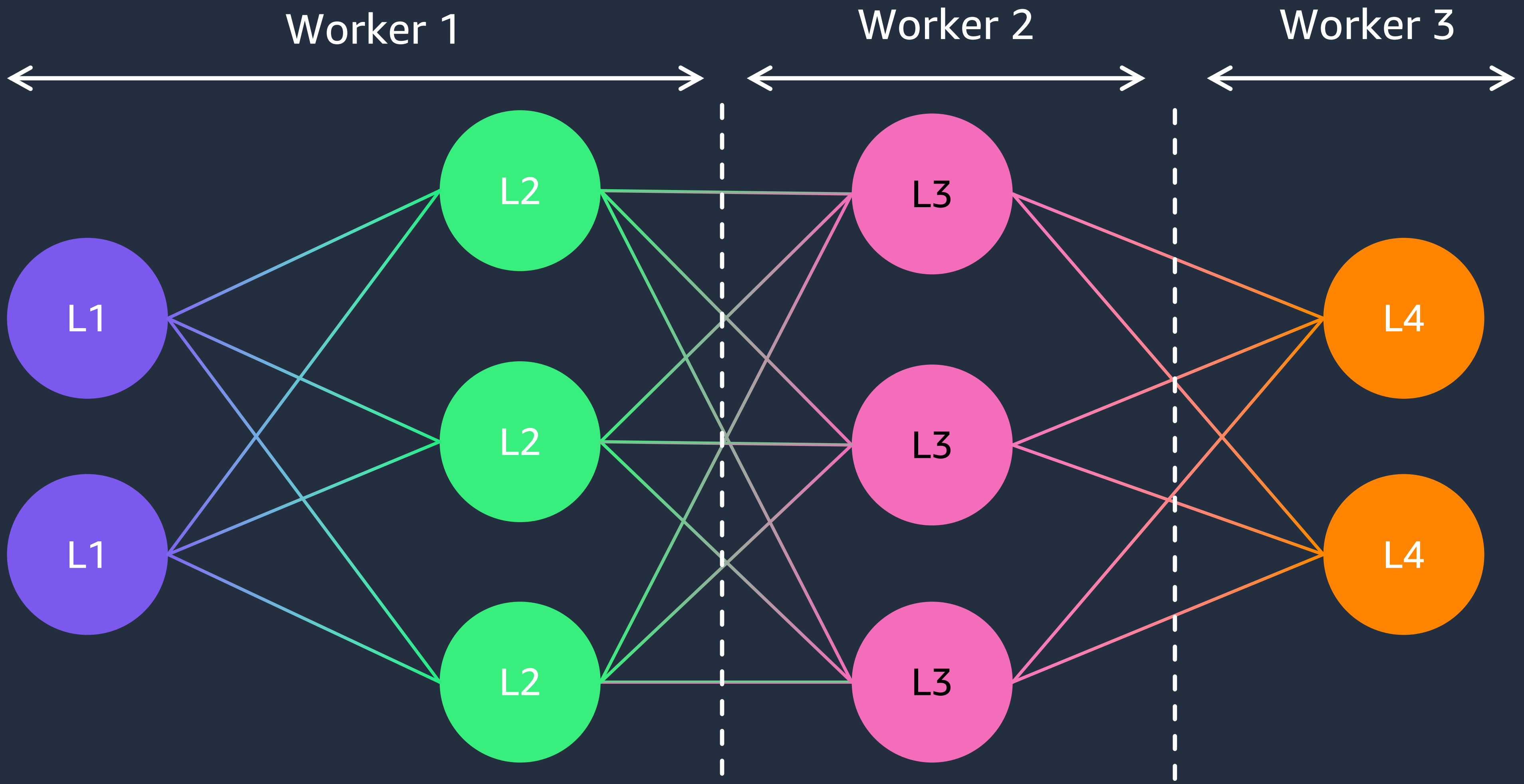
Model parallelism

Pipeline parallelism

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Pipeline parallelism

ALSO CALLED INTER-LAYER MODEL PARALLELISM



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

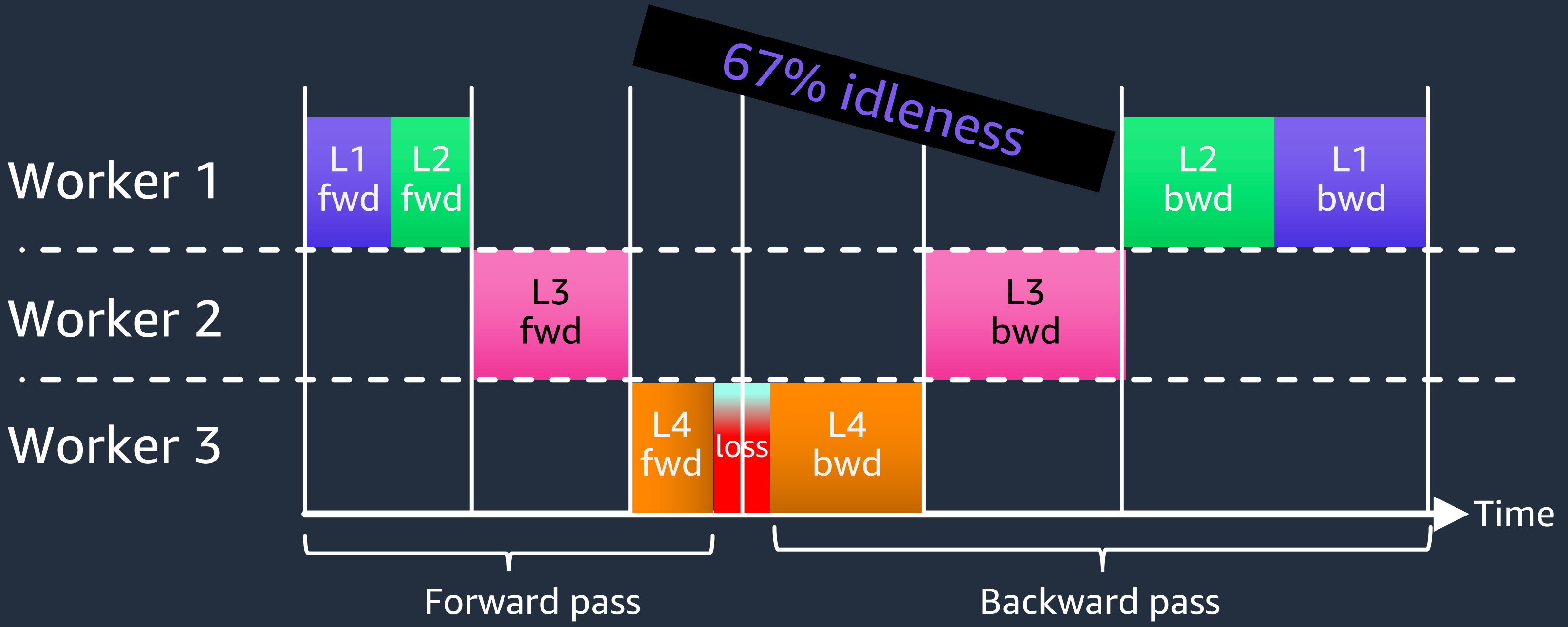
Pipeline parallelism

Benefits

- Fits larger models
- P2P communication

Challenges

- What if a layer doesn't fit?
  → Large memory
- Pipeline idleness reduces scaling efficiency



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Pipeline parallelism

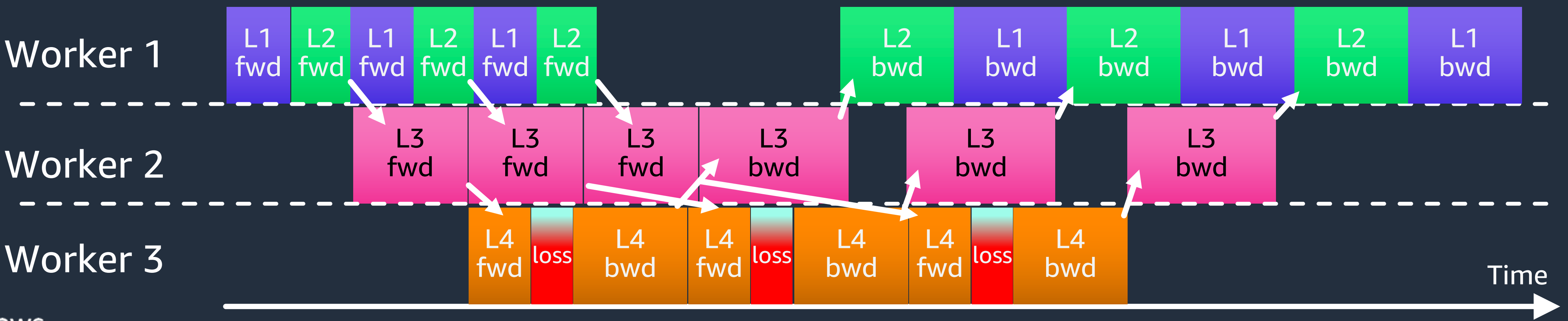
Benefits

- Fits larger models
- P2P communication

Challenges

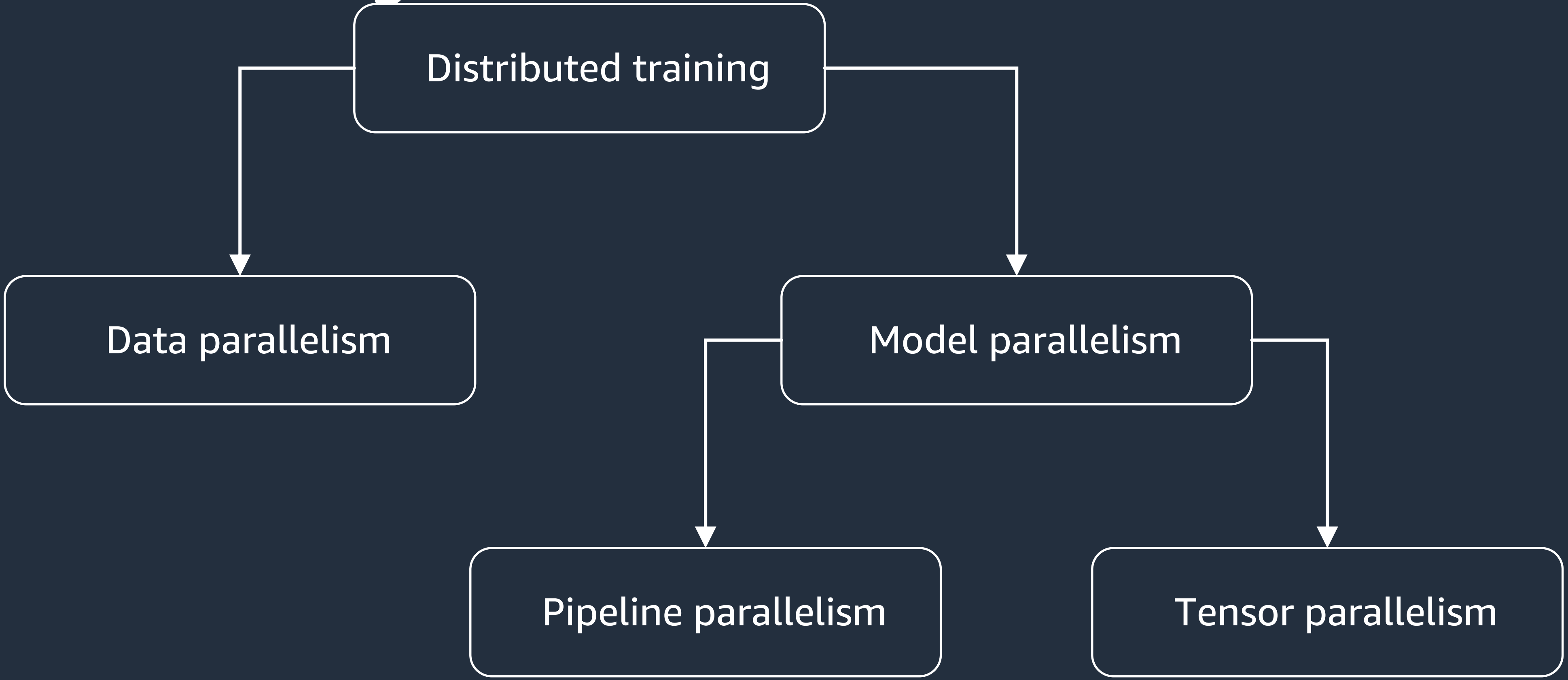
- What if a layer doesn't fit?
  → Large memory
- Pipeline idleness reduces scaling efficiency
  → Overlapped comp/comm

3 microbatches
67% → 40% (idleness)



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

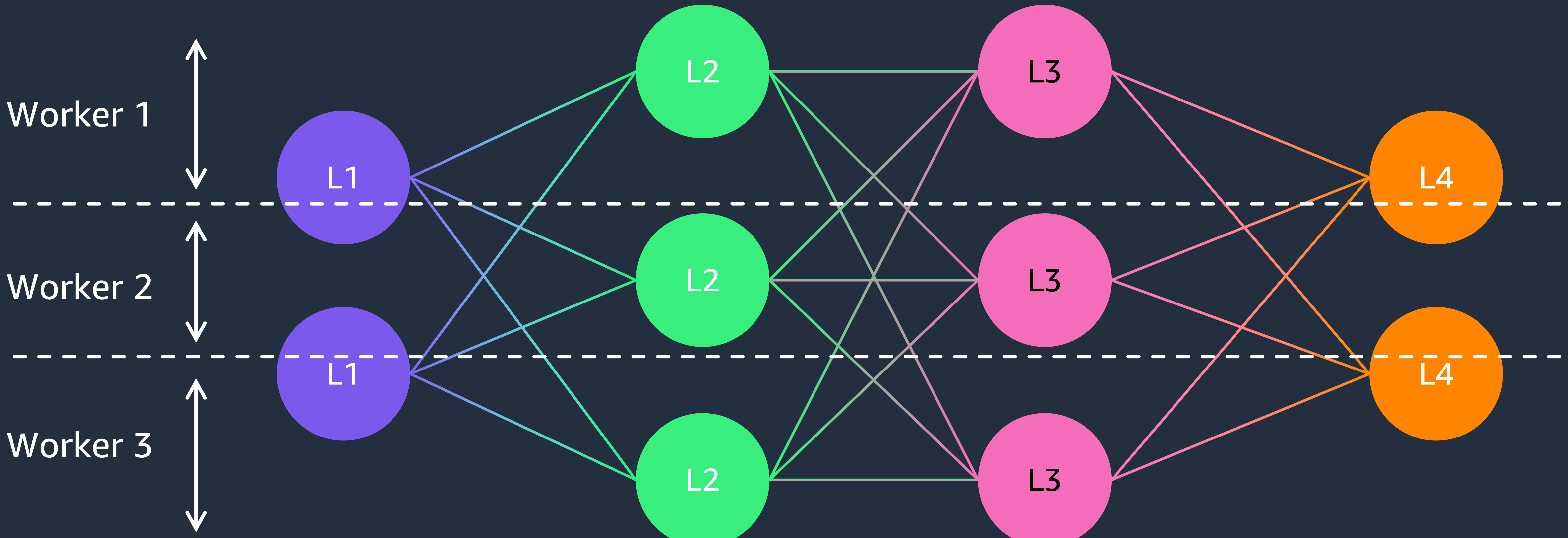
Distributed training



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

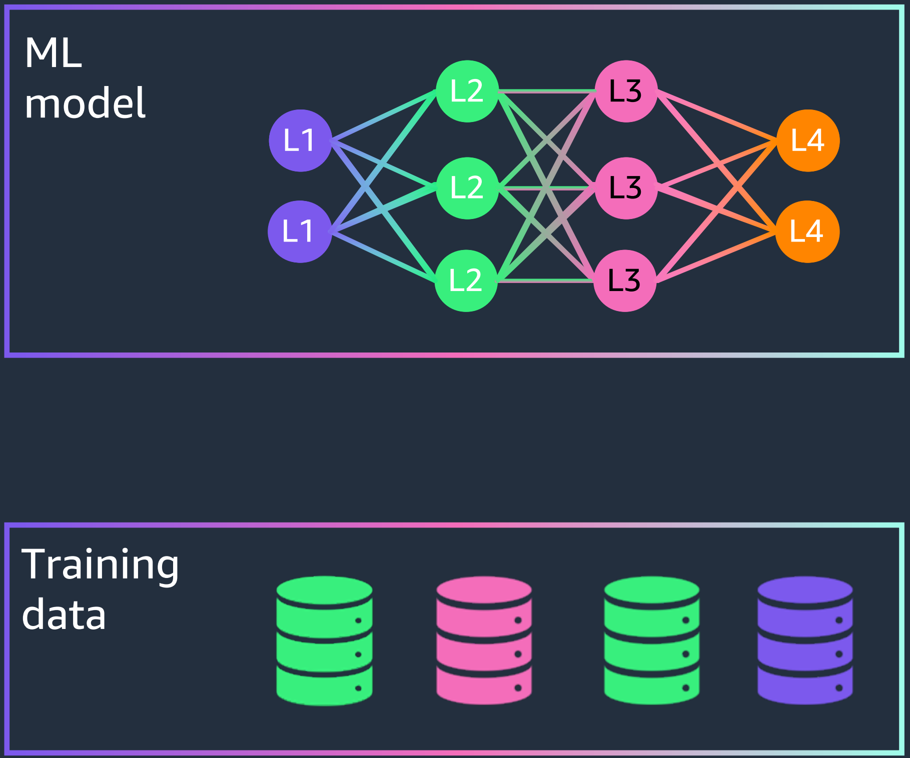
Tensor parallelism (TP)

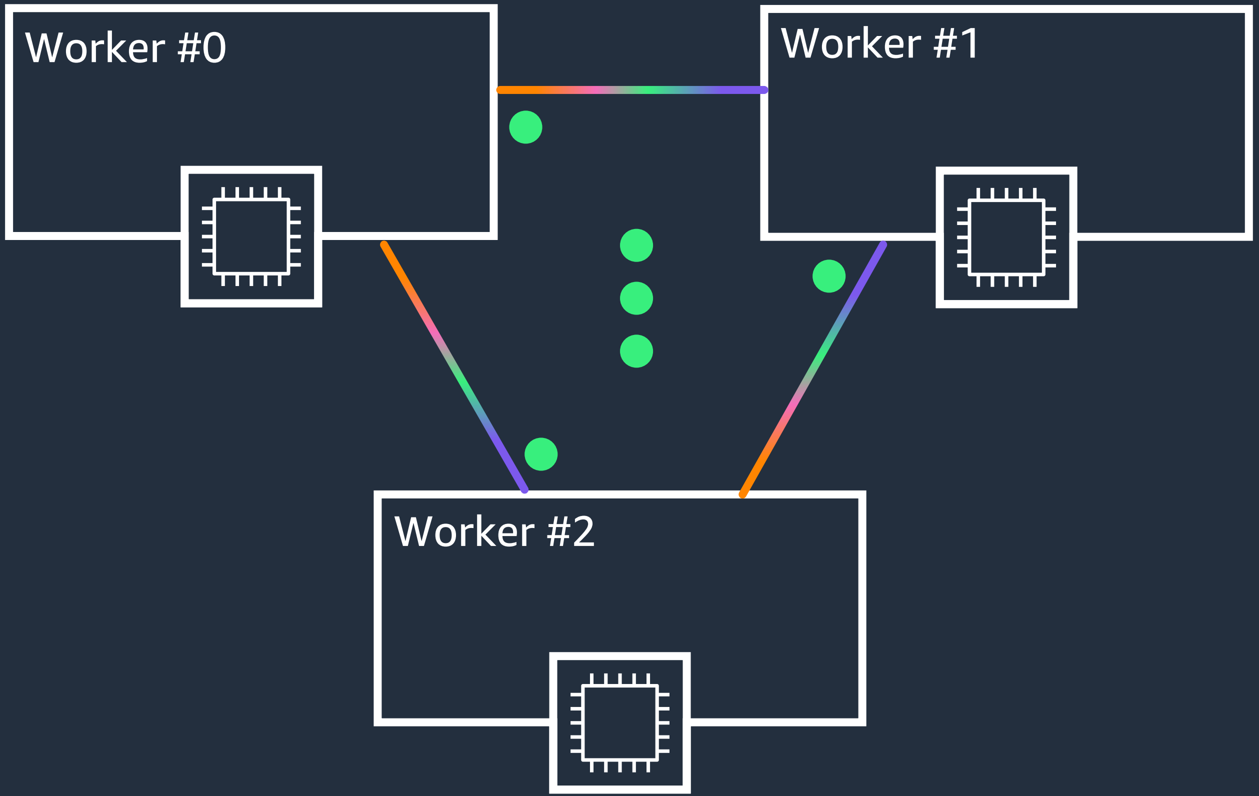
ALSO CALLED INTRA-LAYER MODEL PARALLELISM



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Tensor parallelism (TP)





© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Tensor parallelism (TP)

Benefits

- Fits large models/layers
- Reduces minibatch size

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Tensor parallelism (TP)

Benefits

- Fits large models/layers
- Reduces minibatch size

Challenges

- All-to-all communication
  → High-bandwidth, low-latency interconnect
- Difficult to overlap computation and communication
  → Overlapped comp/comm

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Aspects of Distributed Training

Compute

Storage

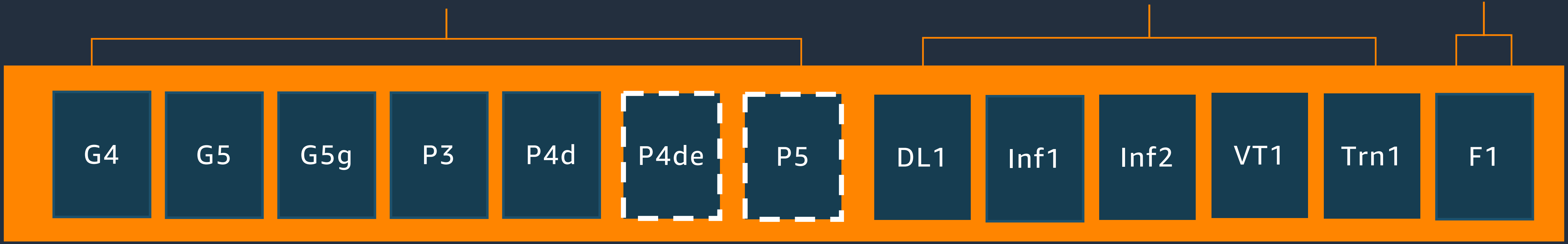
Network

Orchestration

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Broad and deep accelerated computing portfolio

GPU, AWS BUILT ACCELERATORS, AND FPGA-BASED EC2 INSTANCES



GPUs

AI/ML Accelerators and ASICs

FPGAs

G4 | G5 | G5g | P3 | P4d | P4de | P5 | DL1 | Inf1 | Inf2 | VT1 | Trn1 | F1

Preview

Habana Gaudi
accelerator

Radeon GPU
Xilinx FPGA

Graviton CPU
Inferentia accelerator
Trainium accelerator

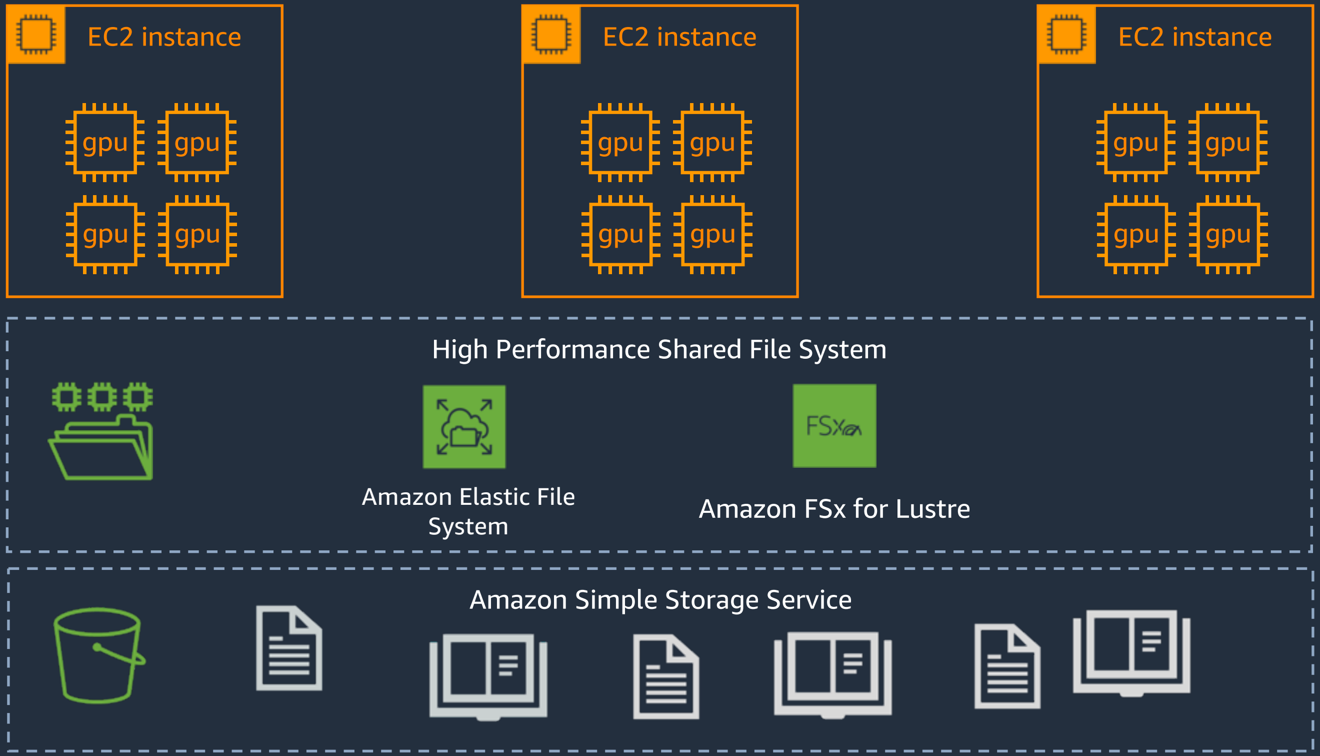
A100, V100,
A10G, T4g GPUs

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Storage & Memory Challenges for ML training

NLU models trained on internet scale datasets:

- Original BERT pre-trained on **16 GB of Wikipedia** text (2500M tokens) & **11k books** (800M tokens)
- T5-XXL: Colossal Clean Crawled Corpus **750 GB**



EC2 instance

gpu gpu

gpu gpu

EC2 instance

gpu gpu

gpu gpu

EC2 instance

gpu gpu

gpu gpu

High Performance Shared File System

Amazon Elastic File System

Amazon FSx for Lustre

Amazon Simple Storage Service

Read intensive job:

Mini batch + data loader strategy during training is key.

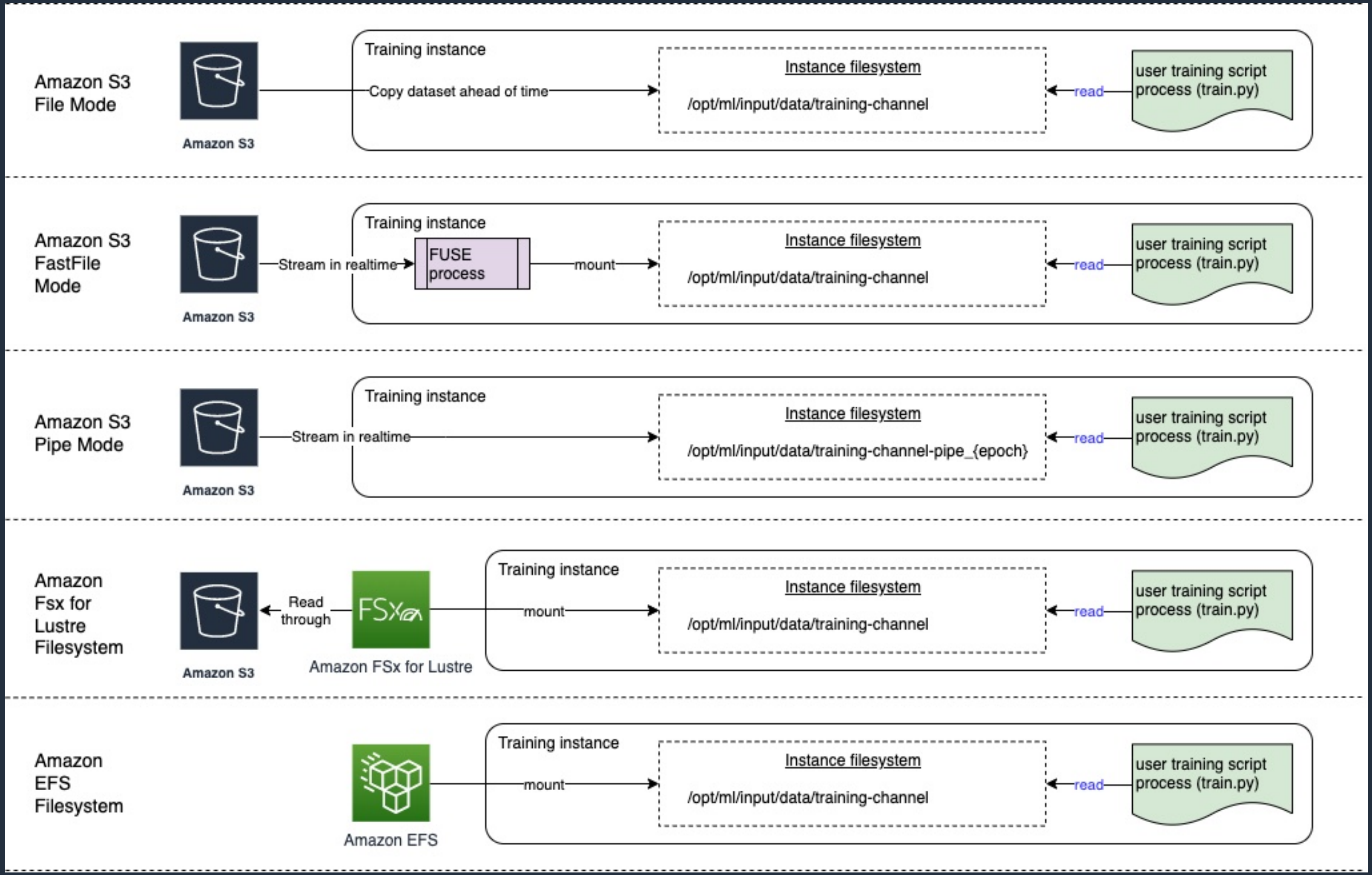
GPU Memory bound:

Device memory limits the amount of sentences (data) at each training step.

