SPRING INTO AI

Building Intelligent Applications in Java with Spring AI

Dan Vega - Spring Developer Advocate @Broadcom

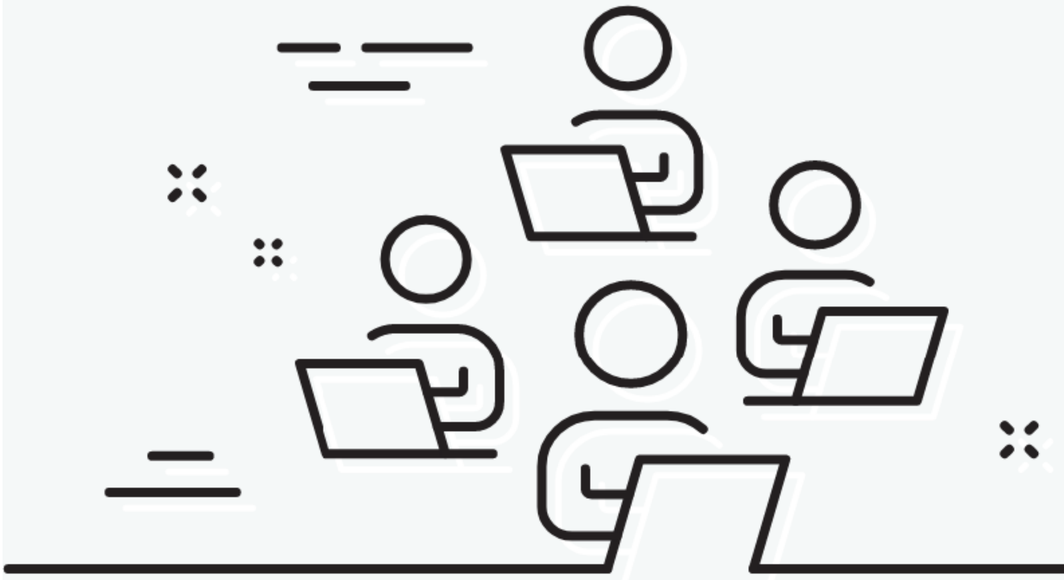
ABOUT ME

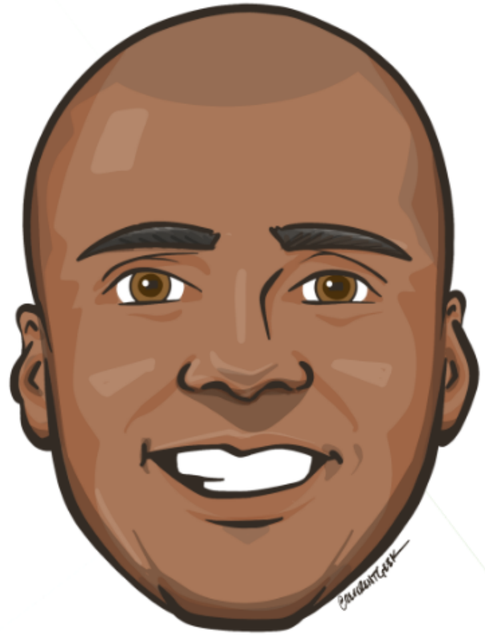
Learn more at danvega.dev

- 👨‍👩‍👧‍👦 Husband & Father
- 🏠 Cleveland
- ☕ Java Champion
- 👨‍💻 Software Development 23 Years
- 🍃 Spring Developer Advocate
- 📝 Content Creator
- 🐋 @therealdanvega
- 🏃‍♂️ Running
- 🏌️‍♂️ Golf

spring

OFFICE HOURS





https://www.springofficehours.io

ByteSized AI

www.bytesizedai.dev

Archive

Search posts...

Artificial Intelligence (AI) | Prompt Engineering | Open AI | Claude | Image Generation

Sep 18, 2024

How to talk to Robots

Learning how to effectively communicate with AI

Dan Vega

Sep 16, 2024 ♥ 2

The AI That Thinks Before It Speaks

Getting to know Open AI's two new models

Dan Vega

Sep 12, 2024

🤖 Why you need to check out Claude's Projects

Getting started with Projects in Claude

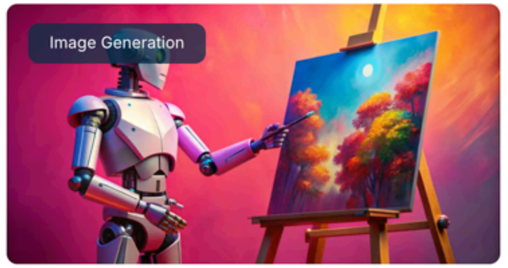
Dan Vega

Sep 06, 2024

What is Artificial Intelligence (AI)

How do you define AI?

Dan Vega



Sep 05, 2024

Generating images with AI

How I generated a logo for my newsletter

Dan Vega

Sep 03, 2024 ♥ 1 2

Welcome to ByteSized AI 🤖

👋🏻 Hello, World!

Dan Vega

AGENDA

What are we going to talk about?

