



GENERATIVE AI IMMERSION DAY

# Implementing Generative AI in Organizations

Challenges and Opportunities

Aris Tsakpinis

AI/ML Specialist Solutions Architect

Heiko Hotz

AI/ML Sr. Specialist Solutions Architect

# Thanks for having us...



**Heiko Hotz**  
AI/ML Sr. Specialist Solutions  
Architect



**Aris Tsakpinis**  
AI/ML Specialist Solutions  
Architect



**Markus Schweier**  
AI/ML Sr. Specialist BDM

# TIMINGS

Time	Duration	Session
10:00-10:15	15min	Kickoff
10:15-11:15	1h	Generative AI 101 Large Language Models 101 LLM Deployment & Inference
11:15-12:15	1h	Lab 1: LLM Deployment & Inference
12:15-13:15	1h	Lunch
13:15-13:45	30min	LLM Finetuning
13:45-15:00	1h 15min	Lab 2: LLM Finetuning
15:00-15:15	15min	Coffee Break
15:15-16:00	45min	Visual Foundation Models 101 Stable Diffusion
16:00-17:15	1h 15min	Lab 3: Stable Diffusion Deployment & Inference
17:15-17:30	15min	Wrapup
17:30		Get together

# 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

# AGENDA

Generative AI – What is it and why the hype?

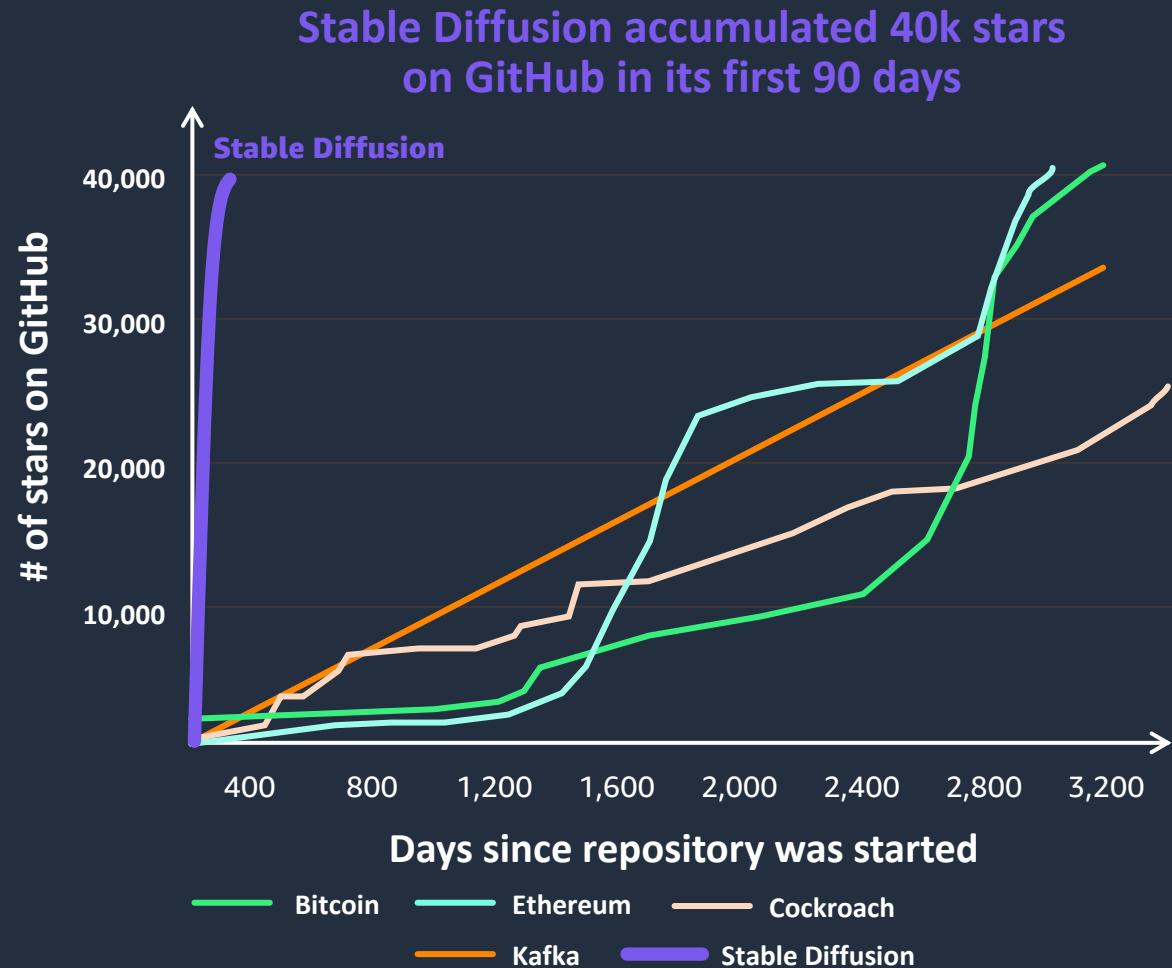
Large Language Models - How the ML works?

Large Language Model Hosting

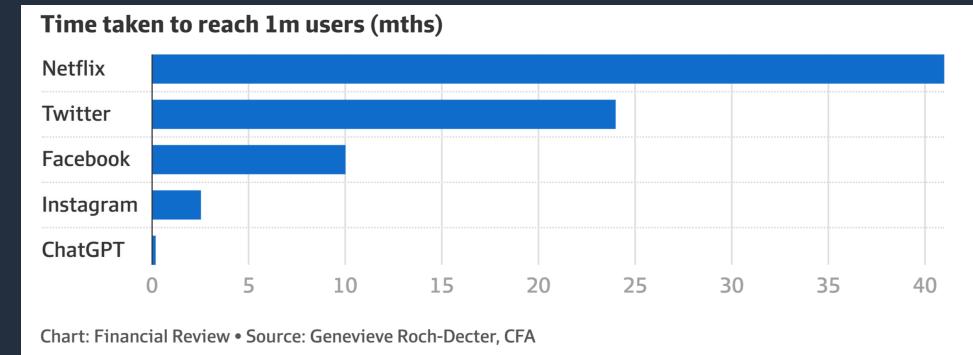
Large Language Model Finetuning

Visual Foundation Models & Stable Diffusion

# Generative AI is the fastest growing trend in AI



ChatGPT reached the 1 million users mark in just 5 days



# What is generative AI?



AI that can produce original content close enough to human generated content for real-world tasks



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



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



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



Reduces time and cost to develop ML models and innovate faster

# Generative AI – what's the perk?

Humans

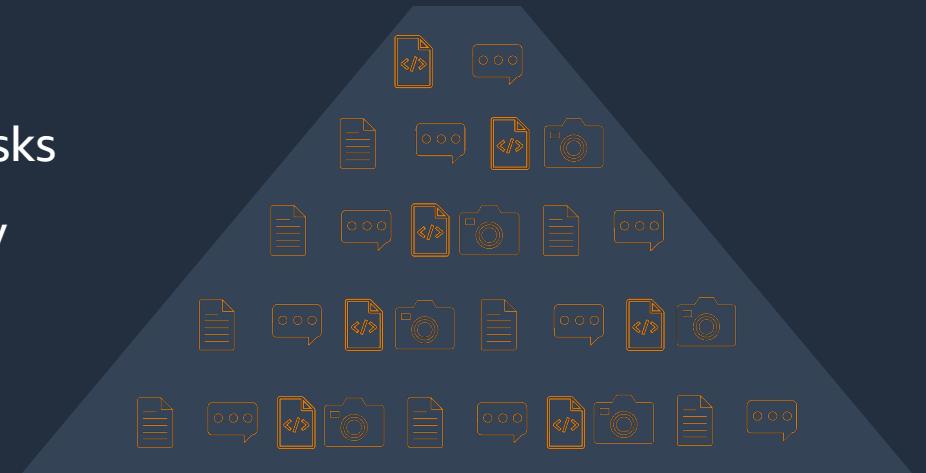


Speed & Breadth

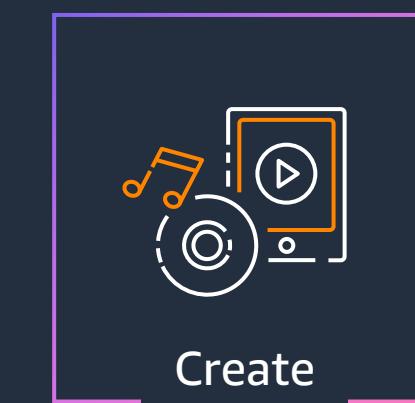
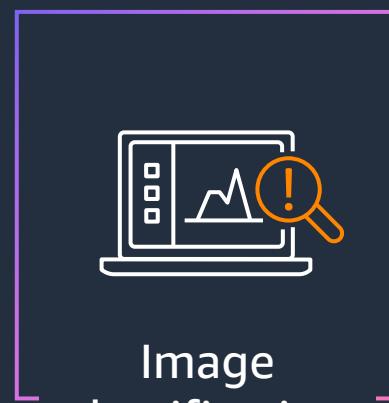
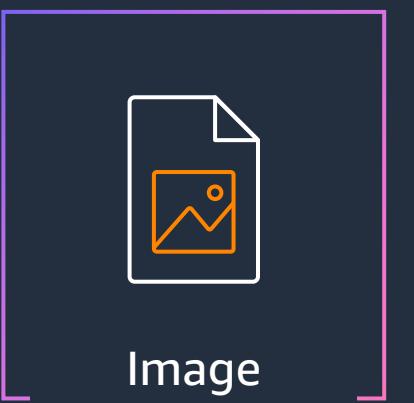
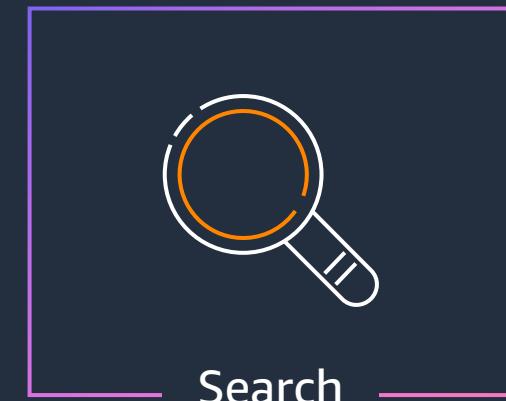
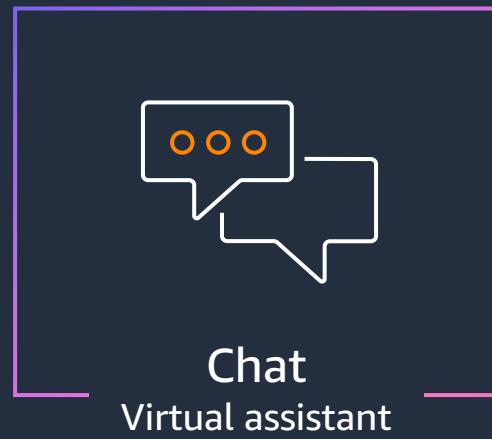
- Generates novel content
- No task specific models
- Applicable to a broad set of tasks
- Outputs easily interpretable by humans



Generative AI



# Generative AI is emerging across a range of domains ...



# ... 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?

G 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
- Lack of common sense
- Inability for (logical) reasoning
- Limited domain-specific knowledge

... but missed crucial aspects for enterprise applications

- Halluzination
- Knowledge cut-off
- Potential need for content moderation
- Missing references to original sources
- No explainability (as of March 2023)
- ...

# One approach to rule them all...?

If you want to ...

... automatically **extract information from documents**, such as e-mails, forms, invoices, ...

... **make information** contained in documents, audio recordings, or videos easily **searchable**

... wants to **convert audio files** into text

... wants to **translate text**

... wants to **build a chat application** (with execution of scripted workflows based on identified user intent)



... you might want to explore



Amazon Textract



Amazon Kendra



Amazon Transcribe



Amazon Translate



Amazon Lex

Machine learning service that automatically extracts text, handwriting, and data from scanned documents.

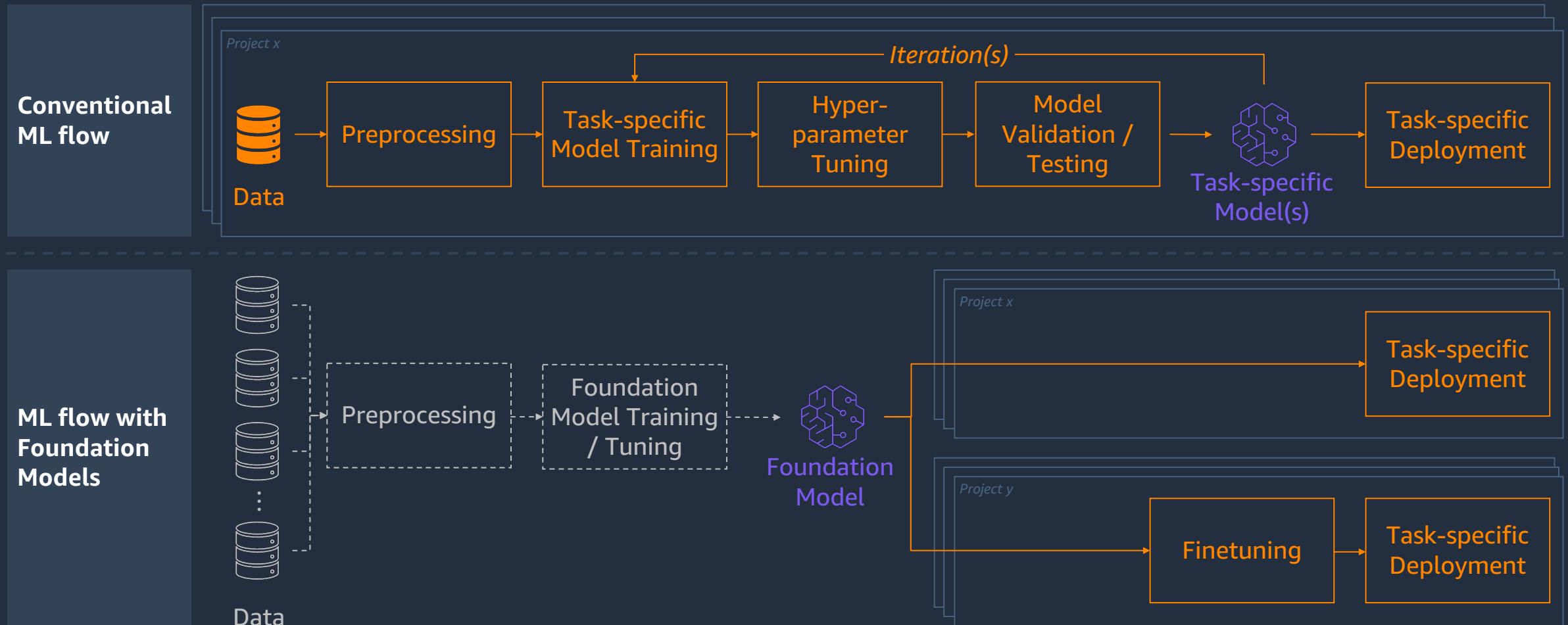
Intelligent enterprise search service that helps you search across different content repositories with built-in connectors.

Automatic speech recognition service that uses machine learning models to convert audio to text.

Neural machine translation service that delivers fast, high-quality, affordable, and customizable language translation.

Fully managed artificial intelligence service with advanced natural language models to design, build, test, and deploy conversational interfaces in applications.

# Foundation models – A paradigm change in the ML flow



# 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 <u>or</u> 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



# How to get started with Gen AI on AWS?

Option 1: Generative AI as a service

Proprietary

Option 2: SageMaker Jumpstart foundation model hub

Open-source

Option 3: SageMaker & open-source model hubs -  
Example: SageMaker built-in HuggingFace integration

# AGENDA

Generative AI – What is it and why the hype?

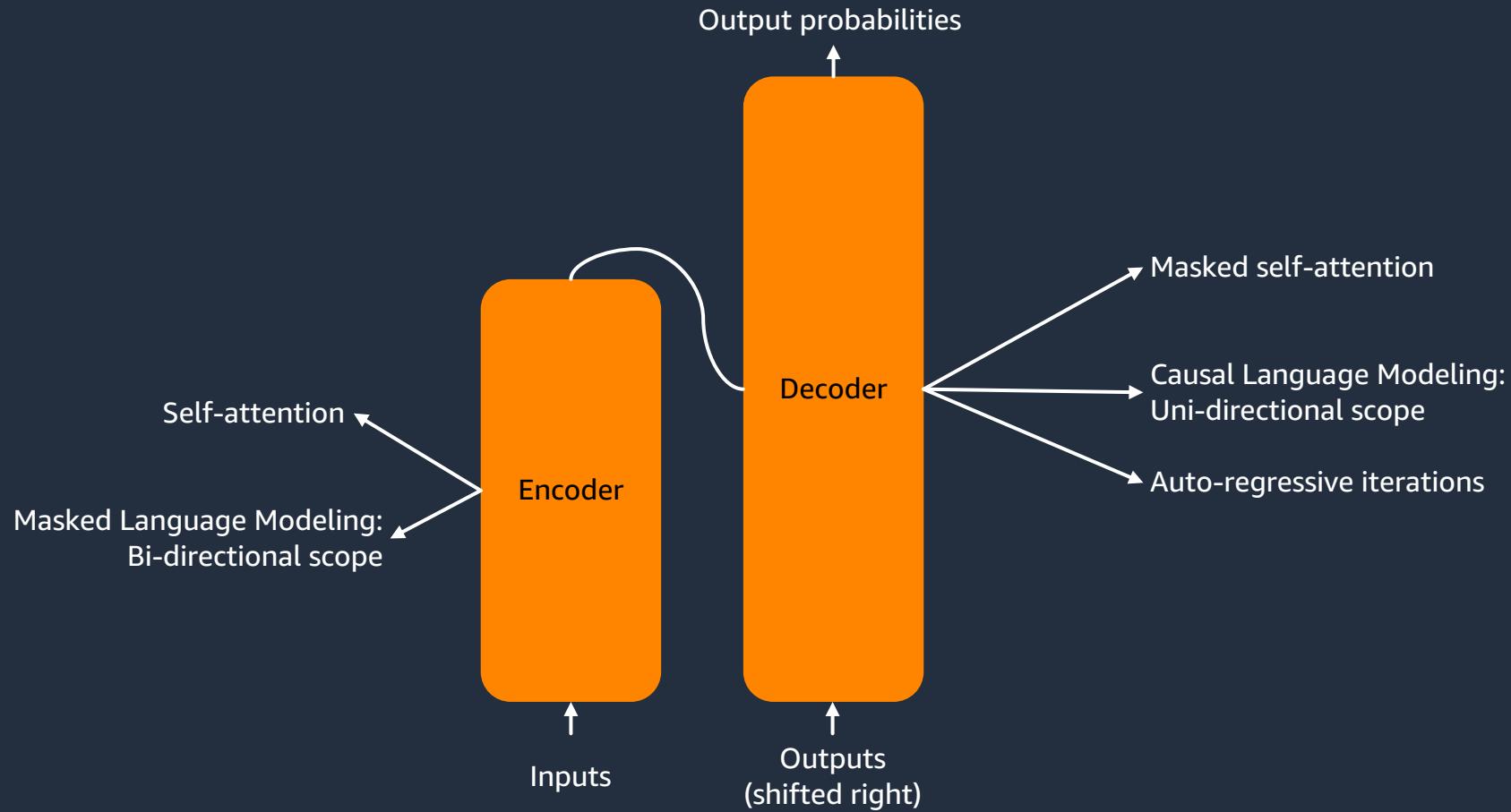
Large Language Models - How the ML works?