Rule of thumb: Larger the batch size, faster the training!

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

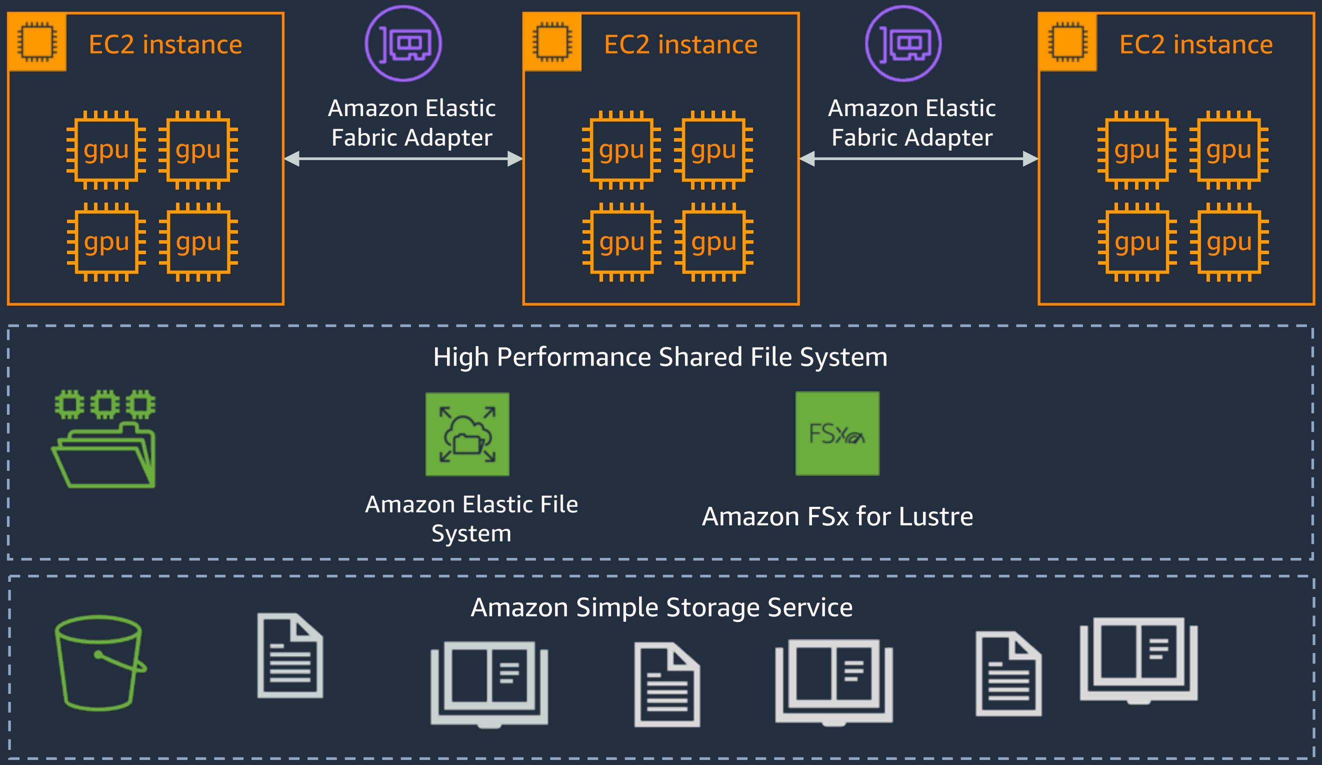
Storage options for SageMaker (and others)


Amazon S3 File Mode
Amazon S3
Training instance
Copy dataset ahead of time
Instance filesystem
/opt/ml/input/data/training-channel
read
user training script process (train.py)
Amazon S3 FastFile Mode
Amazon S3
Training instance
Stream in realtime
FUSE process
mount
Instance filesystem
/opt/ml/input/data/training-channel
read
user training script process (train.py)
Amazon S3 Pipe Mode
Amazon S3
Training instance
Stream in realtime
Instance filesystem
/opt/ml/input/data/training-channel-pipe_{epoch}
read
user training script process (train.py)
Amazon Fsx for Lustre Filesystem
Amazon S3
Read through
FSx
Amazon FSx for Lustre
Training instance
mount
Instance filesystem
/opt/ml/input/data/training-channel
read
user training script process (train.py)
Amazon EFS Filesystem
Amazon EFS
Training instance
mount
Instance filesystem
/opt/ml/input/data/training-channel
read
user training script process (train.py)

Networking Challenges for ML Training

Gradient descent over multi-node multi-gpu architecture:

- Model + mini-batch data fits on single GPU: Data Parallel Training
- Model + mini-batch needs multiple GPUs: Model Parallel Training

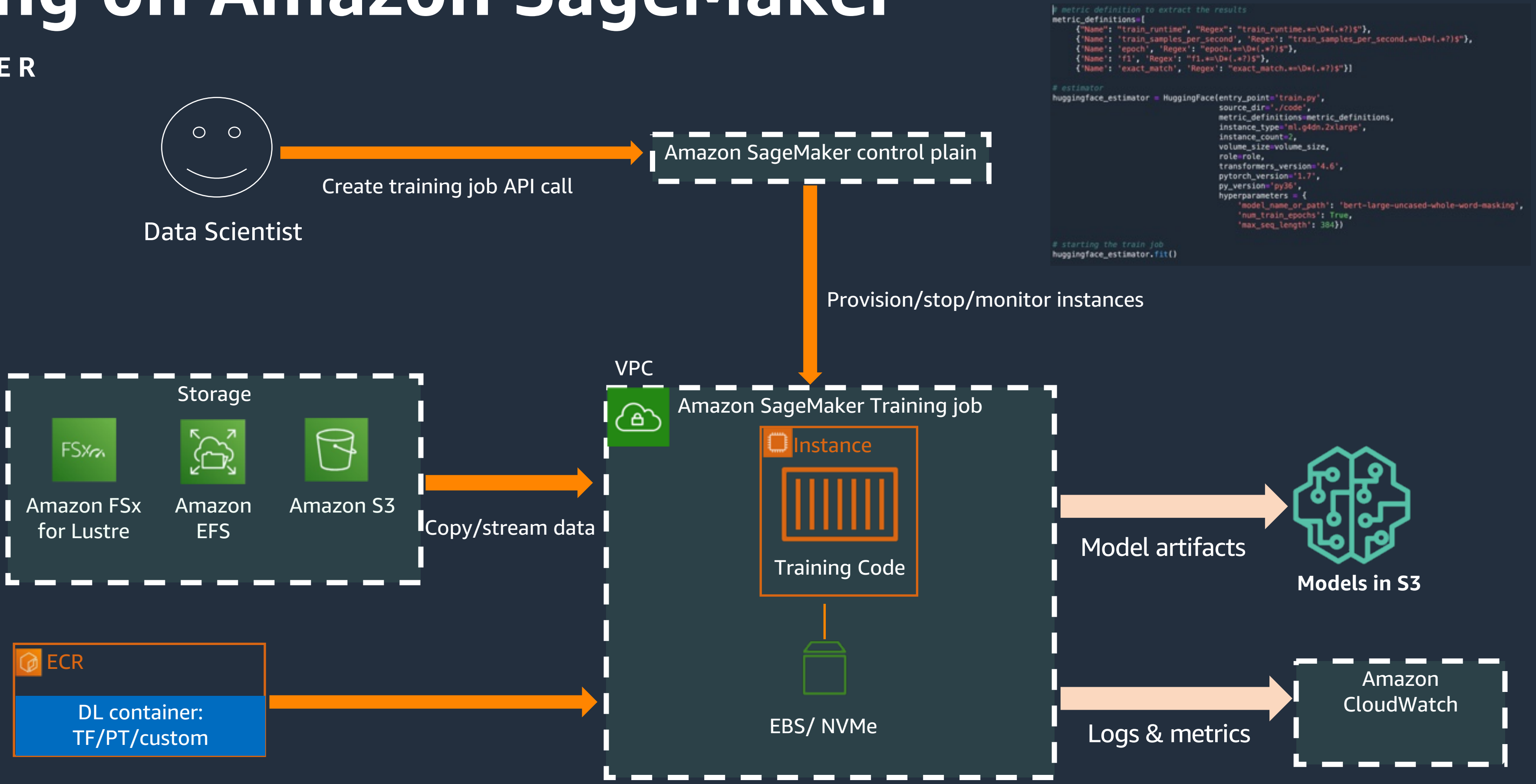

EC2 instance
gpu
gpu
gpu
gpu
Amazon Elastic Fabric Adapter
EC2 instance
gpu
gpu
gpu
gpu
Amazon Elastic Fabric Adapter
EC2 instance
gpu
gpu
gpu
gpu
High Performance Shared File System
Amazon Elastic File System
FSx
Amazon FSx for Lustre
Amazon Simple Storage Service

Fast GPU to GPU communication:
600GB/s in node (Nvlink)
400Gbps networking with EFA

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Training on Amazon SageMaker

A REFRESHER

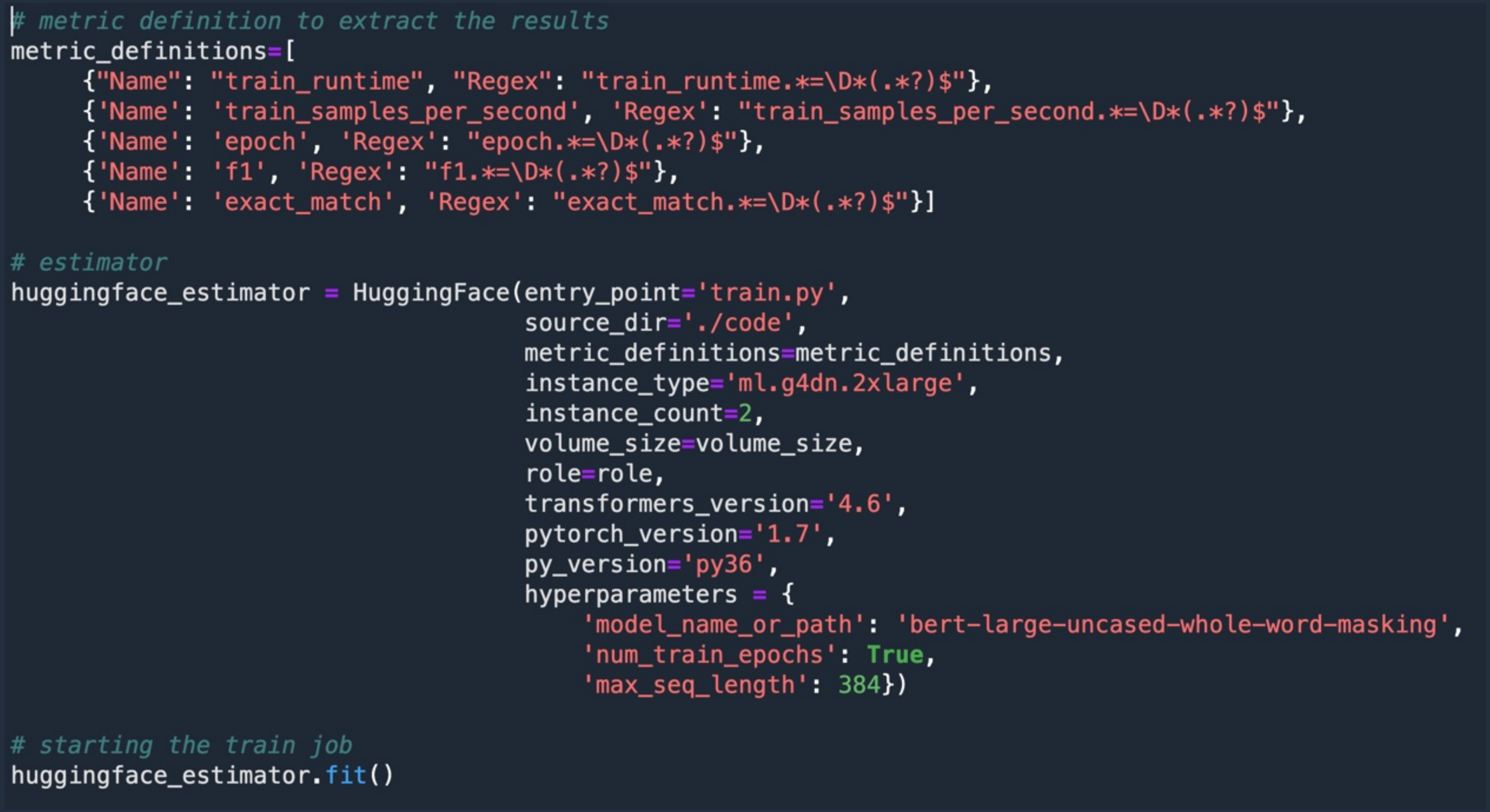


```
# metric definition to extract the results
metric_definitions=[
    {"Name": "train_runtime", "Regex": "train_runtime.*=\D*(.*?)$"},
    {'Name': 'train_samples_per_second', 'Regex': "train_samples_per_second.*=\D*(.*?)$"},
    {'Name': 'epoch', 'Regex': "epoch.*=\D*(.*?)$"},
    {'Name': 'f1', 'Regex': "f1.*=\D*(.*?)$"},
    {'Name': 'exact_match', 'Regex': "exact_match.*=\D*(.*?)$"}]

# estimator
huggingface_estimator = HuggingFace(entry_point='train.py',
                                    source_dir='./code',
                                    metric_definitions=metric_definitions,
                                    instance_type='ml.g4dn.2xlarge',
                                    instance_count=2,
                                    volume_size=volume_size,
                                    role=role,
                                    transformers_version='4.6',
                                    pytorch_version='1.7',
                                    py_version='py36',
                                    hyperparameters = {
                                        'model_name_or_path': 'bert-large-uncased-whole-word-masking',
                                        'num_train_epochs': True,
                                        'max_seq_length': 384})

# starting the train job
huggingface_estimator.fit()
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Training on Amazon SageMaker

Hugging Face estimator



```
# metric definition to extract the results
metric_definitions=[
     {"Name": "train_runtime", "Regex": "train_runtime.*=\D*(.*?)$"},
     {'Name': 'train_samples_per_second', 'Regex': "train_samples_per_second.*=\D*(.*?)$"},
     {'Name': 'epoch', 'Regex': "epoch.*=\D*(.*?)$"},
     {'Name': 'f1', 'Regex': "f1.*=\D*(.*?)$"},
     {'Name': 'exact_match', 'Regex': "exact_match.*=\D*(.*?)$"}]

# estimator
huggingface_estimator = HuggingFace(entry_point='train.py',
                                    source_dir='./code',
                                    metric_definitions=metric_definitions,
                                    instance_type='ml.g4dn.2xlarge',
                                    instance_count=2,
                                    volume_size=volume_size,
                                    role=role,
                                    transformers_version='4.6',
                                    pytorch_version='1.7',
                                    py_version='py36',
                                    hyperparameters = {
                                        'model_name_or_path': 'bert-large-uncased-whole-word-masking',
                                        'num_train_epochs': True,
                                        'max_seq_length': 384})

# starting the train job
huggingface_estimator.fit()
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Training a LLM with HuggingFace

1. Preprocessing Step
    - Download/ingestion of dataset
    - Tokenization
    - Additional preprocessing steps
2. Training Step
    - Download/loading of model
    - Configuration of TrainingArguments
    - Configuration of Trainer
    - Training, evaluation, model serialization and storage

```
raw_datasets = load_dataset(…)
tokenizer = AutoTokenizer.from_pretrained(…)
tokenized_datasets = raw_datasets.map(…)
lm_datasets = tokenized_datasets.map(…)
```

```
model =AutoModelForCausalLM.from_pretrained(‘model-id’)
training_args = TrainingArguments(**kwargs)
trainer = Trainer(
                model=model,
                args=training_args,
                train_dataset=lm_datasets,  …)
trainer.train()
trainer.save_model()
trainer.evaluate()
```

Amazon SageMaker

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Large-scale training on SageMaker

OPTIMIZED DISTRIBUTED TRAINING LIBRARIES & FRAMEWORKS

- TensorFlow
- PyTorch
- Hugging Face
- SageMaker Distributed Training Libraries
- Bring your own library (e.g. DeepSpeed, Megatron)

AMAZON SAGEMAKER TRAINING

Large Scale Cluster Orchestration	NCCL Health Checks	SageMaker Jumpstart for foundational models	SageMaker Compiler	Warm pools	SSH to container
Data loading	Debugger	Profiling	Experiment tracking	Hyperparameter optimization	Pay for what you use

ML COMPUTE INSTANCES & ACCELERATORS

- NVIDIA GPUS A100, V100, K80, T4, A10
- AWS Nitro
- 400/800 Gbps EFA Networking
- CPU instances
- AWS Trainium

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

TensorFlow, the TensorFlow logo and any related marks are trademarks of Google Inc.
PyTorch, the PyTorch logo and any related marks are trademarks of Facebook, Inc.

Lab 2 – LLM fine-tuning

https://github.com/aristsakpinis93/generative-ai-immersion-day

Event Access Code:

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

AGENDA

Generative AI – What is it and why the hype?

Large Language Models - How the ML works?

Large Language Model Hosting

Large Language Model Finetuning

Visual Foundation Models & Stable Diffusion

Engineering GenAI-powered Applications on AWS

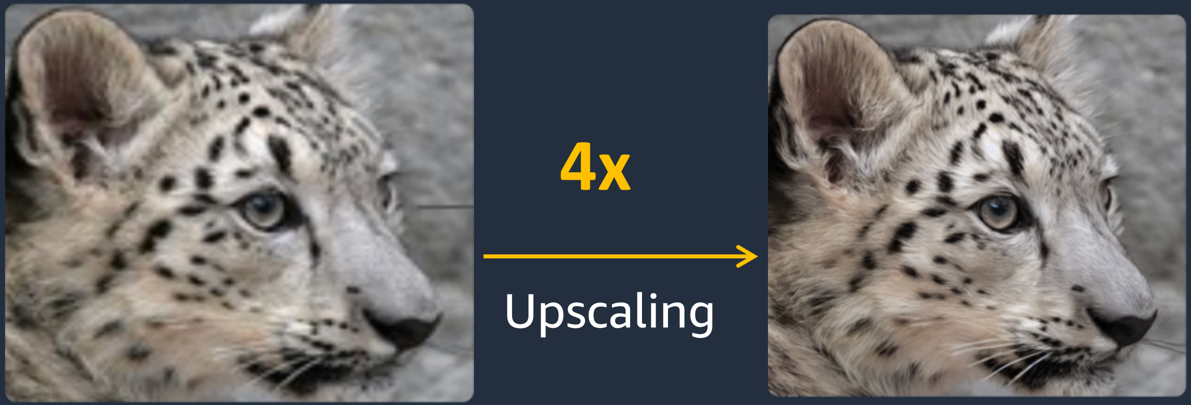
© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Generative AI is transforming AI

IMAGE GENERATION, TRANSFORMATION, UPSCALING

Generated by Stable Diffusion 2.0. This interior does not exist

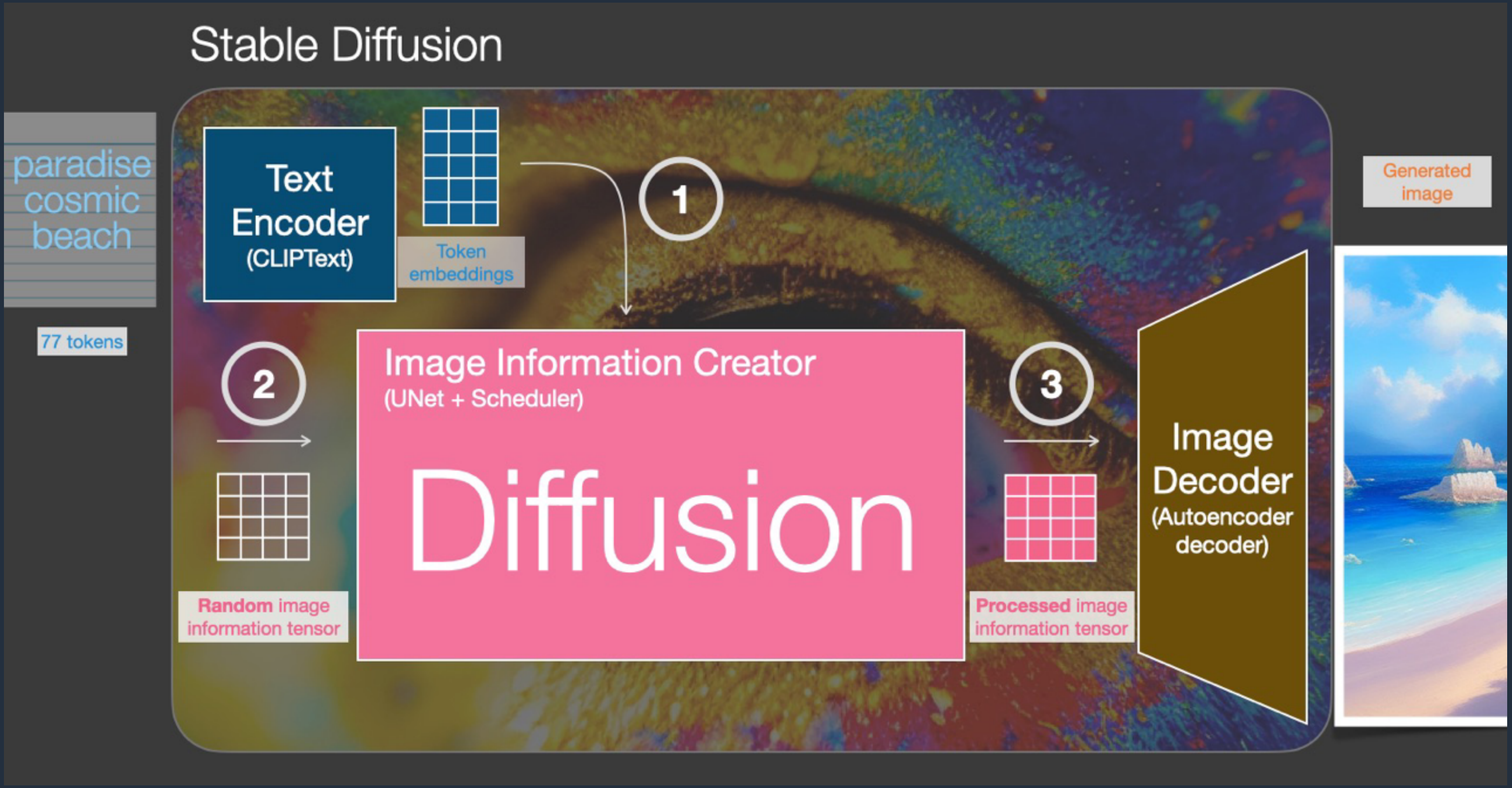
Seamless transformation



Upscaling

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Building blocks of Diffusion


Stable Diffusion
paradise cosmic beach
77 tokens
Text Encoder (CLIPText)
Token embeddings
1
2
Random image information tensor
Image Information Creator (UNet + Scheduler)
Diffusion
3
Processed image information tensor
Image Decoder (Autoencoder decoder)
Generated image

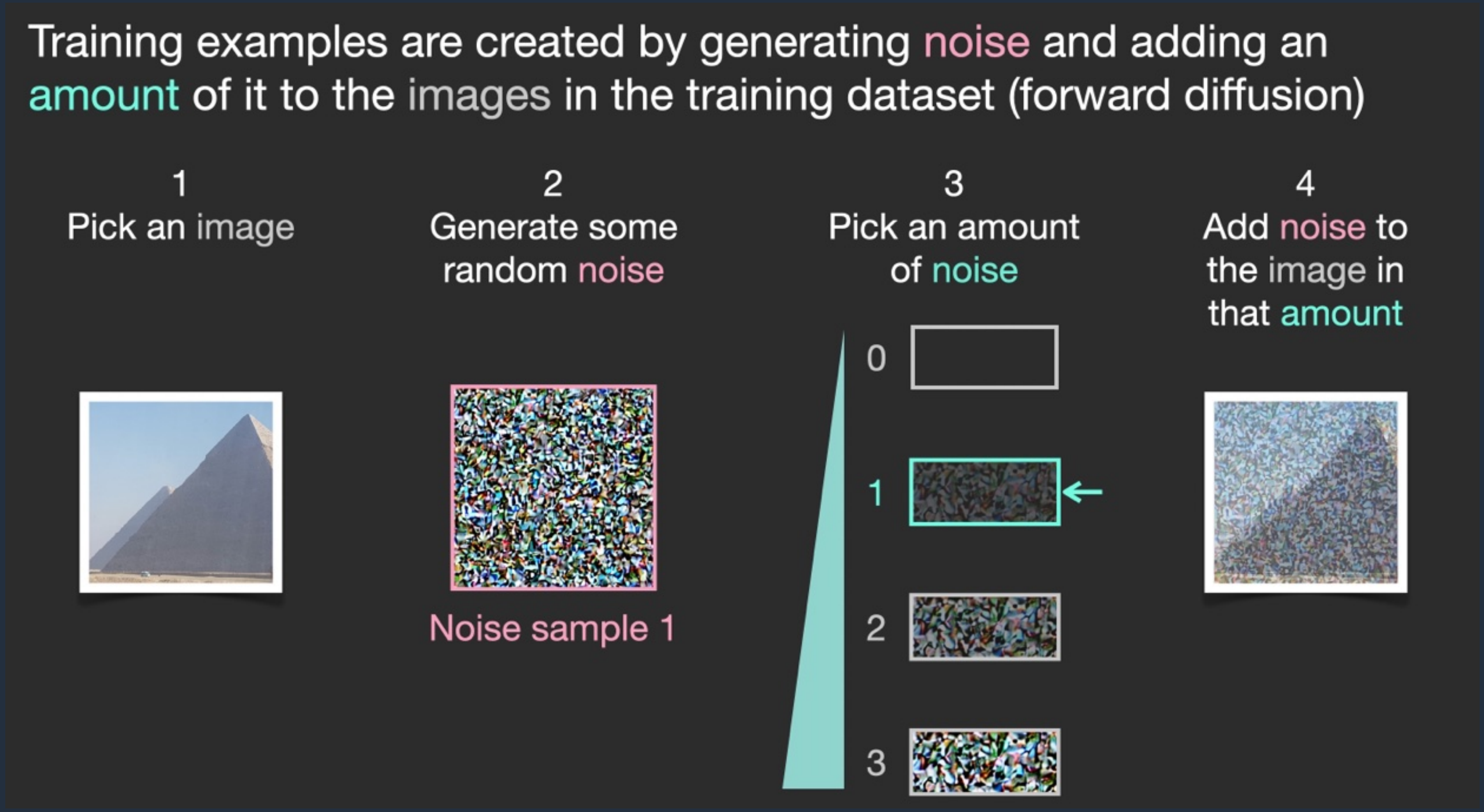
Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Training a diffusion model

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Building blocks of Diffusion



Training examples are created by generating noise and adding an amount of it to the images in the training dataset (forward diffusion)

1
Pick an image

2
Generate some random noise

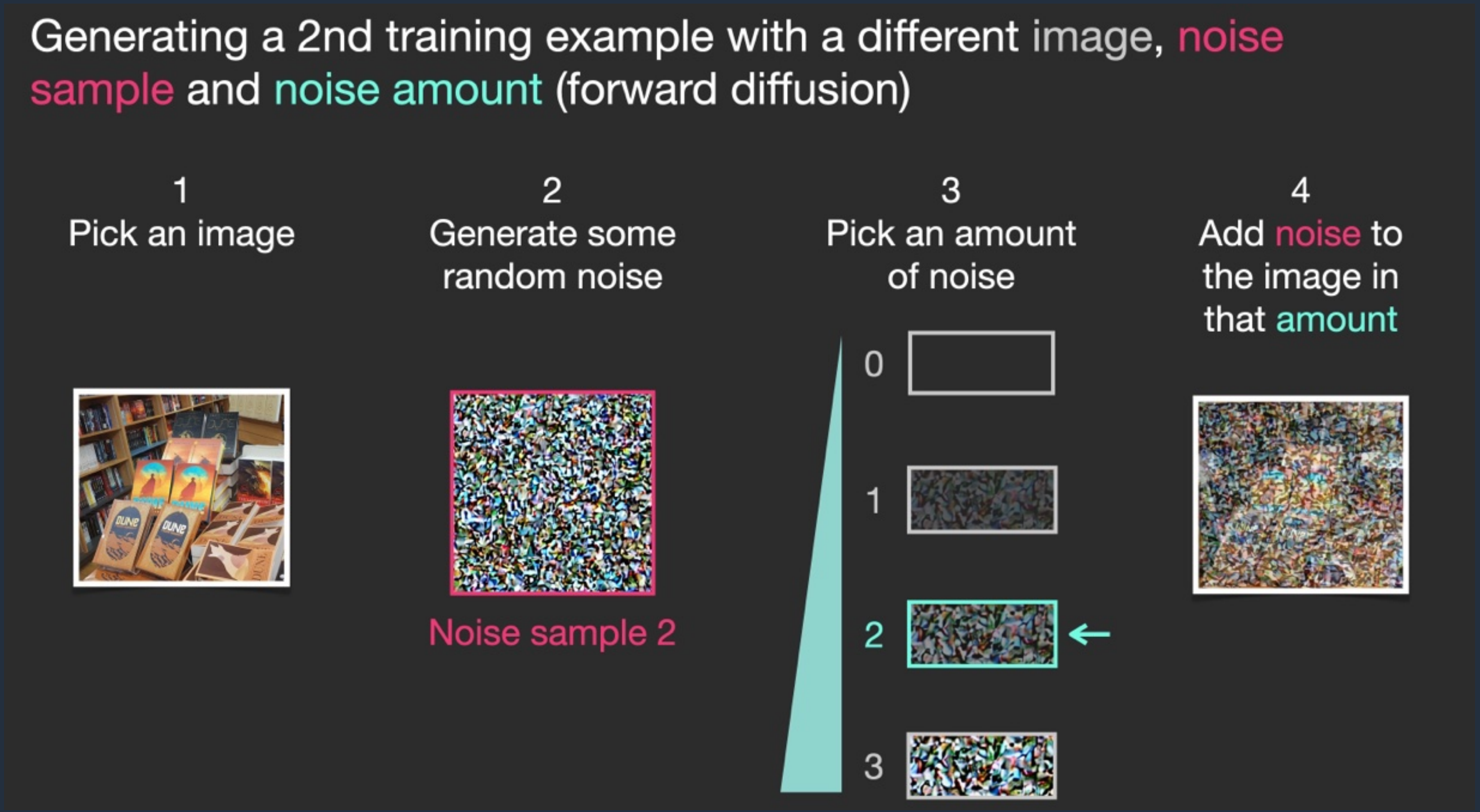
3
Pick an amount of noise

4
Add noise to the image in that amount

Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Building blocks of Diffusion