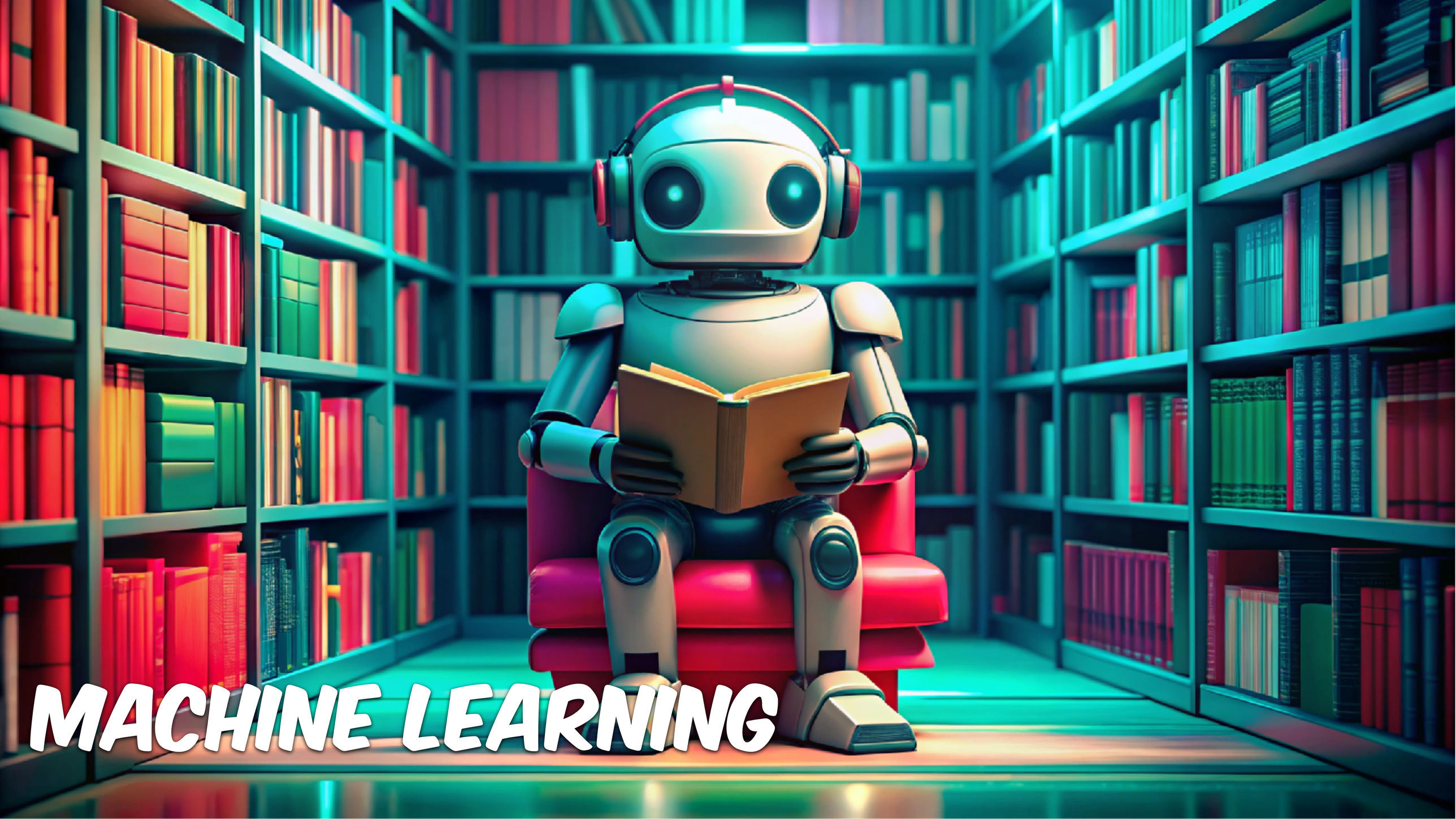
- What is AI?
  - Machine Learning
  - Deep Learning
  - Large Language Models (LLMs)
  - Prompt Engineering
- Java & AI
- Spring AI
- Show me the code


WHAT IS ARTIFICIAL INTELLIGENCE?

WHAT IS ARTIFICIAL INTELLIGENCE

Artificial Intelligence

- It can be categorized based on capabilities such as Narrow AI, General AI and Superintelligent AI
- It can be based on applications such as Artificial Narrow Intelligence (ANI), Artificial General Intelligence (AGI) or Artificial Superintelligence (ASI).
- It can also be based on techniques such as Machine Learning (ML), Deep Learning, Natural Language Processing, Robotics and Expert Systems.