Large Language Model Hosting

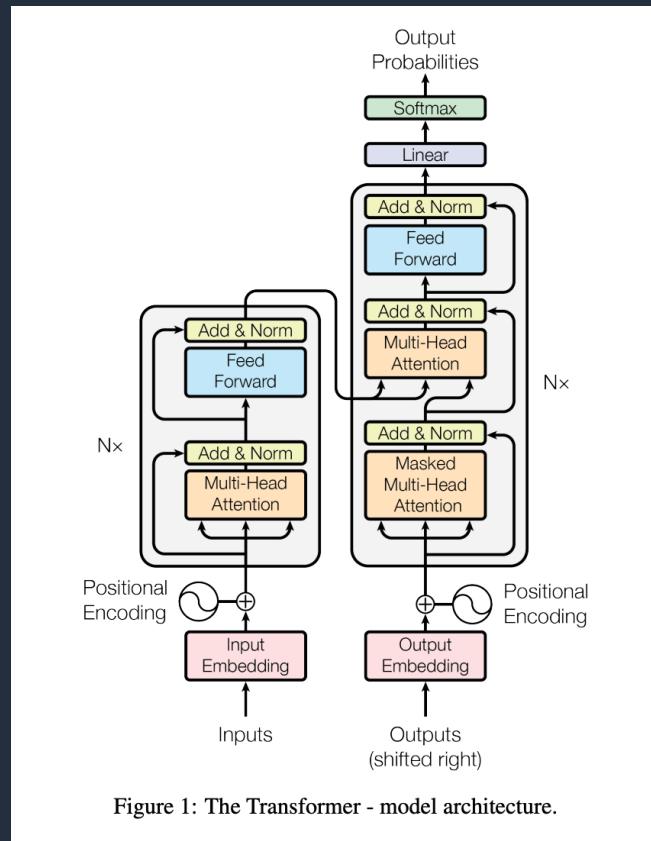
Large Language Model Finetuning

Visual Foundation Models & Stable Diffusion

# Transformer Models – Core characteristics of sequence-to-sequence transforms



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



## Model architecture

### Encoder models

## Common use cases

- Sentence classification
- Named Entity Recognition

## Examples

BERT

### Decoder models

## Common use cases

- Text generation

## Examples

GPT

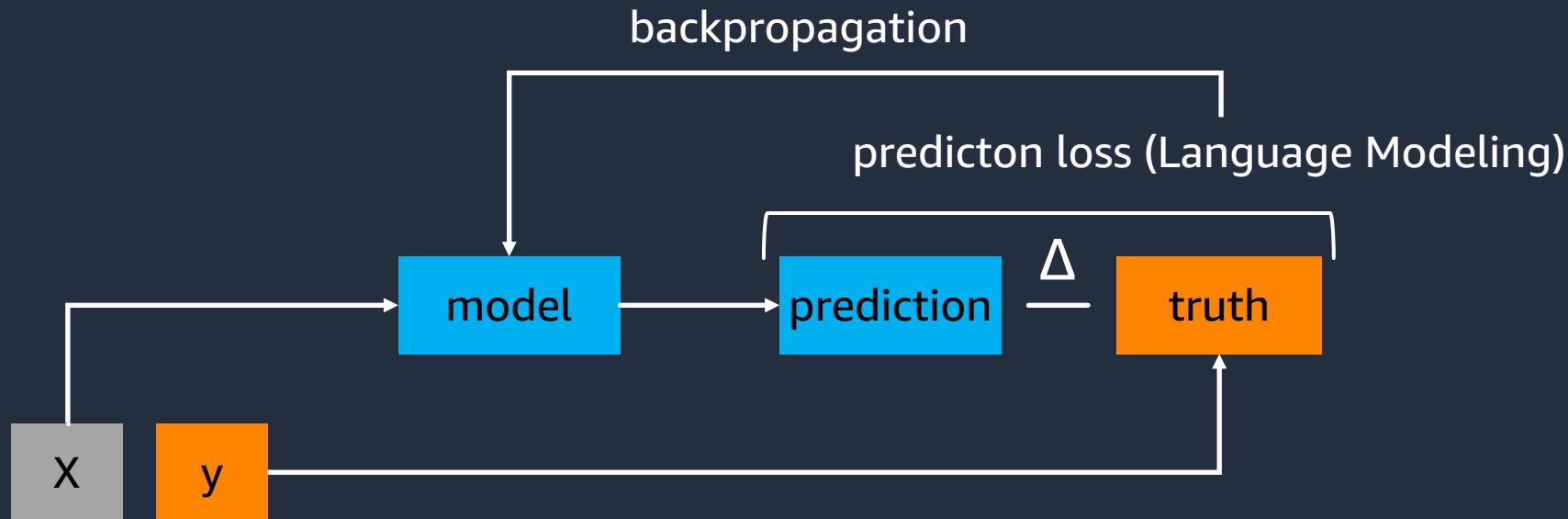
### Encoder-Decoder models

- Summarization
- Translation
- Question answering

## Examples

BART, T5

# How are transformer models trained?



Berlin is the capital of

Switzerland



# Language Modeling Variations

## Masked Language Modeling (MLM)

Berlin is the [ ] of

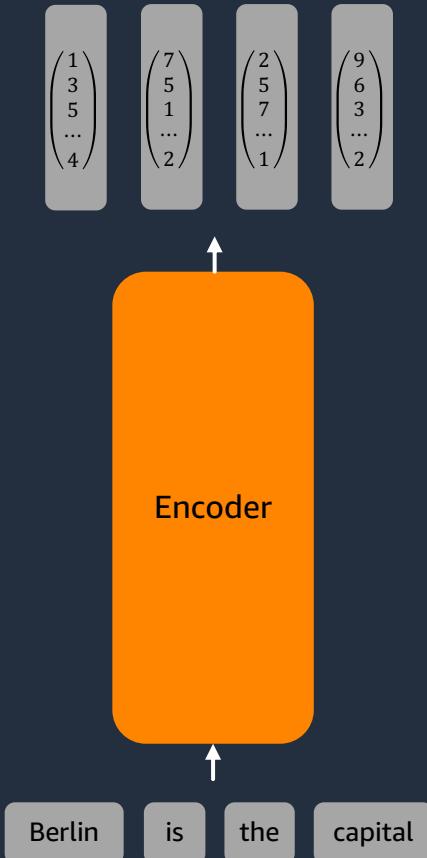
## Causal Language Modeling (CLM)

Berlin is the capital [ ]

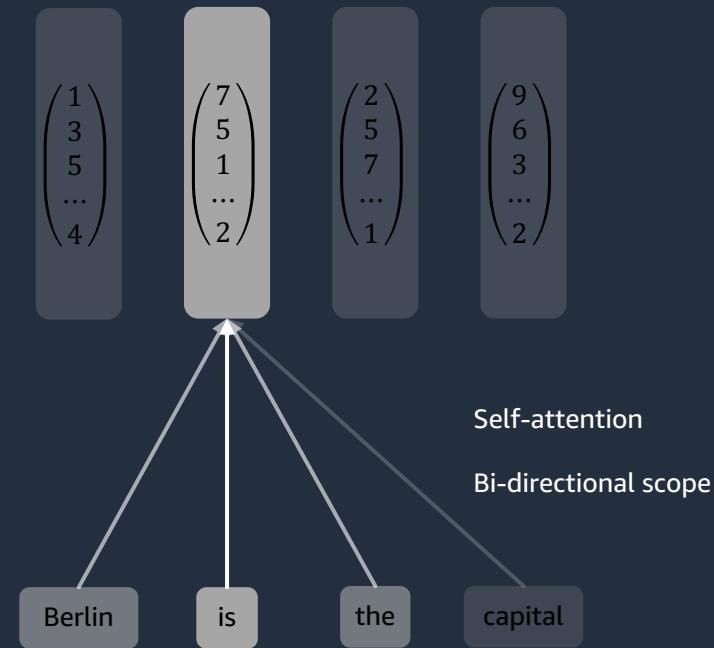
## Permutation Language Modeling (PLM)

of capital Berlin the is  
5 4 1 3 2

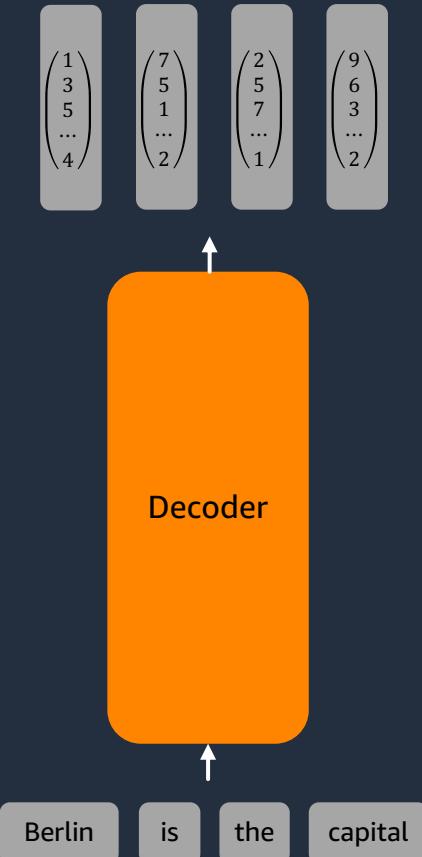
# Transformer Models – How Encoders work



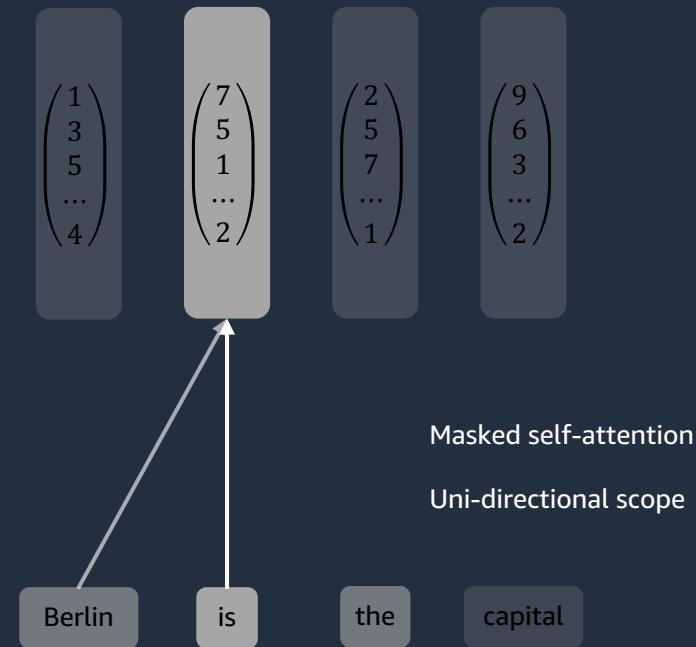
## Masked Language Modeling (MLM)



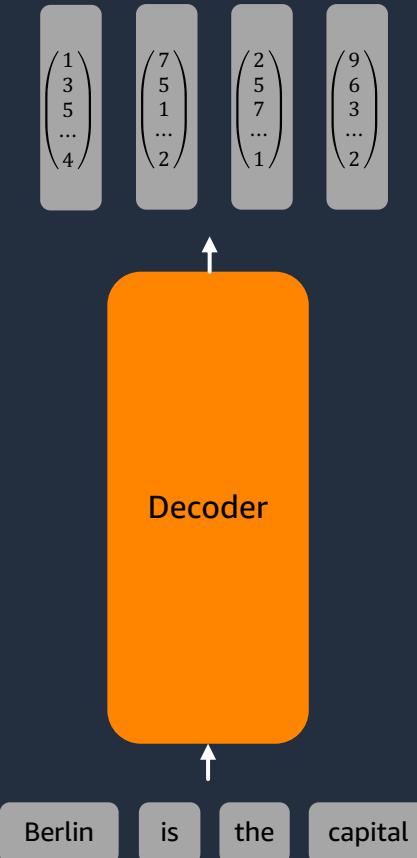
# Transformer Models – How Decoders work



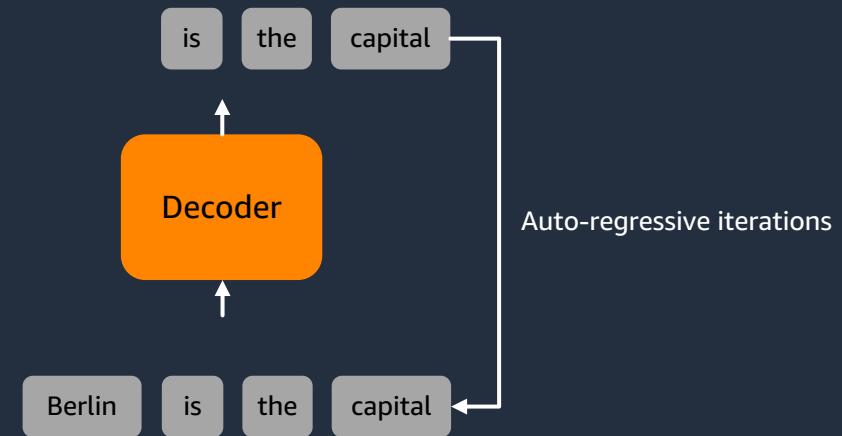
Causal Language Modeling (CLM)



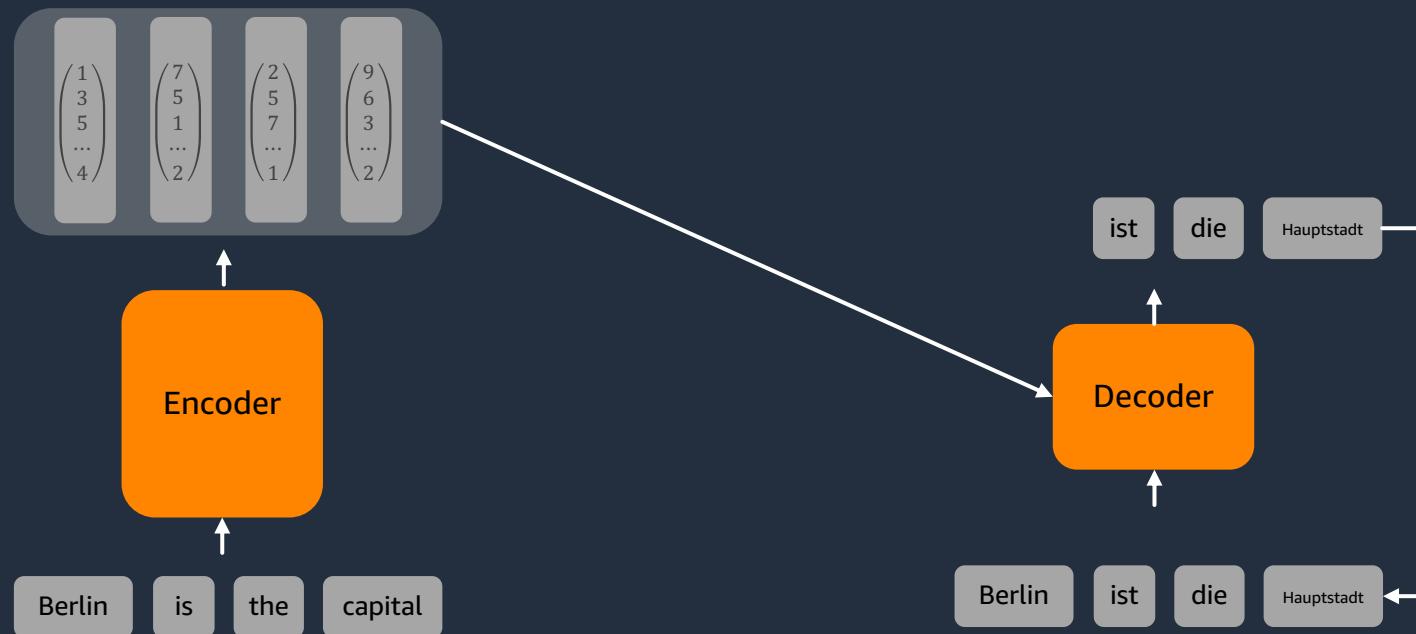
# Transformer Models – How Decoders work