Generating a 2nd training example with a different image, noise sample and noise amount (forward diffusion)

1
Pick an image

2
Generate some random noise

Noise sample 2

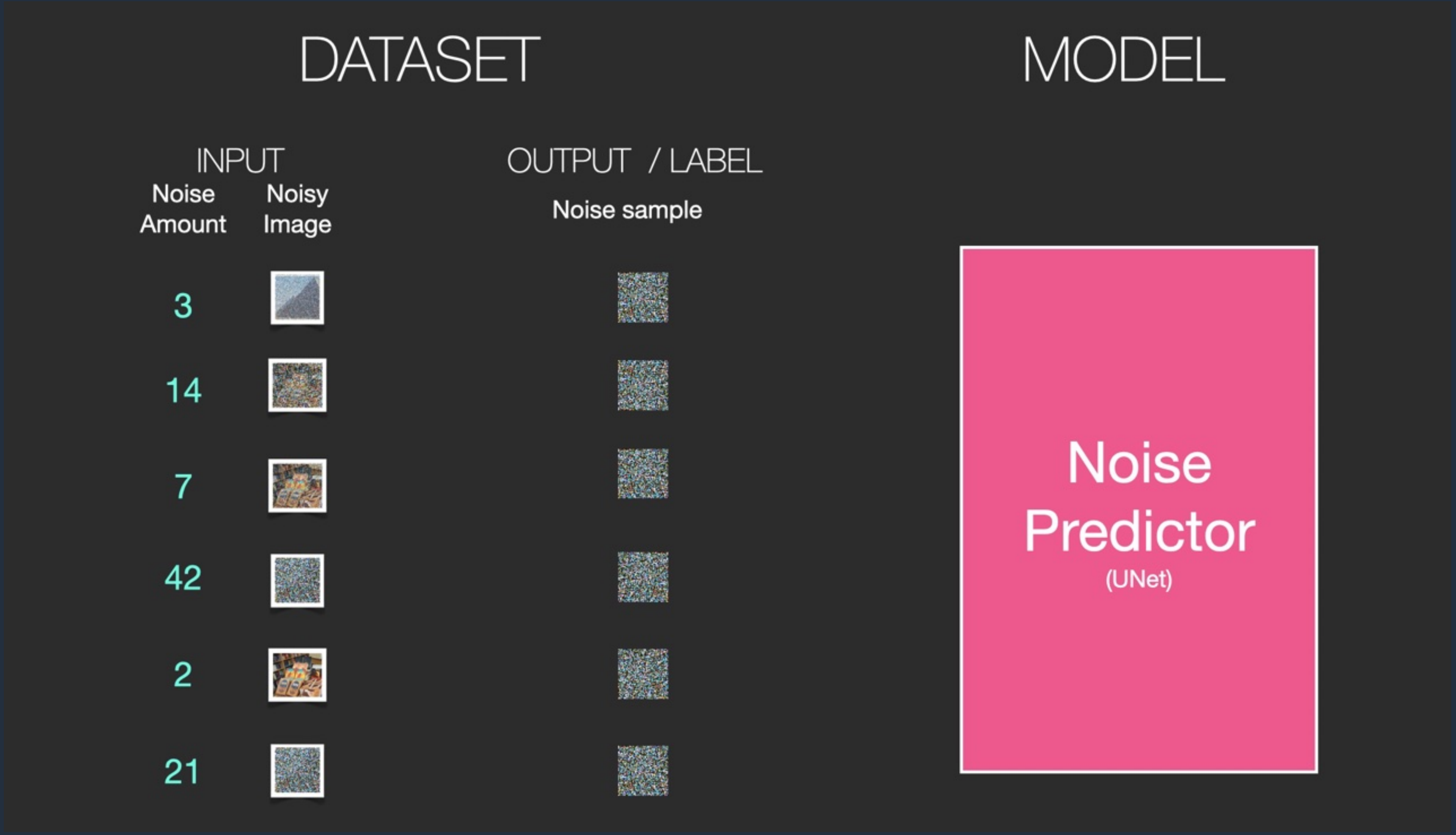
3
Pick an amount of noise

4
Add noise to the image in that amount

Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

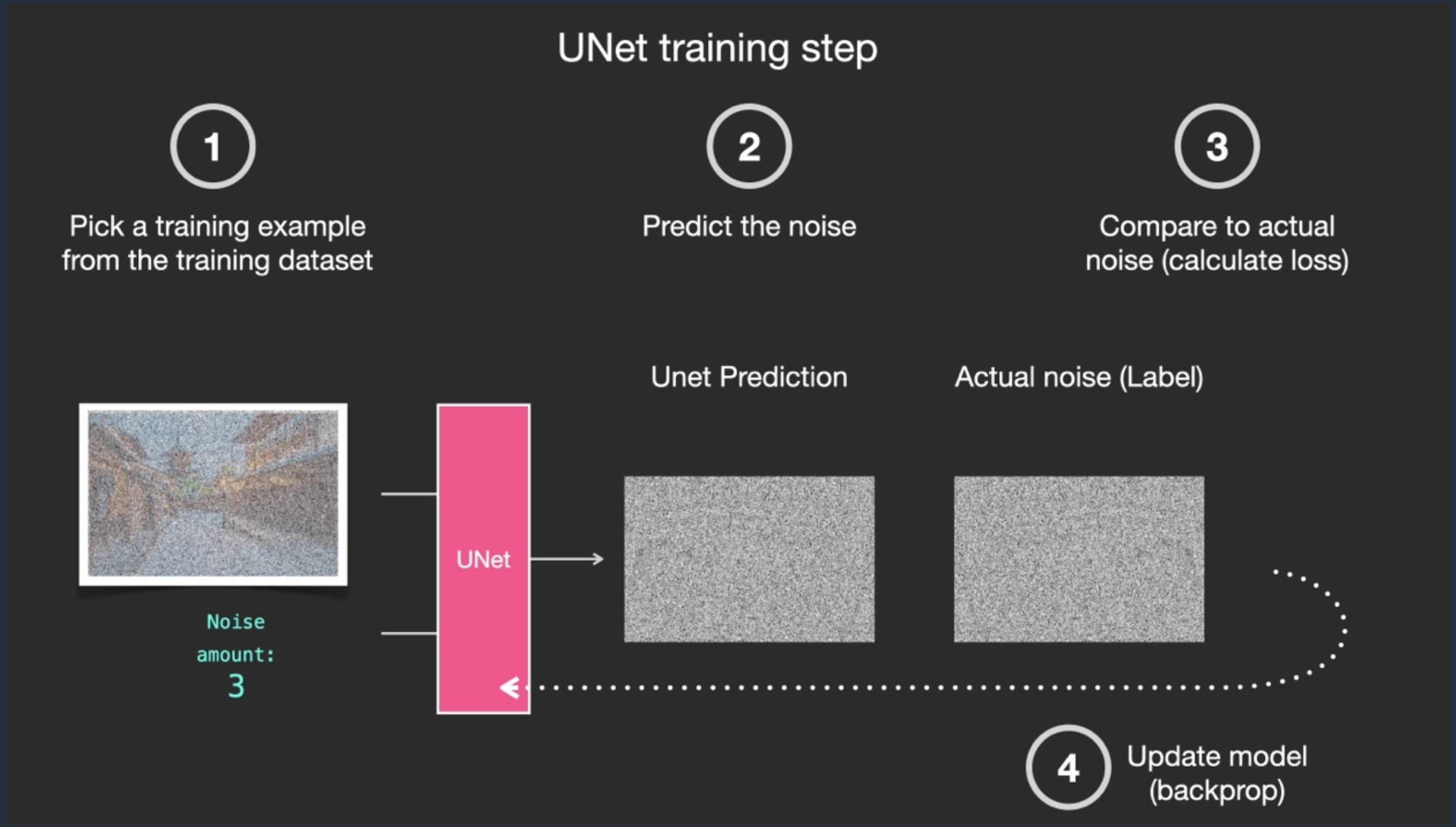
Building blocks of Diffusion



Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Building blocks of Diffusion



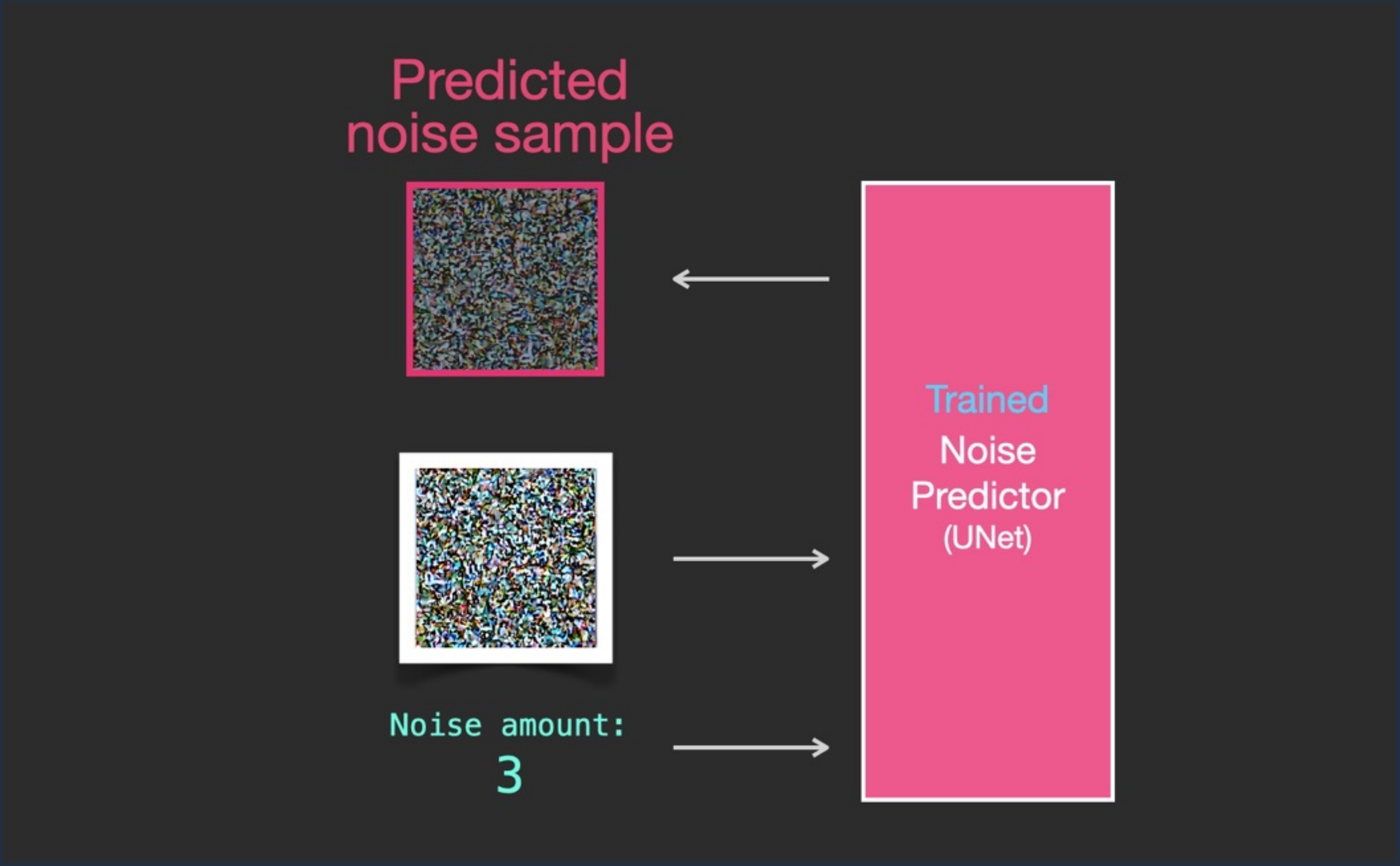
Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Image generation

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

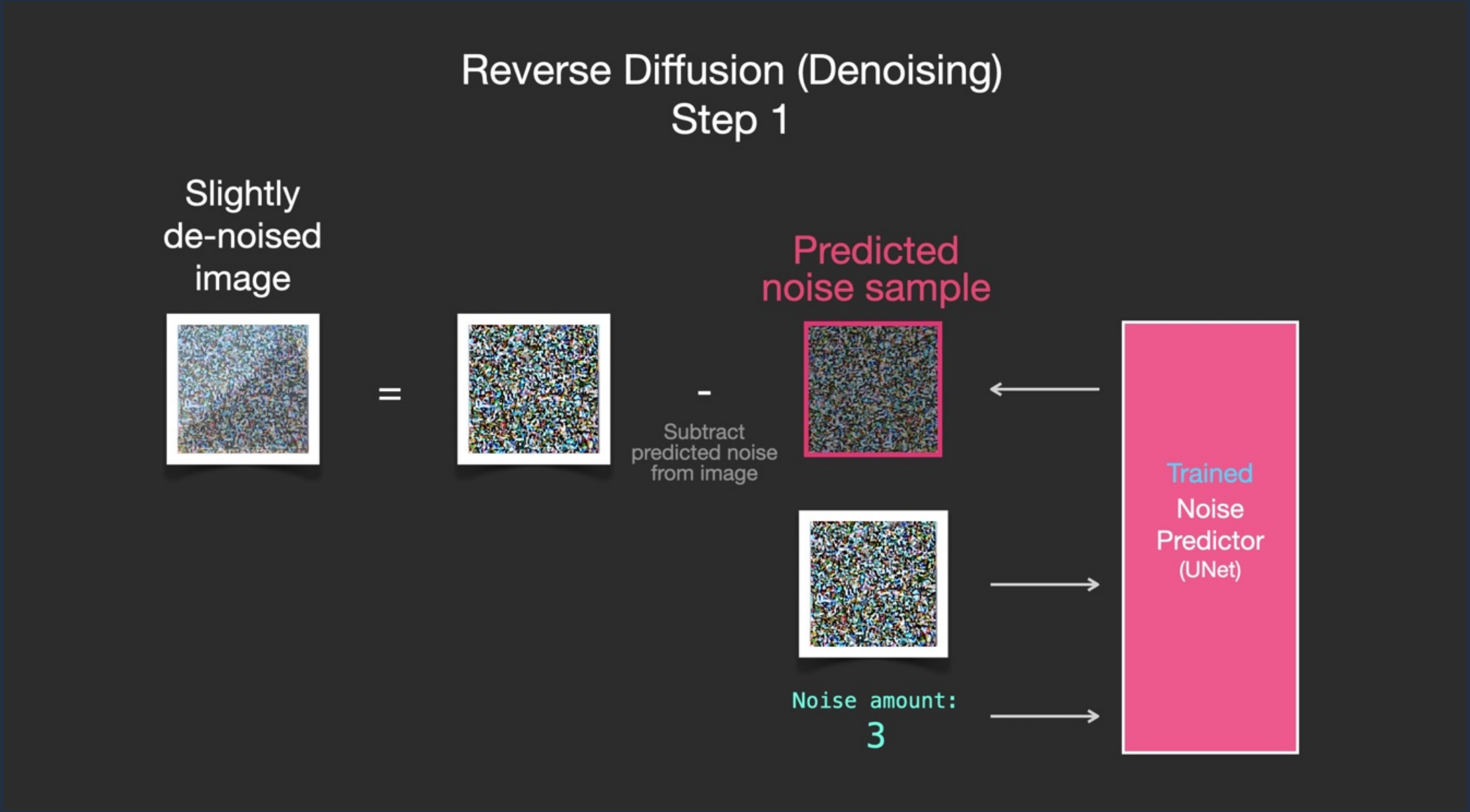
Building blocks of Diffusion



Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

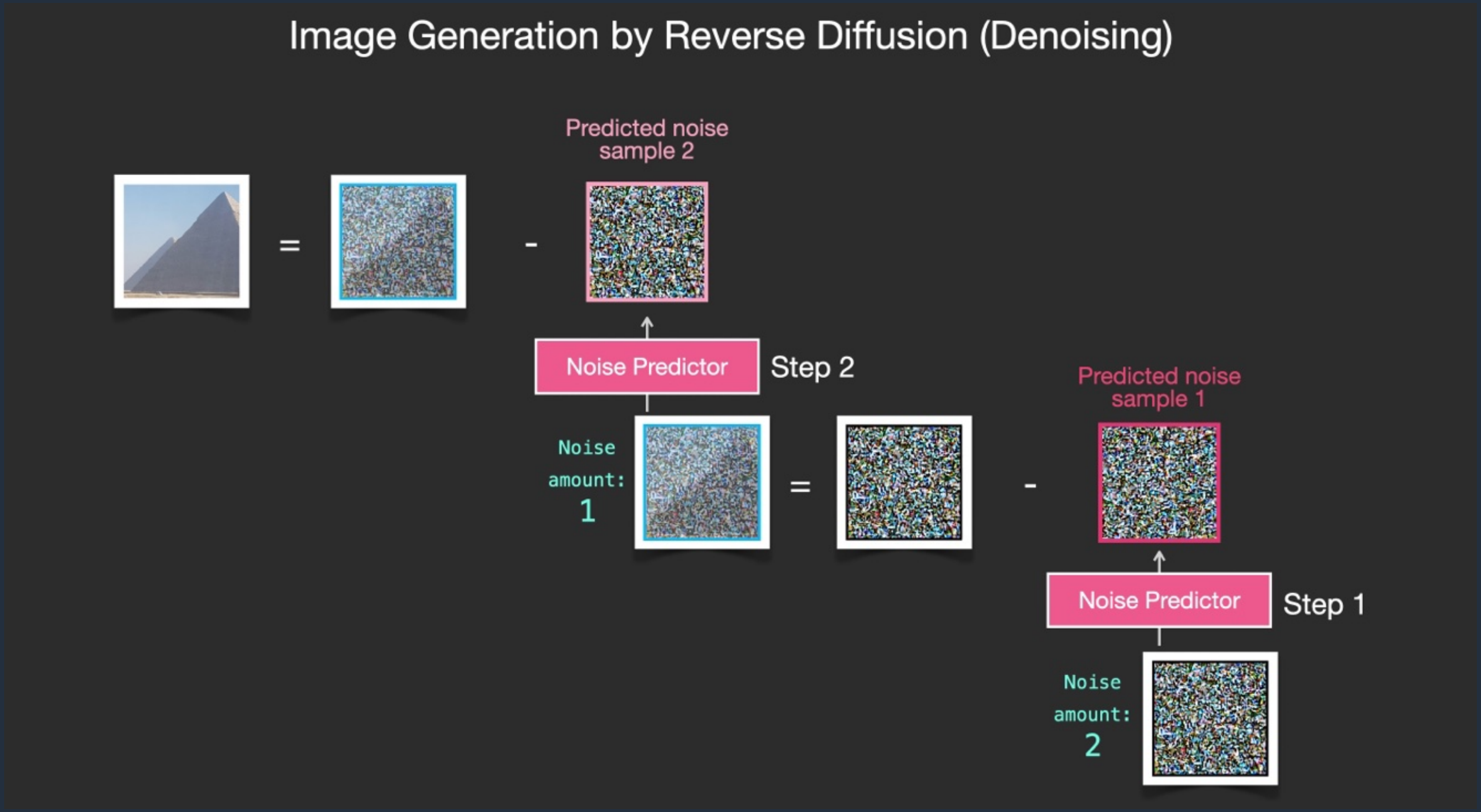
Building blocks of Diffusion


Reverse Diffusion (Denoising)
Step 1
Slightly de-noised image
=
-
Subtract predicted noise from image
Predicted noise sample
Trained Noise Predictor (UNet)
Noise amount:
3

Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

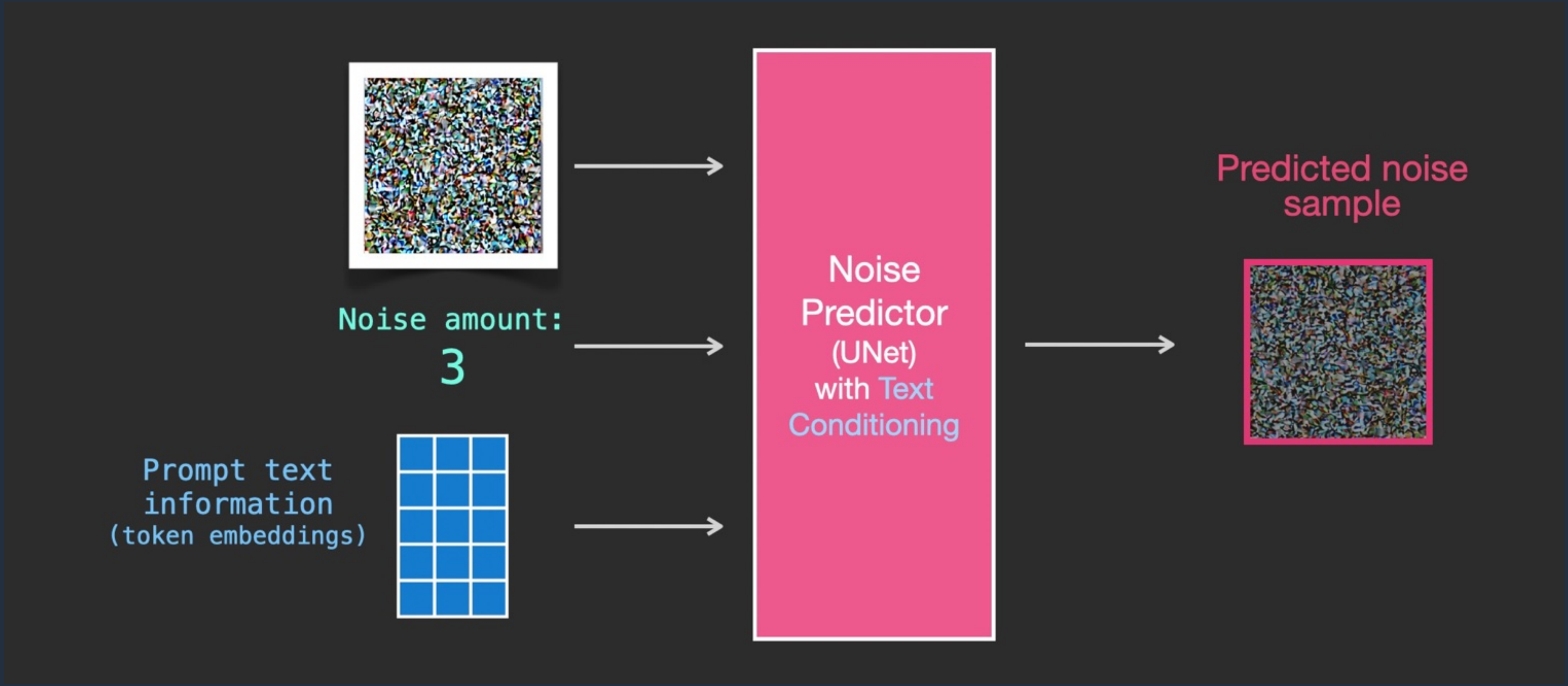
Building blocks of Diffusion



Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

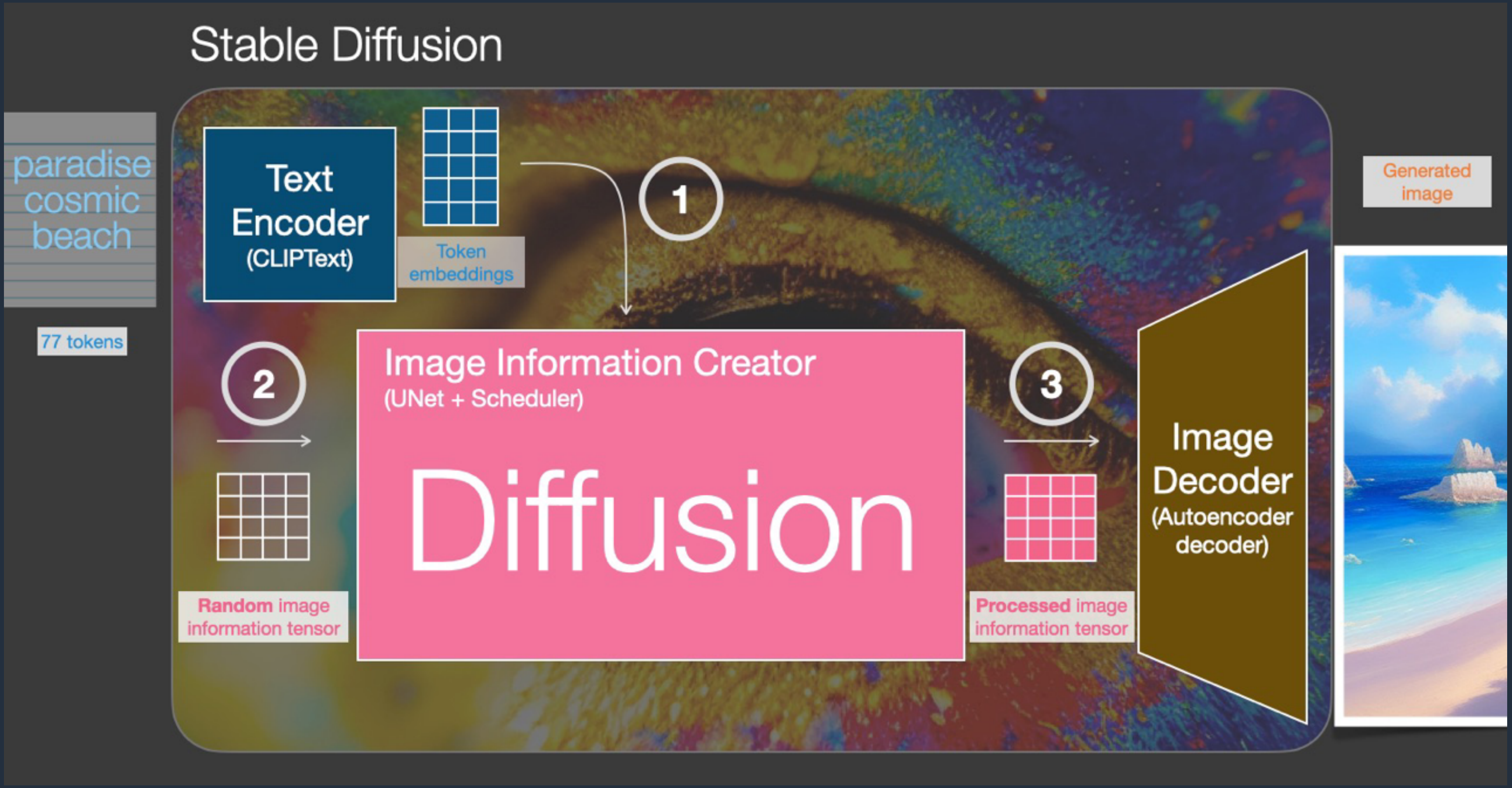
Building blocks of Diffusion



Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Building blocks of Diffusion



Source: https://jalammar.github.io/illustrated-stable-diffusion/

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Lab 3 – Stable Diffusion Deployment & Inference

https://github.com/aristsakpinis93/generative-ai-immersion-day

Event Access Code:

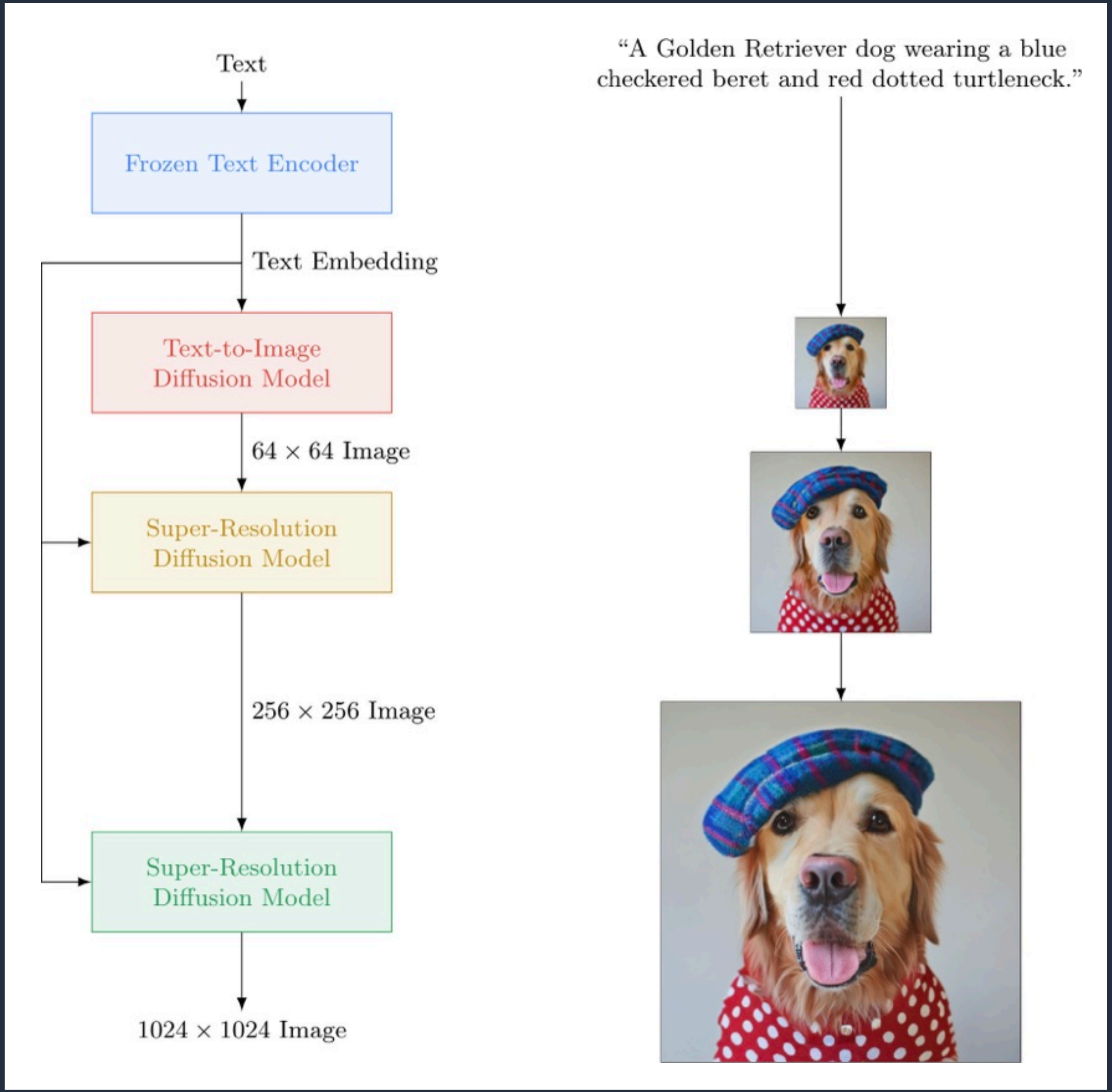
© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Other models and related work

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

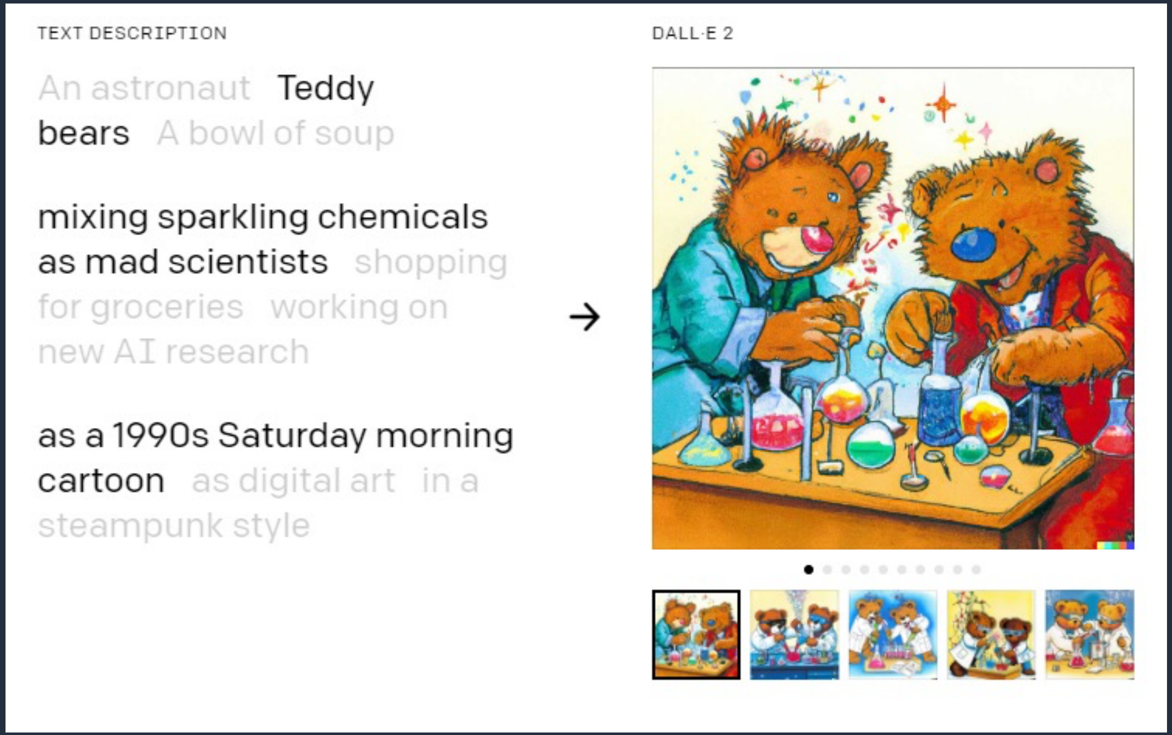
Imagen (Google, 2022)

- Another diffusion model
- Generate at low resolution and use super resolution networks to upscale; doesn't use an autoencoder
- Not available to the public.