MACHINE LEARNING

MACHINE LEARNING

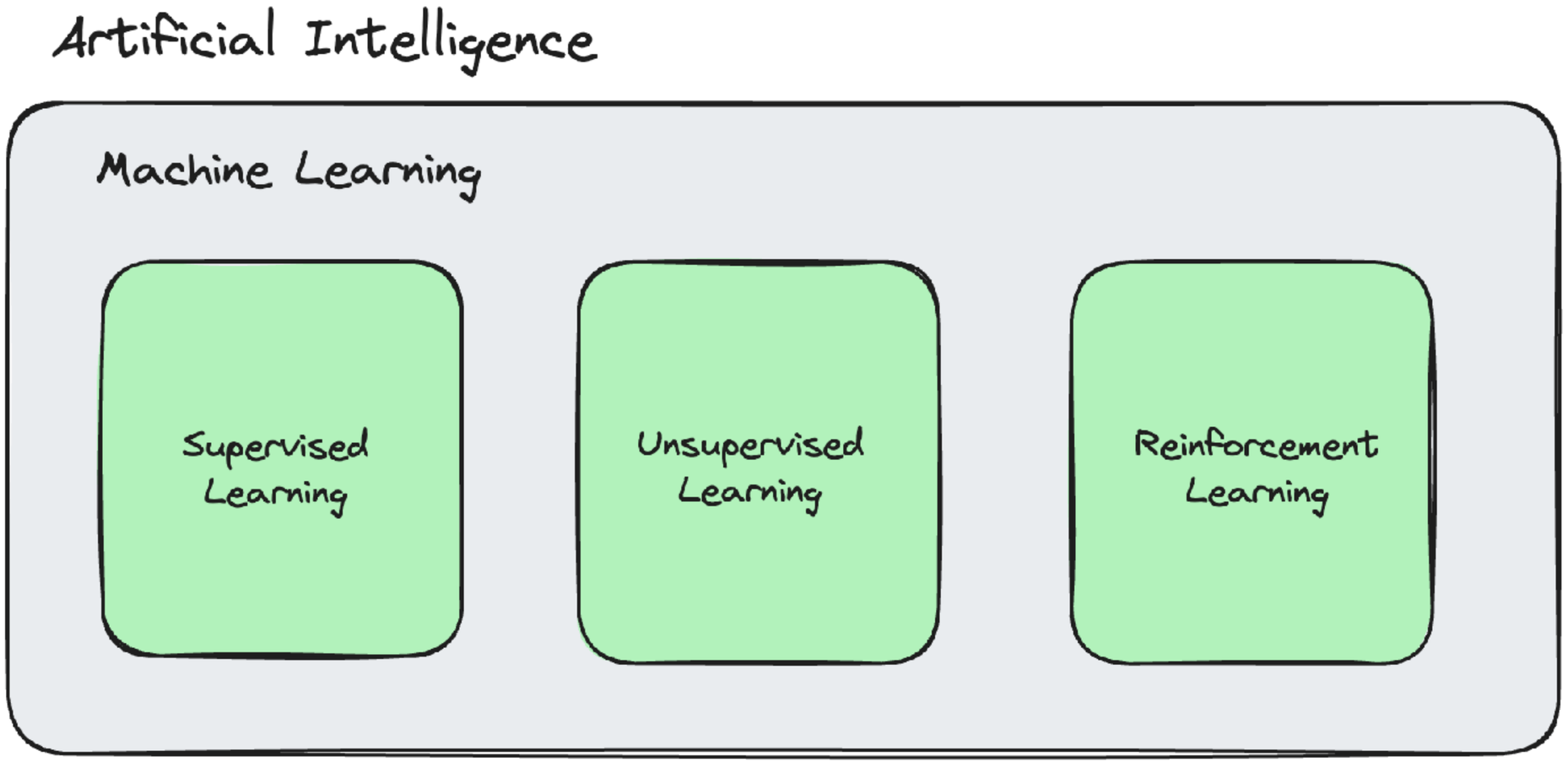
Artificial Intelligence

Machine Learning

Unlike traditional programming, where explicit instructions are provided for every scenario, ML systems learn patterns from data, allowing them to make predictions or decisions without being explicitly programmed for each possibility.

MACHINE LEARNING

Machine Learning encompasses various techniques and types of tasks, including:



MACHINE LEARNING

Use Cases

- Facial Recognition
- Recognize Tumors on x-ray scans
- Abnormality on ultrasounds
- Self-drive mode (recognize stop sign / pedestrian / etc...)
- Fraud Detection
- Product Recommendations (YouTube)
- Spam Filtering