## Causal Language Modeling (CLM)

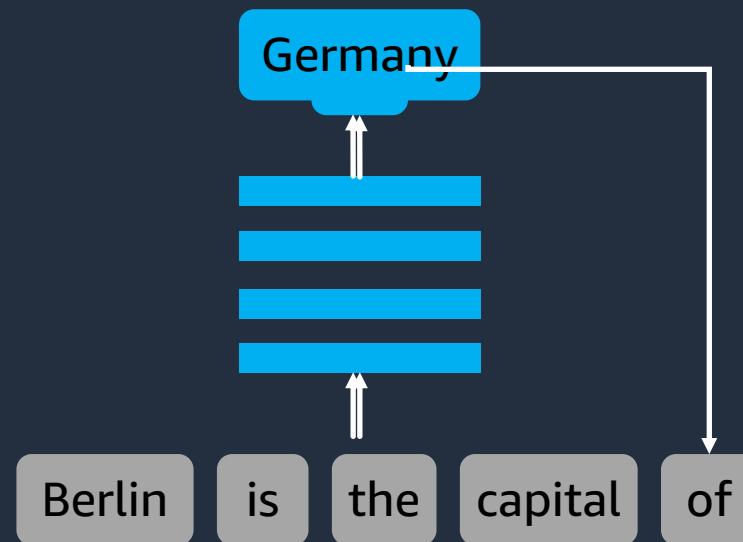


# Transformer Models – How Encoders and Decoders work together



# How do (decoder) LLMs make predictions?

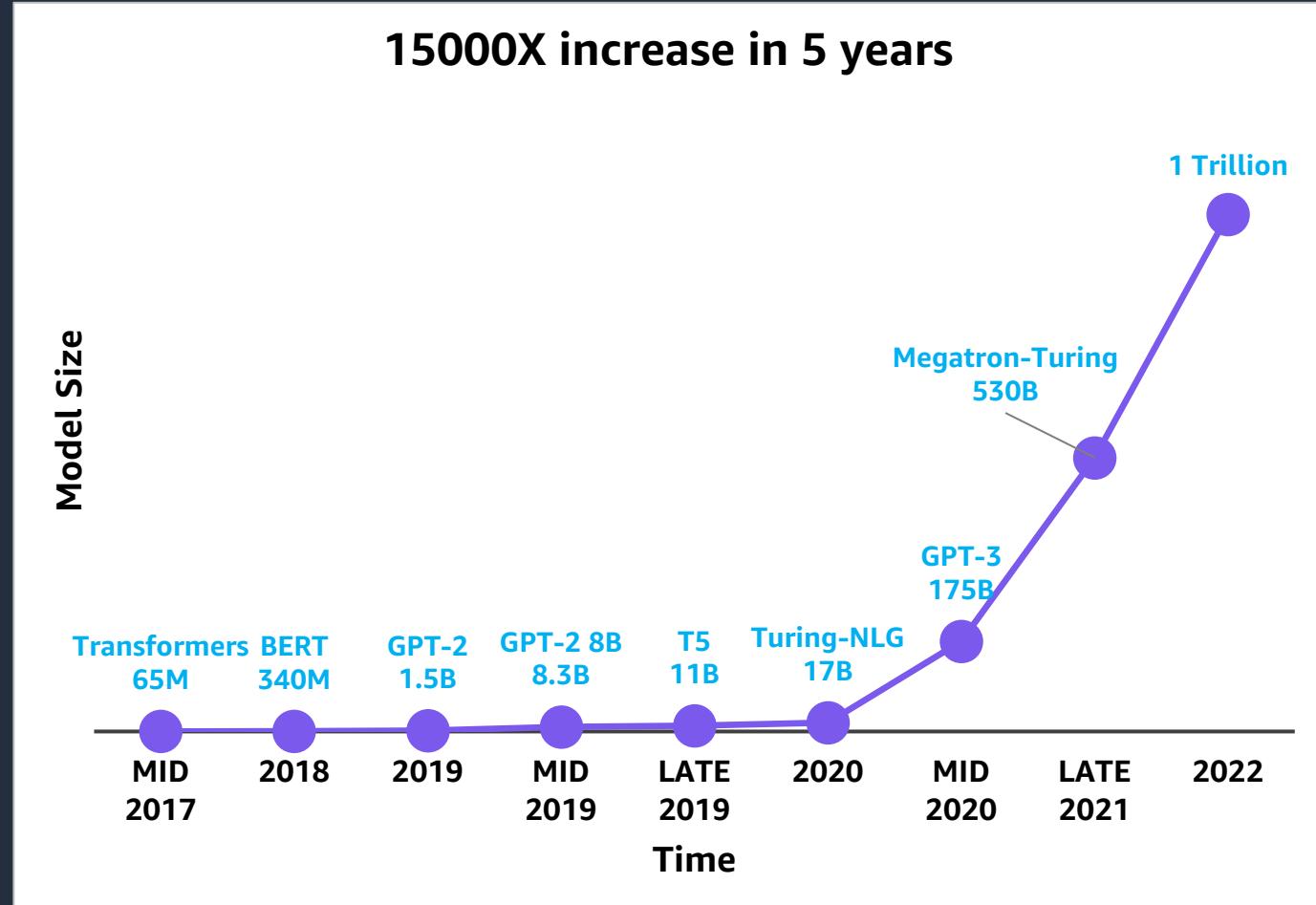
Auto-regressive word-by-word prediction



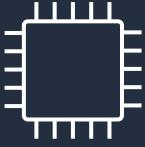
# Key trends in AI/ML

Models are becoming more complex, with end users moving from classical ML to deep learning

State-of-the-art deep learning models are getting larger and larger, as we find that larger models generalize better



# However, when it comes to training and deploying foundation models, there are challenges...



## Hardware



## Health checks



## Orchestration



## Data



## Scaling up



## Cost

# 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

# Large model hosting challenges

## Performance

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



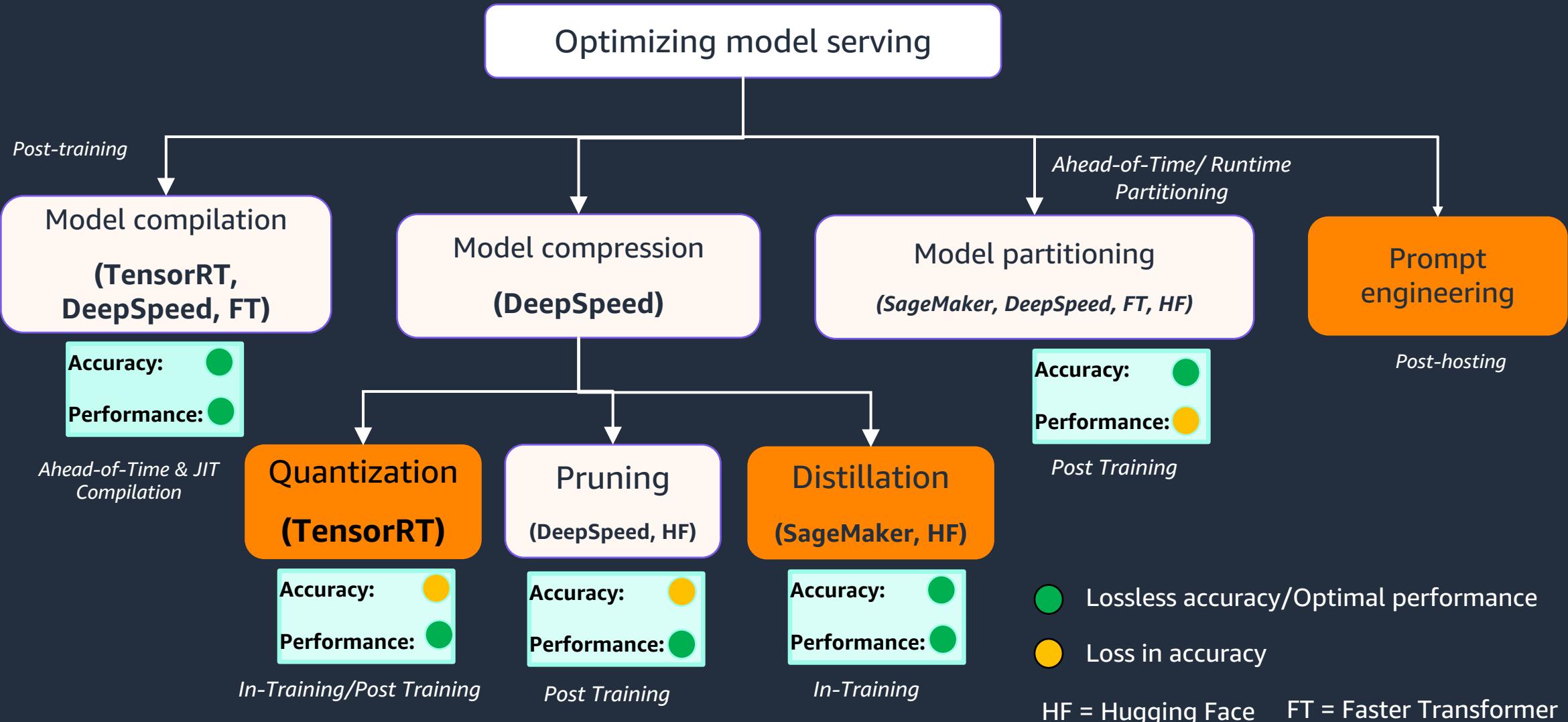
## Complexity

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

## Cost

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

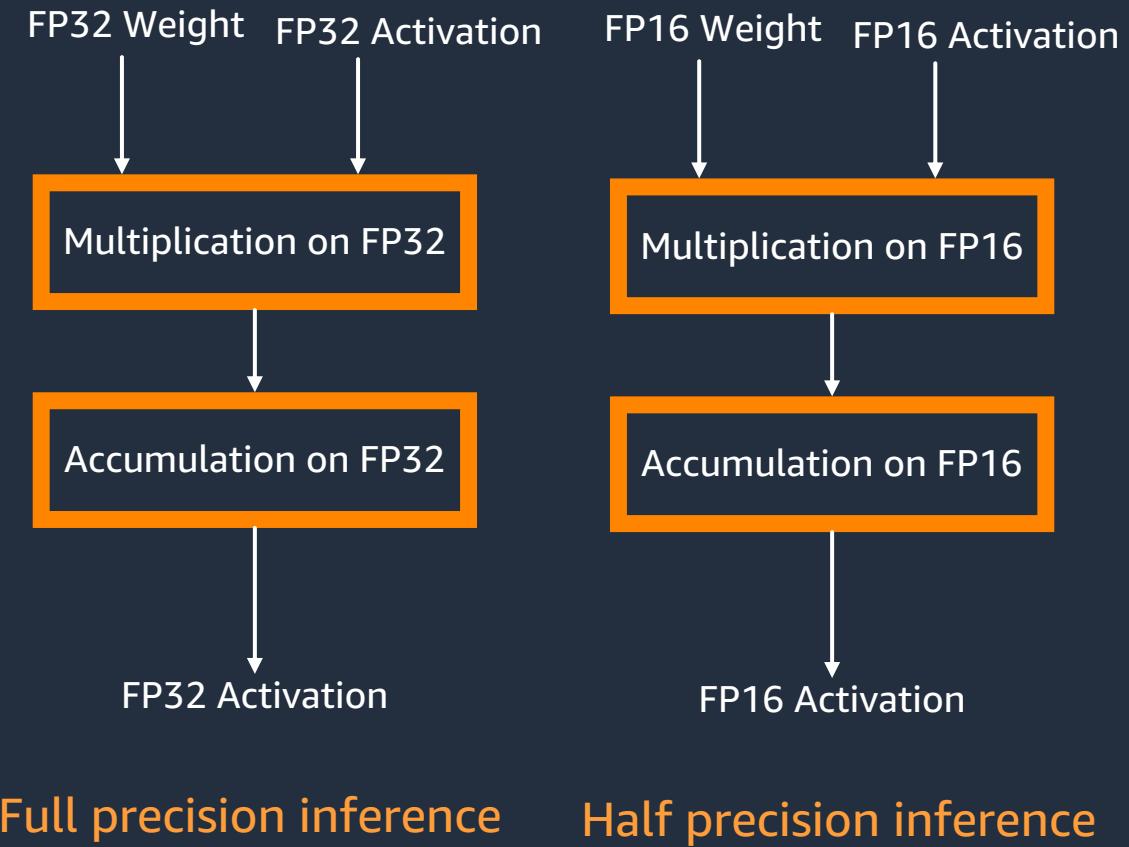
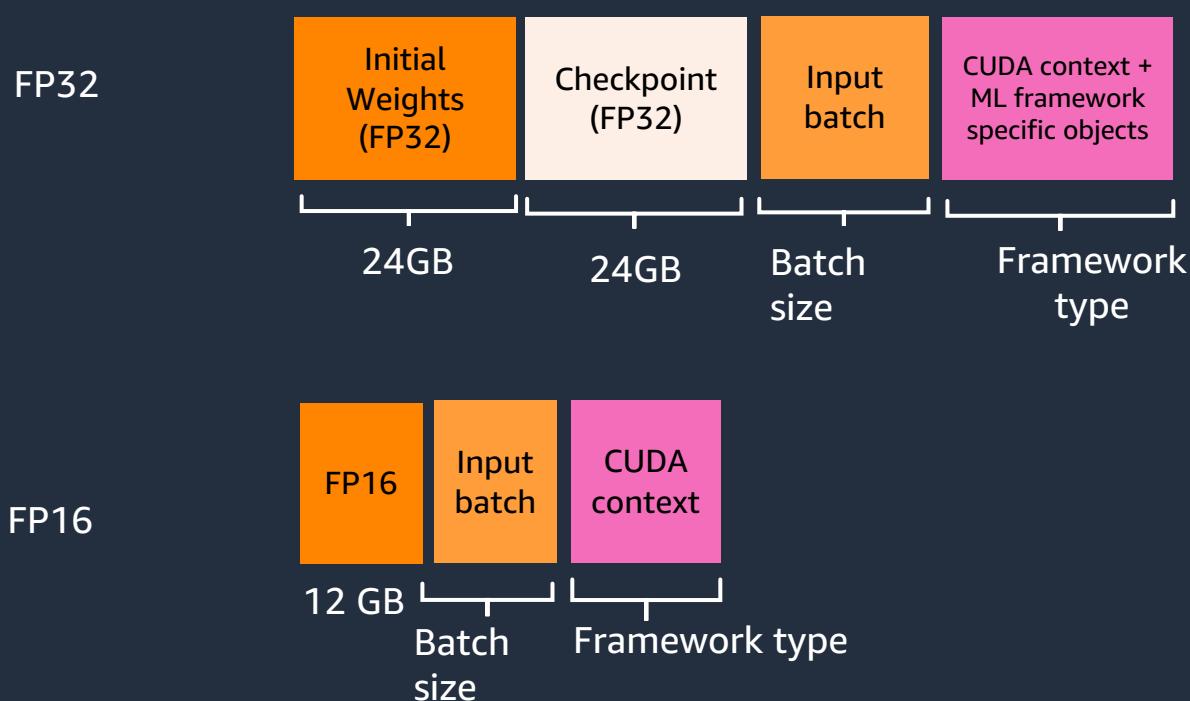
# Large model inference optimization



# Quantization

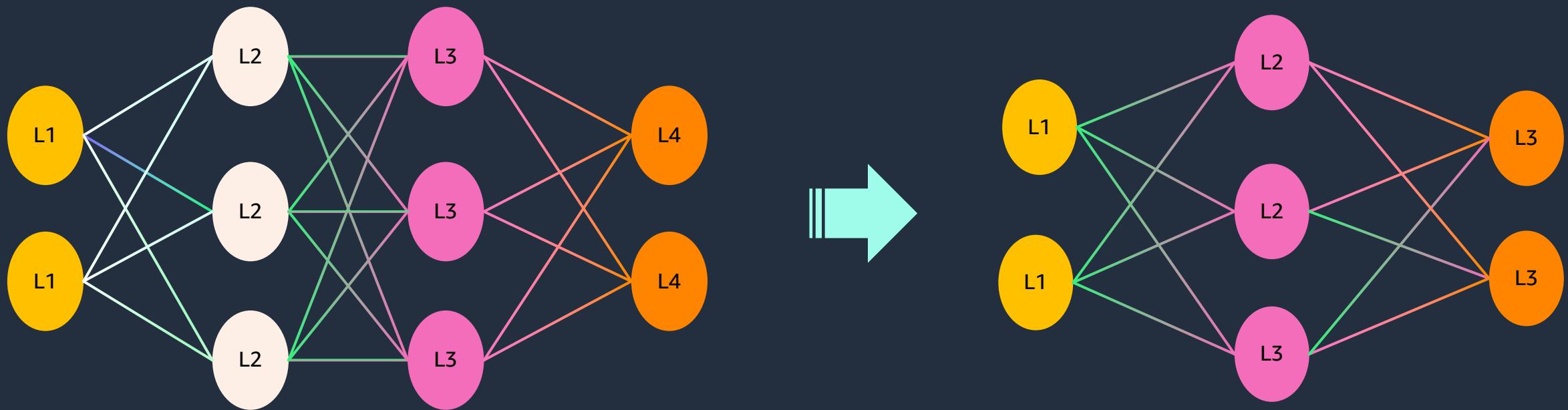
Example: GPT-J-6B (FP16)

**Model serving:**



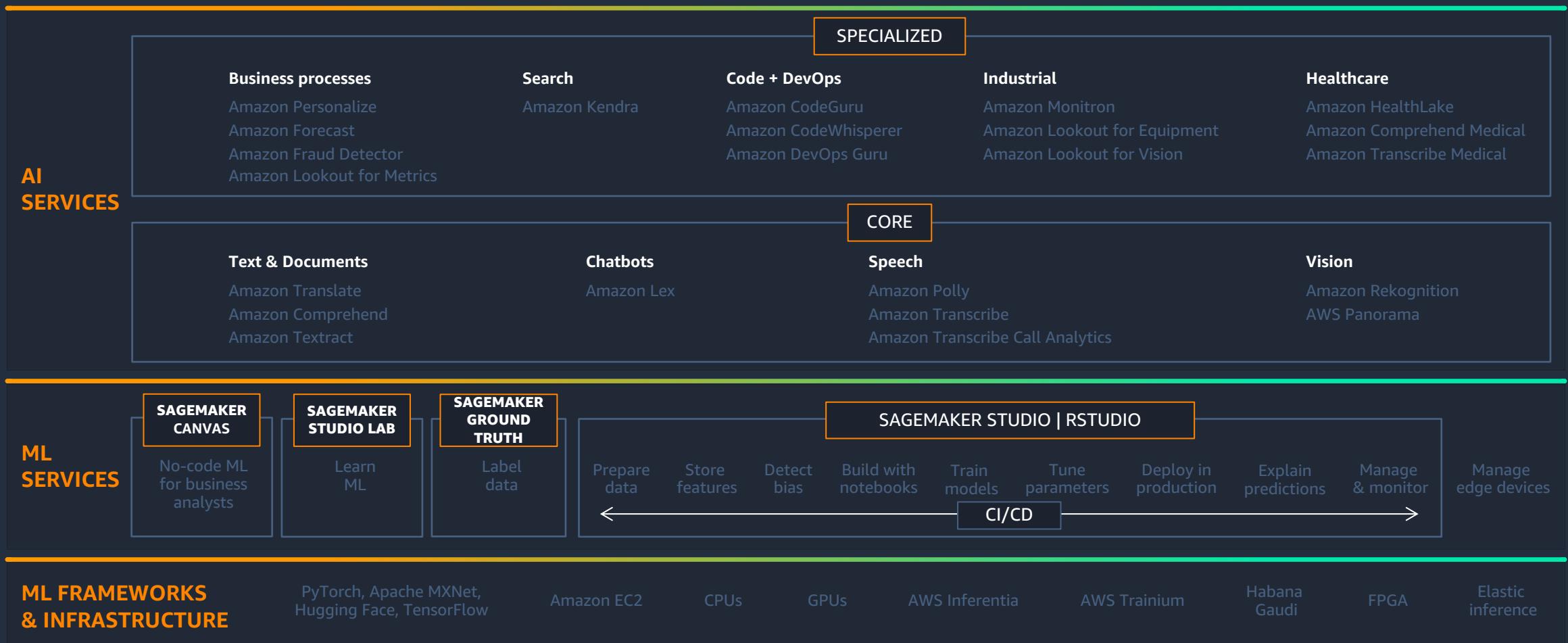
# Knowledge Distillation

EXAMPLE: DISTILGPT2

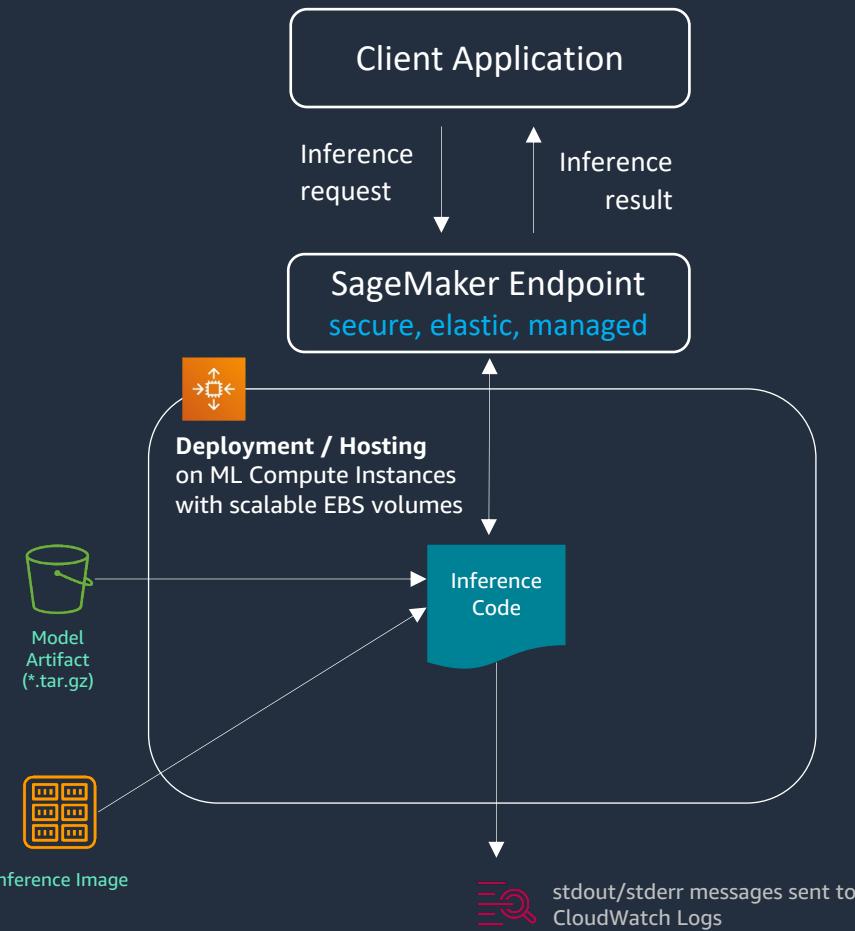


# The AWS ML stack

BROADEST AND MOST COMPLETE SET OF MACHINE LEARNING CAPABILITIES



# How SageMaker Inference uses Docker containers



How serving containers are run by SageMaker

```
docker run [Image] serve
```

Container reserved paths

```
/opt/ml
|---/model ← Model artifact copied from S3
```