Text
Frozen Text Encoder
Text Embedding
Text-to-Image Diffusion Model
64 × 64 Image
Super-Resolution Diffusion Model
256 × 256 Image
Super-Resolution Diffusion Model
1024 × 1024 Image
"A Golden Retriever dog wearing a blue checkered beret and red dotted turtleneck."

Dall-E 2 (OpenAI, 2022)

- Another diffusion model
  - Weights are private but there’s an inference API



TEXT DESCRIPTION

An astronaut Teddy bears A bowl of soup

mixing sparkling chemicals as mad scientists shopping for groceries working on new AI research

as a 1990s Saturday morning cartoon as digital art in a steampunk style

→

DALL·E 2

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Midjourney

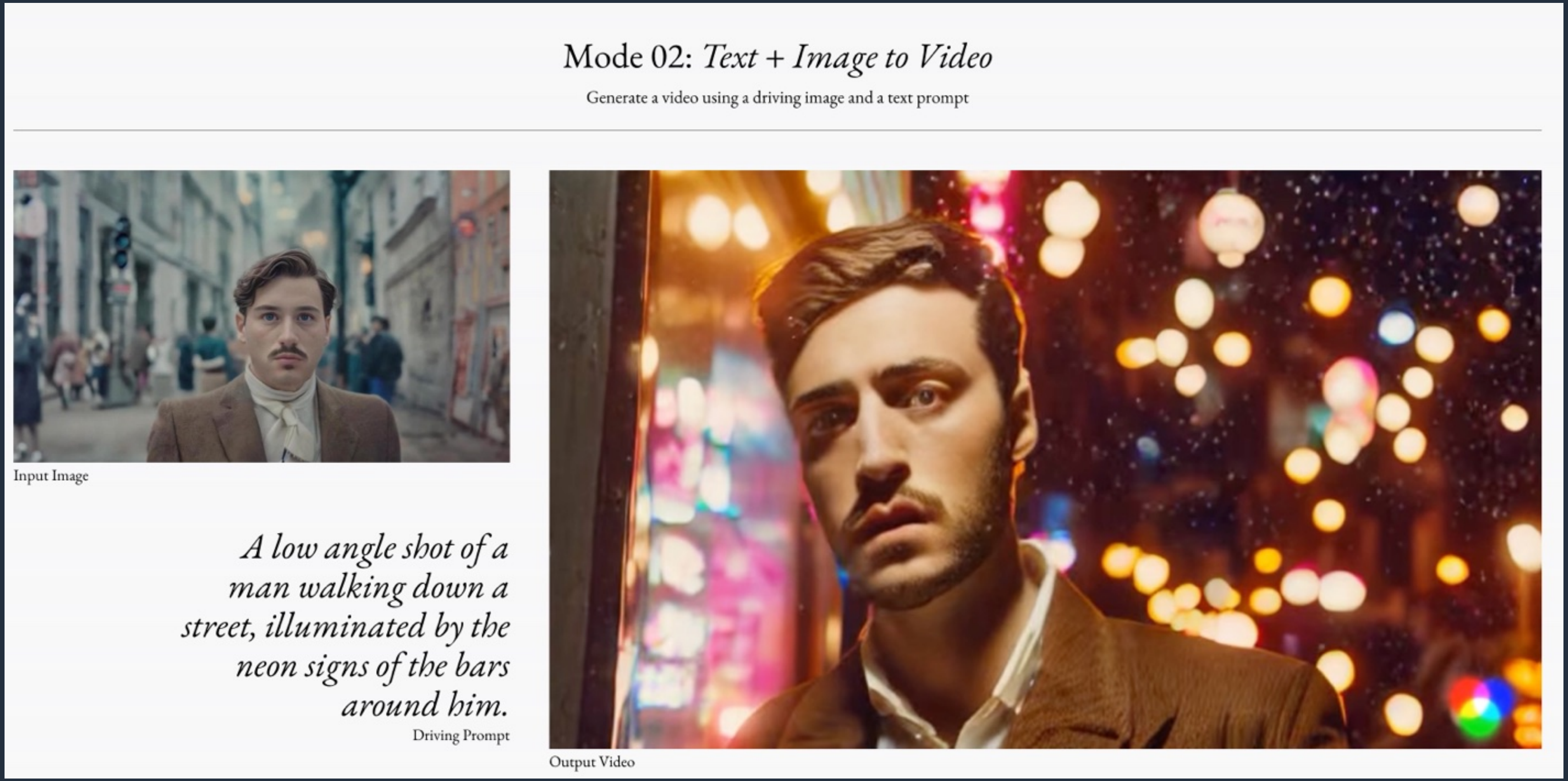
Heavily stylized diffusion model

- Weights are private but there's public access to inference (no API)

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Runway Gen-2

Text-to-Video



Mode 02: *Text + Image to Video*

Generate a video using a driving image and a text prompt

Input Image

A low angle shot of a man walking down a street, illuminated by the neon signs of the bars around him.

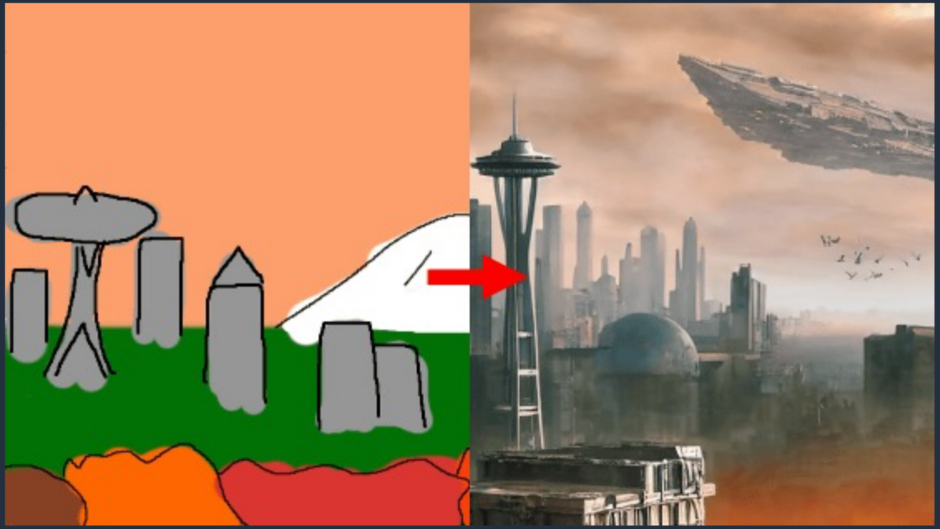
Driving Prompt

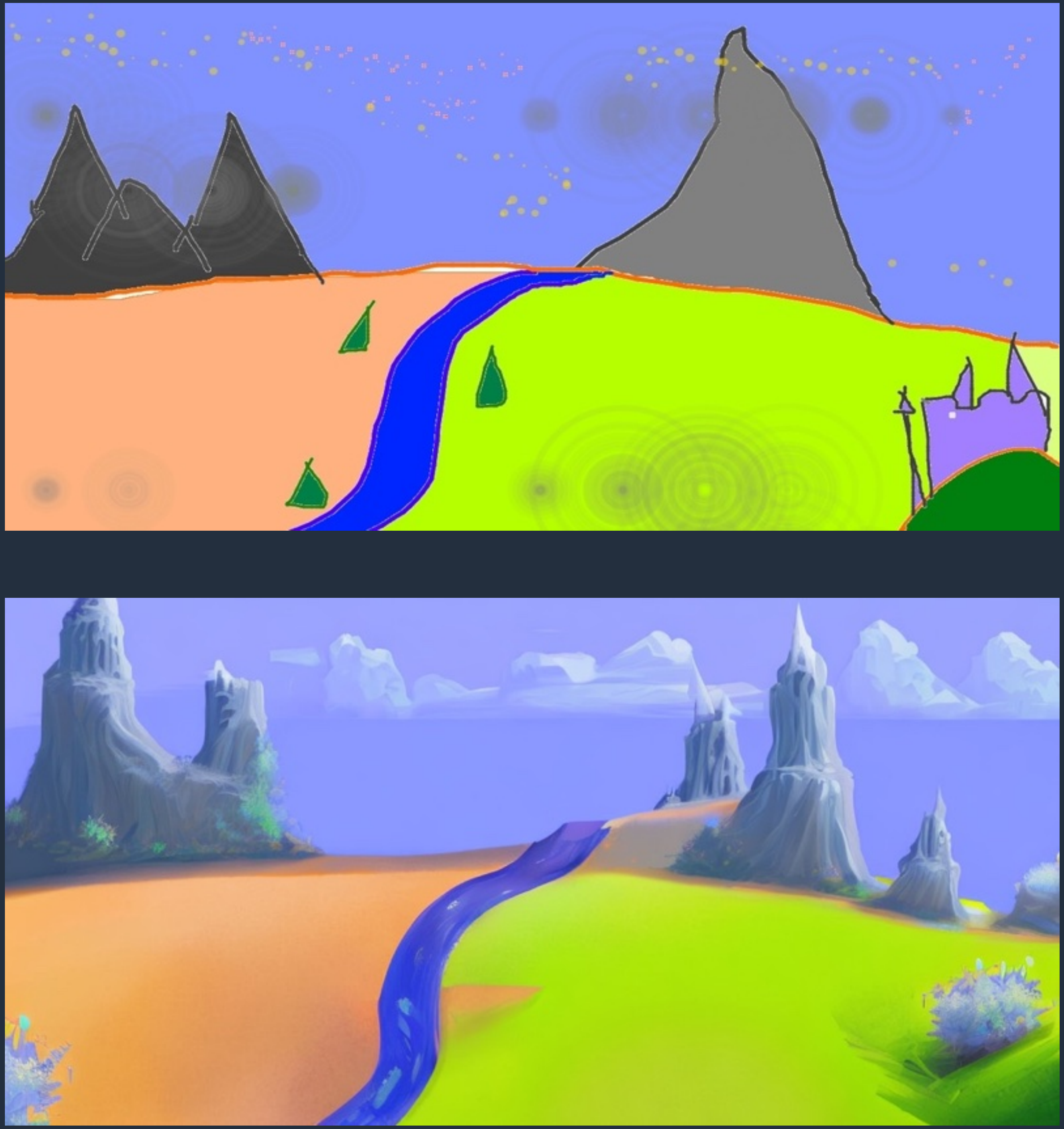
Output Video

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Image-to-image (img2img)

- Add a specified amount of noise to an existing image and start the denoising process

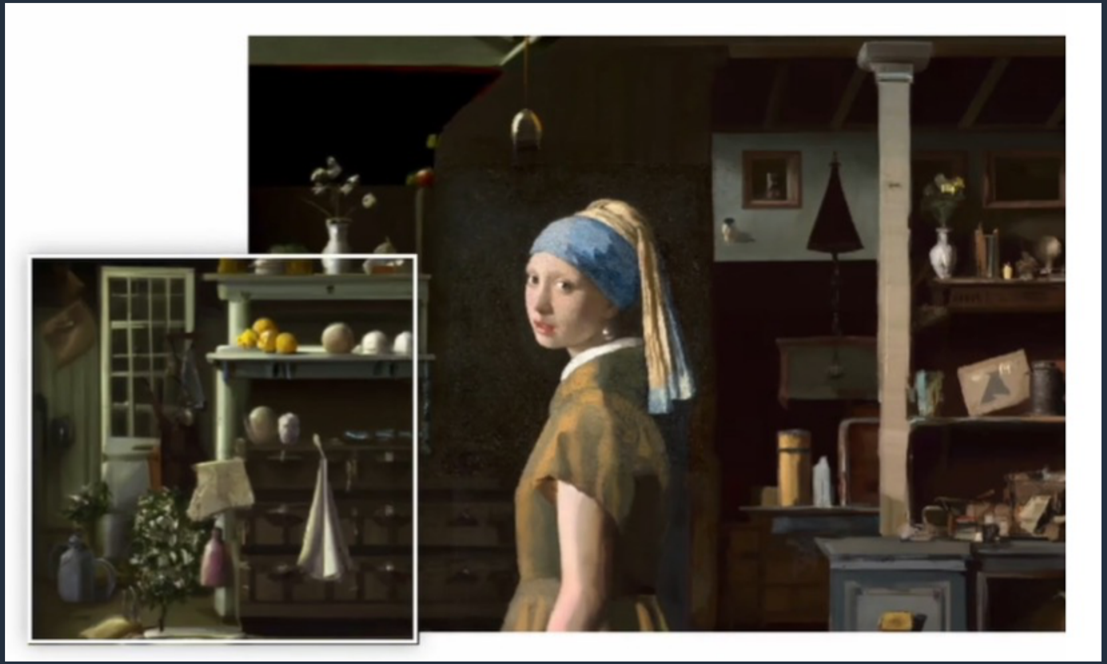


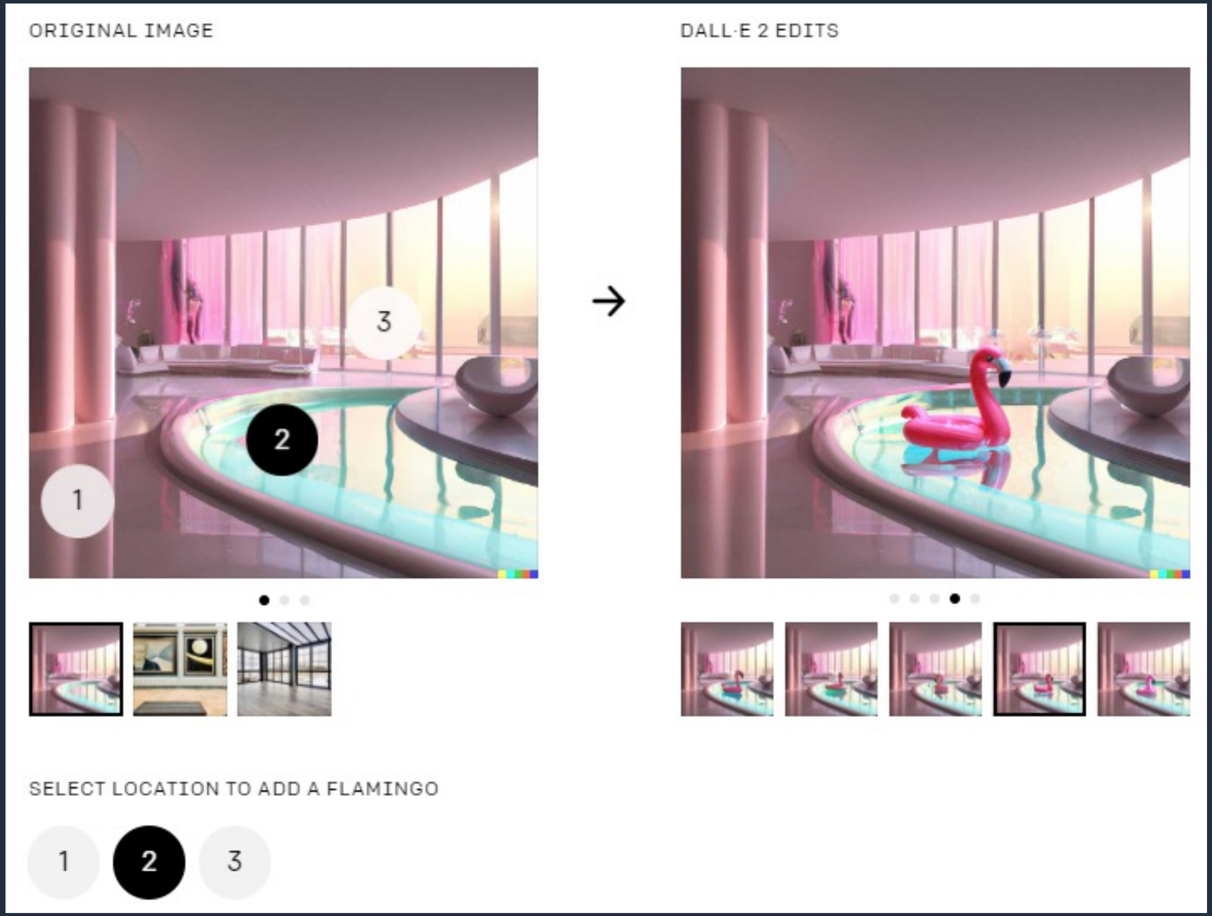


© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Inpainting & outpainting

- Masked image-to-image





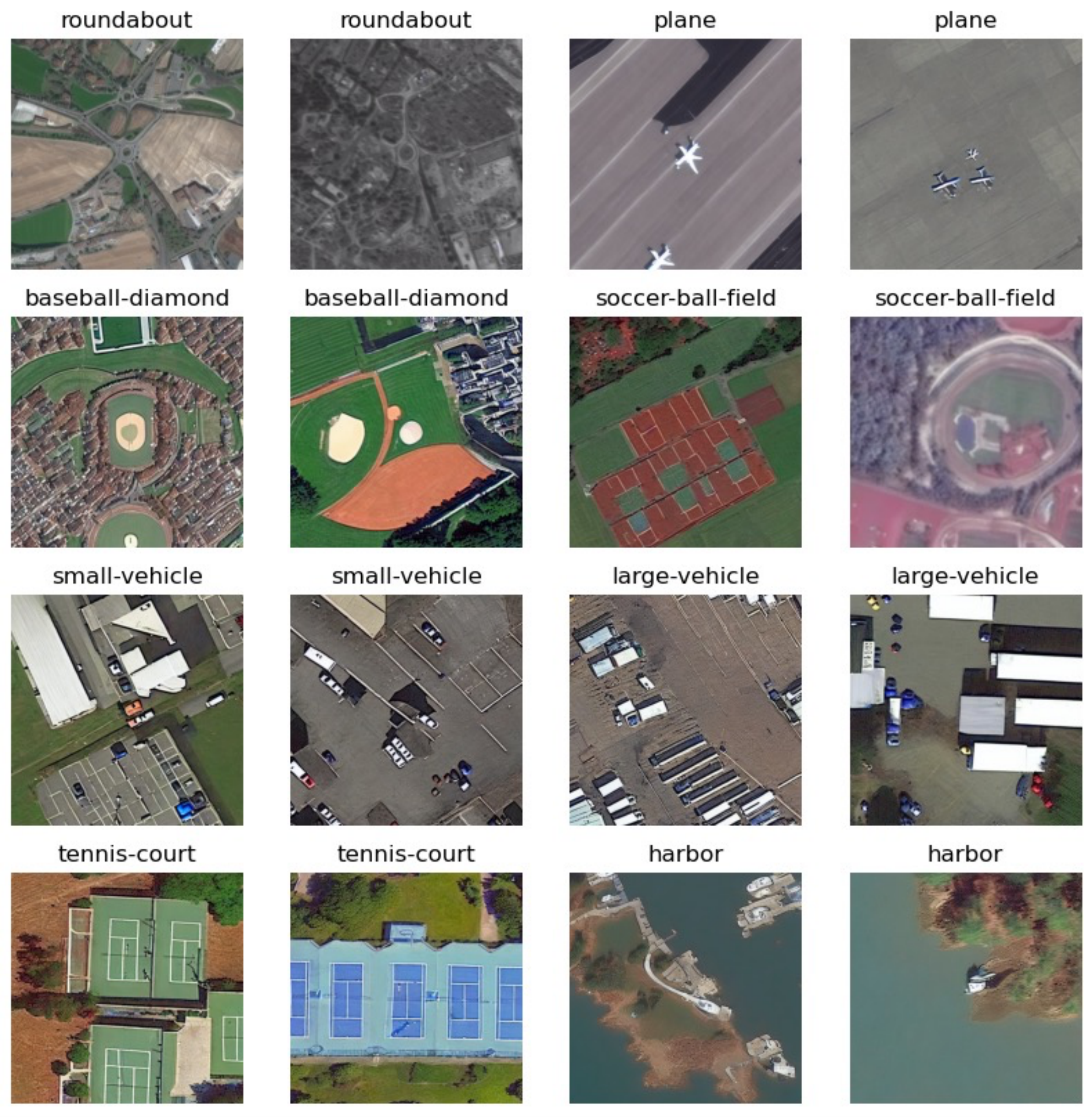
https://openai.com/dall-e-2/

Tuning Stable Diffusion

- Learning a new domain
  - Full scale finetuning
    - Significantly larger data requirement
    - Satellite diffusion model (right) was trained with 2000 image + label pairs
  - LoRA
  - Hypernetworks

More efficient

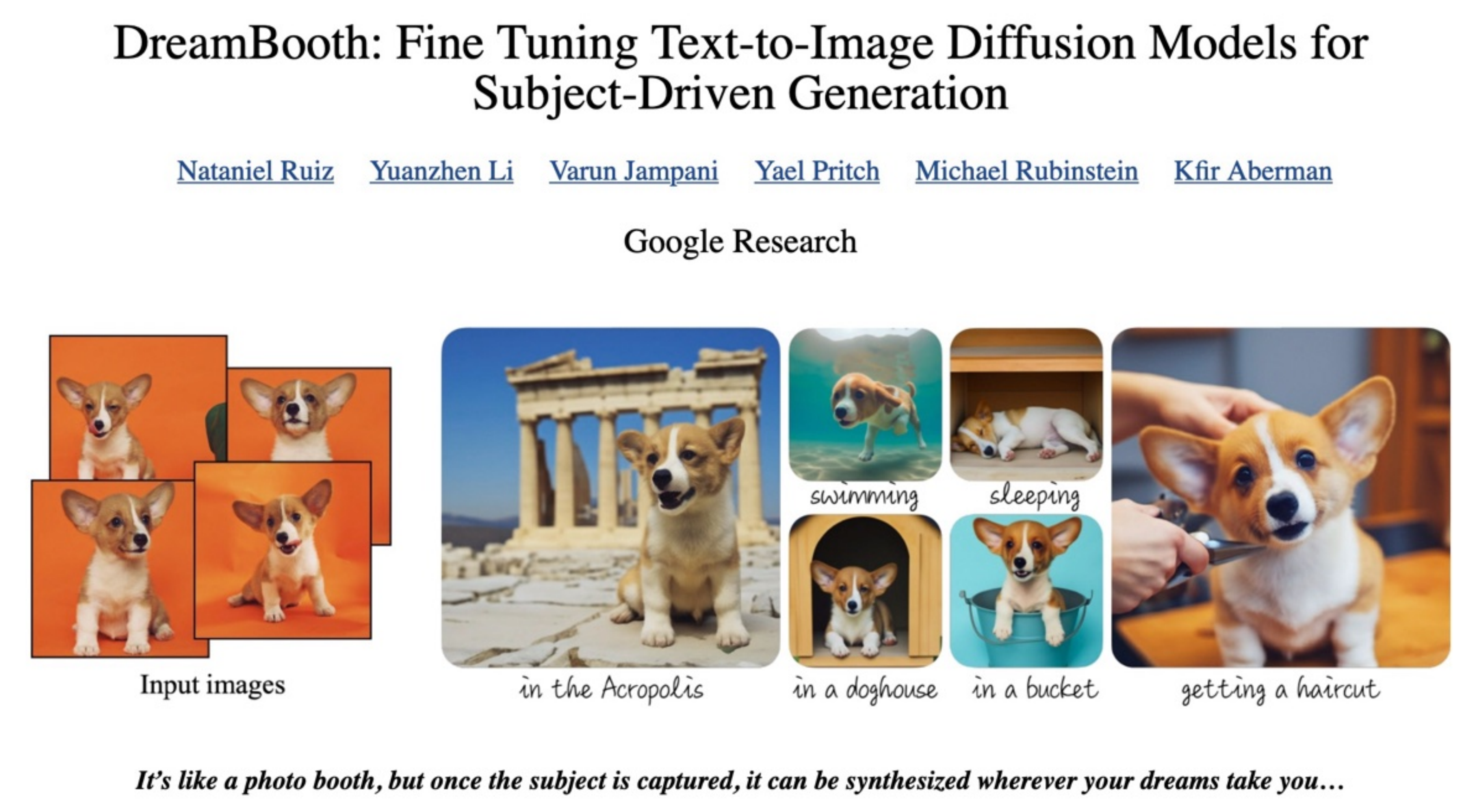
https://gitlab.aws.dev/wwps-natsec/ai-ml-practice/satellite-diffusion



"A satellite image showing a {class_name}"

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Tuning Stable Diffusion



DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation

Nataniel Ruiz Yuanzhen Li Varun Jampani Yael Pritch Michael Rubinstein Kfir Aberman

Google Research

It's like a photo booth, but once the subject is captured, it can be synthesized wherever your dreams take you…

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

AGENDA

Generative AI – What is it and why the hype?

Large Language Models - How the ML works?