SUPERVISED LEARNING

Labeling the training data

Dan Vega

Dan Vega

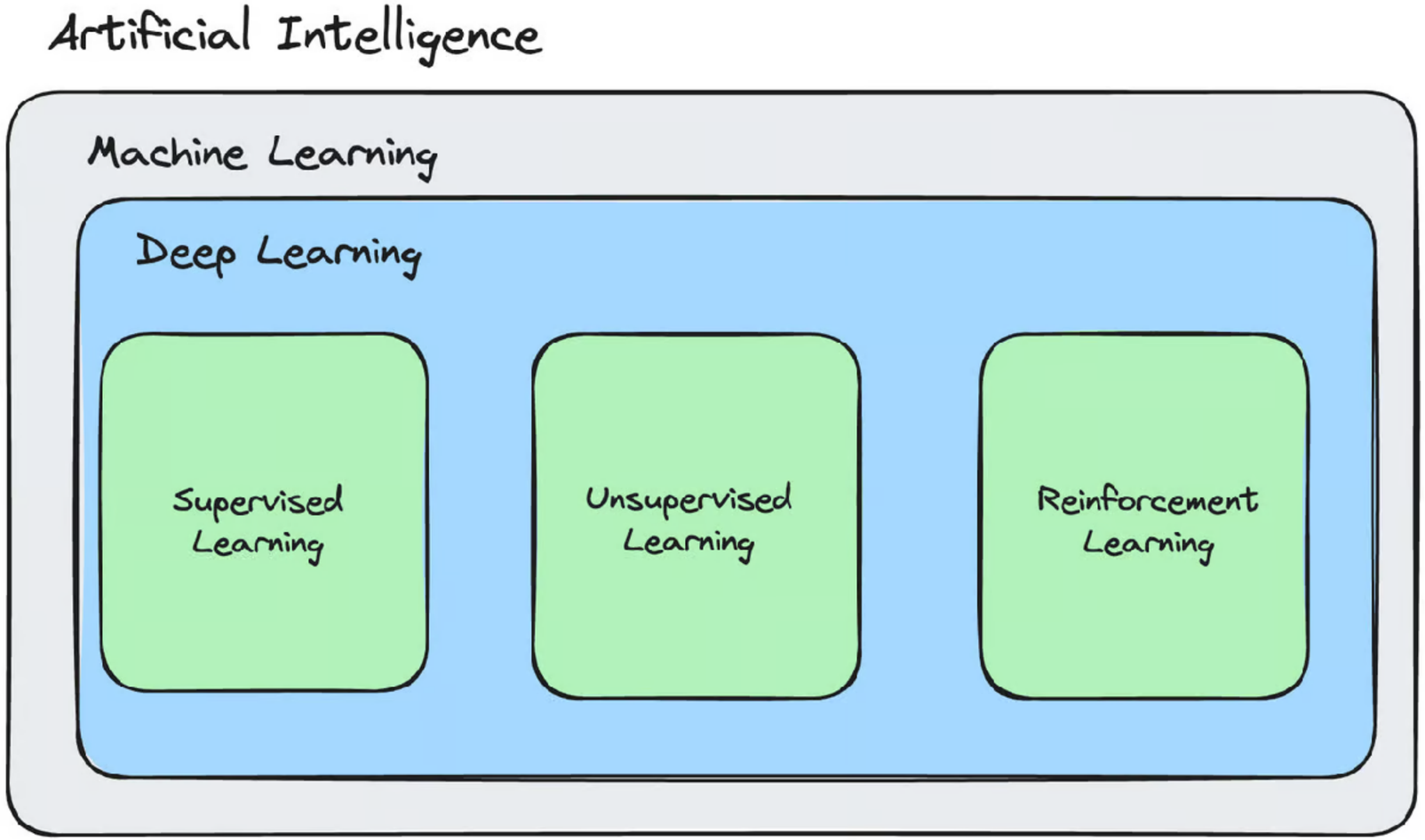
Dan Vega

TRAINING DATA

Supervised Learning of Big Data Companies


DEEP LEARNING

DEEP LEARNING AND NEURAL NETWORKS



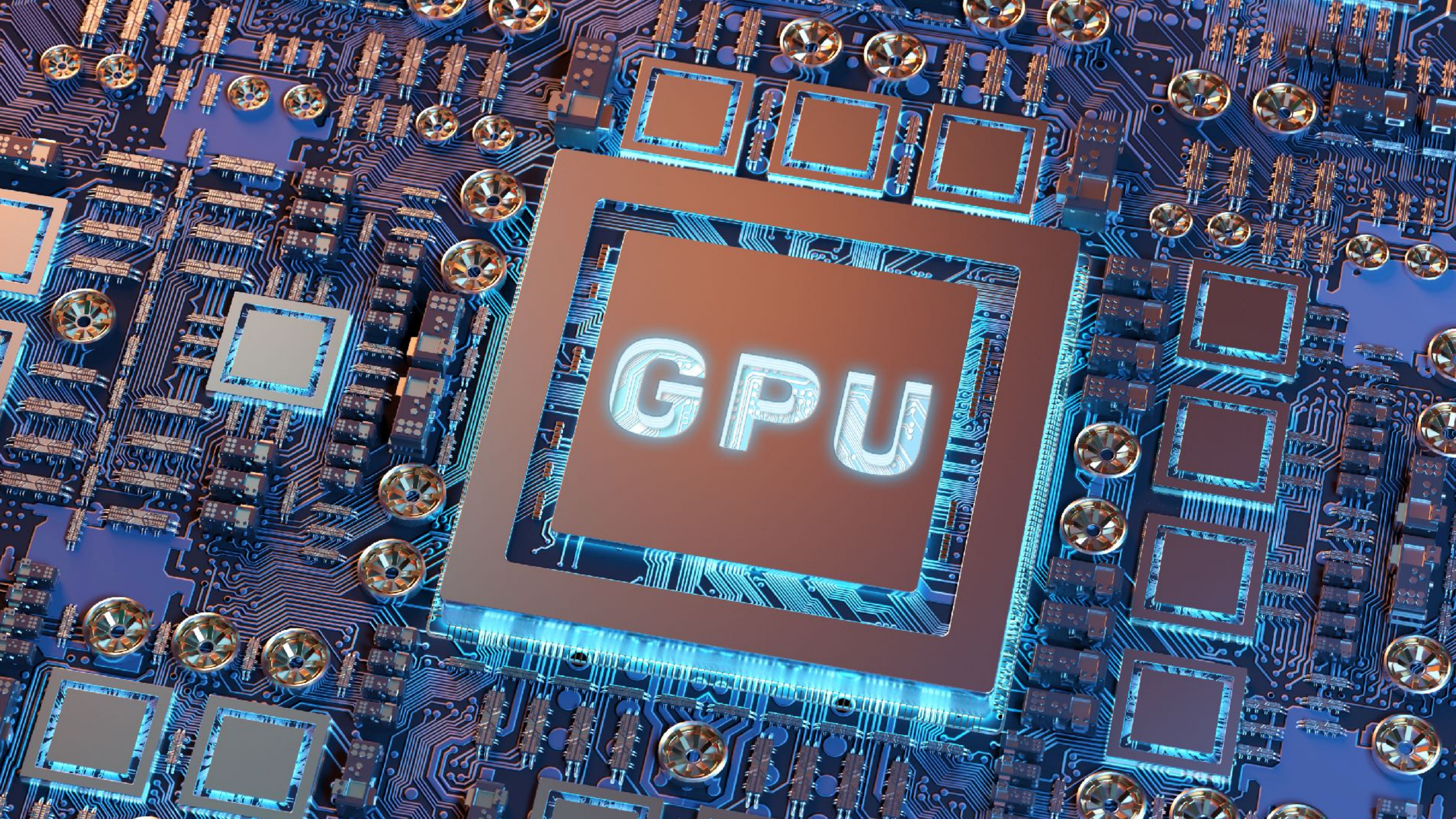
NEURAL NETWORKS

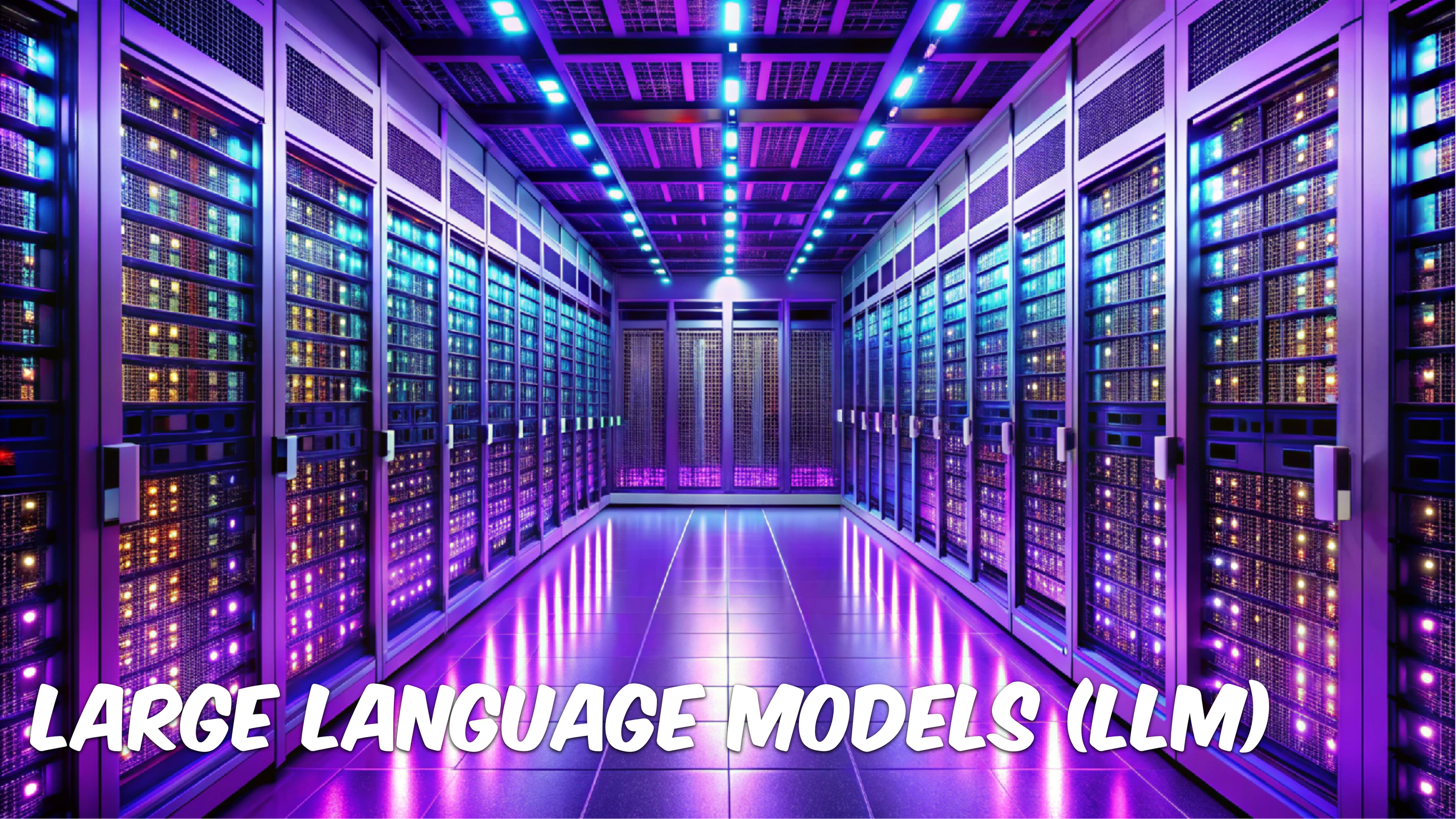
What makes deep learning powerful?

- The first layer might detect edges
- The next layer might recognize shapes
- Deeper layers could identify more complex features like eyes or wheels
- The final layer puts it all together to classify the entire image

ARTIFICIAL NEURAL NETWORK

- Scientific Advances - Deep Learning
- Availability of Big Data (You need data to configure these neural networks)
- Lots of compute power


GPU


LARGE LANGUAGE MODELS (LLM)

ATTENTION IS ALL YOU NEED

Bigger is Better

- Very Large Neural Networks
- Vast Amounts of Training Data
- Huge Compute Power to train data
- General Purpose AI

Attention Is All You Need

Ashish Vaswani[*]
Google Brain
avaswani@google.com

Noam Shazeer[*]
Google Brain
noam@google.com

Niki Parmar[*]
Google Research
nikip@google.com

Jakob Uszkoreit[*]
Google Research
usz@google.com

Llion Jones[*]
Google Research
llion@google.com

Aidan N. Gomez[*] [†]
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser[*]
Google Brain
lukaszkaiser@google.com

Illia Polosukhin[*] [‡]
illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature.

1 Introduction

Recurrent neural networks, long short-term memory [12] and gated recurrent [7] neural networks in particular, have been firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation [29, 2, 5]. Numerous efforts have since continued to push the boundaries of recurrent language models and encoder-decoder architectures [31, 21, 13].

[*]Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

[†]Work performed while at Google Brain.

[‡]Work performed while at Google Research.

ATTENTION IS ALL YOU NEED

Transformer Architecture

- Not Just a big neural network
- Specialized architecture for token prediction
- Key Innovation
  - Attention Mechanisms