Container inference (web) server

*Containers provided by SageMaker come with pre-installed web servers for serving invocation requests, as well as /ping (for health check purposes)*

timeout

2s

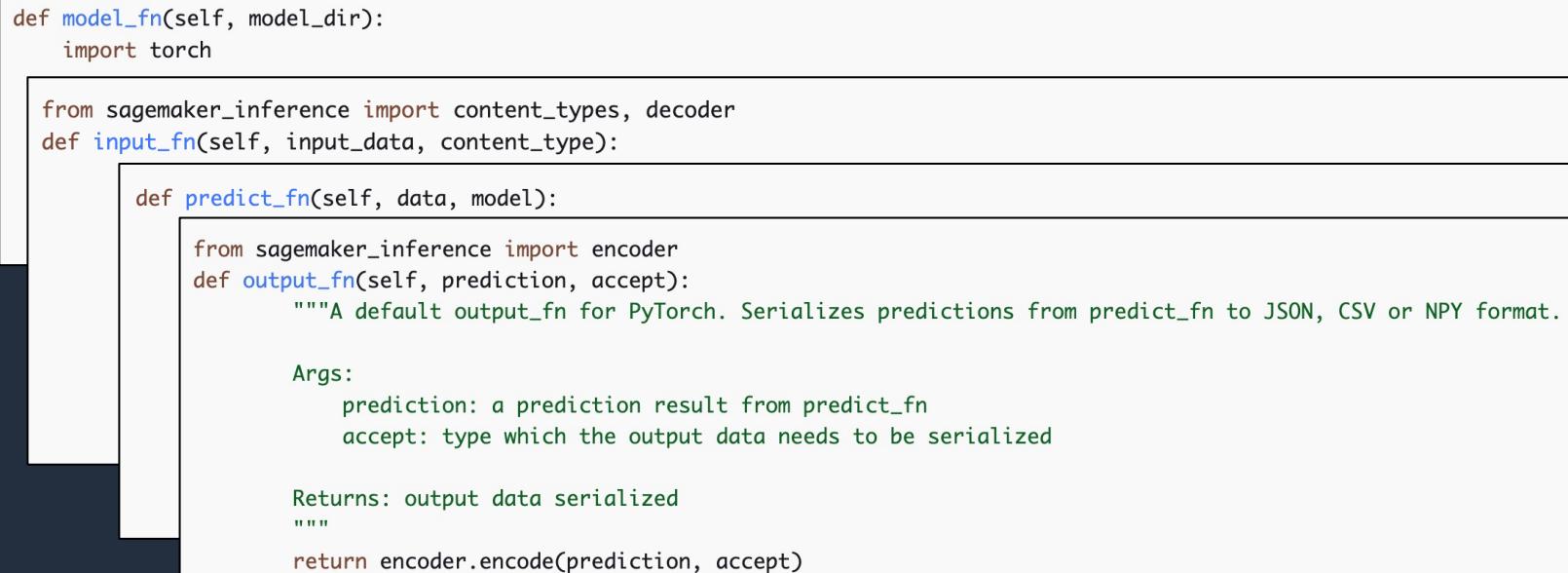
60s

/ping:8080

/invocations:8080

# Customizing SageMaker Inference toolkit lifecycle

```
model.tar.gz/  
|- pytorch_model.bin  
|- ....  
|- code/  
| - inference.py  
| - requirements.txt
```



```
def model_fn(self, model_dir):  
    import torch  
  
    from sagemaker_inference import content_types, decoder  
    def input_fn(self, input_data, content_type):  
  
        def predict_fn(self, data, model):  
  
            from sagemaker_inference import encoder  
            def output_fn(self, prediction, accept):  
                """A default output_fn for PyTorch. Serializes predictions from predict_fn to JSON, CSV or NPY format.  
  
                Args:  
                    prediction: a prediction result from predict_fn  
                    accept: type which the output data needs to be serialized  
  
                Returns: output data serialized  
                """  
  
                return encoder.encode(prediction, accept)
```

# A strong collaboration to make NLP easy and accessible for all

Hugging Face



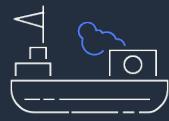
Hugging Face is the most popular open source company providing state-of-the-art NLP technology

AWS



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

# 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



**An example gallery** to find readily usable high-quality samples of Hugging Face scripts on Amazon SageMaker



**Maintained** and supported by AWS

# Large Model Inference (LMI) container

Large ML models  
with 100 billion + parameters

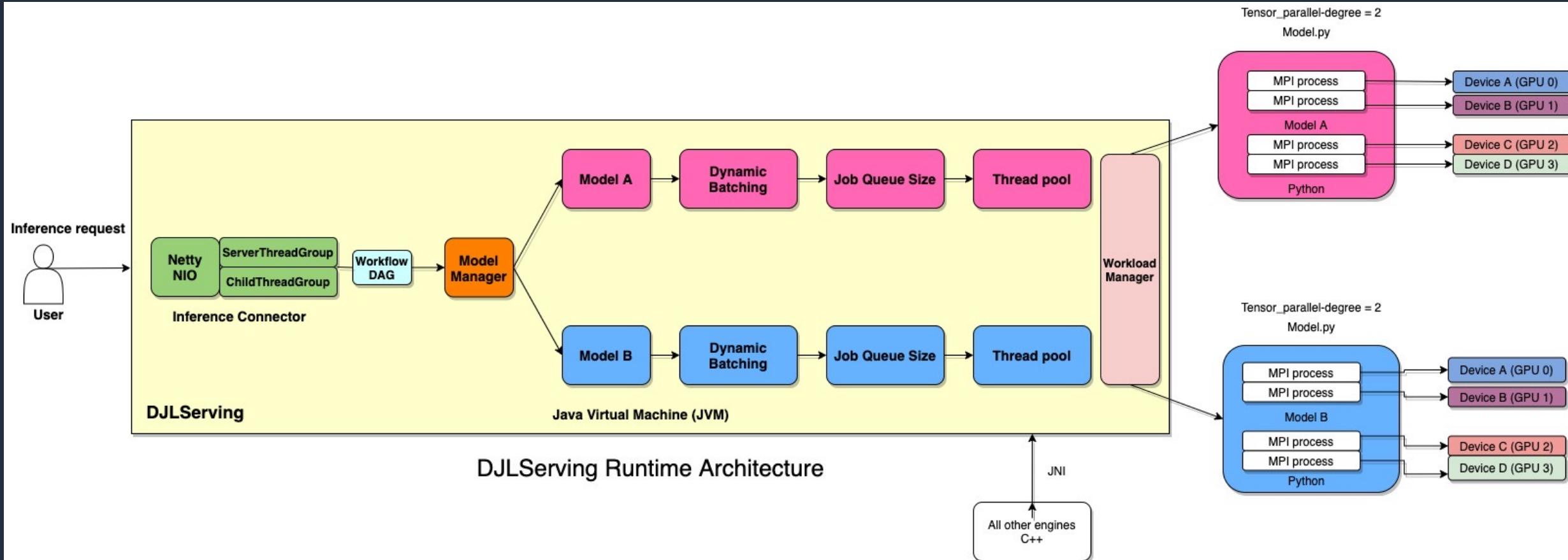


Easily parallelize models across multiple GPUs to fit models into the instance and achieve low latency

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

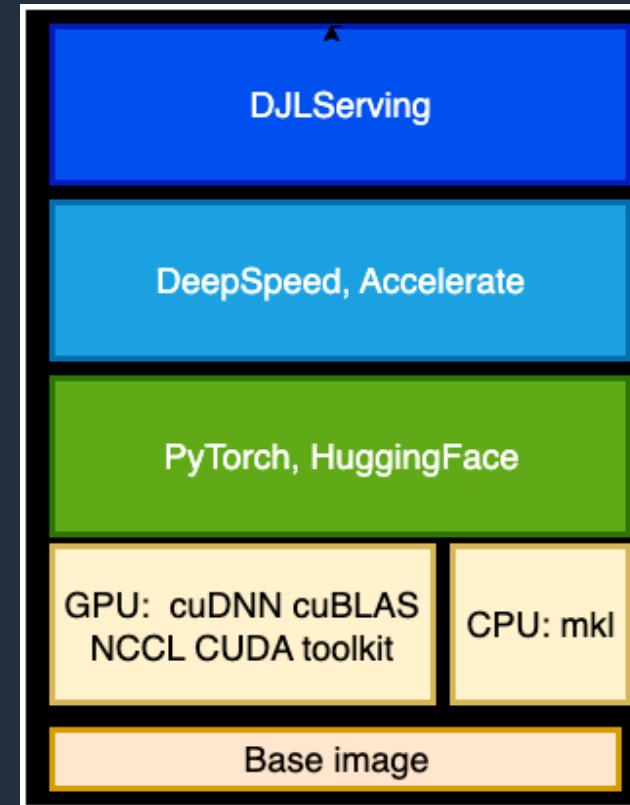
Leverage 500GB of Amazon EBS volume per endpoint

# DJLServing using DeepSpeed on SageMaker



# Large Model Inference Container

- Zero code setup: [DeepSpeed](#), [StableDiffusion](#) 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



# SageMaker model deployment stack

## Amazon SageMaker



Real-time inference    Async inference    Serverless inference    Batch inference    Multi-model endpoints

## SAGEMAKER STUDIO IDE

Multi-container endpoints    Inference DAG and pipelines  
SageMaker JumpStart

Manage and version models    Large model inference containers    Model monitoring    Metrics and logging in CloudWatch

## FRAMEWORKS



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



# LLM inference – Zero shot learning

cohere

## 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?

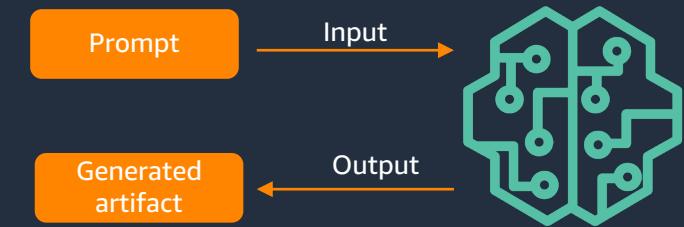
▶ Generate text

### 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 by adding context

cohere

**Cohere Generate Model - Medium**

By Cohere

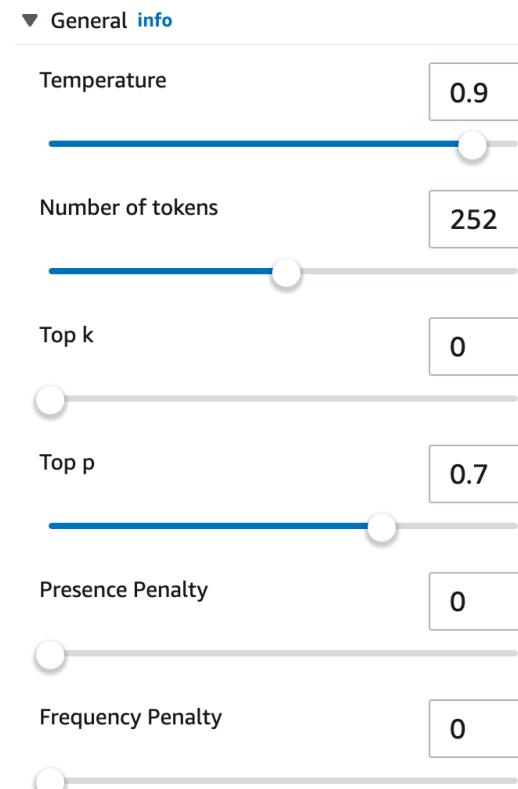
Try a product demo of the capabilities of this model from Cohere. Do not upload any confidential or sensitive information. Use of this feature is for demonstration purposes only. This demo may not accurately represent the actual response times of the product.

**Prompt**

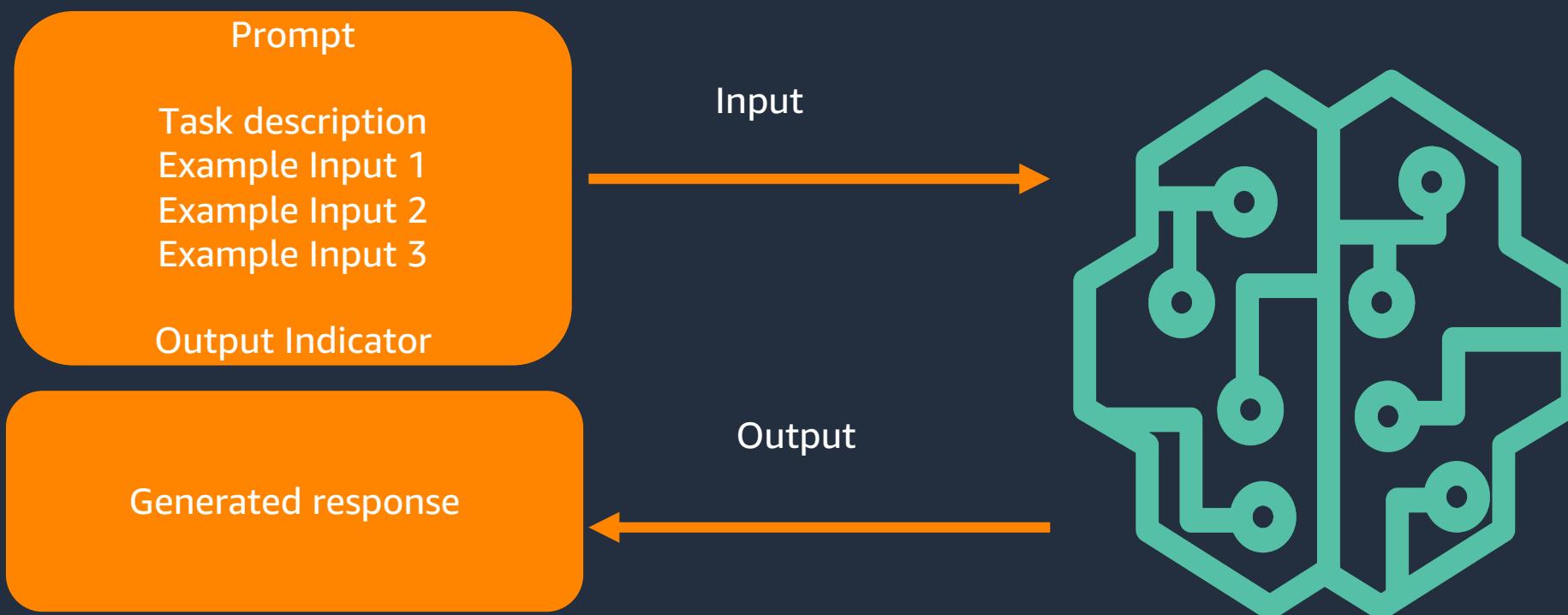
Question: I have a degree in data science. How do I earn more money?