Large Language Model Hosting

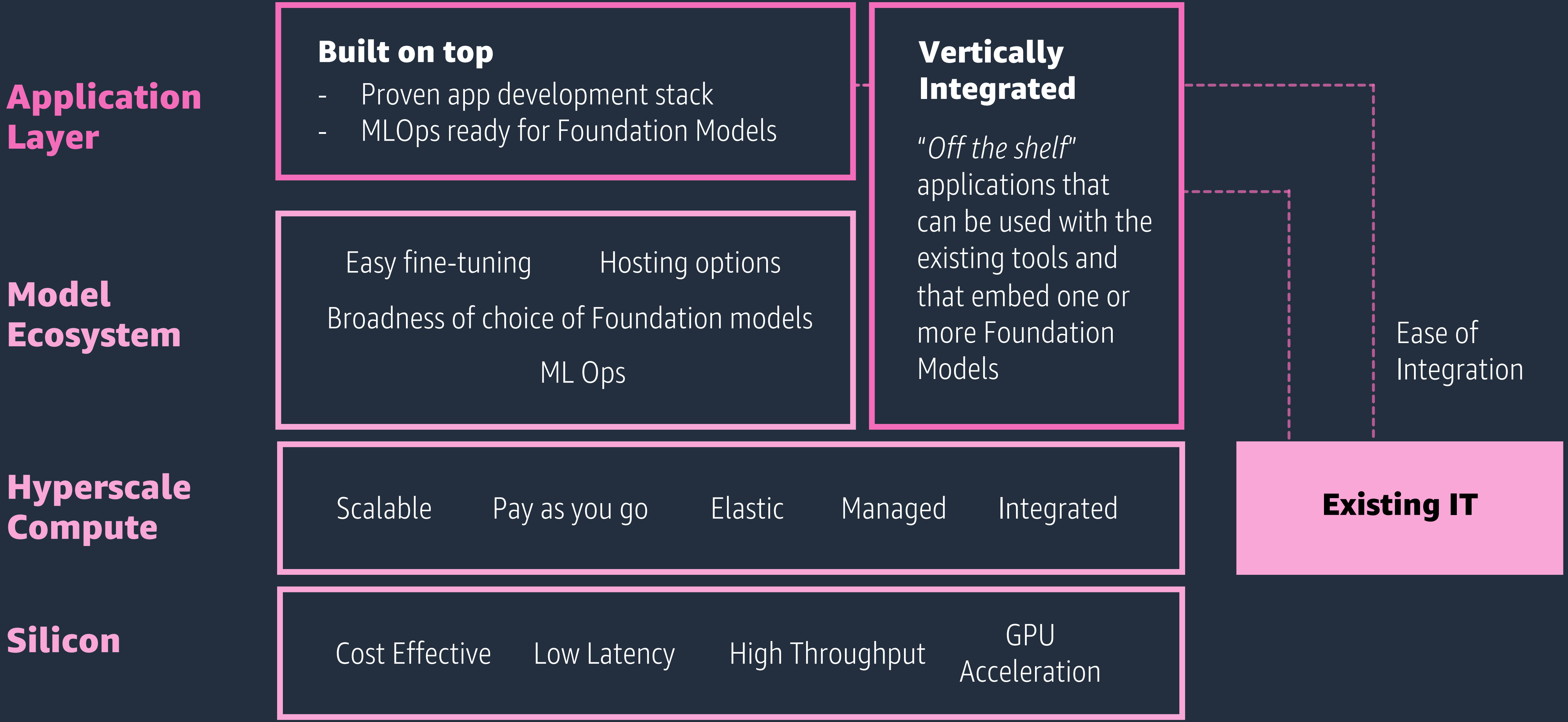
Large Language Model Finetuning

Visual Foundation Models & Stable Diffusion

Engineering GenAI-powered Applications on AWS

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Successfully building GenAI Applications



Application Layer

Built on top
- Proven app development stack
- MLOps ready for Foundation Models

Vertically Integrated

"Off the shelf" applications that can be used with the other existing tools and that embed one or more Foundation Models

Model Ecosystem

Easy fine-tuning Hosting options

Broadness of choice of Foundation models

ML Ops

Hyperscale Compute

Scalable Pay as you go Elastic Managed Integrated

Silicon

Cost Effective Low Latency High Throughput GPU Acceleration

Ease of Integration

Existing IT

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Many roads to FMs – model choice is key!

Proprietary models

Publicly available models

Building your own model from scratch

Different use cases and applications will require different (families) of FMs!

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

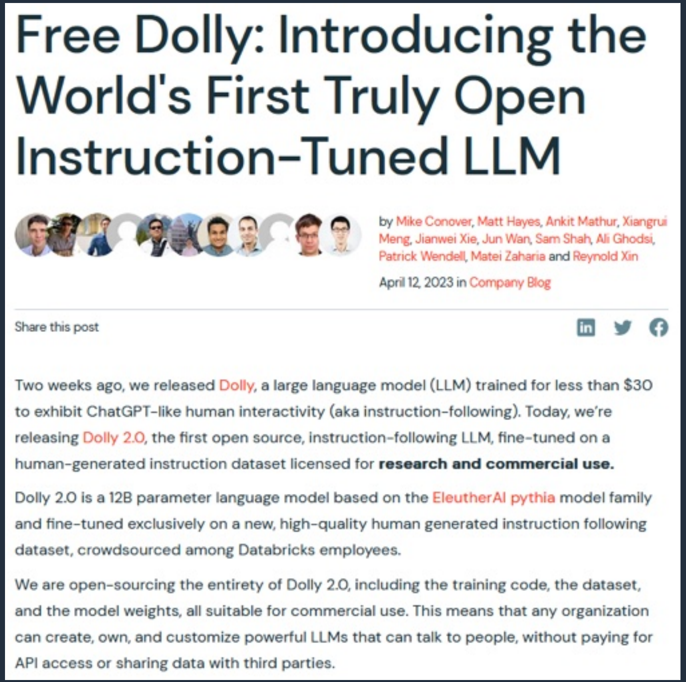
GENERATIVE AI IMMERSION DAY

The rise of open source

Foundation Model

Major companies and advanced startups are open-sourcing state of the art foundation models!

x n



Free Dolly: Introducing the World's First Truly Open Instruction-Tuned LLM

by Mike Conover, Matt Hayes, Ankit Mathur, Xiangrui Meng, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia and Reynold Xin

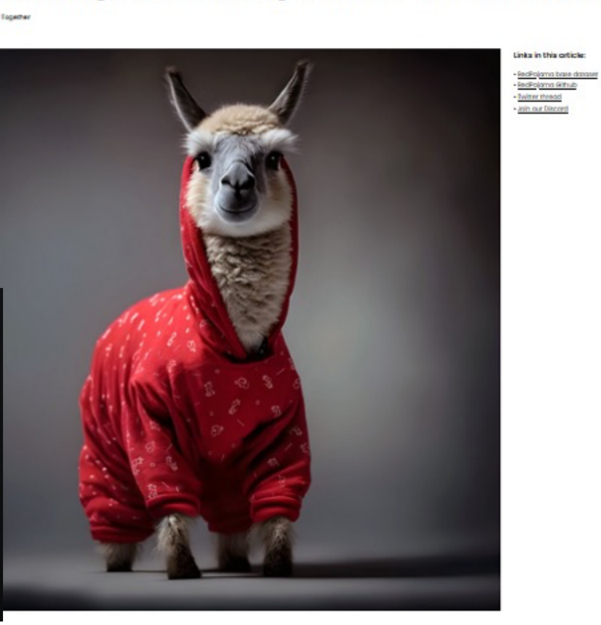
April 12, 2023 in Company Blog

Share this post

Two weeks ago, we released Dolly, a large language model (LLM) trained for less than $30 to exhibit ChatGPT-like human interactivity (aka instruction-following). Today, we're releasing Dolly 2.0, the first open source, instruction-following LLM, fine-tuned on a human-generated instruction dataset licensed for **research and commercial use.**

Dolly 2.0 is a 12B parameter language model based on the EleutherAI pythia model family and fine-tuned exclusively on a new, high-quality human generated instruction following dataset, crowdsourced among Databricks employees.

We are open-sourcing the entirety of Dolly 2.0, including the training code, the dataset, and the model weights, all suitable for commercial use. This means that any organization can create, own, and customize powerful LLMs that can talk to people, without paying for API access or sharing data with third parties.

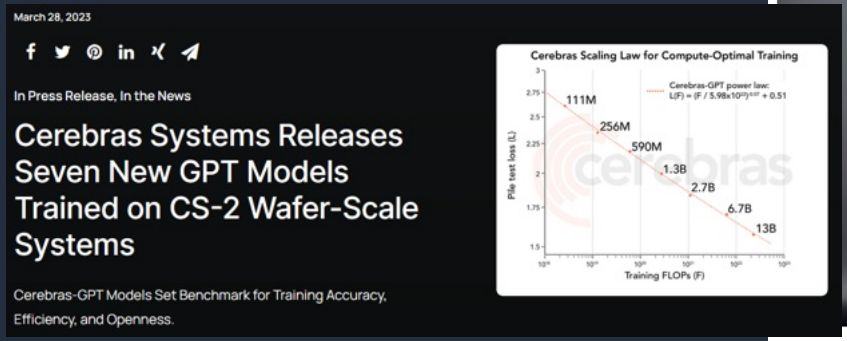


RedPajama, a project to create leading open-source models, starts by reproducing LLaMA training dataset of over 1.2 trillion tokens

Apr 17 • Written by Together

Links in this article:

- RedPajama base dataset
- RedPajama Github
- Twitter thread
- Join our Discord

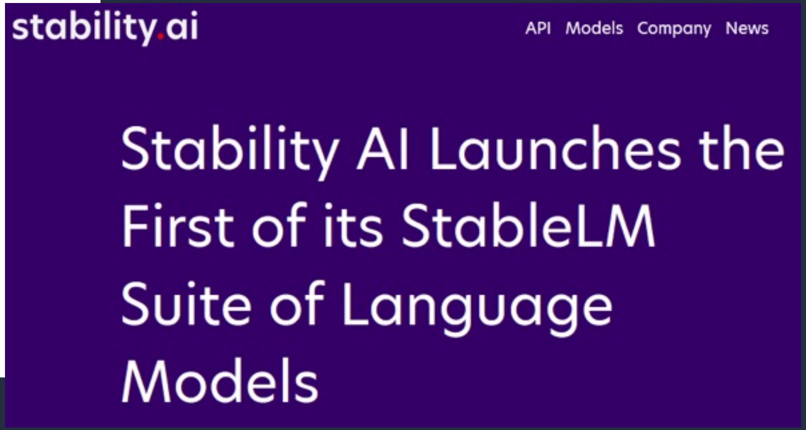


March 28, 2023

In Press Release, In the News

Cerebras Systems Releases Seven New GPT Models Trained on CS-2 Wafer-Scale Systems

Cerebras-GPT Models Set Benchmark for Training Accuracy, Efficiency, and Openness.



Stability AI Launches the First of its StableLM Suite of Language Models

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Many roads to FMs – hosting choice is key!

Serverless, fully managed

Marketplace, one-click usage

SageMaker Custom Hosting

Different workloads and usage patterns will require different FM hosting options!

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Caution!

Identity and access management

Detective controls

Infrastructure protection

Data protection

Incident response

Compliance

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

NEW

Amazon Bedrock

The easiest way to build and scale generative AI applications with FMs

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Bedrock supports a wide range of foundation models

FMs from Amazon

- Titan Text
- Titan Embeddings

FMs from AI21 Labs, Anthropic, and Stability AI

- Jurassic-2
- Claude
- Stable Diffusion

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Amazon SageMaker JumpStart with Foundation Models

Machine Learning Hub for SageMaker

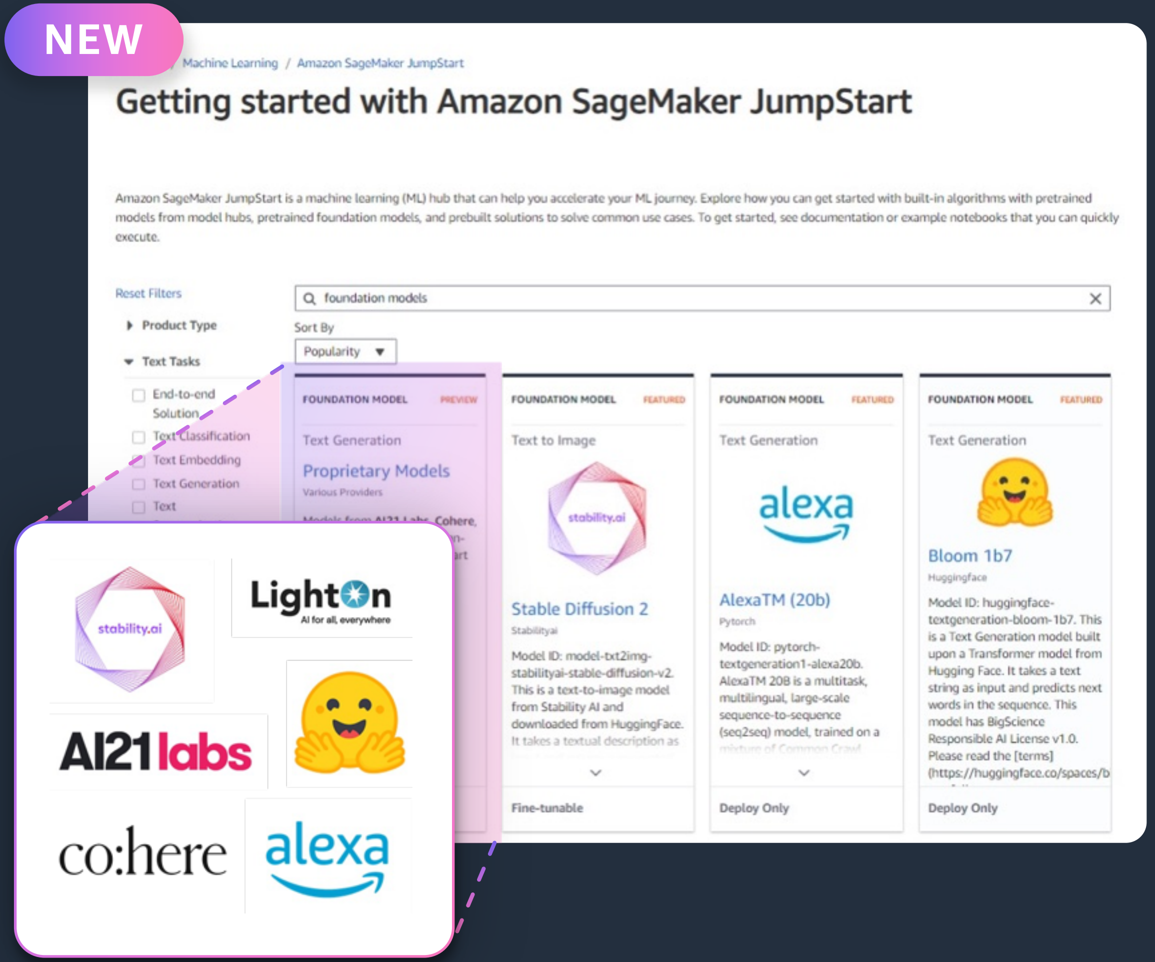
Browse through ~400 contents including, built-in algorithms with pre-trained models, (New) Foundation Models, solution templates, and example notebooks

UI as well as API based machine learning

Use the User Interface for single click model deployment or the API for Python SDK based workflow

(New) Share and collaborate within an organization

Share models and notebooks with others within your organization, and allow them to train with their own data or deploy as-is for inferencing



Machine Learning / Amazon SageMaker JumpStart

Getting started with Amazon SageMaker JumpStart

Amazon SageMaker JumpStart is a machine learning (ML) hub that can help you accelerate your ML journey. Explore how you can get started with built-in algorithms with pretrained models from model hubs, pretrained foundation models, and prebuilt solutions to solve common use cases. To get started, see documentation or example notebooks that you can quickly execute.

Reset Filters

foundation models

Sort By: Popularity

▸ Product Type

▾ Text Tasks

☐ End-to-end Solution
☐ Text Classification
☐ Text Embedding
☐ Text Generation
☐ Text

FOUNDATION MODEL — PREVIEW
Text Generation
Proprietary Models
Various Providers

FOUNDATION MODEL — FEATURED
Text to Image
Stable Diffusion 2
Stabilityai
Model ID: model-txt2img-stabilityai-stable-diffusion-v2. This is a text-to-image model from Stability AI and downloaded from HuggingFace. It takes a textual description as
Fine-tunable

FOUNDATION MODEL — FEATURED
Text Generation
AlexaTM (20b)
Pytorch
Model ID: pytorch-textgeneration1-alexa20b. AlexaTM 20B is a multitask, multilingual, large-scale sequence-to-sequence (seq2seq) model, trained on a
Deploy Only

FOUNDATION MODEL — FEATURED
Text Generation
Bloom 1b7
Huggingface
Model ID: huggingface-textgeneration-bloom-1b7. This is a Text Generation model built upon a Transformer model from Hugging Face. It takes a text string as input and predicts next words in the sequence. This model has BigScience Responsible AI License v1.0. Please read the [terms] (https://huggingface.co/spaces/b
Deploy Only

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Base components and design patterns

Models

Prompts

Memory

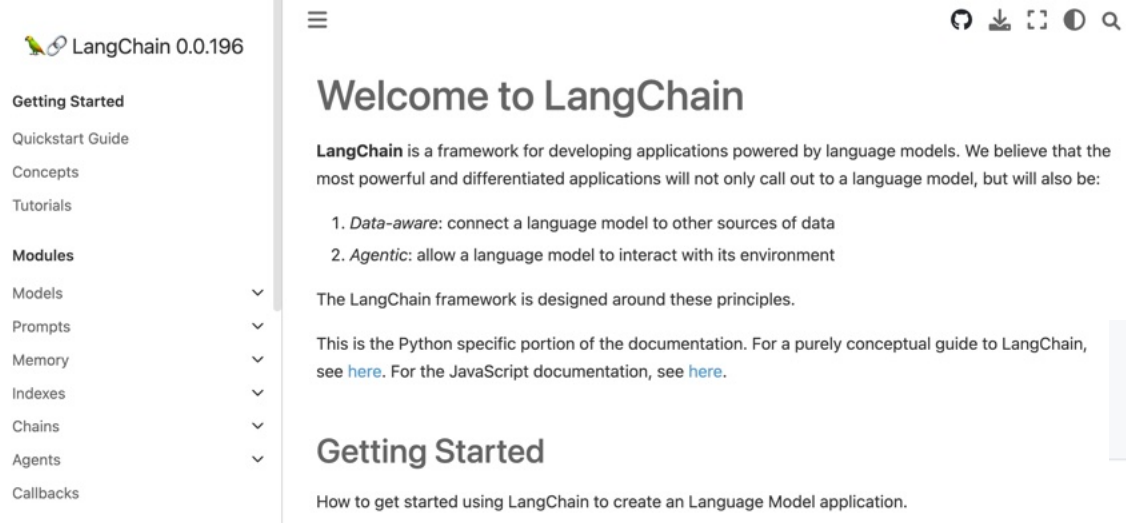
Chains

Tools

Agents

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

LangChain



Welcome to LangChain

LangChain is a framework for developing applications powered by language models. We believe that the most powerful and differentiated applications will not only call out to a language model, but will also be:

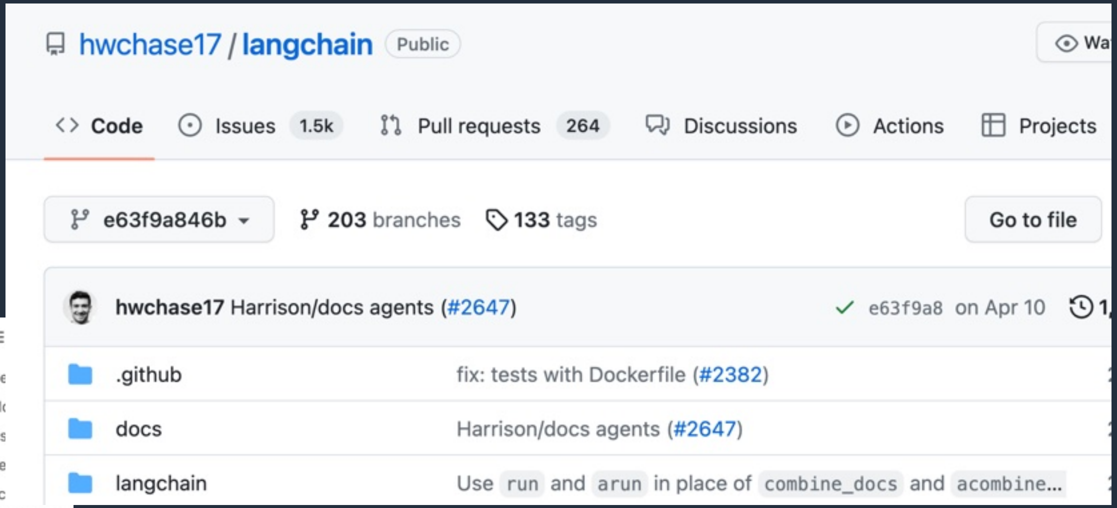
1. *Data-aware*: connect a language model to other sources of data
2. *Agentic*: allow a language model to interact with its environment

The LangChain framework is designed around these principles.

This is the Python specific portion of the documentation. For a purely conceptual guide to LangChain, see here. For the JavaScript documentation, see here.

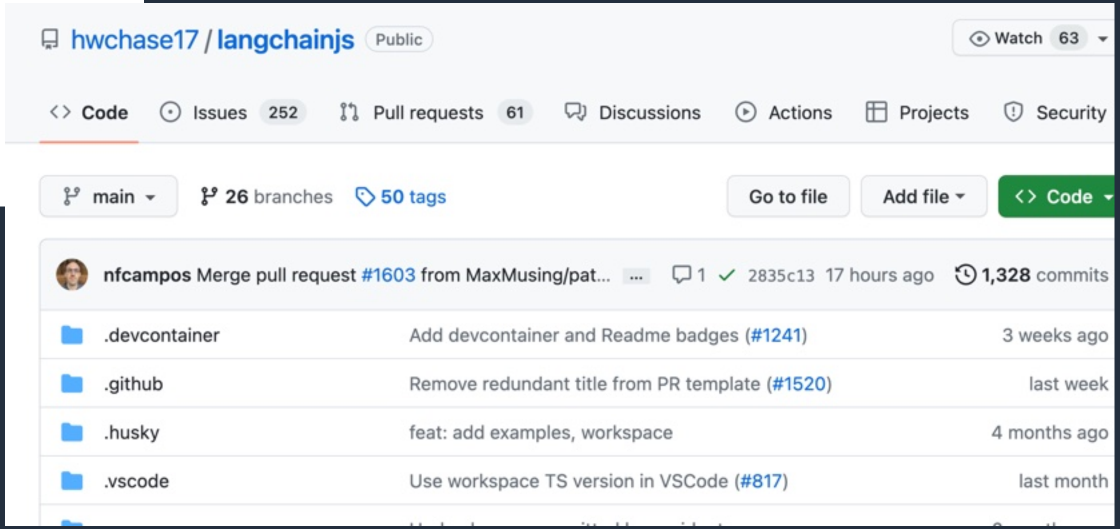
Getting Started

How to get started using LangChain to create an Language Model application.



hwchase17 Harrison/docs agents (#2647) ✓ e63f9a8 on Apr 10

.github	fix: tests with Dockerfile (#2382)
docs	Harrison/docs agents (#2647)
langchain	Use `run` and `arun` in place of `combine_docs` and `acombine...`



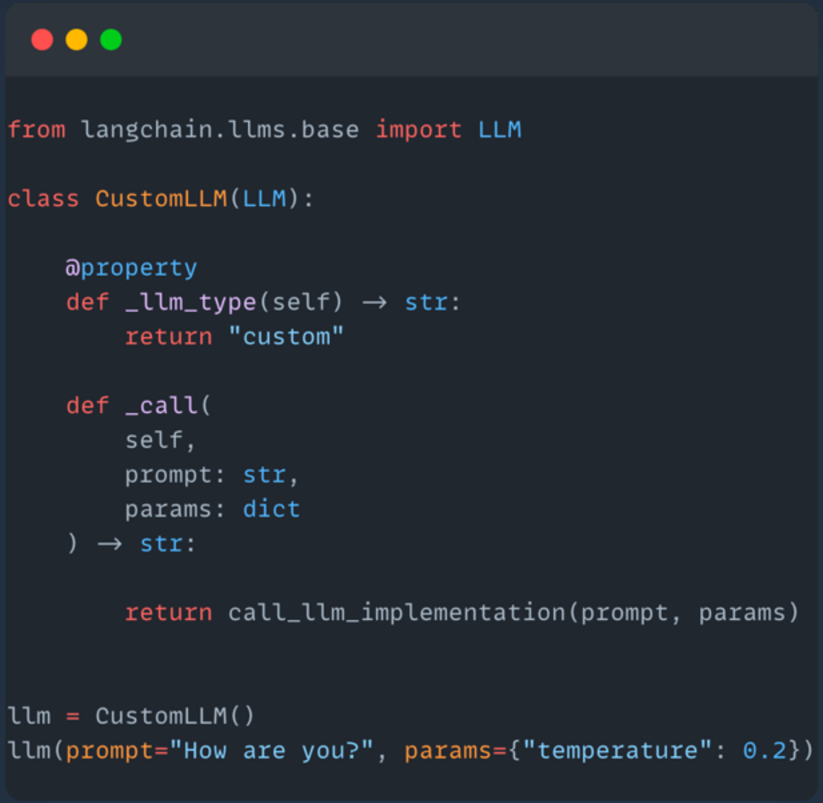
nfcampos Merge pull request #1603 from MaxMusing/pat... 1 ✓ 2835c13 17 hours ago 1,328 commits

.devcontainer	Add devcontainer and Readme badges (#1241)	3 weeks ago
.github	Remove redundant title from PR template (#1520)	last week
.husky	feat: add examples, workspace	4 months ago
.vscode	Use workspace TS version in VSCode (#817)	last month

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Models

- Language generation models
- Text Embedding Models
- Purpose-fine-tuned models (chat, …)

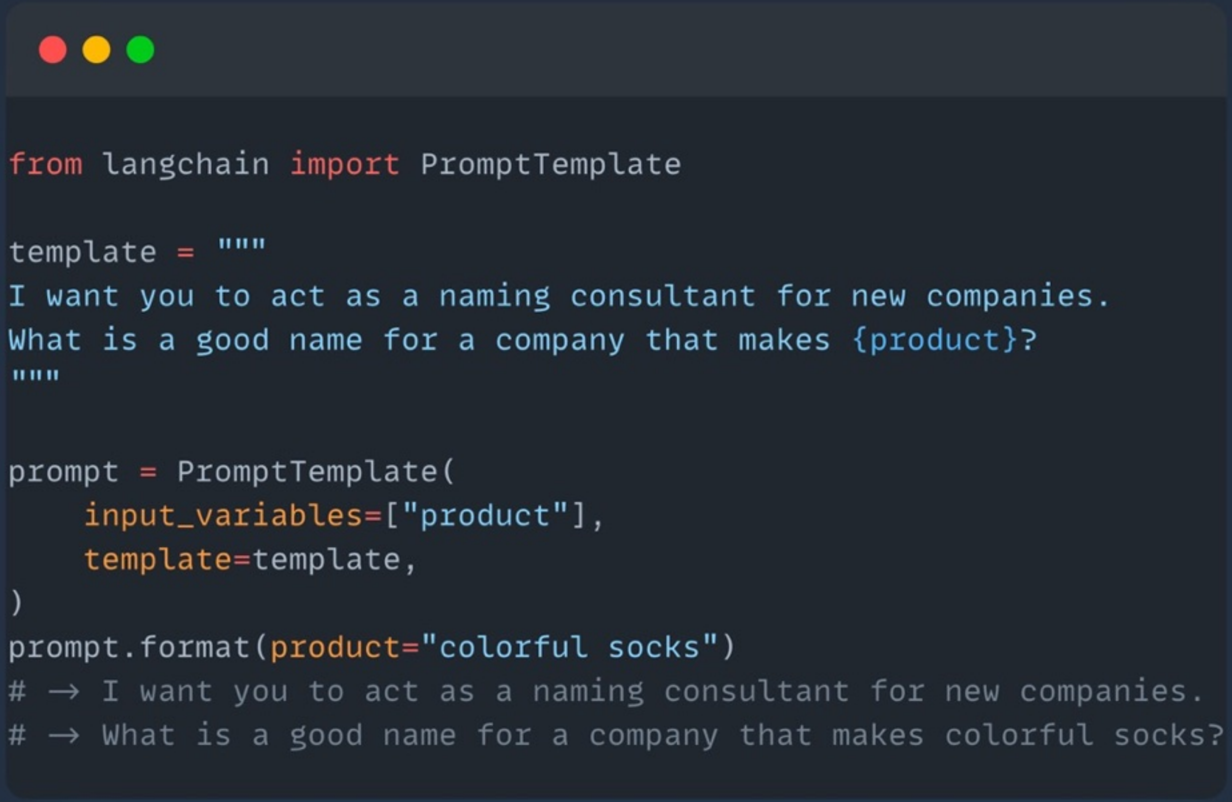


```
from langchain.llms.base import LLM

class CustomLLM(LLM):

    @property
    def _llm_type(self) -> str:
        return "custom"

    def _call(
        self,
        prompt: str,
        params: dict
    ) -> str:

        return call_llm_implementation(prompt, params)


llm = CustomLLM()
llm(prompt="How are you?", params={"temperature": 0.2})
```

Pre-built implementations available for multiple model providers and hosting options!