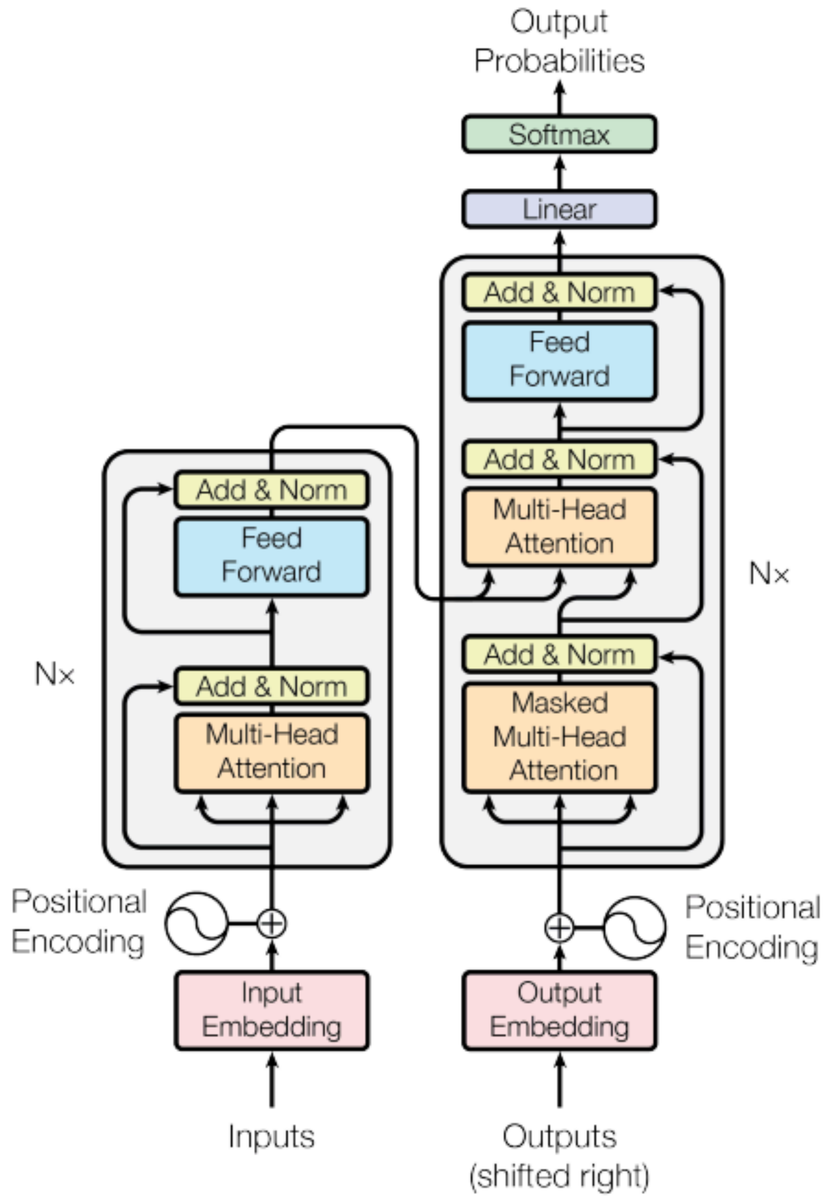


Figure 1: The Transformer - model architecture.

LARGE LANGUAGE MODELS

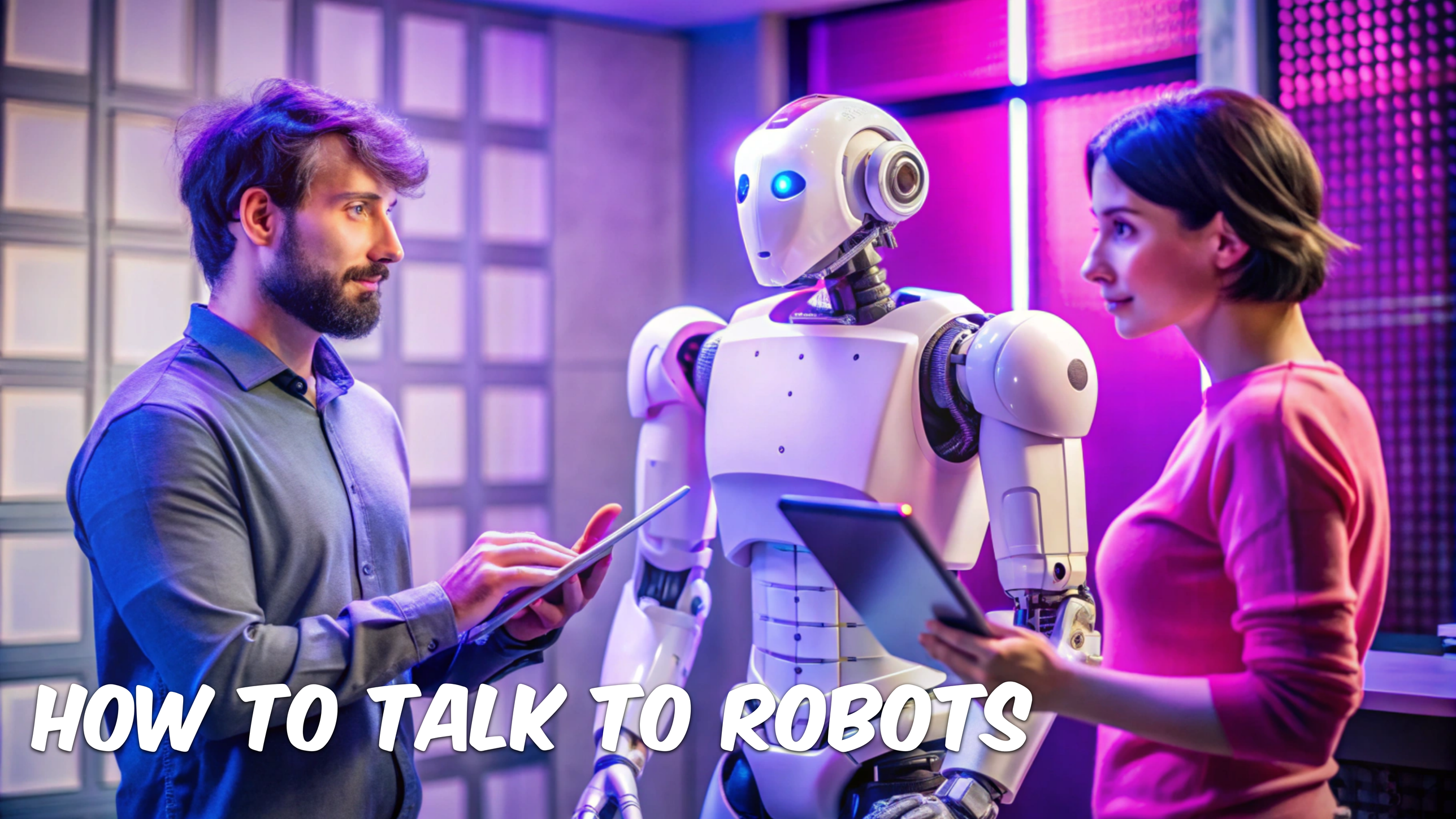
What sets LLMs apart

- Their immense scale and their ability to process and generate language in a way that often seems remarkably human-like
- These models are trained on vast amounts of text data, allowing them to capture intricate patterns and nuances in languages.
- To give you an idea of their scale:
  - GPT-3, introduced in 2020, contained **175 Billion** parameters
  - More recent models, like GPT-4 and some from other companies, are believed to be even larger, though the exact sizes aren’t always disclosed.