**► Generate text****Output**

Answer: In general, I think the main difference between someone with a CS degree and someone with a degree in data science is that a CS degree will have more (technical) programming skills. I would suggest you look at:  
<https://www.kaggle.com/jobs> <https://www.data-science.com/articles/data-science-careers-and-salaries>  
<https://www.gigaom.com/data-science-careers-and-salaries/> <https://www.quora.com/What-is-the-salary-for-data-scientist-jobs> <https://www.accenture.com/us-en/how-to-create-jobs-that-pay-more-diversity-and-inclusion-with-diversity-and-inclusion-guide> <https://www.theatlantic.com/technology/archive/2015/07/data-scientists-computer-science-talent/391276/> <http://www.usnews.com/best-jobs/data-scientist-bachelor-s-in-computer-science-2016-data-scientist-salary-data-science-jobs>



# LLM inference – Improving performance with few shot learning



# 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

Task Description

Examples

Input



Output

Positive

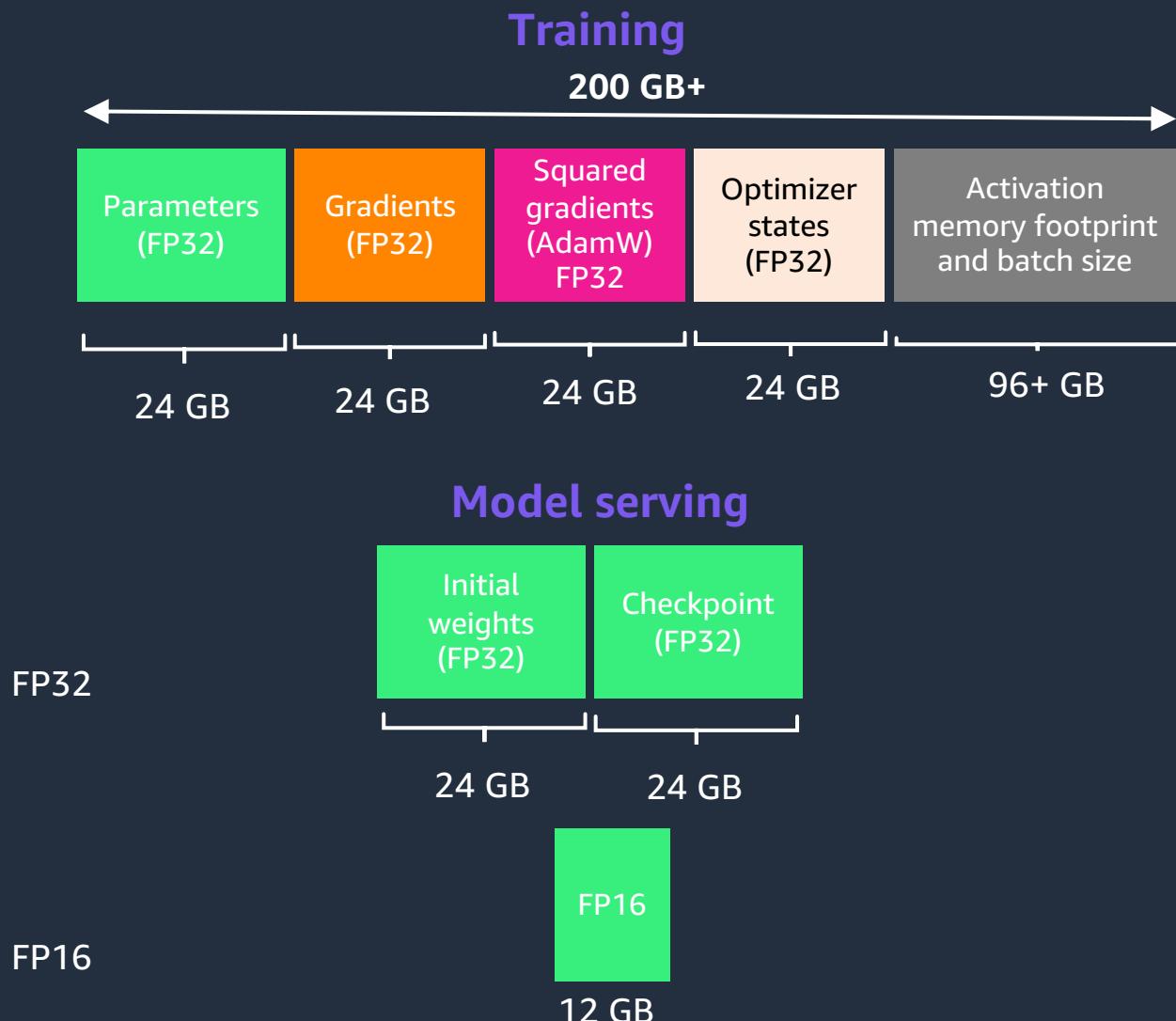
Model

Output indicator

# 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



# Lab 1 – LLM inference

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

EventHash:



# AGENDA

Generative AI – What is it and why the hype?

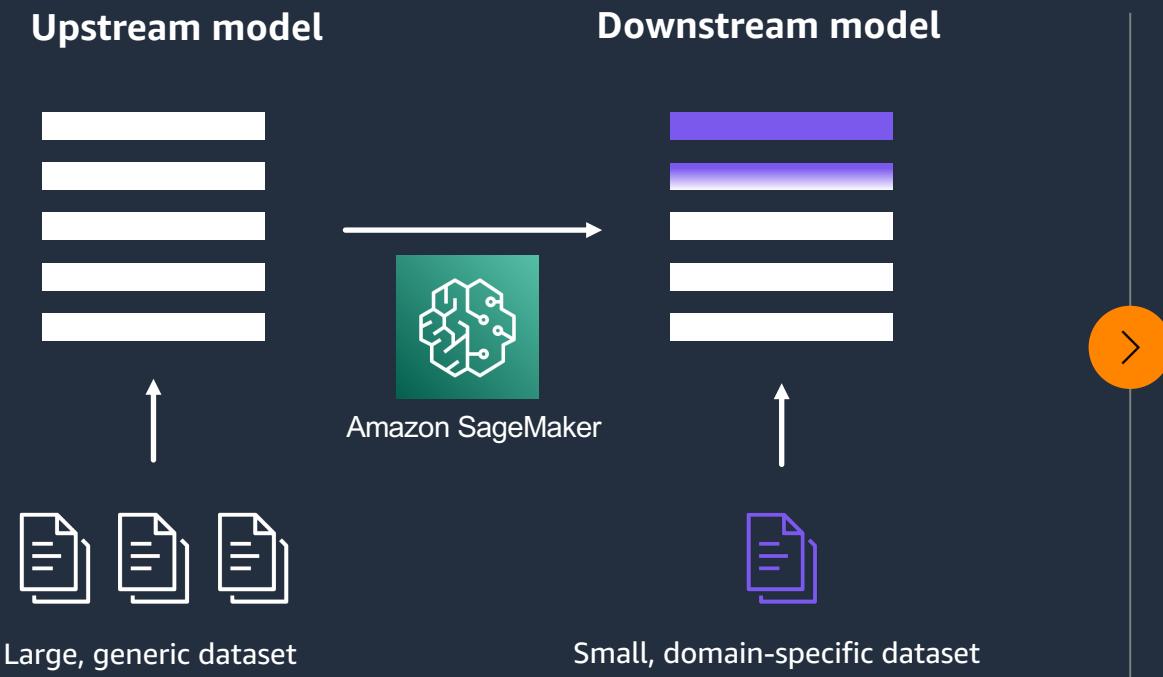
Large Language Models - How the ML works?

Large Language Model Hosting

Large Language Model Finetuning

Visual Foundation Models & Stable Diffusion

# Improving LLM performance by fine-tuning



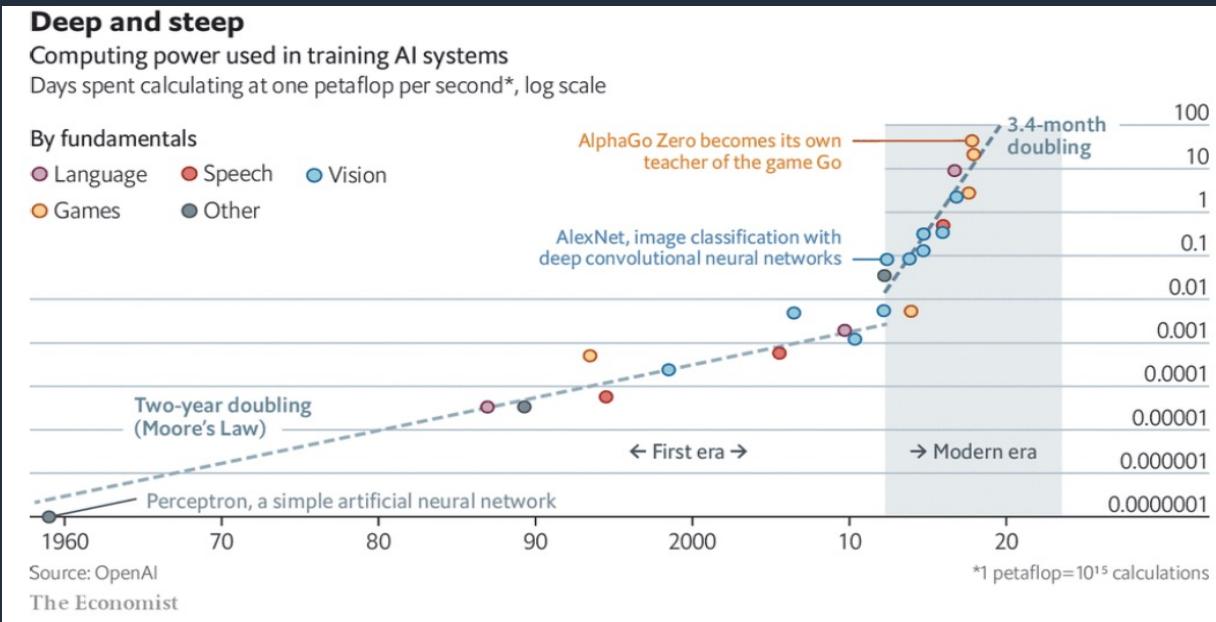
- Transfer learning of domain-specific knowledge into a foundation model at reasonable cost
- Update of weights in the network, while architecture is kept
- Fine-tuning is task-specific (e.g., MLM, CLM, PLM, translation, classification, ...)

# Challenges with large model training

MODELS GROW FASTER THAN HARDWARE, LEADING TO BOTTLENECKS

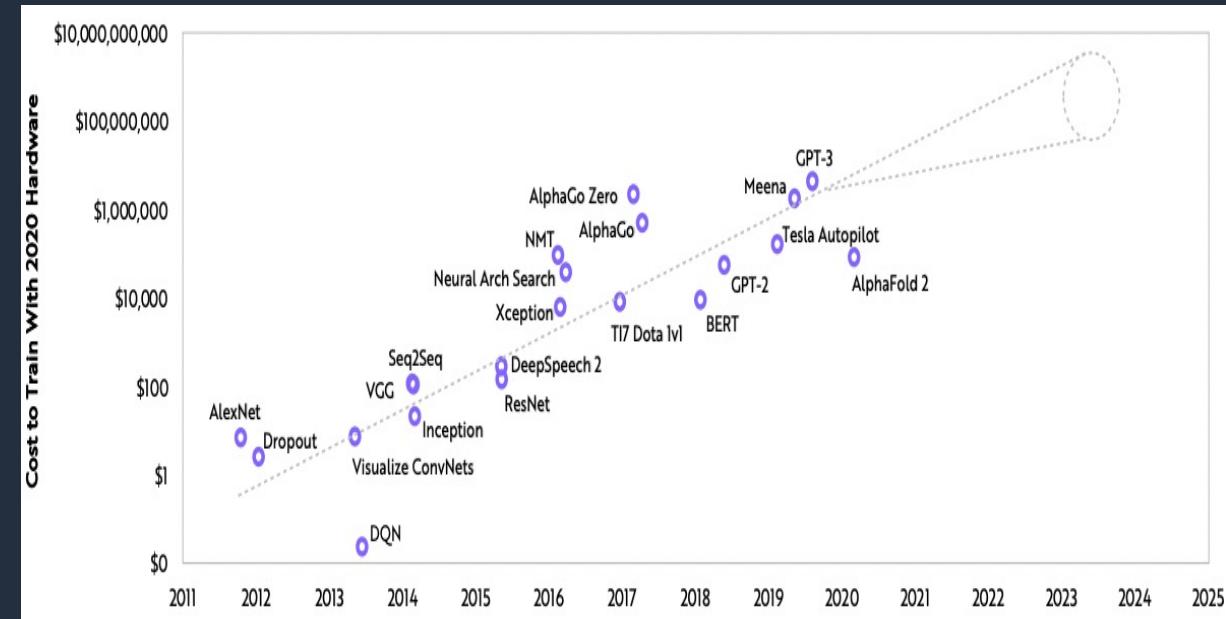
## INCREASING COMPLEXITY

Compute power ~ 2x every 3.4 months

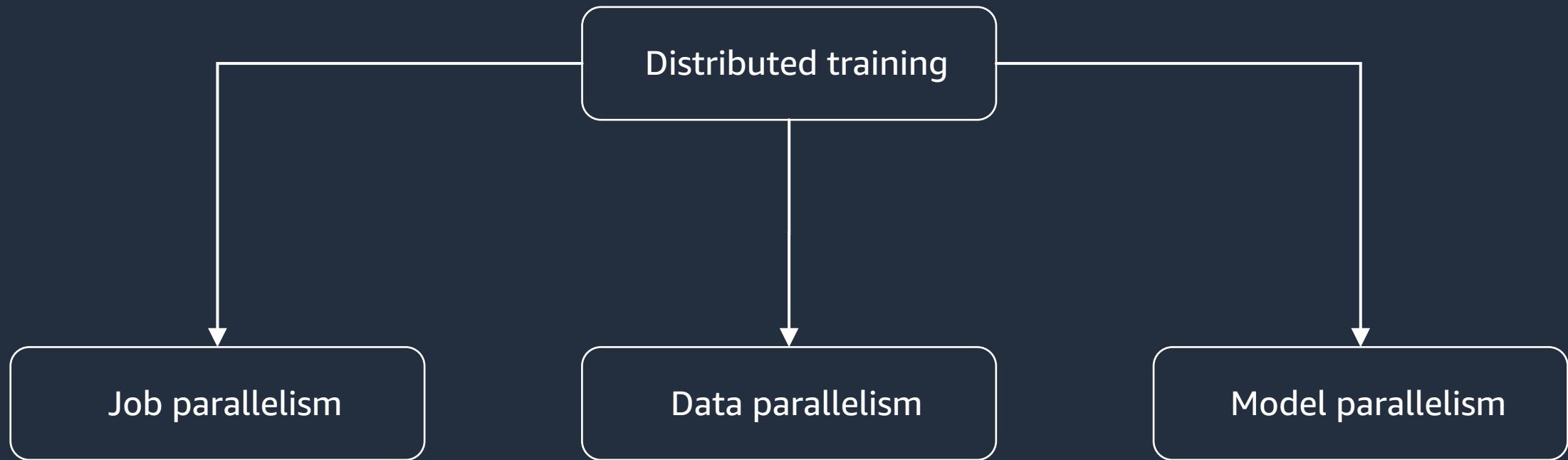


## INCREASING COSTS

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



# SageMaker distributed training options

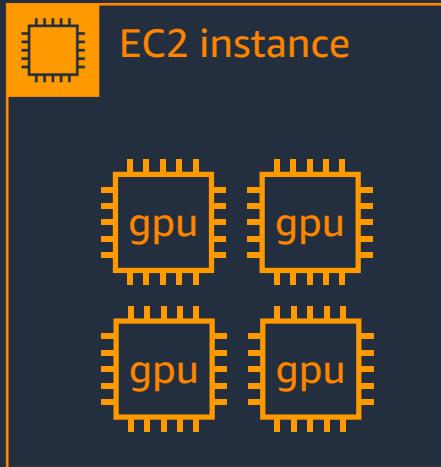


# Compute Challenges for ML training

## NLU Deep Learning models sizes:

- BERT Base – Neural Net with 12 layers, 768 hidden layer dimension, total of 110M parameters
- GPT-3: 175B parameters (2020)
- T5-XXL: 1.6T parameters (2021)

**Hardware acceleration is a must:**



BERT Training time:

**7.5 days on 8 NVIDIA V100**

→ 1 p3dn.24xlarge

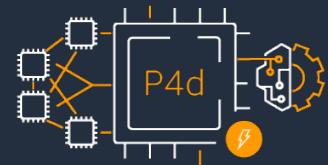
**Or 62 minutes on 2048 NVIDIA V100**

→ 256 p3dn.24xlarge

**Up to 3.1x speed up with 8 NVIDIA A100**

→ 1 p4d.24xlarge

# EC2 Accelerated Instances - Training



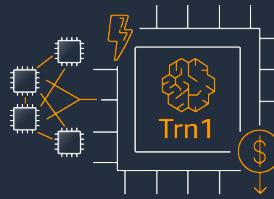
## P4d: GPU compute instance

- Designed for lowest-cost/highest-performance single- and multi-node distributed model training jobs
- 8x A100 40 GB, 400 Gbps



## P3/P3dn: GPU compute instance

- Midrange/single-node ML training
- 8x V100 16/32 GB, 25/100 Gbps



## Trn1: Custom ML acceleration

- Fastest and most cost-efficient DL Training in the cloud
- 16x Trn1 chips 32 GB, 800 Gbps



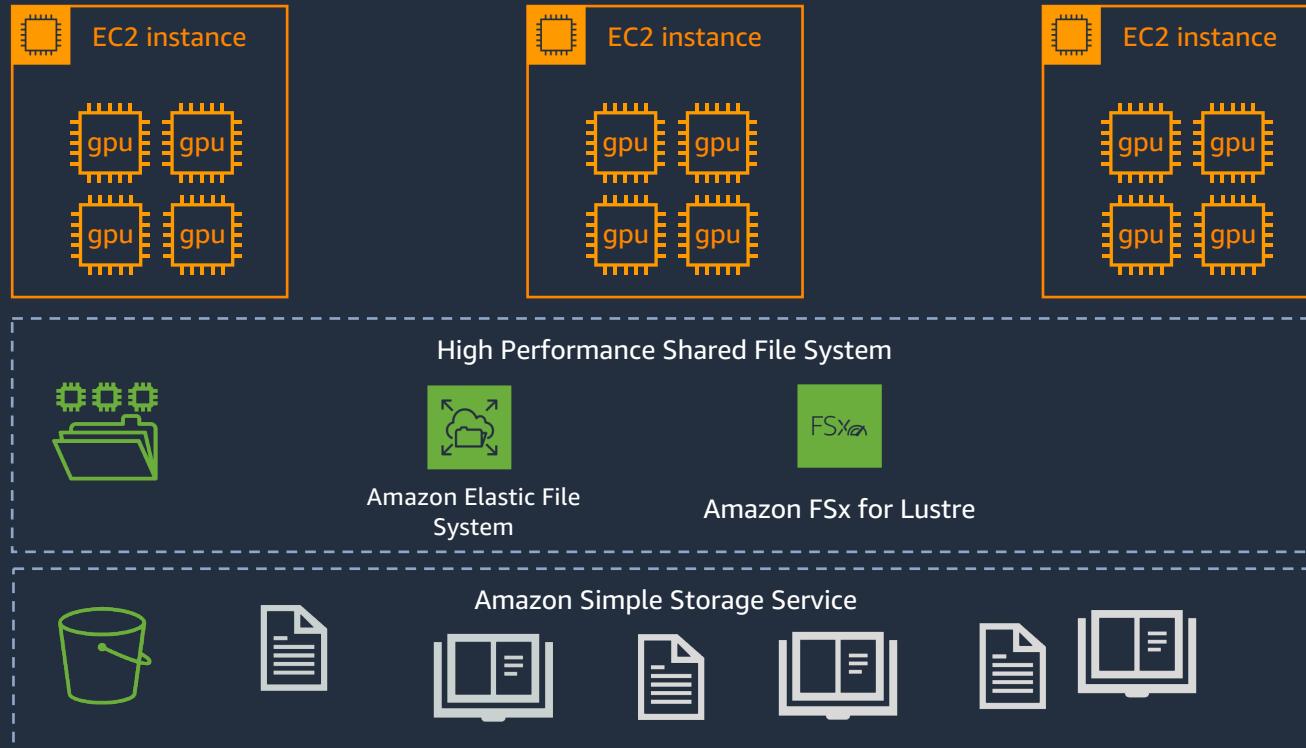
## DL1: Custom ML acceleration

- Cost-efficient training for small to medium scale training workloads
- 8x Habana Gaudi Accelerators, 400 Gbps