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

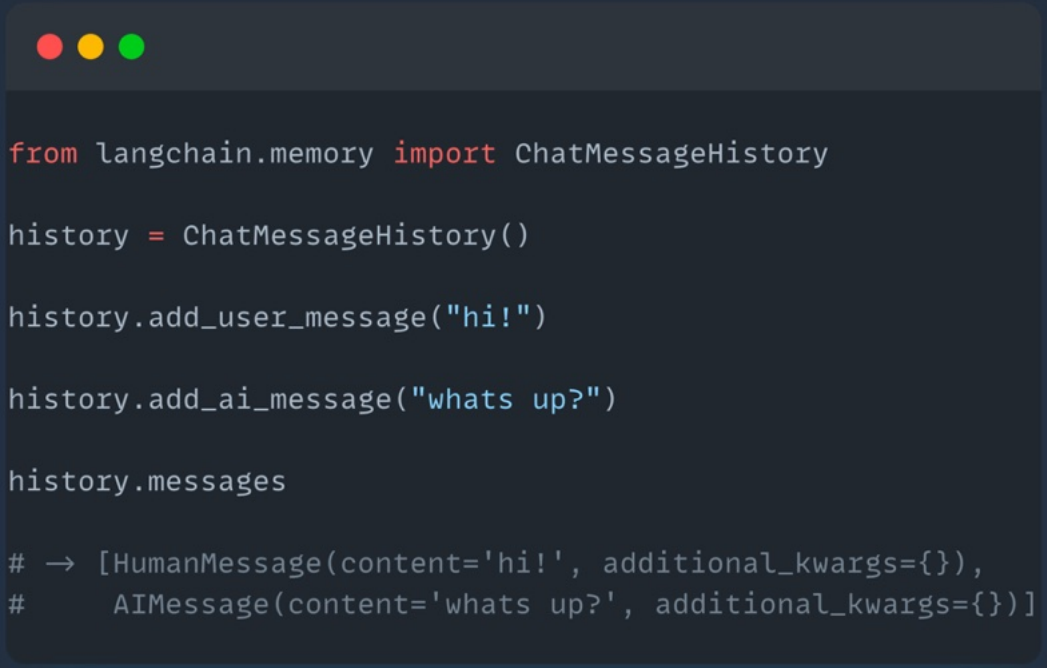
Prompts: dynamic prompting through PromptTempates



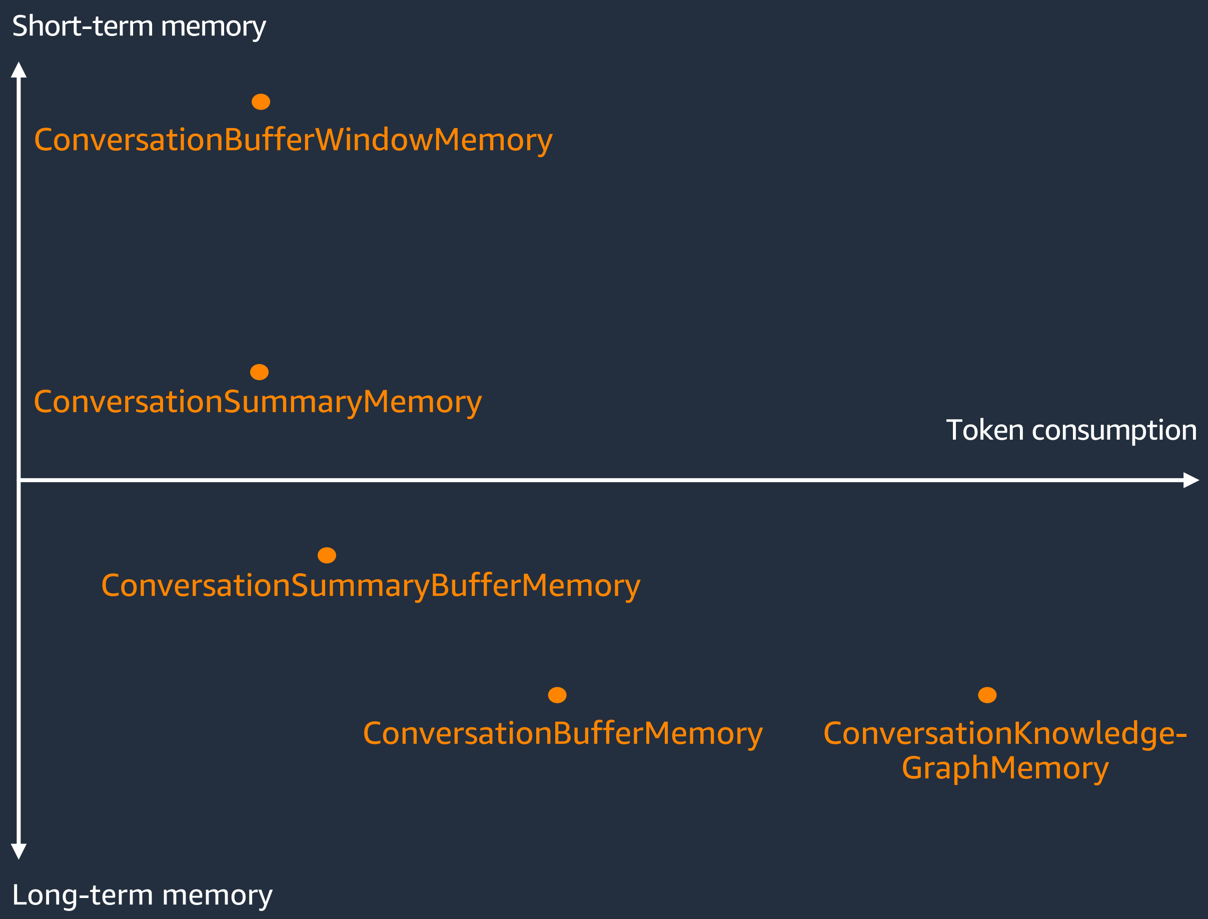
```
from langchain import PromptTemplate

template = """
I want you to act as a naming consultant for new companies.
What is a good name for a company that makes {product}?
"""

prompt = PromptTemplate(
    input_variables=["product"],
    template=template,
)
prompt.format(product="colorful socks")
# -> I want you to act as a naming consultant for new companies.
# -> What is a good name for a company that makes colorful socks?
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Memory: keeping track of conversation history



```
from langchain.memory import ChatMessageHistory

history = ChatMessageHistory()

history.add_user_message("hi!")

history.add_ai_message("whats up?")

history.messages

# → [HumanMessage(content='hi!', additional_kwargs={}),
#     AIMessage(content='whats up?', additional_kwargs={})]
```



Short-term memory

ConversationBufferWindowMemory

ConversationSummaryMemory

Token consumption

ConversationSummaryBufferMemory

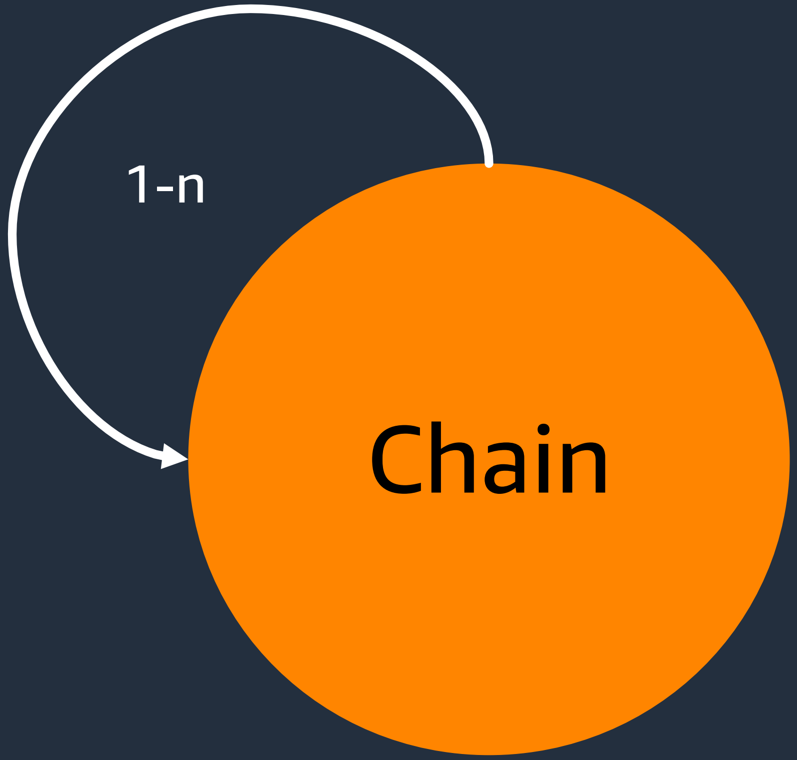
ConversationBufferMemory

ConversationKnowledge-GraphMemory

Long-term memory

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

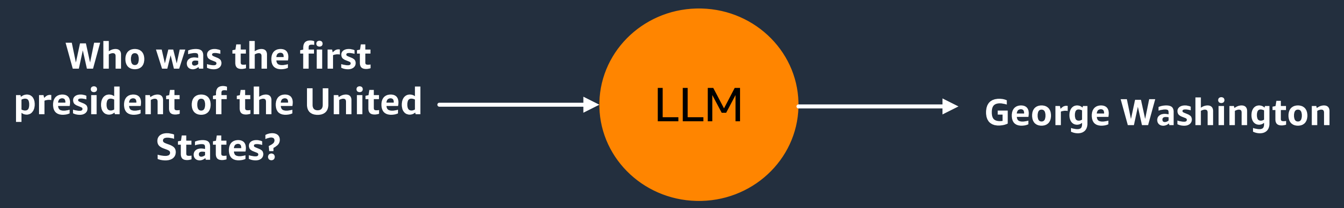
LangChains 😉



Pre-built implementations available in multiple fashions!

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

LLMChain



Who was the first president of the United States?

George Washington

Answer the following question: {question}
If you don’t know the answer, just say “I don’t know”.

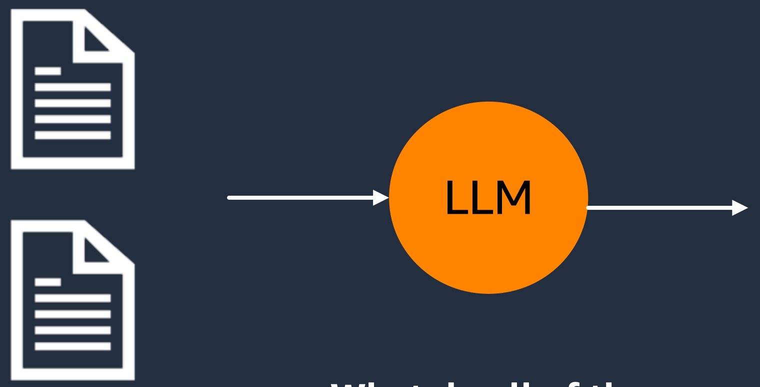
● Model Node

◆ Decision Node

▬ Business Logic Node

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

StuffDocumentsChain



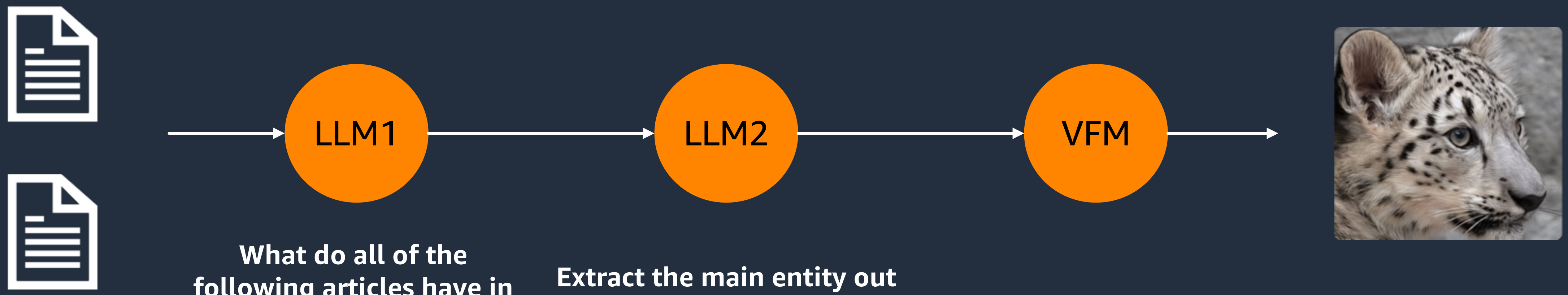
They are about predatory big cats called leopards.

What do all of the following articles have in common?
Article 1: {article1}
Article 2: {article2}
…

● Model Node
◆ Decision Node
■ Business Logic Node

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

SequentialChain



What do all of the following articles have in common?
Article 1: {article1}
Article 2: {article2}
…

Extract the main entity out of the input.
Input: {input}

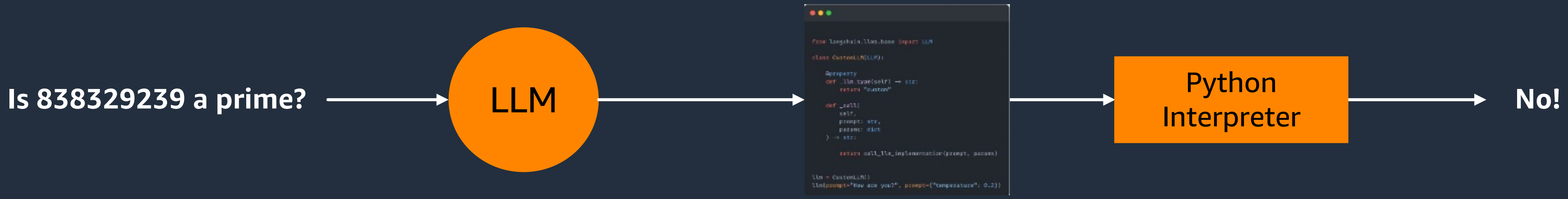
● Model Node

◆ Decision Node

■ Business Logic Node

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

TransformationChain



```python
from langchain.llms.base import LLM

class CustomLLM(LLM):

    @property
    def _llm_type(self) -> str:
        return "custom"

    def _call(
        self,
        prompt: str,
        params: dict
    ) -> str:

        return call_llm_implementation(prompt, params)


llm = CustomLLM()
llm(prompt="How are you?", prompt={"temperature": 0.2})
```

**Translate this question into Python source code.
Question: {question}**

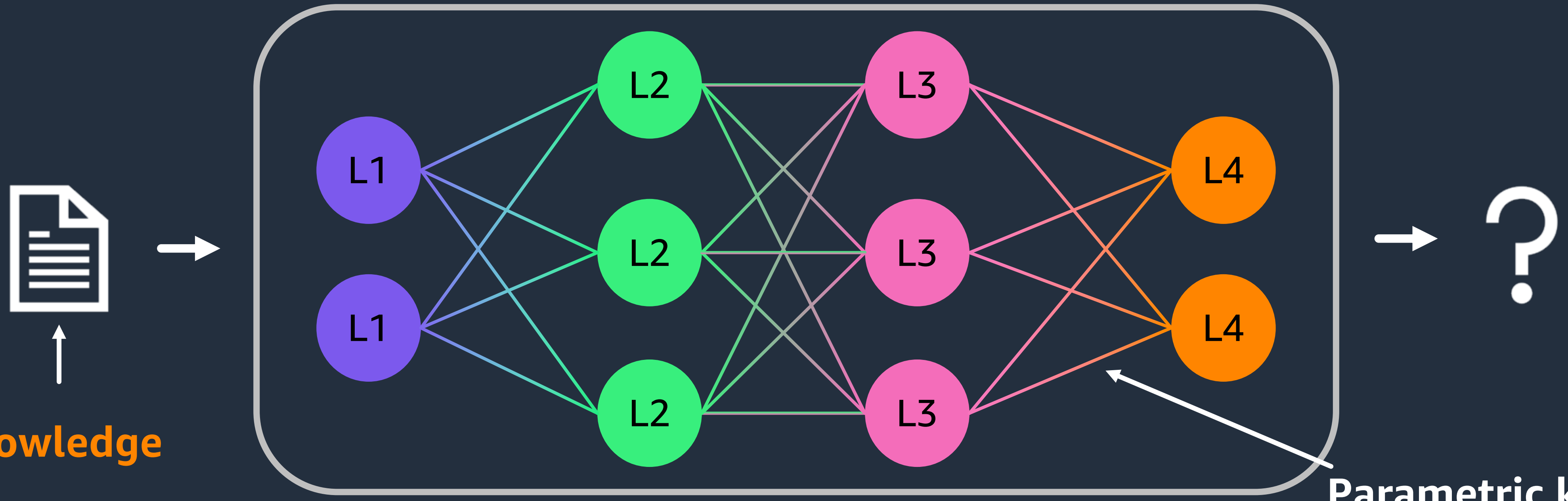
● Model Node

◆ Decision Node

■ Business Logic Node

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Domain-specific knowledge infusion: forms of knowledge



Source knowledge

Adjustable though prompt enrichment ($)

Dynamic data (live-systems, (semi-)frequently changing knowledge bases, …)

Minimizes risk of halluzination, adds tracability of results

Parametric knowledge

Adjustable through fine-tuning ($$$)

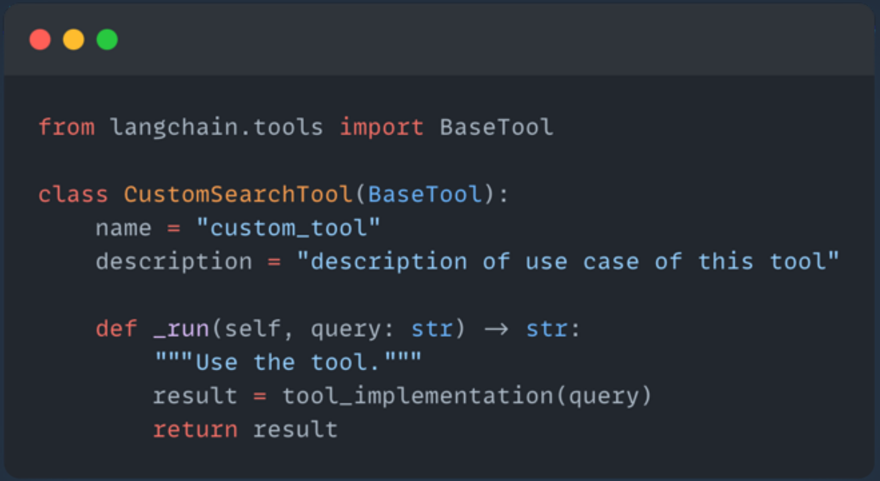
Static data (language foundations , domain-specific vocabulary, writing style, chat/instruction-fine-tuning, …)

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

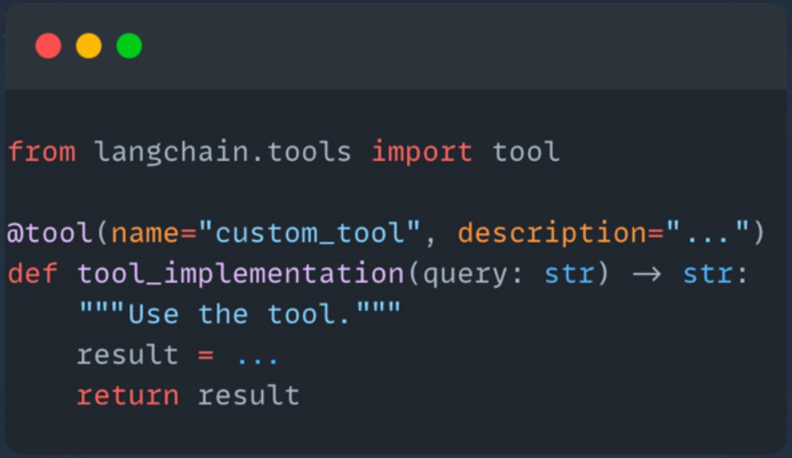
Domain-specific knowledge infusion through Tools

*A Tool is "a function that performs a **specific duty**. (...) The interface for a tool is currently a function that is expected to have a string as an input, with a string as an output."*

"These tools can be generic utilities (...), other chains, or even other agents."



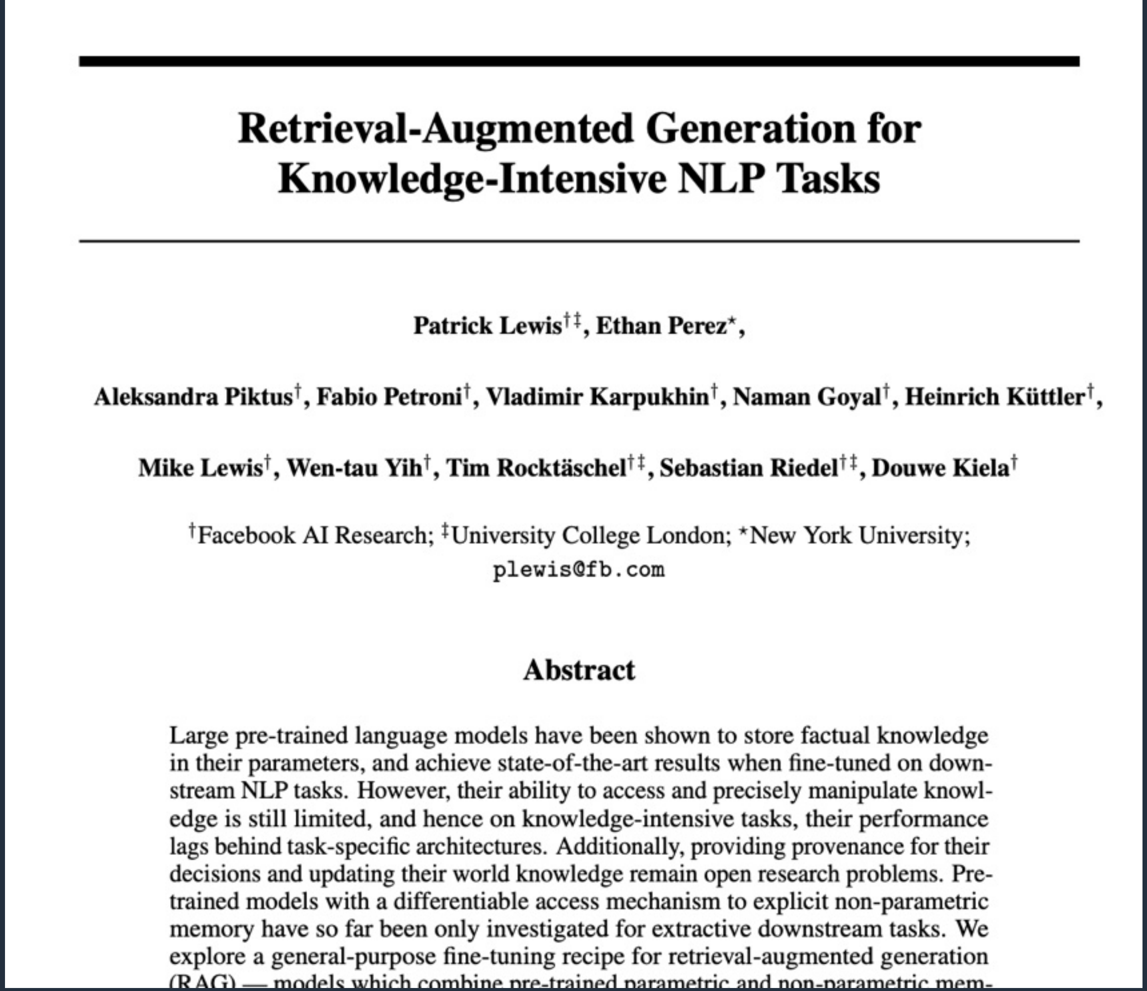
```python
from langchain.tools import BaseTool

class CustomSearchTool(BaseTool):
    name = "custom_tool"
    description = "description of use case of this tool"

    def _run(self, query: str) -> str:
        """Use the tool."""
        result = tool_implementation(query)
        return result
```



```python
from langchain.tools import tool

@tool(name="custom_tool", description="...")
def tool_implementation(query: str) -> str:
    """Use the tool."""
    result = ...
    return result
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Tools: RAG



Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis[†‡], **Ethan Perez**[*],

Aleksandra Piktus[†], **Fabio Petroni**[†], **Vladimir Karpukhin**[†], **Naman Goyal**[†], **Heinrich Küttler**[†],

Mike Lewis[†], **Wen-tau Yih**[†], **Tim Rocktäschel**[†‡], **Sebastian Riedel**[†‡], **Douwe Kiela**[†]

[†]Facebook AI Research; [‡]University College London; [*]New York University;
`plewis@fb.com`

Abstract

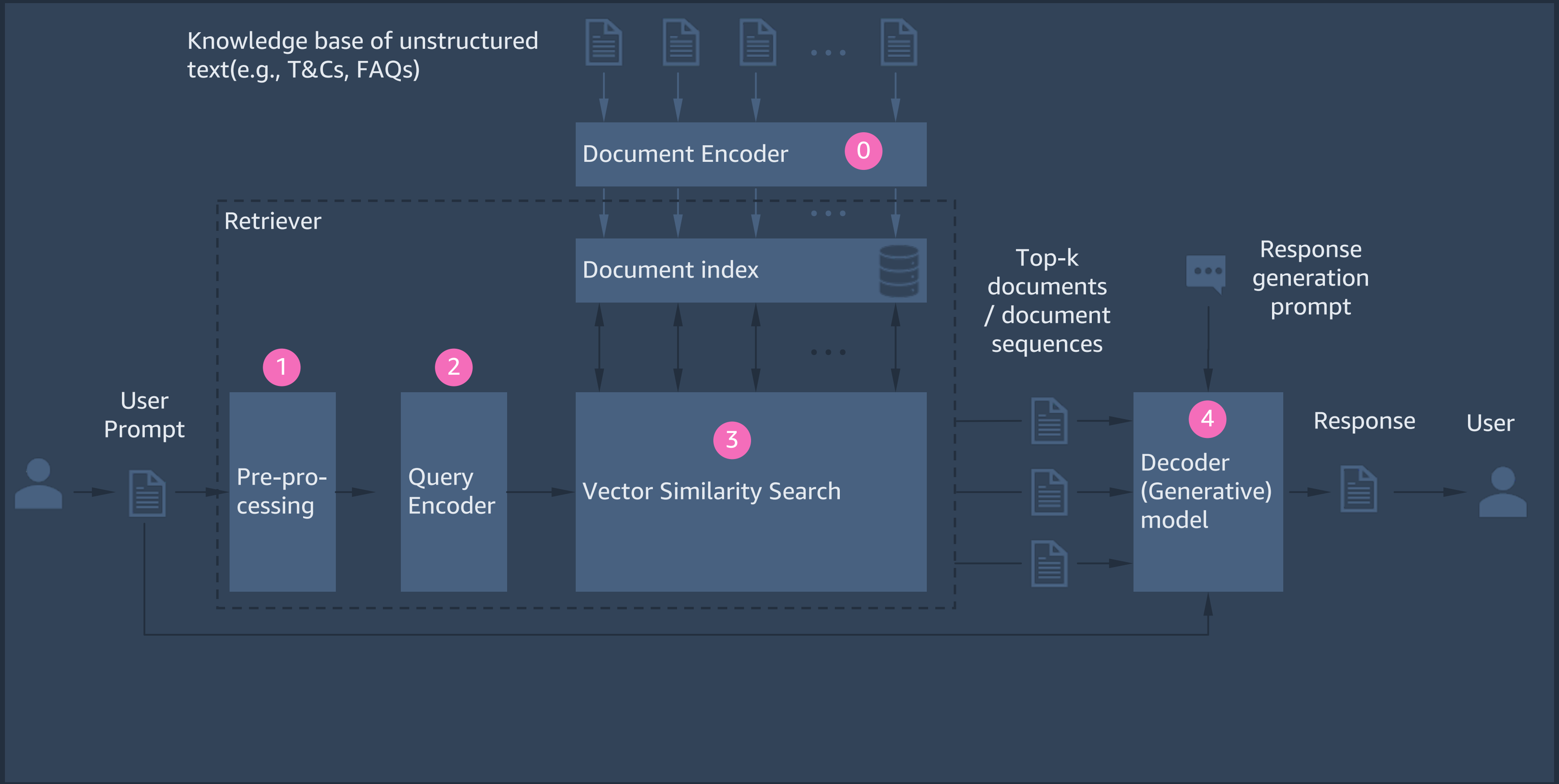
Large pre-trained language models have been shown to store factual knowledge in their parameters, and achieve state-of-the-art results when fine-tuned on downstream NLP tasks. However, their ability to access and precisely manipulate knowledge is still limited, and hence on knowledge-intensive tasks, their performance lags behind task-specific architectures. Additionally, providing provenance for their decisions and updating their world knowledge remain open research problems. Pre-trained models with a differentiable access mechanism to explicit non-parametric memory have so far been only investigated for extractive downstream tasks. We explore a general-purpose fine-tuning recipe for retrieval-augmented generation (RAG) — models which combine pre-trained parametric and non-parametric mem-

- Retrieval-augmented generation
- Access to knowledge base of unstructured text
- Implemented through two-step chain powered by two different LLMs

Source: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, Patrick Lewis et al., 2021, https://arxiv.org/pdf/2005.11401.pdf

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Tools: RAG


Knowledge base of unstructured text(e.g., T&Cs, FAQs)
Document Encoder
0
Retriever
Document index
Top-k documents / document sequences
Response generation prompt
User Prompt
1
2
3
4
Pre-pro-cessing
Query Encoder
Vector Similarity Search
Decoder (Generative) model
Response
User

Source: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, Patrick Lewis et al., 2021, https://arxiv.org/pdf/2005.11401.pdf

Concept

⓪ Knowledge documents / document sequences are encoded and ingested into a vector database.