GENERATIVE AI

Generative Pre-Trained Transformer (GPT)

Gemini


HOW TO TALK TO ROBOTS

PROMPT ENGINEERING

Learn how to effectively communicate with AI

- Not a huge fan of the term “Prompt Engineering”
- Communication is important in the real world
  - Same applies to interacting with LLMs
- Write me a blog post on “Topic”
- https://www.bytesizedai.dev/p/how-to-talk-to-robots

MESSAGE TYPES

There are several types of messages that can be sent to a language model in a conversation.

- User: Messages from the human or client user interacting with the model.
- Assistant: Responses generated by the AI assistant (the language model).
- System: Instructions or context provided to the model at the beginning of a conversation to guide its behavior and set parameters. These are typically not visible to the end user.
- Function: Used in function calling scenarios, where the model can call pre-defined functions to perform specific tasks or retrieve information.


JAVA & AI

JAVA & AI

Leveraging Artificial Intelligence in Java Applications

- Why AI + Java?
  - The world of software is experiencing widespread adoption of Artificial Intelligence
  - Java is the language of enterprises, creating Java + AI apps is a new requirement
- Spring AI
  - Provides the necessary API access and components for developer AI applications
  - Abstraction similar to Spring Data
- Use Cases
  - Q&A Over docs
  - Documentation Summarization
  - Text, Code, Image, Audio and Video Generation

```
#!/bin/bash
echo "Calling Open AI..."
MY_OPENAI_KEY=“YOUR_API_KEY_HERE”
PROMPT=“Tell me an interesting fact about Java"

curl https://api.openai.com/v1/chat/completions \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer $MY_OPENAI_KEY" \
  -d '{"model": "gpt-4o", "messages": [{"role":"user", "content": "'"${PROMPT}"'"}] }'
```

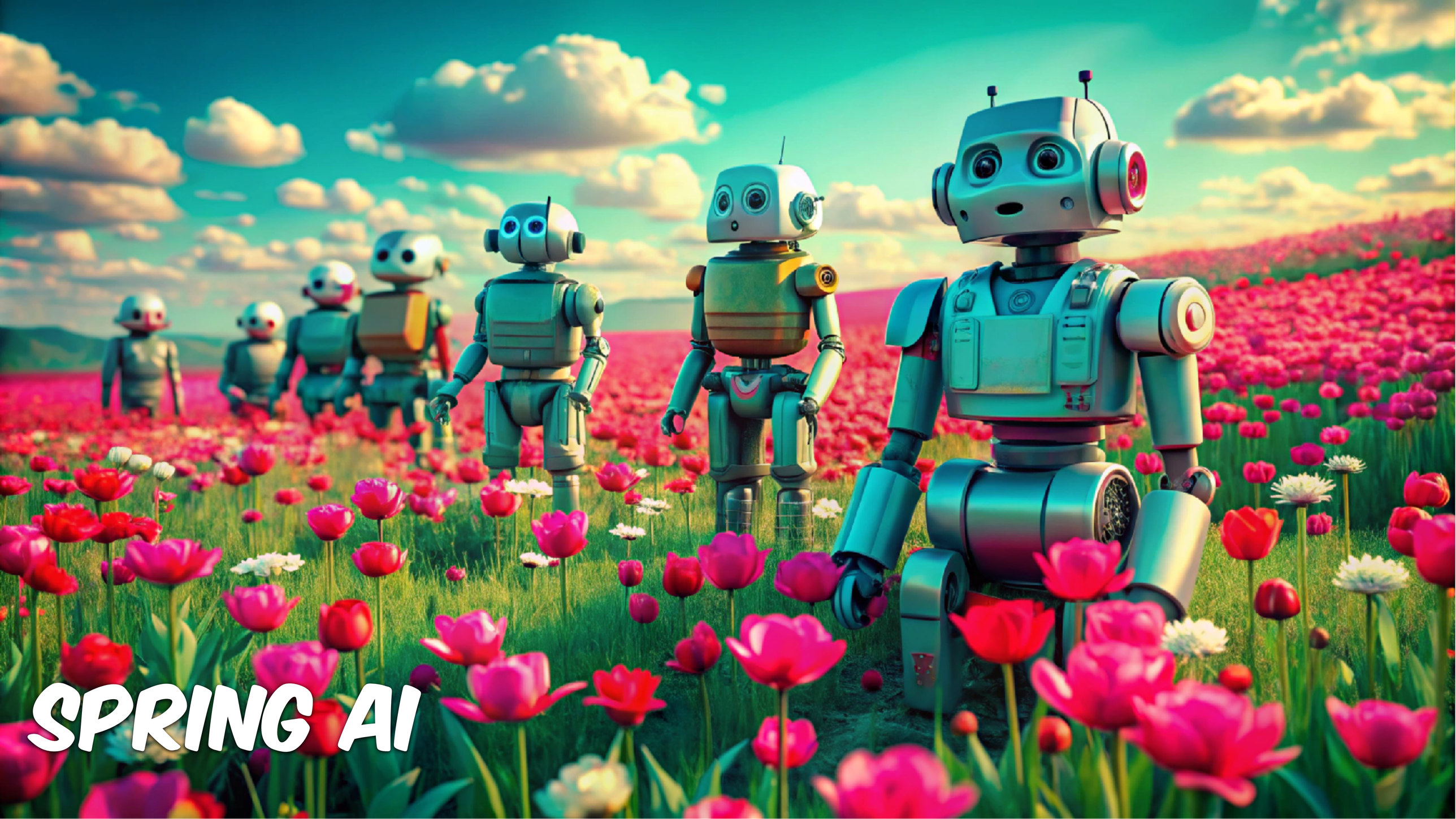
```
{
  "id": "chatcmpl-ABNbjZ5oRbo72OevnCX2arPufJCYK",
  "object": "chat.completion",
  "created": 1727275719,
  "model": "gpt-4o-2024-05-13",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "Sure! Did you know that Java was initially designed with interactive television in mind? James Gosling and his team at Sun Microsystems started the project in 1991 under the name \"Oak.\" The name was later changed to \"Java\" after discovering there was already a programming language called Oak. Java's versatility has made it one of the most popular programming languages for a wide range of applications, far beyond its initial intended use for TV set-top boxes!",
        "refusal": null
      },
      "logprobs": null,
      "finish_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 14,
    "completion_tokens": 90,
    "total_tokens": 104,
    "completion_tokens_details": {
      "reasoning_tokens": 0
    }
  },
  "system_fingerprint": "fp_e375328146"
```

```java
public static void main(String[] args) throws IOException, InterruptedException {
    var apiKey = "YOUR_API_KEY_HERE";
    var body = """
            {
                "model": "gpt-4o",
                "messages": [
                    {
                        "role": "user",
                        "content": "Tell me an interesting fact about Java"
                    }
                ]
            }""";

    HttpRequest request = HttpRequest.newBuilder()
            .uri(URI.create("https://api.openai.com/v1/chat/completions"))
            .header("Content-Type", "application/json")
            .header("Authorization", "Bearer " + apiKey)
            .POST(HttpRequest.BodyPublishers.ofString(body))
            .build();

    var client = HttpClient.newHttpClient();
    var response = client.send(request, HttpResponse.BodyHandlers.ofString());
    System.out.println(response.body());
}
```