# 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**



## Read intensive job:

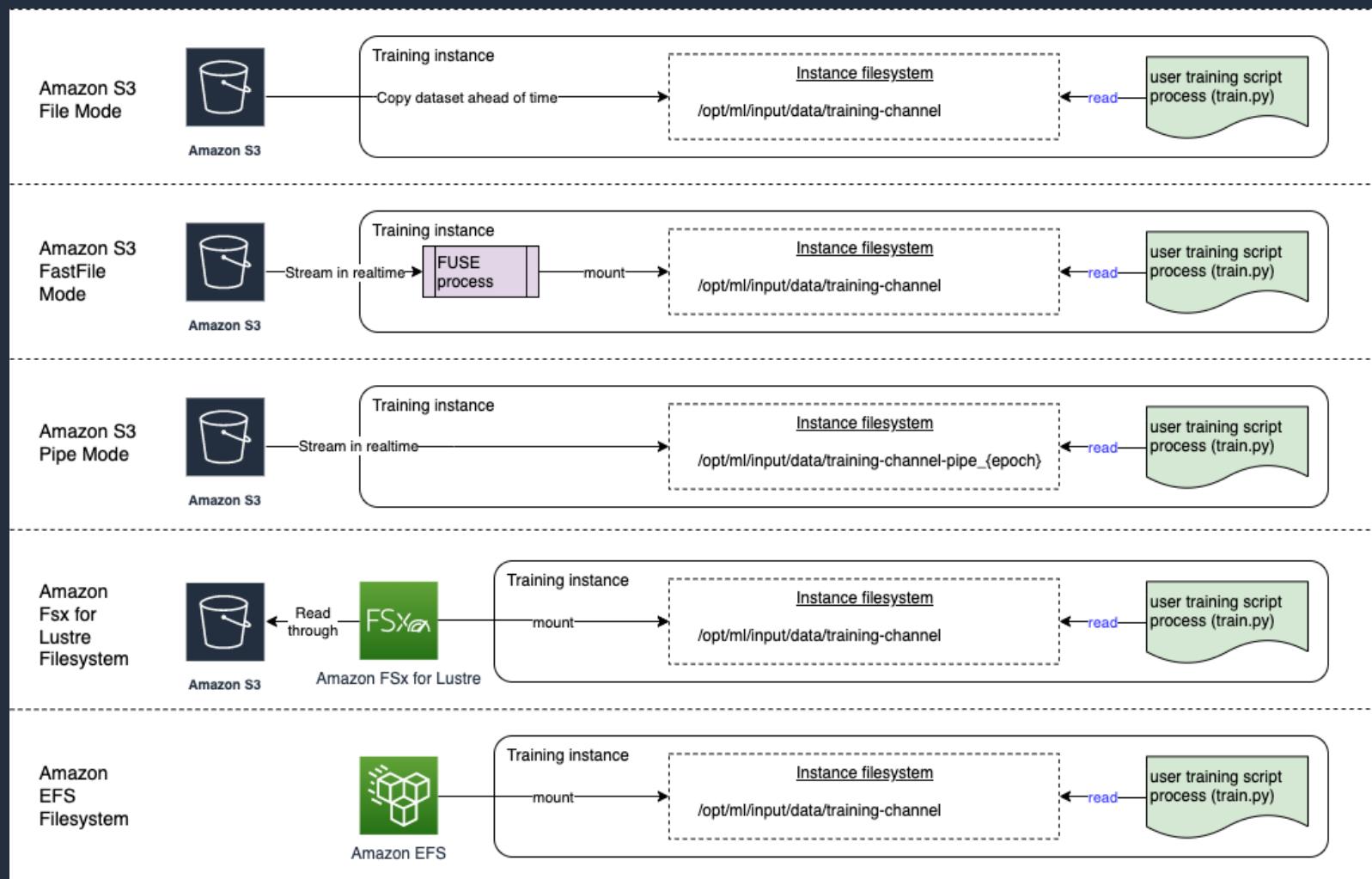
Mini batch + data loader strategy during training is key.

## 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!*

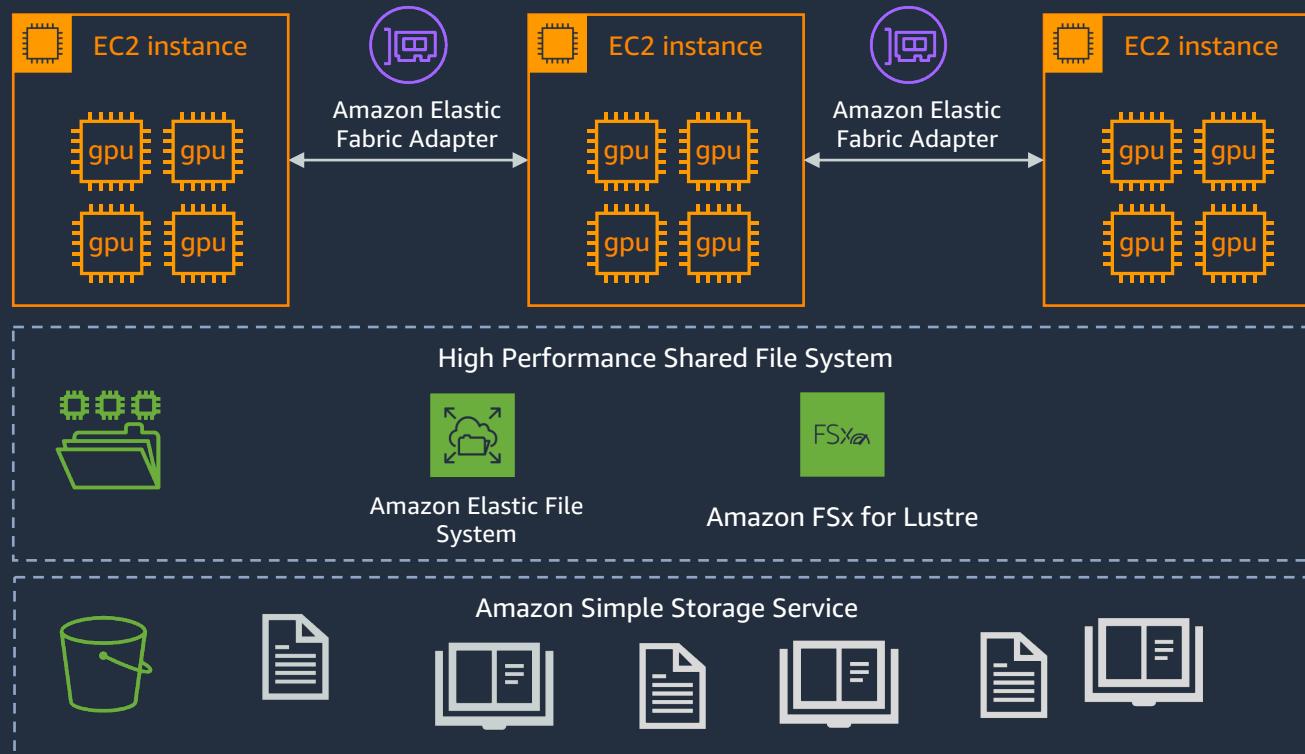
# Pick the right data option for SageMaker Training



# Networking Challenges for ML Training

Gradient descent over multi-node multi-gpu architecture:

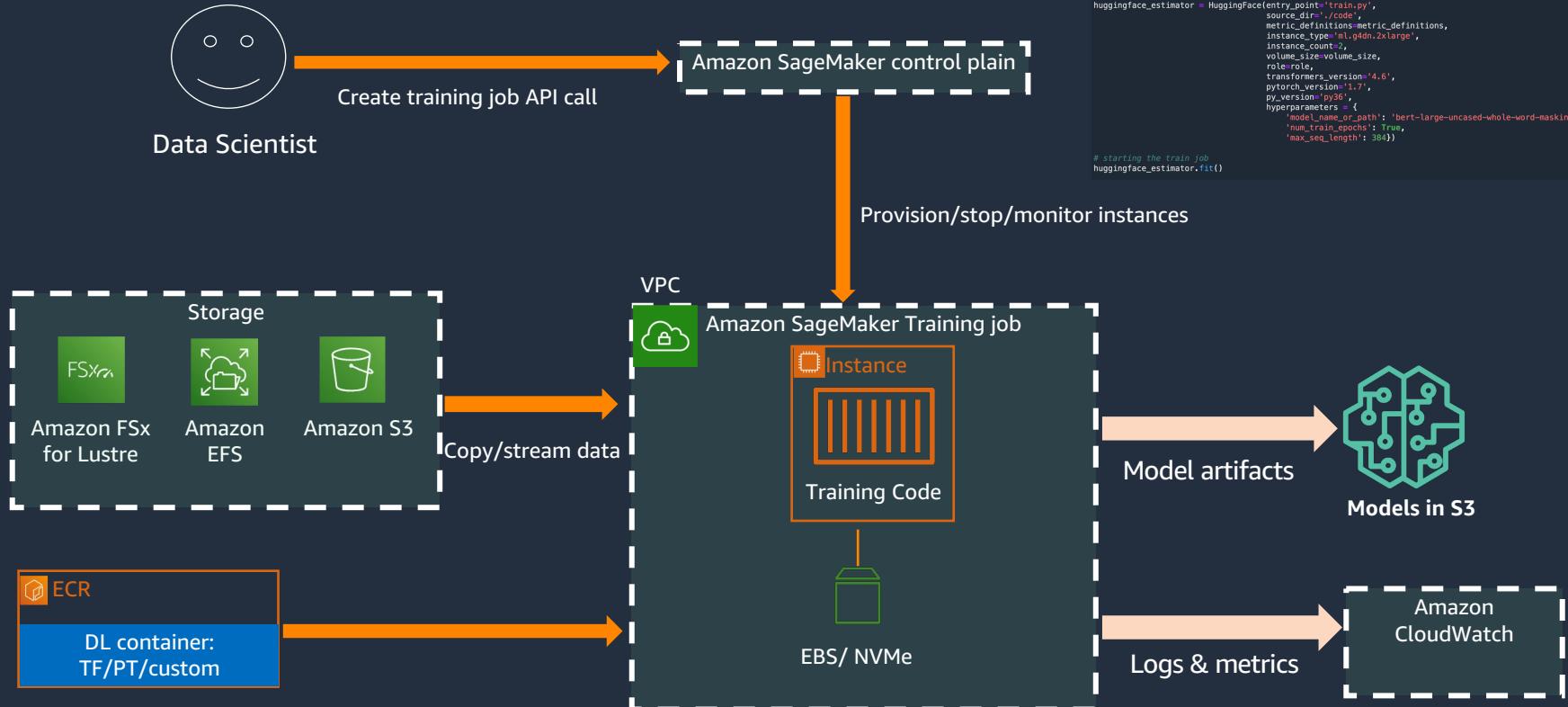
- Model + mini-batch data fits on single GPU: **Data Parallel Training**
- Model + mini-batch needs multiple GPUs: **Model Parallel Training**