① Customer e-mail query is pre-processed and/or tokenized

② Tokenized input query is encoded

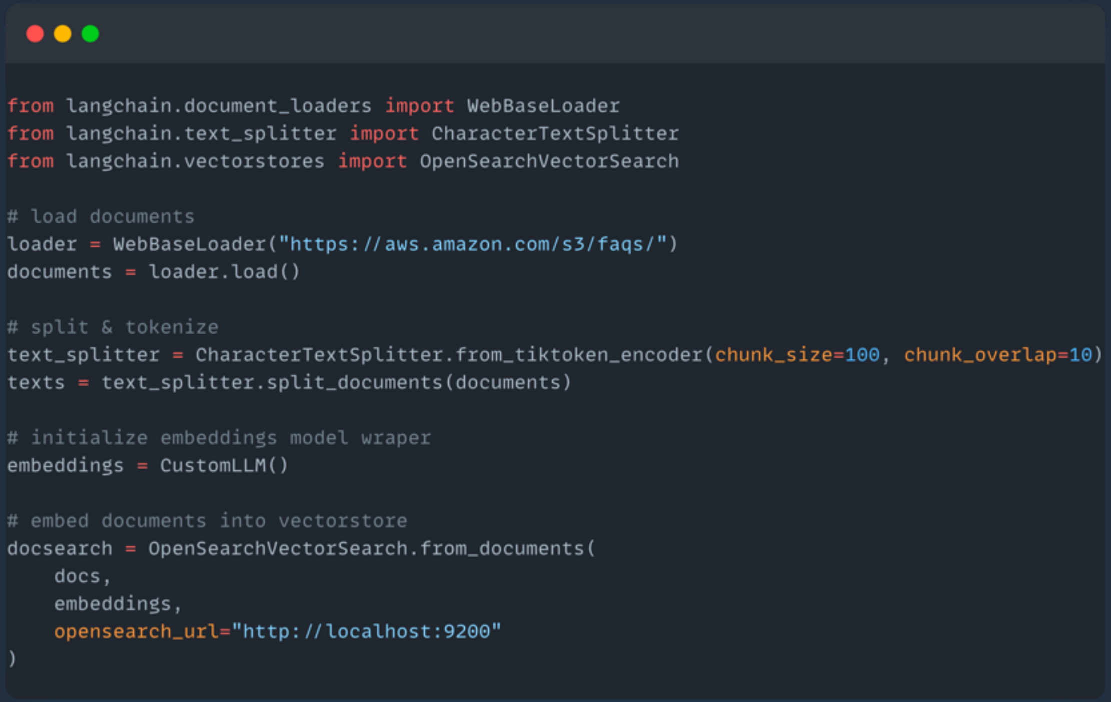
③ Encoded query is used to retrieve most similar text passages in document index using vector similarity search (e.g., Mixed Inner Product Search)

④ Top-k retrieved documents/text passages in combination with original customer e-mail query and e-mail generation prompt are fed into Generator model (Encoder-Decoder) to generate response e-mail

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

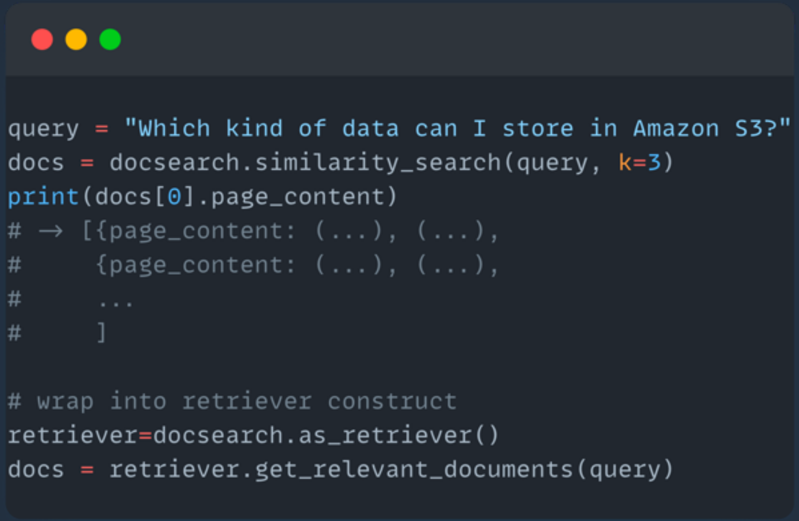
Tools: RAG

Document embedding & vectorstore ingestion



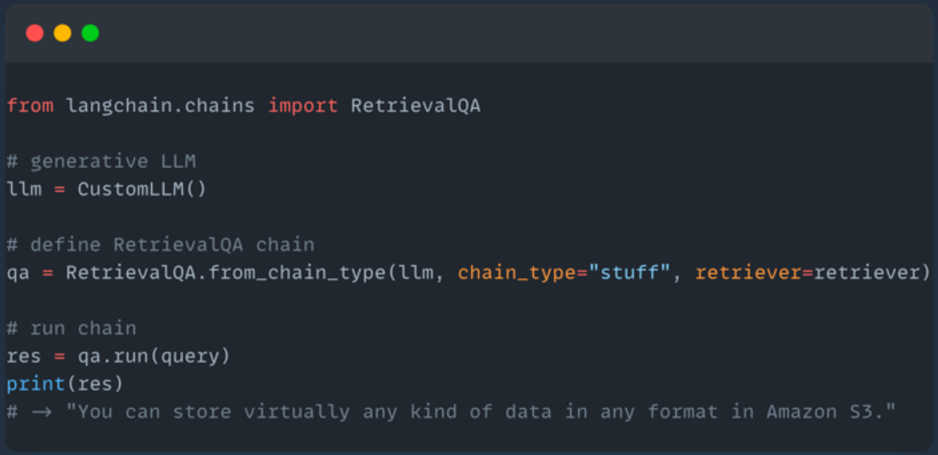
```
from langchain.document_loaders import WebBaseLoader
from langchain.text_splitter import CharacterTextSplitter
from langchain.vectorstores import OpenSearchVectorSearch

# load documents
loader = WebBaseLoader("https://aws.amazon.com/s3/faqs/")
documents = loader.load()

# split & tokenize
text_splitter = CharacterTextSplitter.from_tiktoken_encoder(chunk_size=100, chunk_overlap=10)
texts = text_splitter.split_documents(documents)

# initialize embeddings model wraper
embeddings = CustomLLM()

# embed documents into vectorstore
docsearch = OpenSearchVectorSearch.from_documents(
    docs,
    embeddings,
    opensearch_url="http://localhost:9200"
)
```

Retrievers



```
query = "Which kind of data can I store in Amazon S3?"
docs = docsearch.similarity_search(query, k=3)
print(docs[0].page_content)
# → [{page_content: (...), (...),
#     {page_content: (...), (...),
#     ...
#     ]

# wrap into retriever construct
retriever=docsearch.as_retriever()
docs = retriever.get_relevant_documents(query)
```

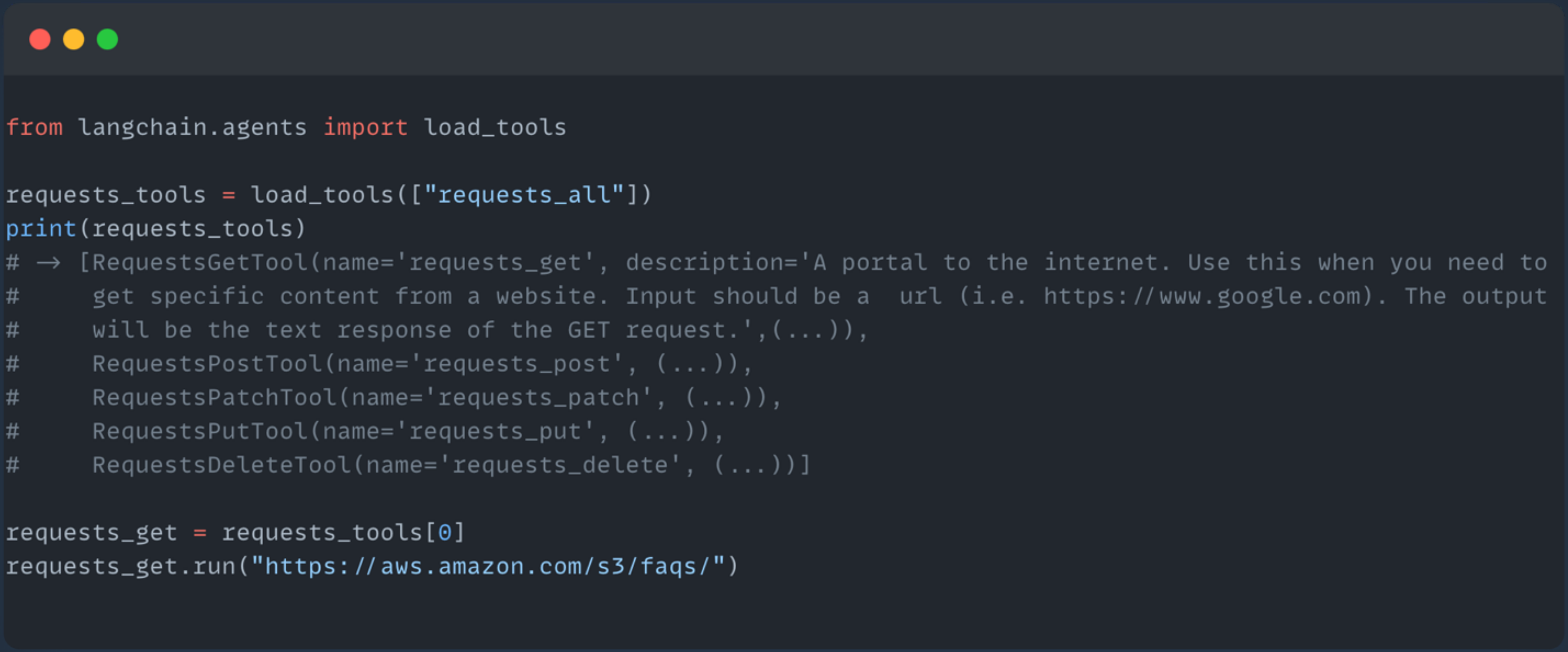
RetrievalQAChain



```
from langchain.chains import RetrievalQA

# generative LLM
llm = CustomLLM()

# define RetrievalQA chain
qa = RetrievalQA.from_chain_type(llm, chain_type="stuff", retriever=retriever)

# run chain
res = qa.run(query)
print(res)
# → "You can store virtually any kind of data in any format in Amazon S3."
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

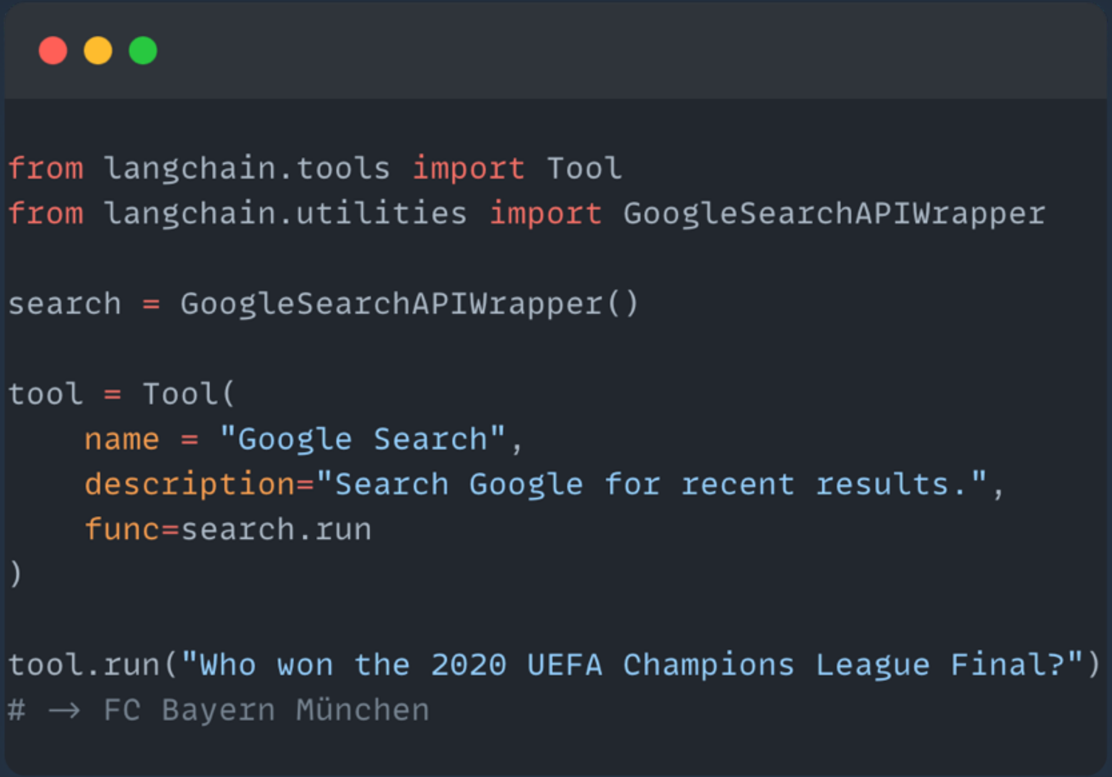
Tools: Requests



```
from langchain.agents import load_tools

requests_tools = load_tools(["requests_all"])
print(requests_tools)
# -> [RequestsGetTool(name='requests_get', description='A portal to the internet. Use this when you need to
#     get specific content from a website. Input should be a  url (i.e. https://www.google.com). The output
#     will be the text response of the GET request.',(...)),
#     RequestsPostTool(name='requests_post', (...)),
#     RequestsPatchTool(name='requests_patch', (...)),
#     RequestsPutTool(name='requests_put', (...)),
#     RequestsDeleteTool(name='requests_delete', (...))]

requests_get = requests_tools[0]
requests_get.run("https://aws.amazon.com/s3/faqs/")
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

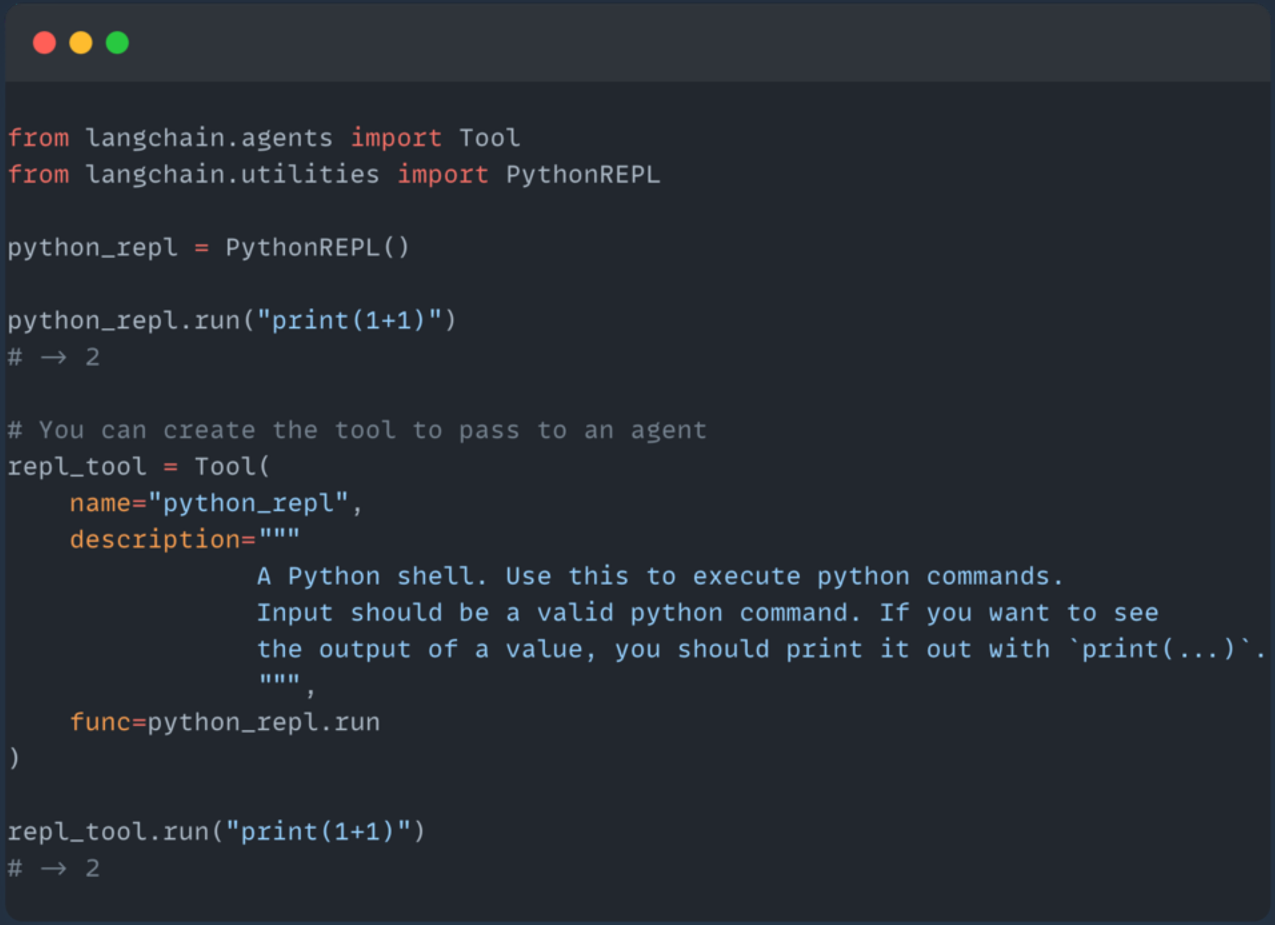
Tools: Google Search



```python
from langchain.tools import Tool
from langchain.utilities import GoogleSearchAPIWrapper

search = GoogleSearchAPIWrapper()

tool = Tool(
    name = "Google Search",
    description="Search Google for recent results.",
    func=search.run
)

tool.run("Who won the 2020 UEFA Champions League Final?")
# → FC Bayern München
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

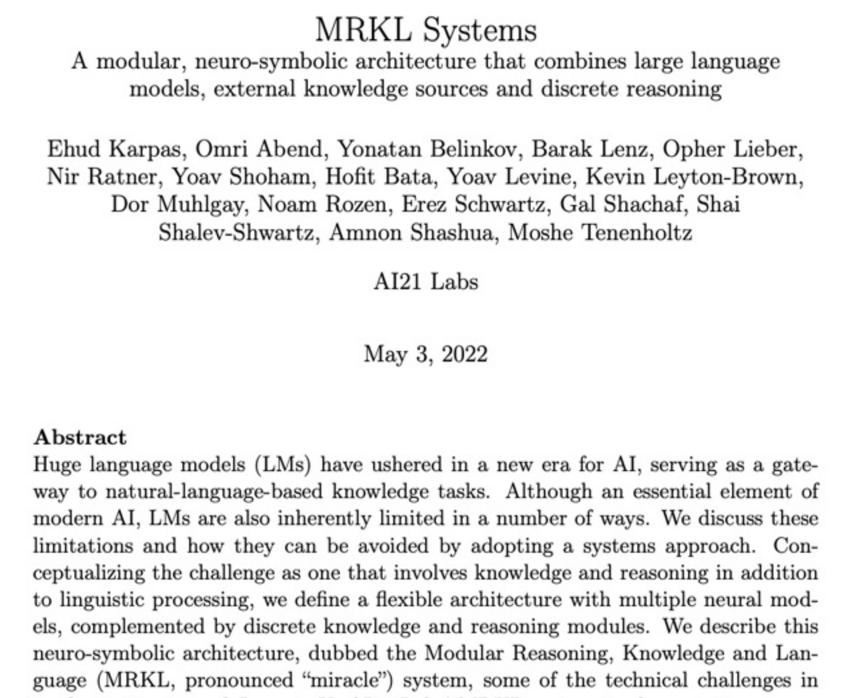
Tools: Python REPL



```python
from langchain.agents import Tool
from langchain.utilities import PythonREPL

python_repl = PythonREPL()

python_repl.run("print(1+1)")
# → 2

# You can create the tool to pass to an agent
repl_tool = Tool(
    name="python_repl",
    description="""
                A Python shell. Use this to execute python commands.
                Input should be a valid python command. If you want to see
                the output of a value, you should print it out with `print(...)`.
                """,
    func=python_repl.run
)

repl_tool.run("print(1+1)")
# → 2
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Agents: MRKL



MRKL Systems

A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning

Ehud Karpas, Omri Abend, Yonatan Belinkov, Barak Lenz, Opher Lieber,
Nir Ratner, Yoav Shoham, Hofit Bata, Yoav Levine, Kevin Leyton-Brown,
Dor Muhlgay, Noam Rozen, Erez Schwartz, Gal Shachaf, Shai
Shalev-Shwartz, Amnon Shashua, Moshe Tenenholtz

AI21 Labs

May 3, 2022

Abstract

Huge language models (LMs) have ushered in a new era for AI, serving as a gateway to natural-language-based knowledge tasks. Although an essential element of modern AI, LMs are also inherently limited in a number of ways. We discuss these limitations and how they can be avoided by adopting a systems approach. Conceptualizing the challenge as one that involves knowledge and reasoning in addition to linguistic processing, we define a flexible architecture with multiple neural models, complemented by discrete knowledge and reasoning modules. We describe this neuro-symbolic architecture, dubbed the Modular Reasoning, Knowledge and Language (MRKL, pronounced "miracle") system, some of the technical challenges in

Source: MRKL Systems – A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning, Ehud Karpas et al, 2022, https://arxiv.org/pdf/2205.00445.pdf

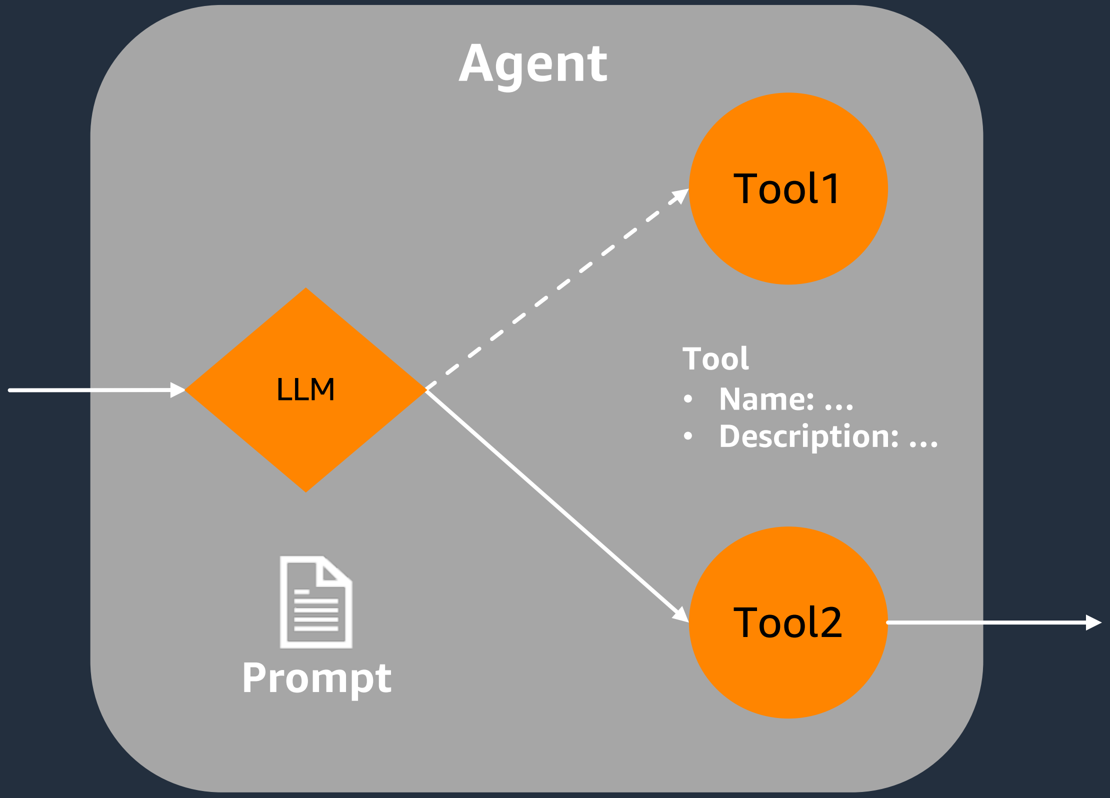
Idea: System routes between 1-n out of multiple tools, which can be *neural* or *symbolic*.

Benefits:

- Proprietary knowledge
- Up-to-date information
- Interpretability
- Composability
- Robust extensibility
- Safe fallback

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Agents: basic components

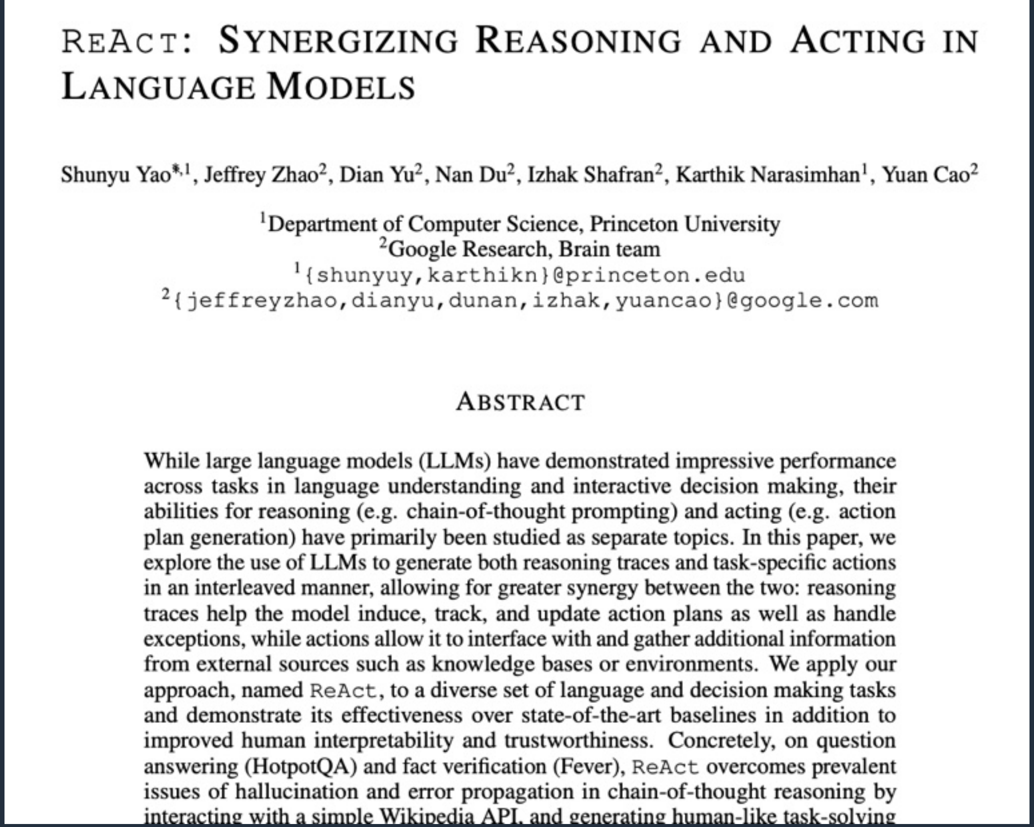


Tool

- **Name: …**
- **Description: …**

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Agents: ReAct



REACT: SYNERGIZING REASONING AND ACTING IN LANGUAGE MODELS

Shunyu Yao[*,1], Jeffrey Zhao[2], Dian Yu[2], Nan Du[2], Izhak Shafran[2], Karthik Narasimhan[1], Yuan Cao[2]

[1]Department of Computer Science, Princeton University
[2]Google Research, Brain team
[1]`{shunyuy,karthikn}@princeton.edu`
[2]`{jeffreyzhao,dianyu,dunan,izhak,yuancao}@google.com`

ABSTRACT

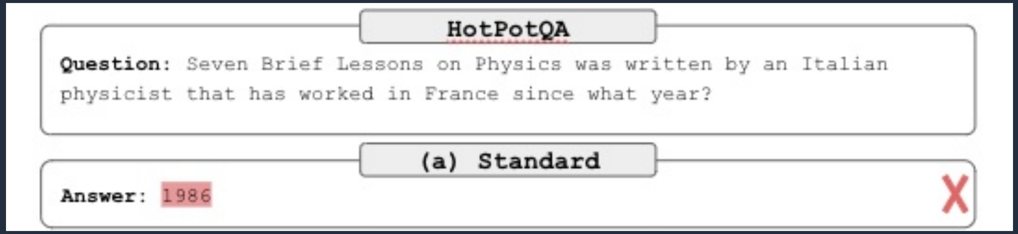
While large language models (LLMs) have demonstrated impressive performance across tasks in language understanding and interactive decision making, their abilities for reasoning (e.g. chain-of-thought prompting) and acting (e.g. action plan generation) have primarily been studied as separate topics. In this paper, we explore the use of LLMs to generate both reasoning traces and task-specific actions in an interleaved manner, allowing for greater synergy between the two: reasoning traces help the model induce, track, and update action plans as well as handle exceptions, while actions allow it to interface with and gather additional information from external sources such as knowledge bases or environments. We apply our approach, named `ReAct`, to a diverse set of language and decision making tasks and demonstrate its effectiveness over state-of-the-art baselines in addition to improved human interpretability and trustworthiness. Concretely, on question answering (HotpotQA) and fact verification (Fever), `ReAct` overcomes prevalent issues of hallucination and error propagation in chain-of-thought reasoning by interacting with a simple Wikipedia API, and generating human-like task-solving

Source: ReAct: Synergizing reasoning and acting in language models,
Shunyu Yao et al, 2023, https://arxiv.org/pdf/2210.03629.pdf

- Logical and modular reasoning of GenAI-powered systems (chain-of-thought)
- Execution of task-specific actions (action plan generation)
- Implemented through chain of recursive steps against a powerful LLM

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

ReAct (Reasoning + Action) Prompting



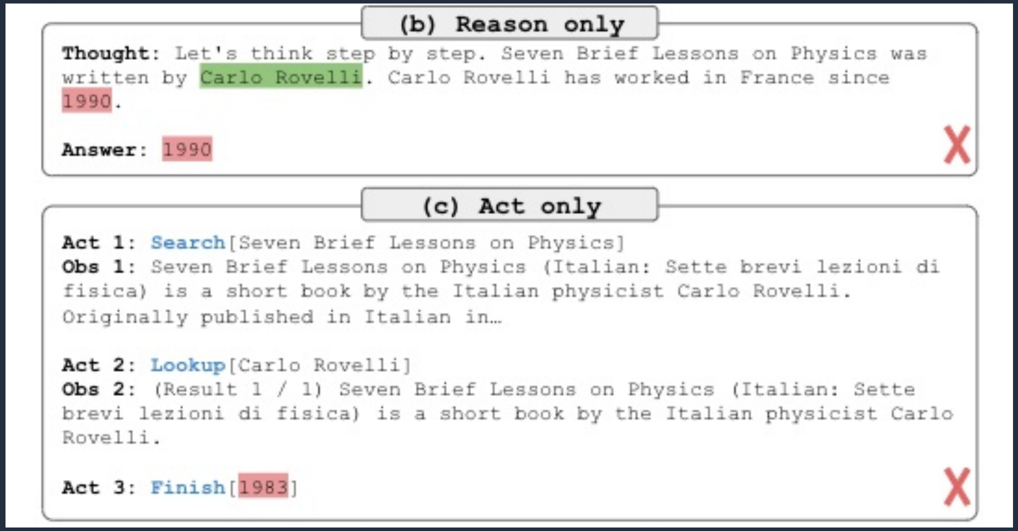
HotPotQA

Question: Seven Brief Lessons on Physics was written by an Italian physicist that has worked in France since what year?

(a) Standard

Answer: 1986

✗



(b) Reason only

Thought: Let's think step by step. Seven Brief Lessons on Physics was written by Carlo Rovelli. Carlo Rovelli has worked in France since 1990.

Answer: 1990

✗

(c) Act only

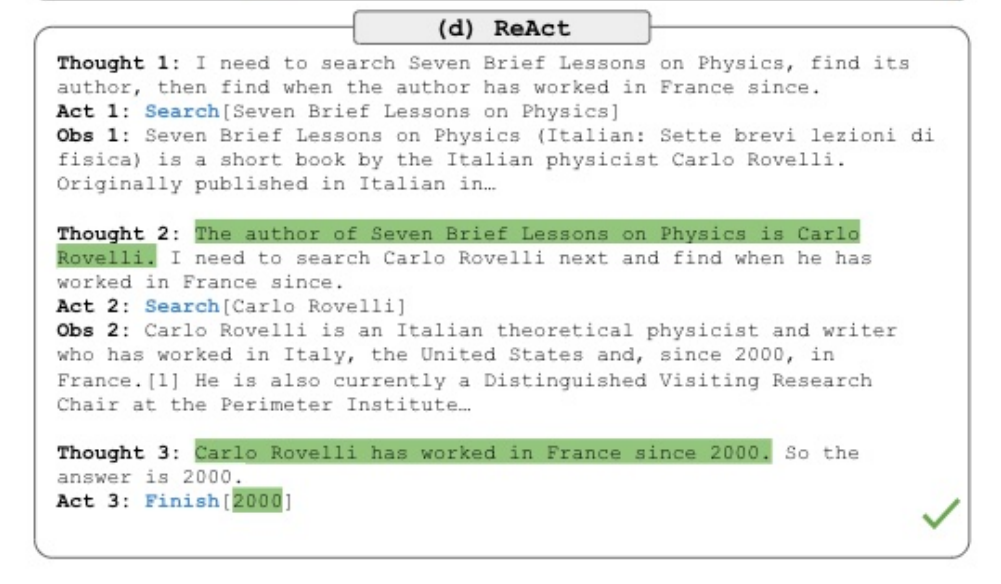
Act 1: Search[Seven Brief Lessons on Physics]
Obs 1: Seven Brief Lessons on Physics (Italian: Sette brevi lezioni di fisica) is a short book by the Italian physicist Carlo Rovelli. Originally published in Italian in…

Act 2: Lookup[Carlo Rovelli]
Obs 2: (Result 1 / 1) Seven Brief Lessons on Physics (Italian: Sette brevi lezioni di fisica) is a short book by the Italian physicist Carlo Rovelli.

Act 3: Finish[1983]

✗

Combine text reasoning and actions in a single model



(d) ReAct

Thought 1: I need to search Seven Brief Lessons on Physics, find its author, then find when the author has worked in France since.
Act 1: Search[Seven Brief Lessons on Physics]
Obs 1: Seven Brief Lessons on Physics (Italian: Sette brevi lezioni di fisica) is a short book by the Italian physicist Carlo Rovelli. Originally published in Italian in…

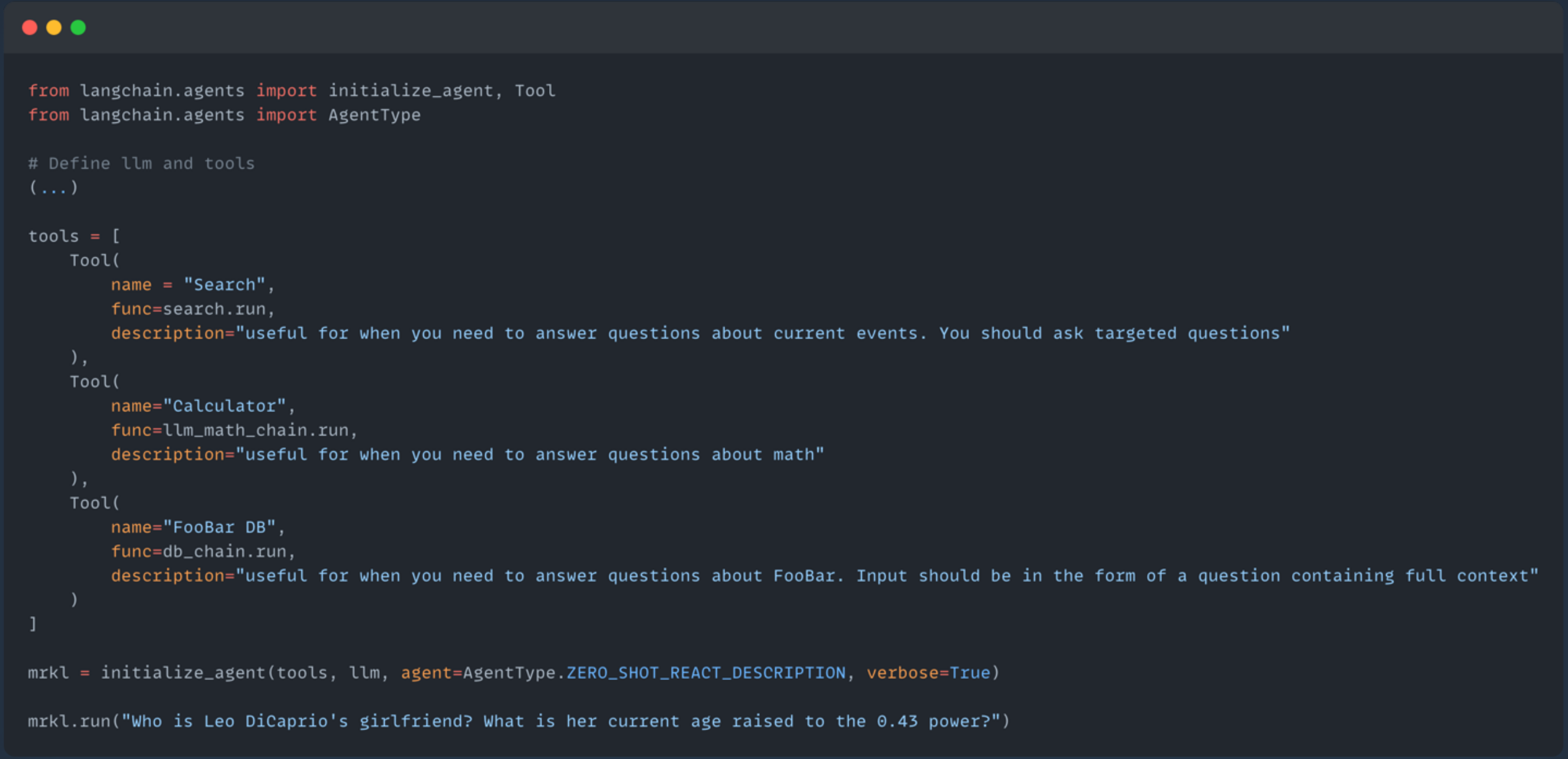
Thought 2: The author of Seven Brief Lessons on Physics is Carlo Rovelli. I need to search Carlo Rovelli next and find when he has worked in France since.
Act 2: Search[Carlo Rovelli]
Obs 2: Carlo Rovelli is an Italian theoretical physicist and writer who has worked in Italy, the United States and, since 2000, in France.[1] He is also currently a Distinguished Visiting Research Chair at the Perimeter Institute…

Thought 3: Carlo Rovelli has worked in France since 2000. So the answer is 2000.
Act 3: Finish[2000]

✓

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Agents: MRKL agent implementation

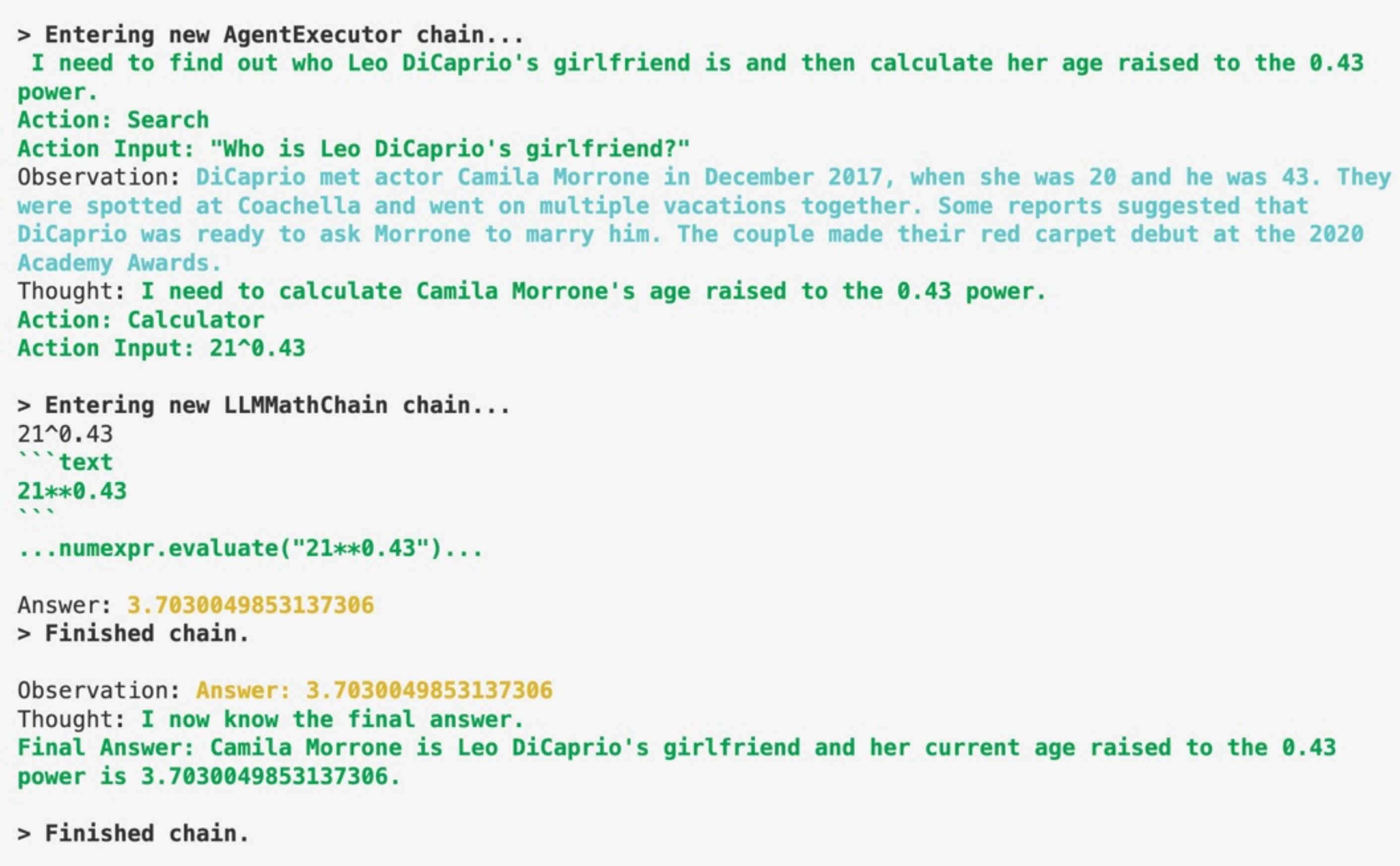


```python
from langchain.agents import initialize_agent, Tool
from langchain.agents import AgentType

# Define llm and tools
(...)

tools = [
    Tool(
        name = "Search",
        func=search.run,
        description="useful for when you need to answer questions about current events. You should ask targeted questions"
    ),
    Tool(
        name="Calculator",
        func=llm_math_chain.run,
        description="useful for when you need to answer questions about math"
    ),
    Tool(
        name="FooBar DB",
        func=db_chain.run,
        description="useful for when you need to answer questions about FooBar. Input should be in the form of a question containing full context"
    )
]

mrkl = initialize_agent(tools, llm, agent=AgentType.ZERO_SHOT_REACT_DESCRIPTION, verbose=True)

mrkl.run("Who is Leo DiCaprio's girlfriend? What is her current age raised to the 0.43 power?")
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Agents: MRKL agent implementation

“Who is Leo DiCaprio's girlfriend? What is her current age raised to the 0.43 power?”



~~~
> Entering new AgentExecutor chain...
 I need to find out who Leo DiCaprio's girlfriend is and then calculate her age raised to the 0.43
power.
Action: Search
Action Input: "Who is Leo DiCaprio's girlfriend?"
Observation: DiCaprio met actor Camila Morrone in December 2017, when she was 20 and he was 43. They
were spotted at Coachella and went on multiple vacations together. Some reports suggested that
DiCaprio was ready to ask Morrone to marry him. The couple made their red carpet debut at the 2020
Academy Awards.
Thought: I need to calculate Camila Morrone's age raised to the 0.43 power.
Action: Calculator
Action Input: 21^0.43

> Entering new LLMMathChain chain...
21^0.43
```text
21**0.43
```
...numexpr.evaluate("21**0.43")...

Answer: 3.7030049853137306
> Finished chain.

Observation: Answer: 3.7030049853137306
Thought: I now know the final answer.
Final Answer: Camila Morrone is Leo DiCaprio's girlfriend and her current age raised to the 0.43
power is 3.7030049853137306.

> Finished chain.
~~~

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

How to get started on AWS?

❤️

aws

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Langchain ❤️ AWS: Orchestration Layer

JS

AWS Lambda

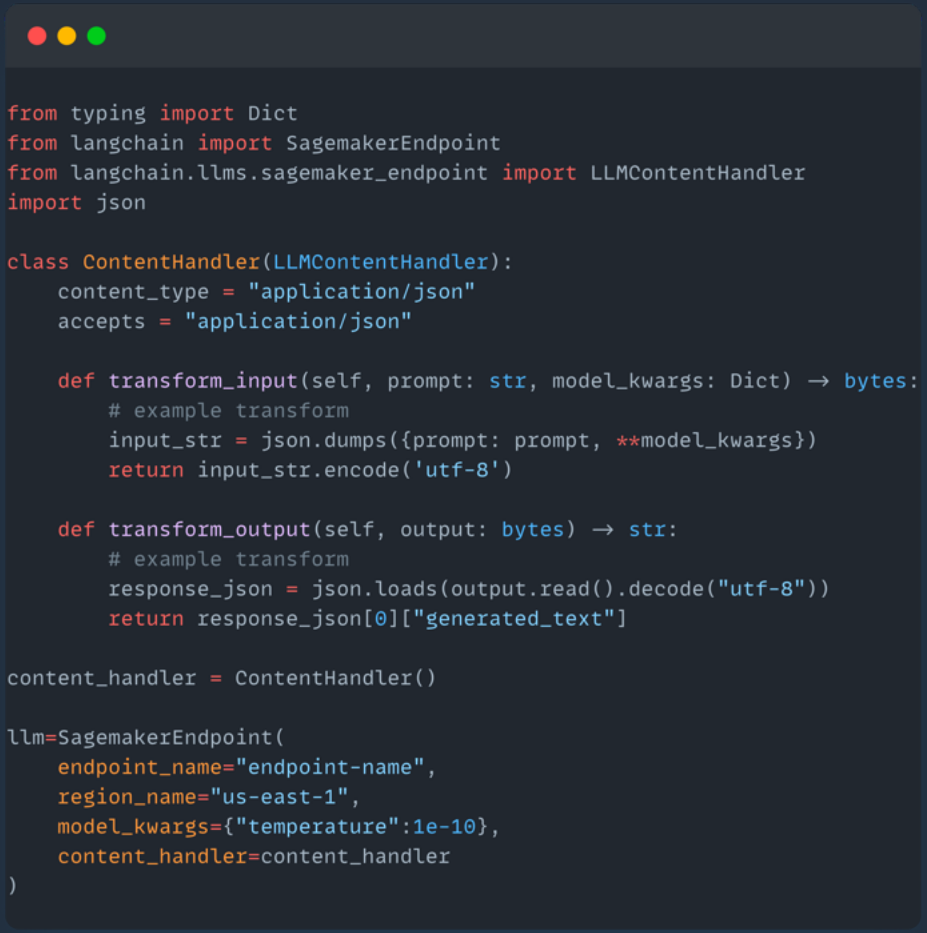
Amazon ECS

Amazon EC2

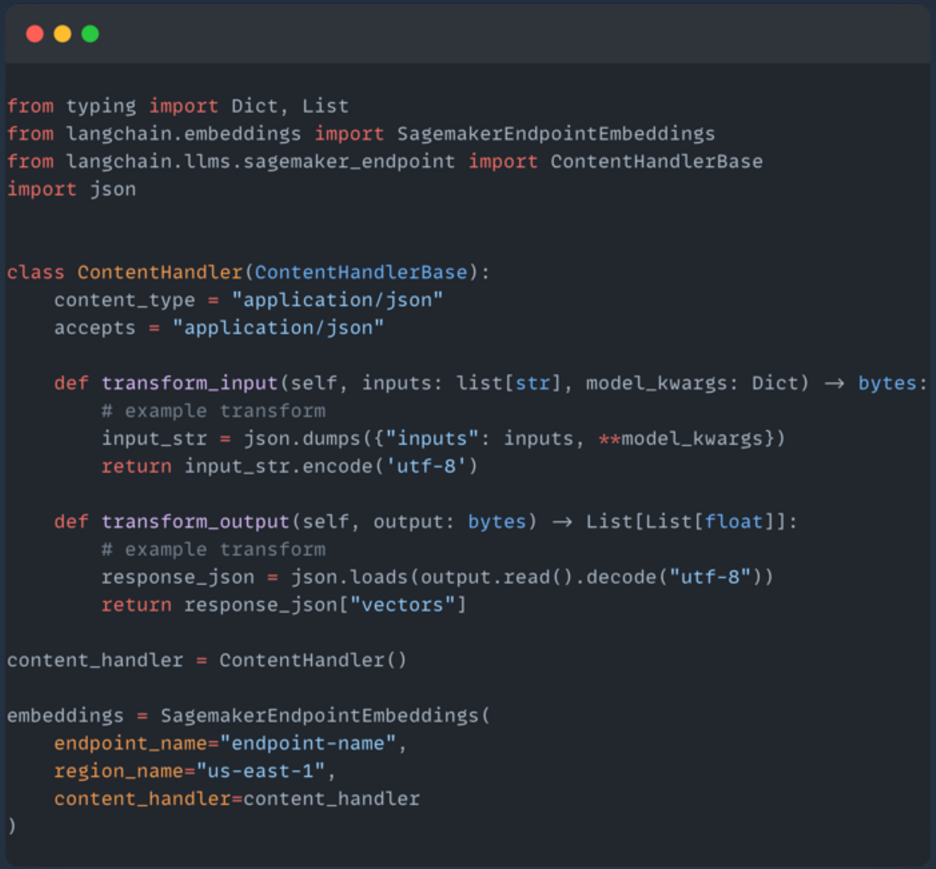
…

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

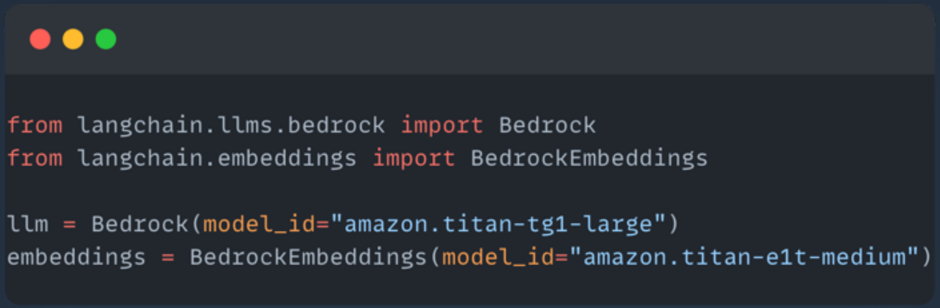
Langchain ❤️ AWS: Models



```python
from typing import Dict
from langchain import SagemakerEndpoint
from langchain.llms.sagemaker_endpoint import LLMContentHandler
import json

class ContentHandler(LLMContentHandler):
    content_type = "application/json"
    accepts = "application/json"

    def transform_input(self, prompt: str, model_kwargs: Dict) -> bytes:
        # example transform
        input_str = json.dumps({prompt: prompt, **model_kwargs})
        return input_str.encode('utf-8')

    def transform_output(self, output: bytes) -> str:
        # example transform
        response_json = json.loads(output.read().decode("utf-8"))
        return response_json[0]["generated_text"]

content_handler = ContentHandler()

llm=SagemakerEndpoint(
    endpoint_name="endpoint-name",
    region_name="us-east-1",
    model_kwargs={"temperature":1e-10},
    content_handler=content_handler
)
```



```python
from typing import Dict, List
from langchain.embeddings import SagemakerEndpointEmbeddings
from langchain.llms.sagemaker_endpoint import ContentHandlerBase
import json


class ContentHandler(ContentHandlerBase):
    content_type = "application/json"
    accepts = "application/json"

    def transform_input(self, inputs: list[str], model_kwargs: Dict) -> bytes:
        # example transform
        input_str = json.dumps({"inputs": inputs, **model_kwargs})
        return input_str.encode('utf-8')

    def transform_output(self, output: bytes) -> List[List[float]]:
        # example transform
        response_json = json.loads(output.read().decode("utf-8"))
        return response_json["vectors"]

content_handler = ContentHandler()

embeddings = SagemakerEndpointEmbeddings(
    endpoint_name="endpoint-name",
    region_name="us-east-1",
    content_handler=content_handler
)
```



```python
from langchain.llms.bedrock import Bedrock
from langchain.embeddings import BedrockEmbeddings

llm = Bedrock(model_id="amazon.titan-tg1-large")
embeddings = BedrockEmbeddings(model_id="amazon.titan-e1t-medium")
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Langchain ❤️ AWS: RAG

Amazon SageMaker Processing

…

Preprocessing

Amazon Kendra

Amazon OpenSearch Service

Amazon RDS Postgres

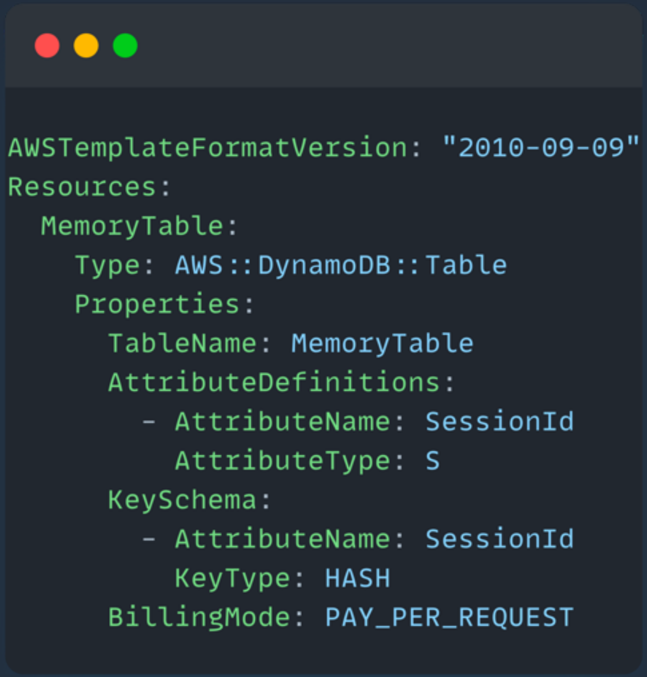
AWS Marketplace

…

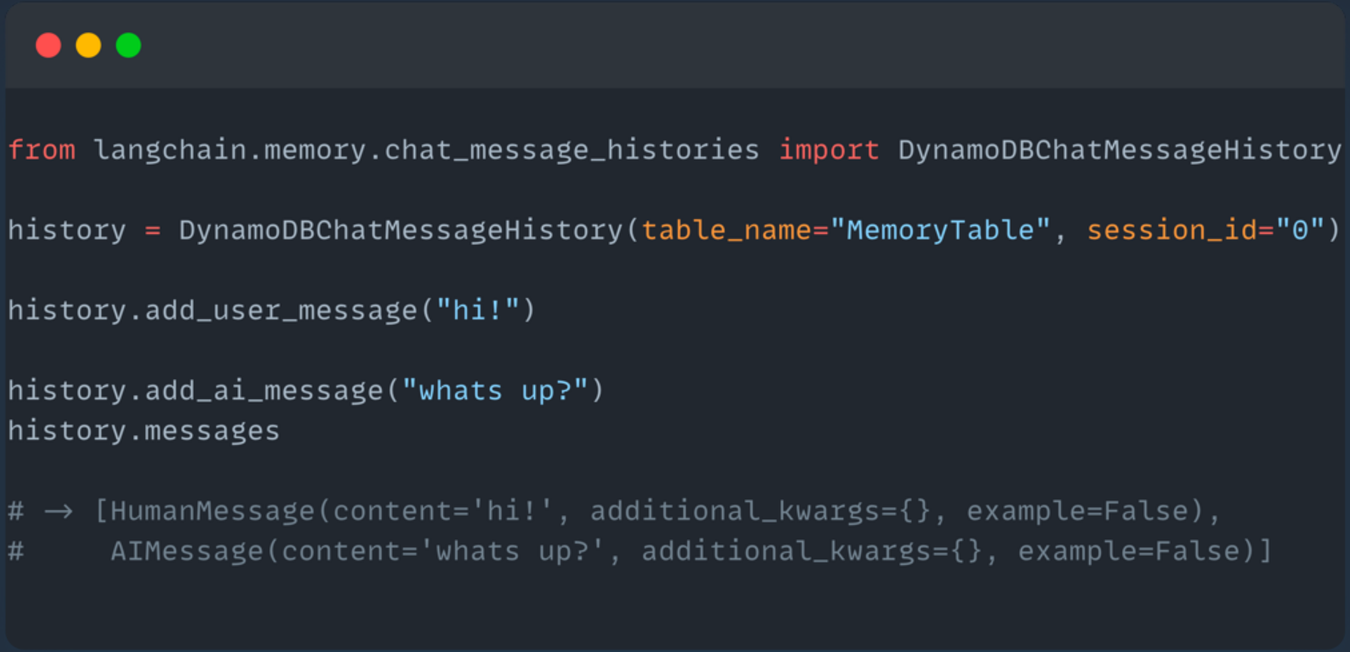
Document store & retrieval

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Langchain ❤️ AWS: DynamoDBChatMessageHistory



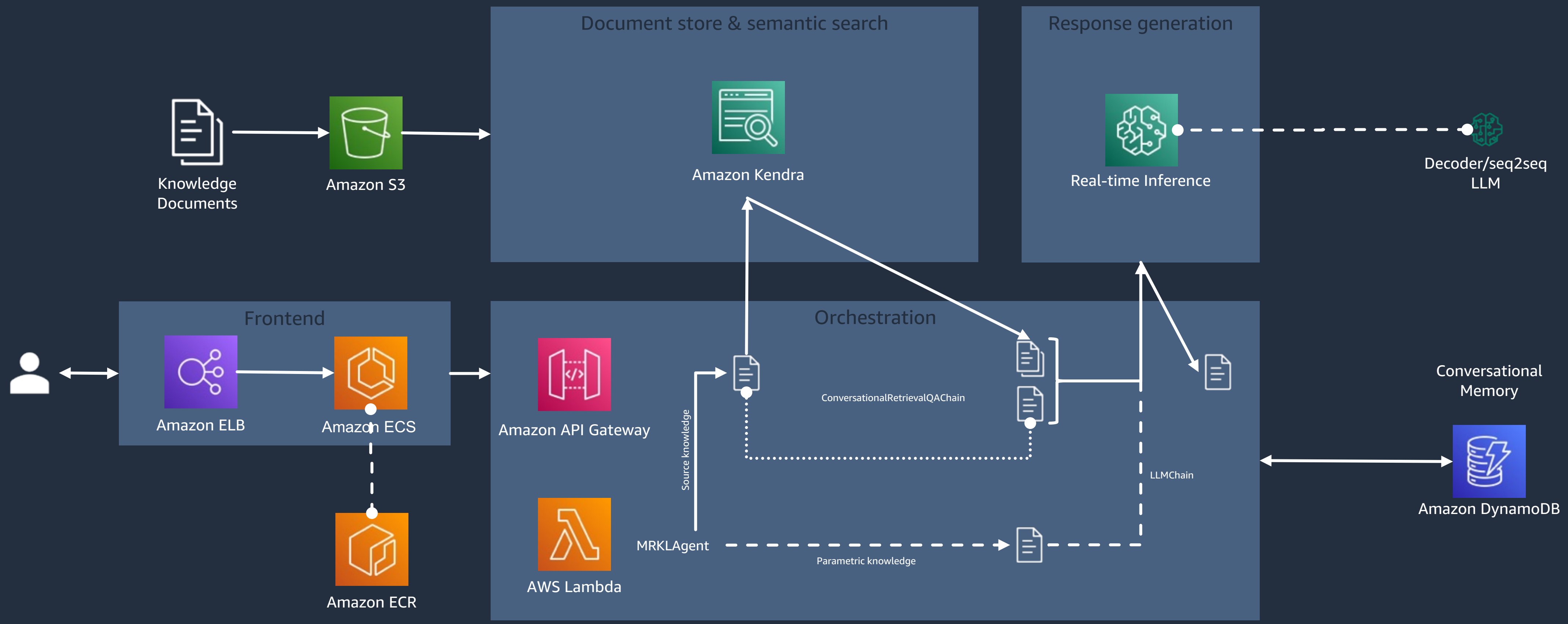
```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  MemoryTable:
    Type: AWS::DynamoDB::Table
    Properties:
      TableName: MemoryTable
      AttributeDefinitions:
        - AttributeName: SessionId
          AttributeType: S
      KeySchema:
        - AttributeName: SessionId
          KeyType: HASH
      BillingMode: PAY_PER_REQUEST
```



```python
from langchain.memory.chat_message_histories import DynamoDBChatMessageHistory

history = DynamoDBChatMessageHistory(table_name="MemoryTable", session_id="0")

history.add_user_message("hi!")

history.add_ai_message("whats up?")
history.messages

# → [HumanMessage(content='hi!', additional_kwargs={}, example=False),
#     AIMessage(content='whats up?', additional_kwargs={}, example=False)]
```

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Langchain ❤️ AWS: E2E Architecture



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Overview of Falcon40b-instruct model

- Decoder only model built by TII in the UAE
- Largest version of Falcon model family
- Fine-tuned on a mixture of Baize chat dataset mixed with RefinedWeb dataset
- Currently best open-source model available according to HuggingFace Open-LLM Leaderboard
- Available under Apache 2.0 license
- Available through SageMaker JumpStart (FP16)
- Optimized deployment with HuggingFace LLM DLC for SageMaker

Hyperparameters	Value
Parameters	40 billion
Layers	60

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Lab 4 – LLM-powered chatbot with RAG-capabilities and short-term memory

https://github.com/aristsakpinis93/generative-ai-immersion-day

Event Access Code:

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Thank you!

Aris Tsakpinis

tsaris@amazon.de

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.