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

# Training on Amazon SageMaker

## A REFRESHER



# 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()
```



# Training a LLM with HuggingFace

```
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()
```

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



Amazon  
SageMaker

# Large-scale training on SageMaker

## OPTIMIZED DISTRIBUTED TRAINING LIBRARIES & FRAMEWORKS



SageMaker Distributed Training Libraries

Bring your own library (e.g. DeepSpeed, Megatron)

## AMAZON SAGEMAKER TRAINING

Large Scale Cluster Orchestration

Data loading

NCCL Health Checks

Debugger

SageMaker Jumpstart for foundational models

Profiling

SageMaker Compiler

Experiment tracking

Warm pools

SSH to container

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

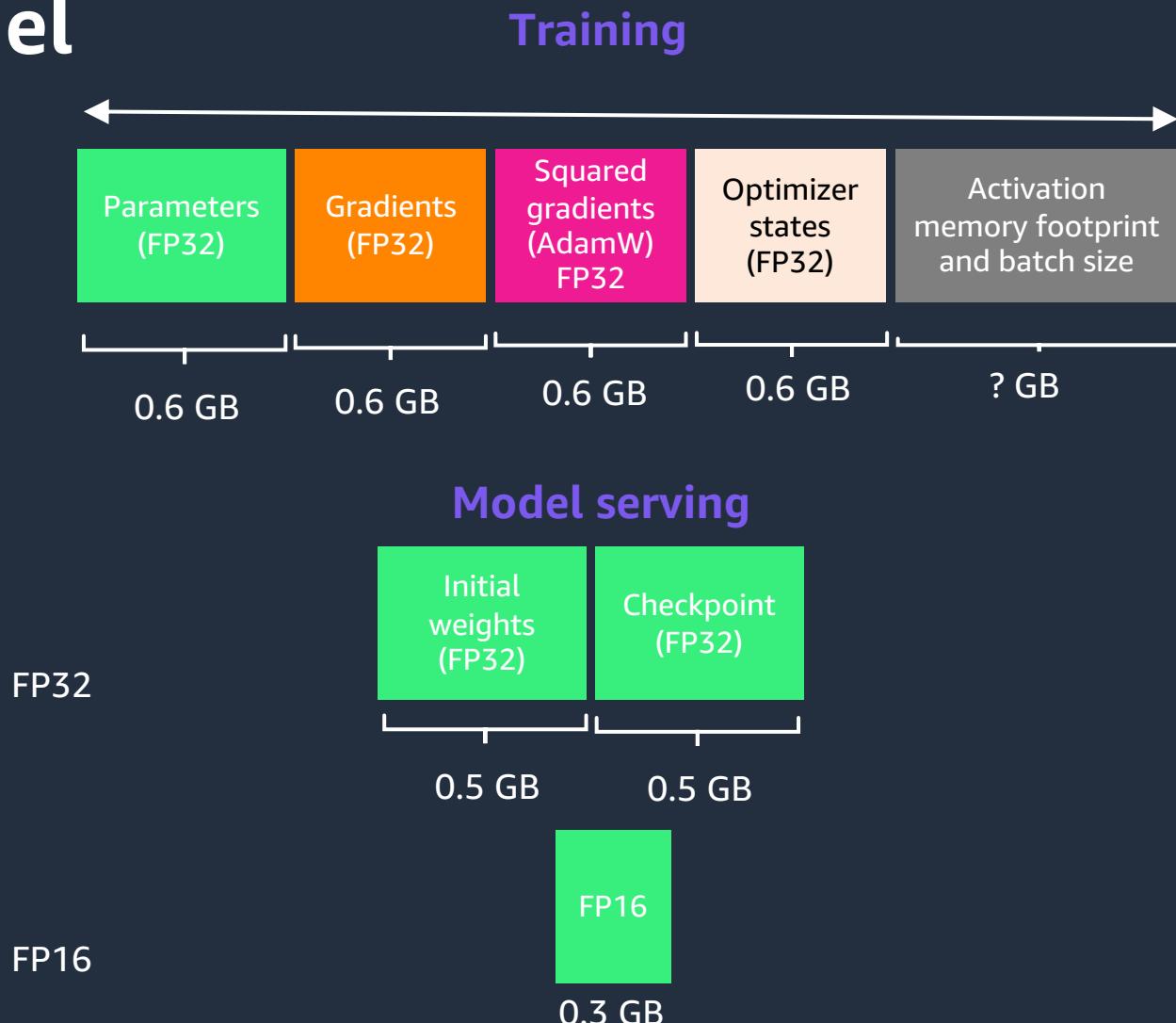
AWS Trainium



# Overview of (distil)GPT2 model

- Open-source predecessor of OpenAI's GPT-3/4
- Mainly used for predicting the next token
- Model released by OpenAI and distilled by HuggingFace
- Transformer model based on this [paper](#)
- Trained on the WebText dataset and contains data from [these](#) domains. It can perform various tasks in language processing

Hyperparameters	Value
Parameters	82 million (124 million)
Layers	6 (12)



# Lab 2 – LLM fine-tuning

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

EventHash:



# 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

# Generative AI is transforming AI

IMAGE GENERATION, TRANSFORMATION, UPSCALING



Generated by Stable Diffusion  
2.0. This interior does not exist



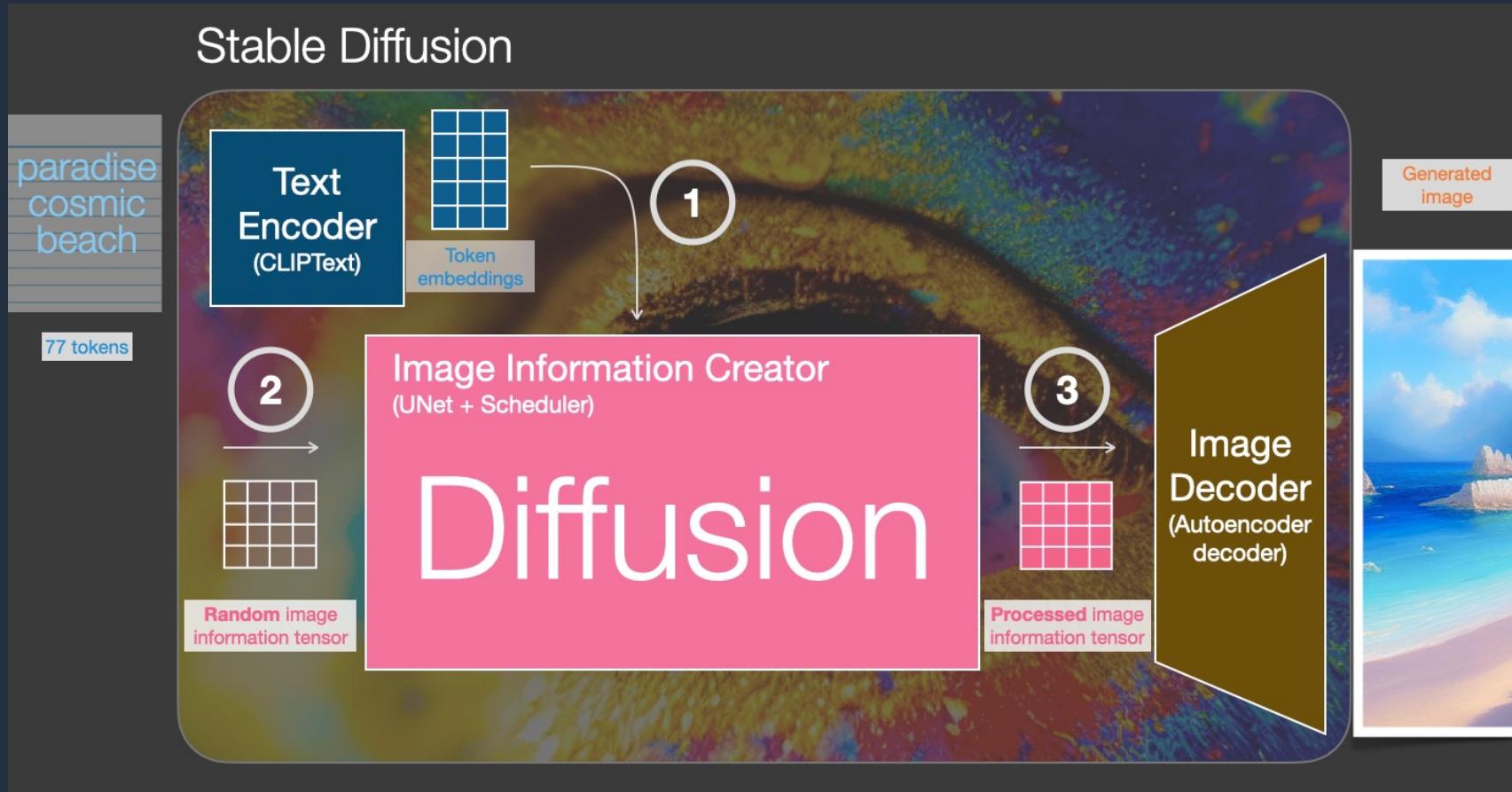
Seamless  
transformation



4x  
→  
Upscaling



# Building blocks of Diffusion



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

# Training a diffusion model

# 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)

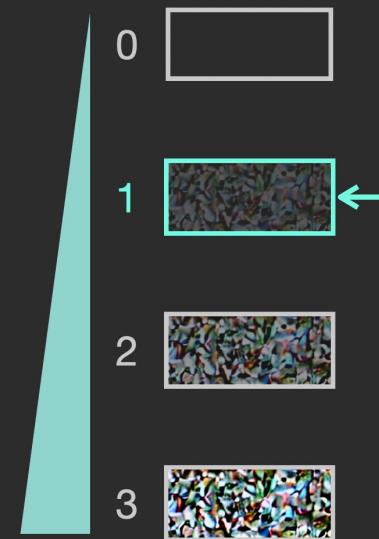
- 1  
Pick an image



- 2  
Generate some random **noise**



- 3  
Pick an amount of **noise**



- 4  
Add **noise** to the image in that **amount**



Noise sample 1

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

# 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

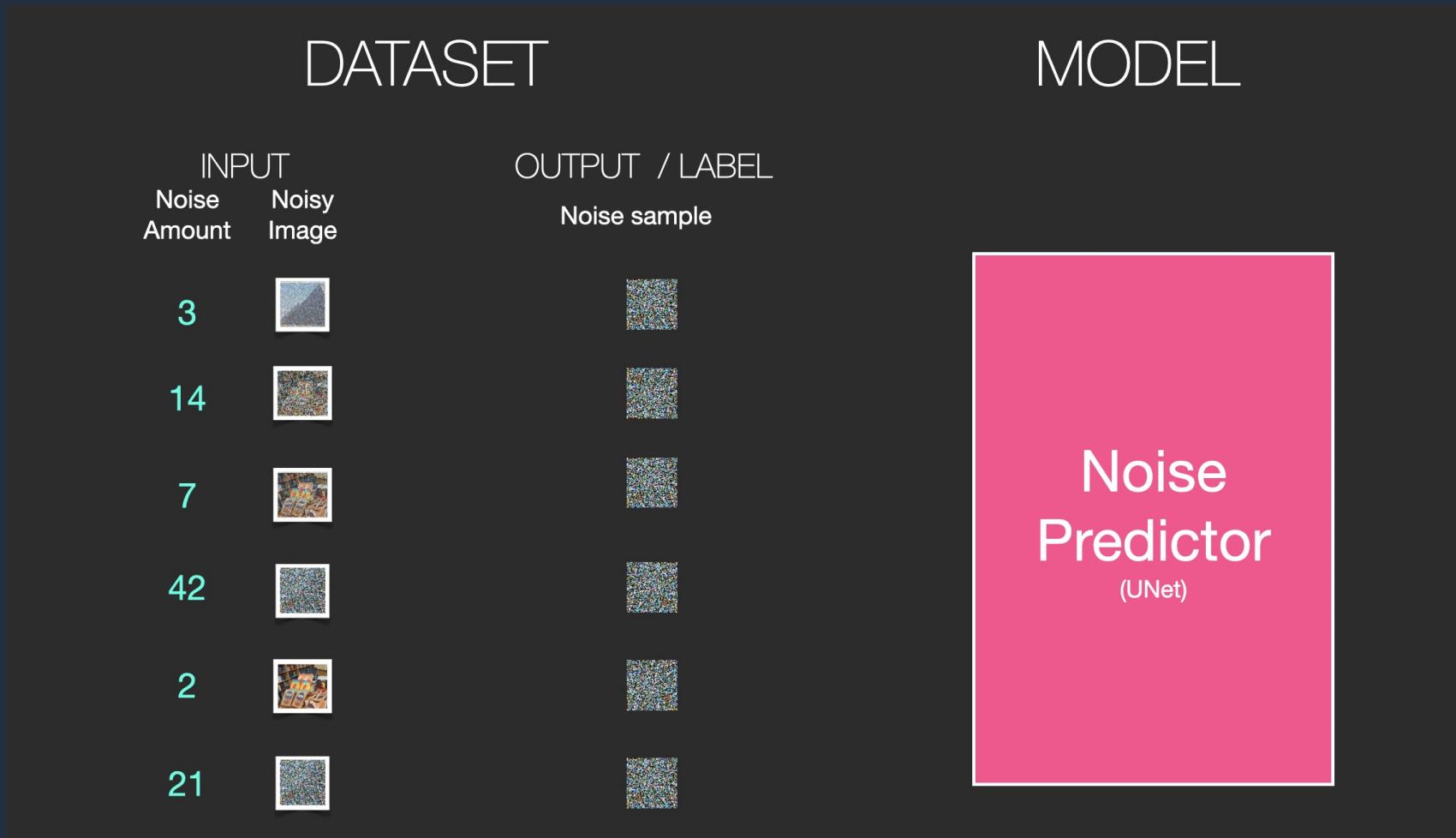


- 4  
Add **noise** to the image in that **amount**



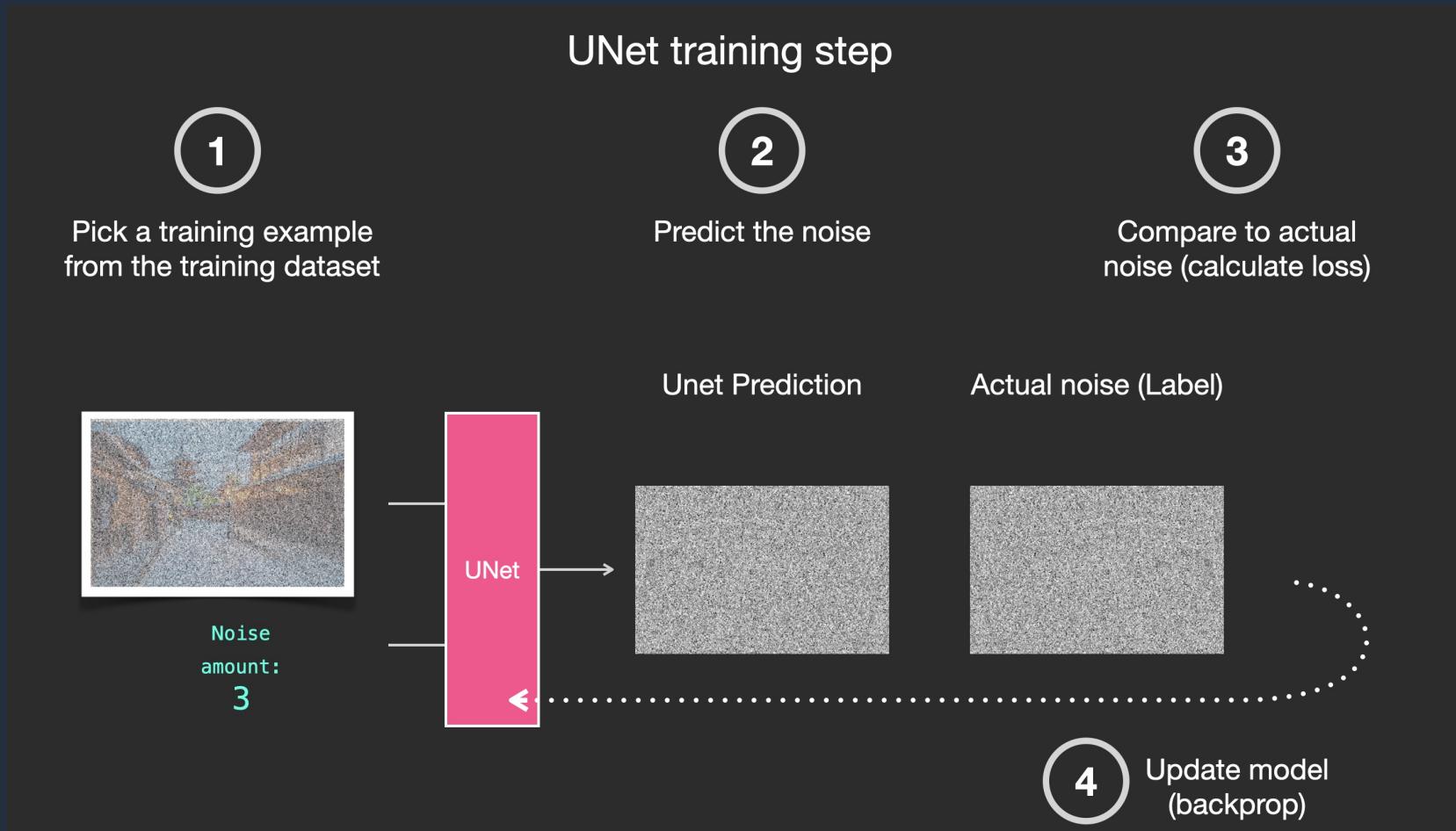
Source: <https://jalammar.github.io/illustrated-stable-diffusion/>

# Building blocks of Diffusion



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

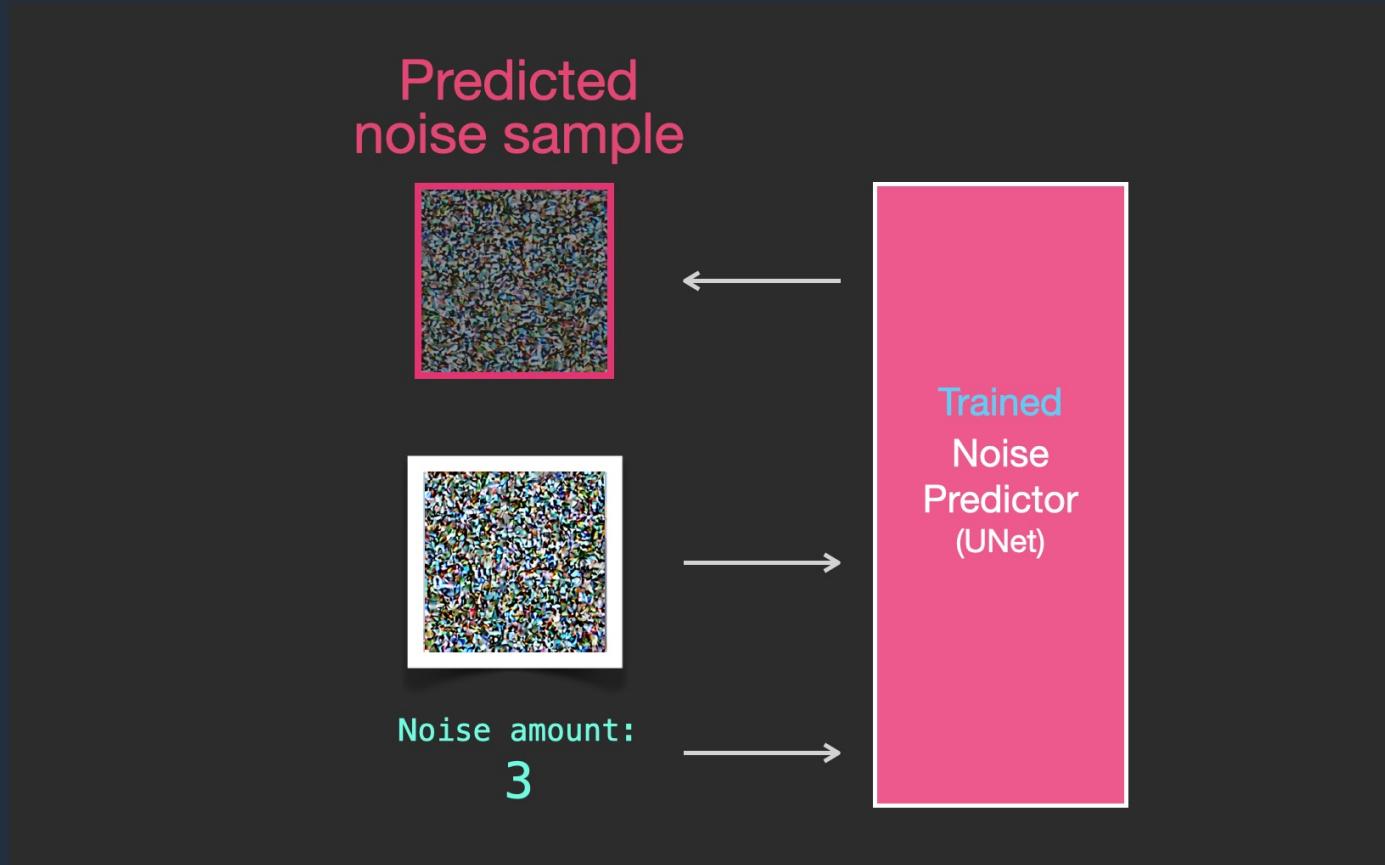
# Building blocks of Diffusion



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

# Image generation

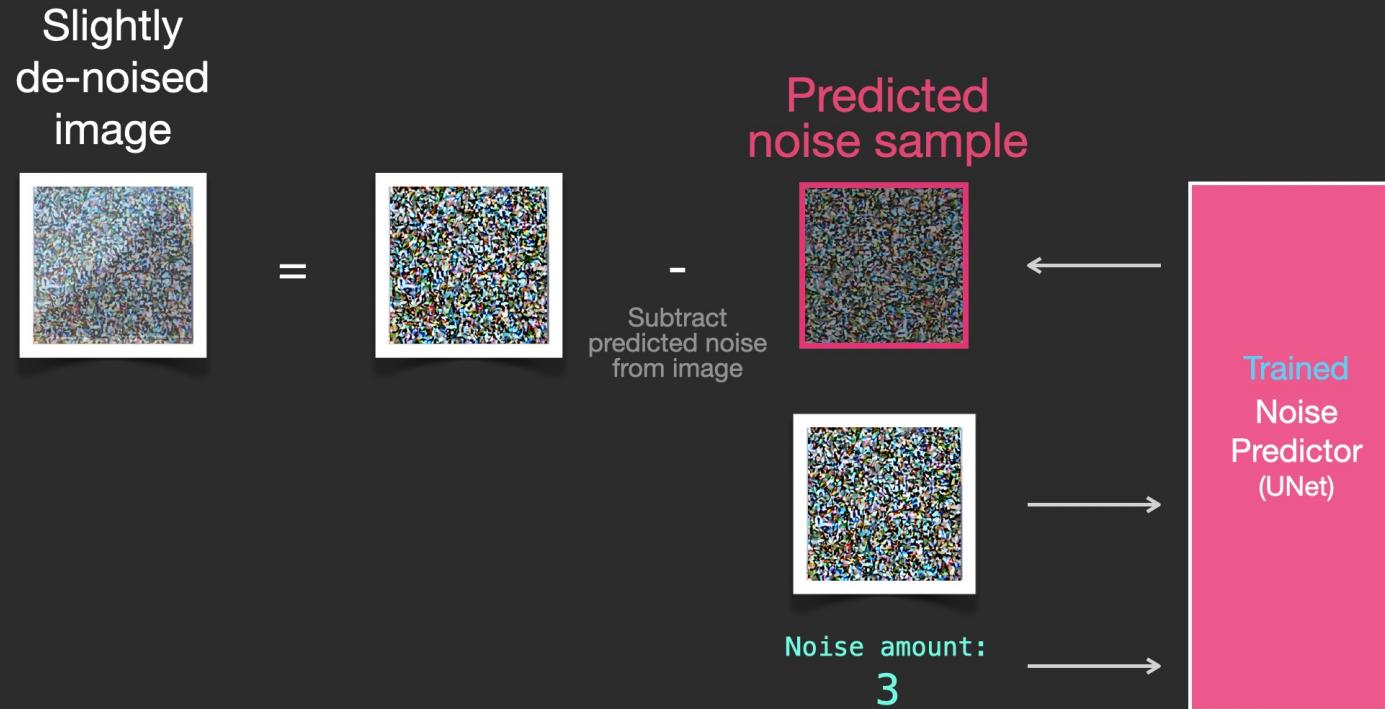
# Building blocks of Diffusion



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

# Building blocks of Diffusion

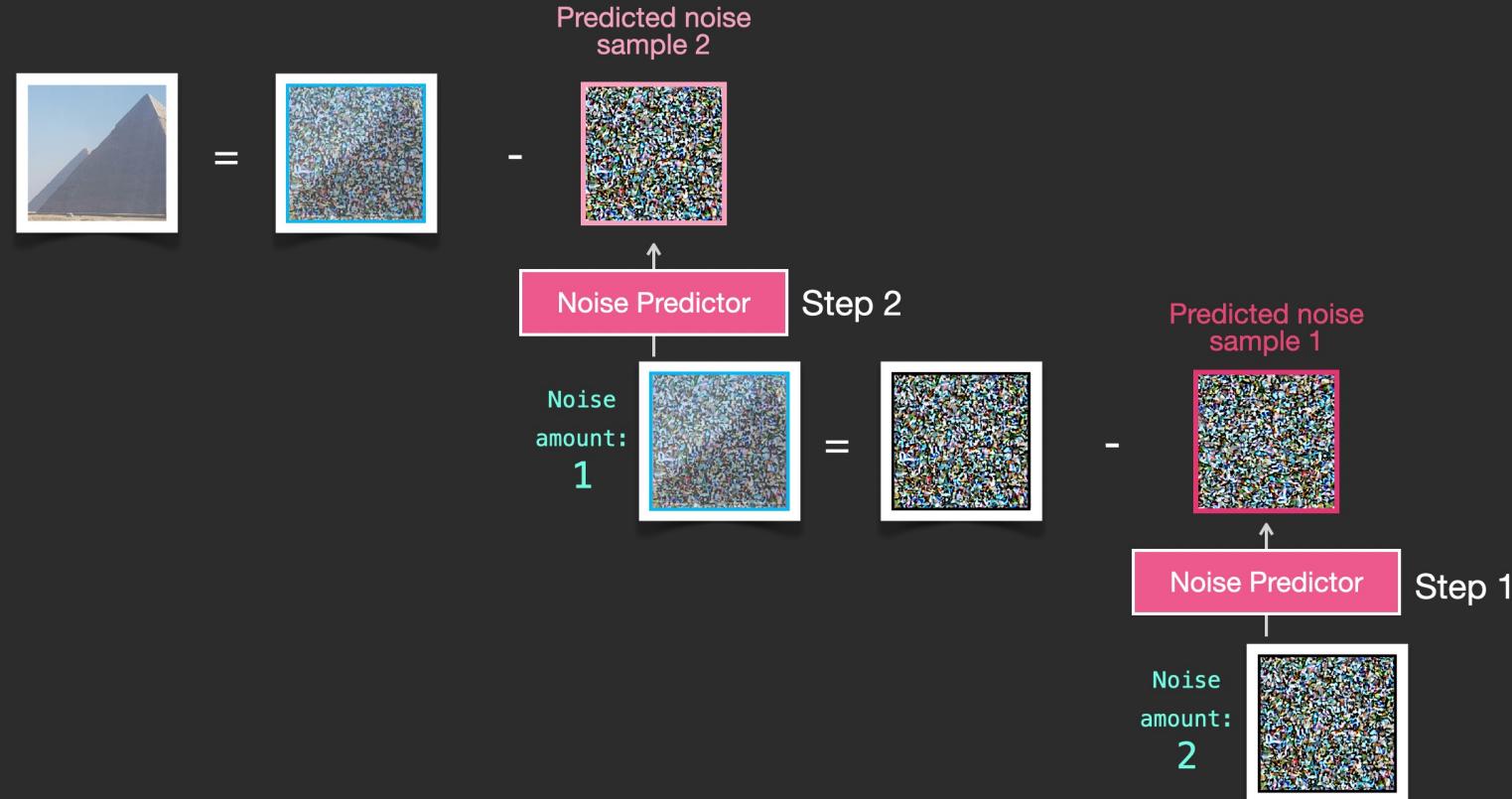
## Reverse Diffusion (Denoising) Step 1



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

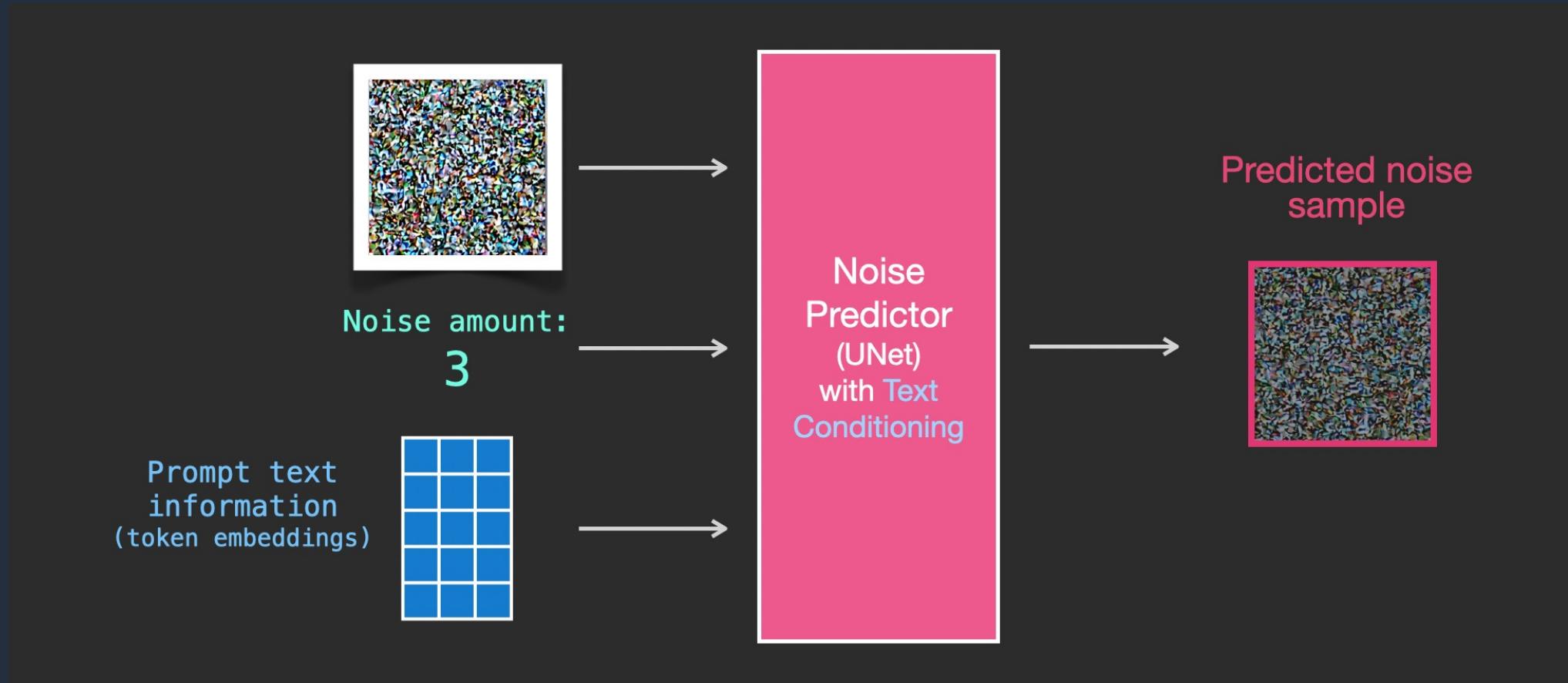
# Building blocks of Diffusion

## Image Generation by Reverse Diffusion (Denoising)



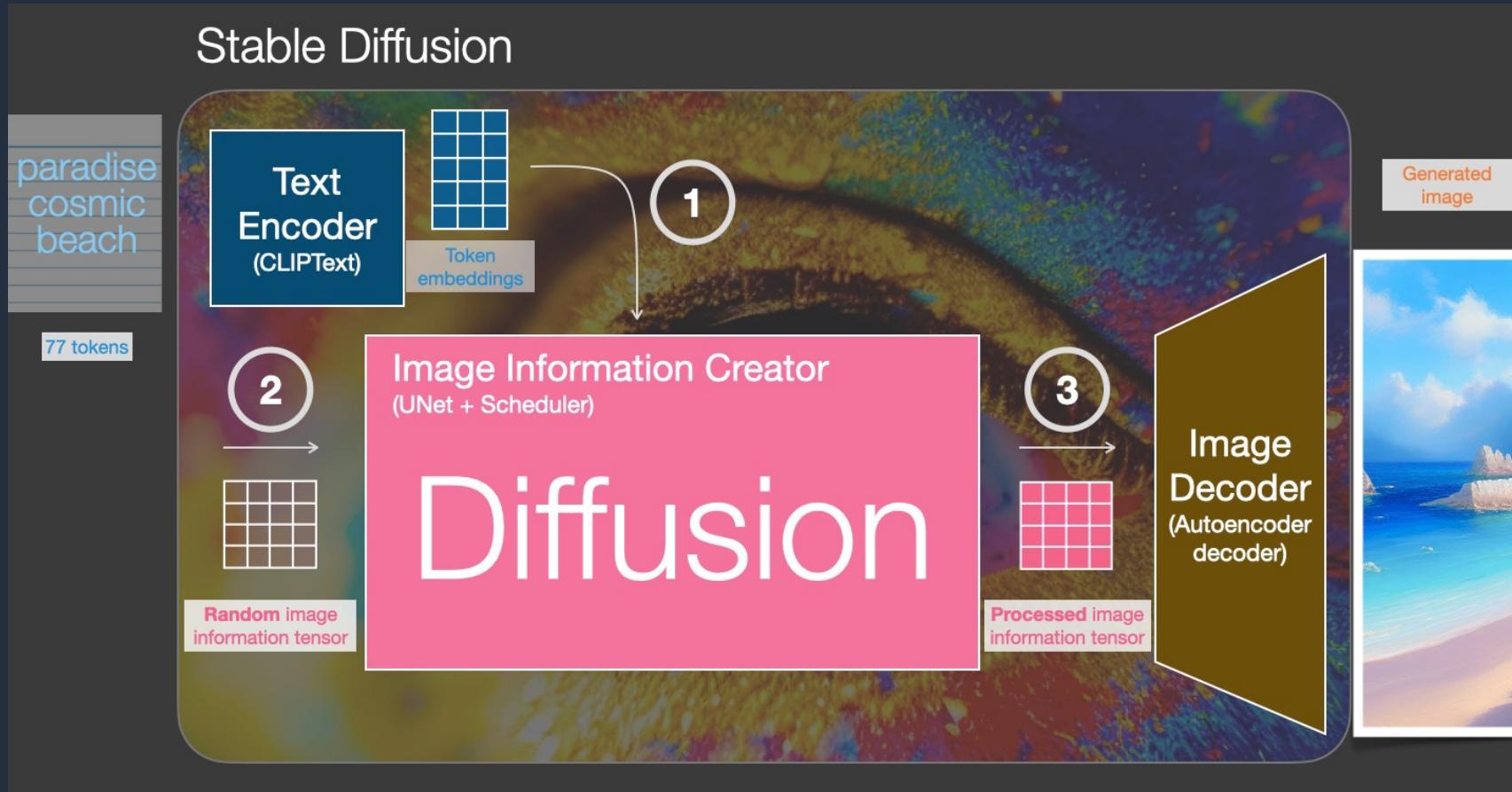
Source: <https://jalammar.github.io/illustrated-stable-diffusion/>

# Building blocks of Diffusion



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

# Building blocks of Diffusion



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

# Lab 3 – Stable Diffusion Deployment & Inference

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

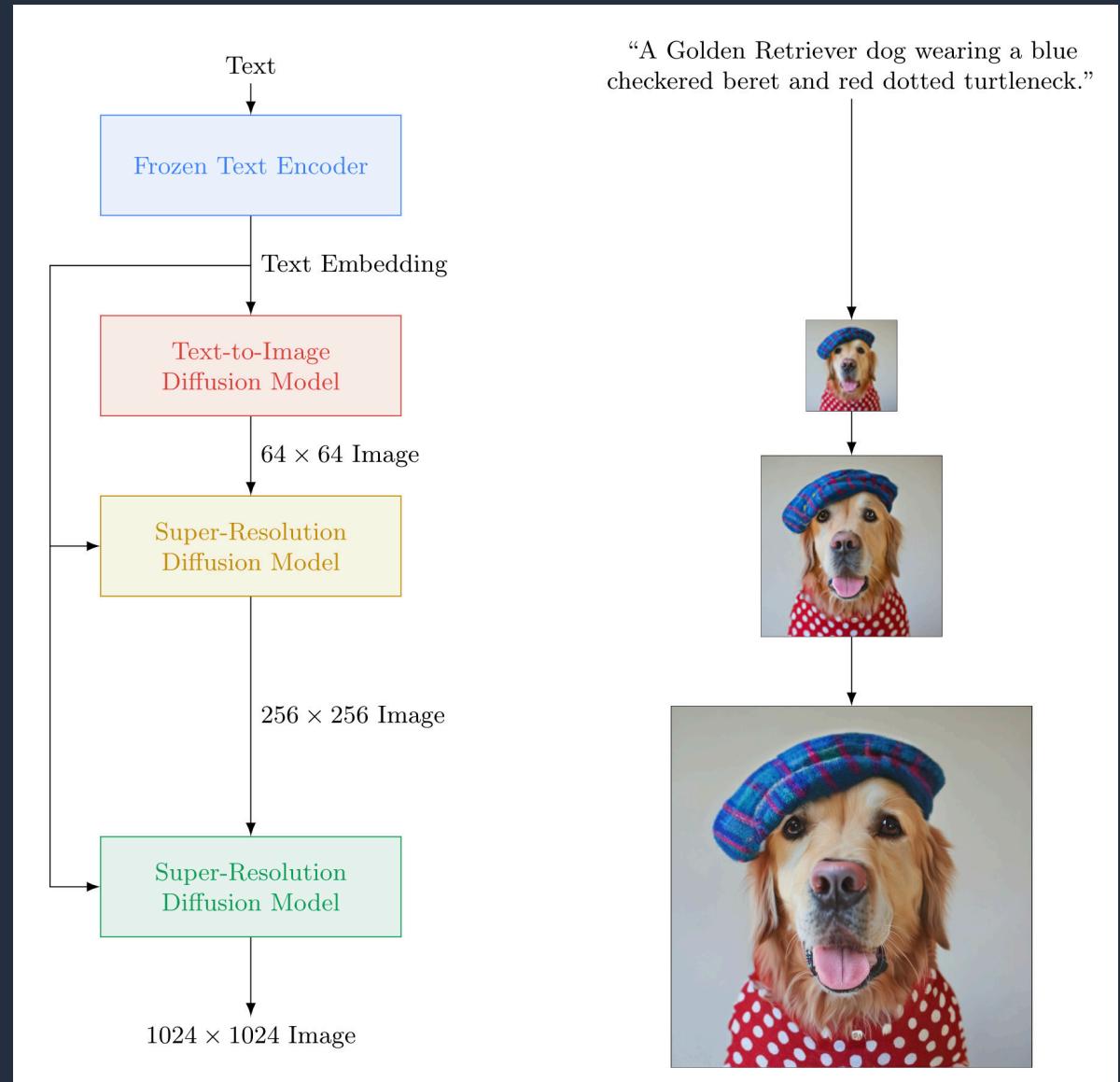
EventHash:



# Other models and related work

# 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.



# 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



# Midjourney

Heavily stylized diffusion model

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



# Runway Gen-2

## Text-to-Video

Mode 02: *Text + Image to Video*  
Generate a video using a driving image and a text prompt

---

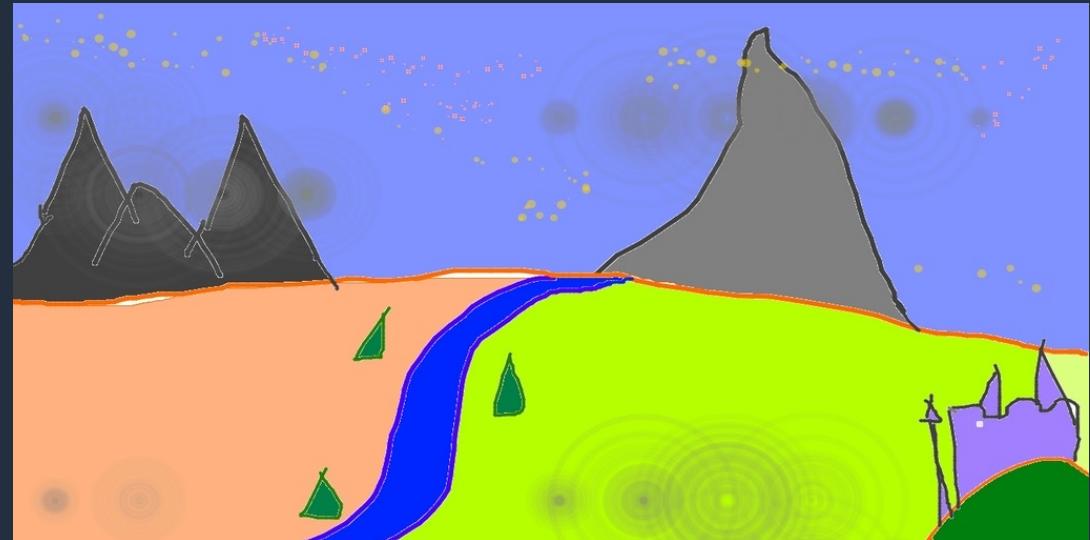
 Input Image

*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

# Image-to-image (img2img)

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



# Inpainting & outpainting

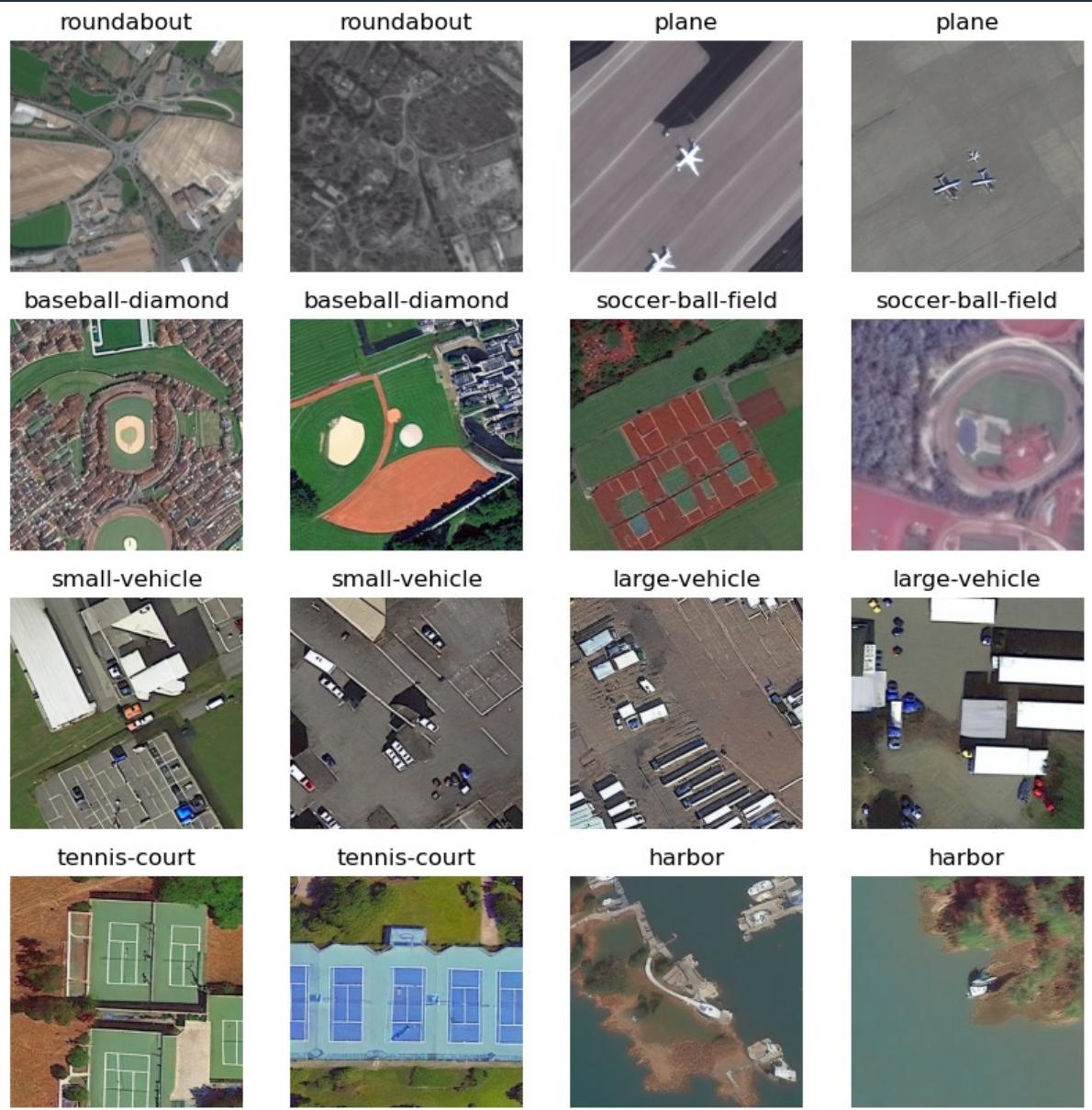
- Masked image-to-image



# 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

- More efficient {
  - LoRA
  - Hypernetworks



"A satellite image showing a {class\_name}"

# 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

Input images

in the Acropolis   swimming   sleeping  
in a doghouse   in a bucket   getting a haircut

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

# Q&A + CSAT – Please provide feedback!



# Thank you!

Aris Tsakpinis

[tsaris@amazon.de](mailto:tsaris@amazon.de)

Heiko Hotz

[hotzh@amazon.co.uk](mailto:hotzh@amazon.co.uk)

Dr. Markus Schweier

[mschweie@amazon.de](mailto:mschweie@amazon.de)