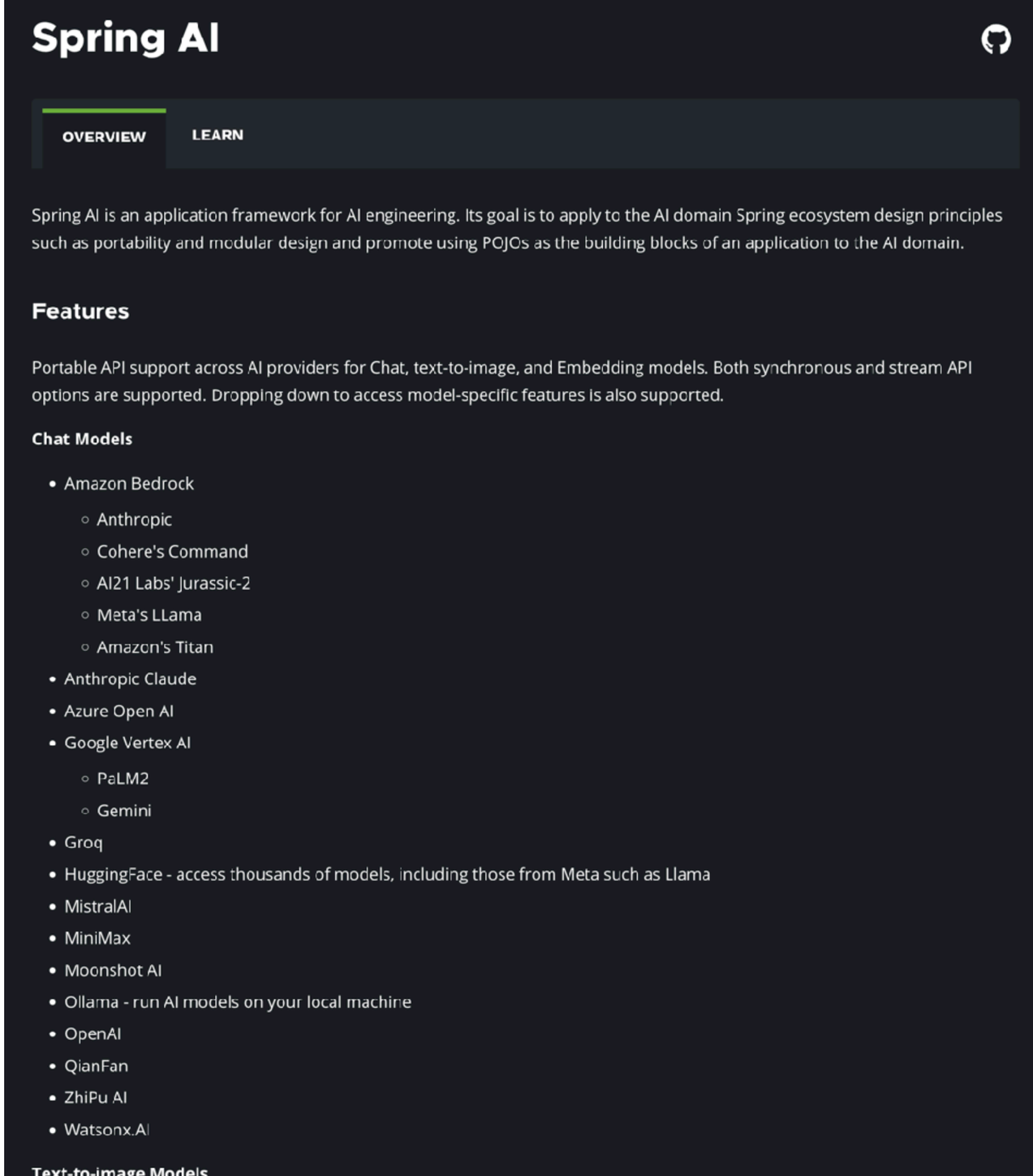
SPRING AI PROVIDES US SO MUCH MORE THAN A FACILITY FOR MAKING REST API CALLS


SPRING AI

SPRING AI

AI for Spring Developers

- https://spring.io/projects/spring-ai
  - Dr. Mark Pollack
  - Current Version 1.0.0-M2
- Portable API support across AI providers for Chat, Image & Audio
- Synchronous & Streaming API options
- Inspired by Python Projects
  - LangChain
  - LlamaIndex



Spring AI

OVERVIEW LEARN

Spring AI is an application framework for AI engineering. Its goal is to apply to the AI domain Spring ecosystem design principles such as portability and modular design and promote using POJOs as the building blocks of an application to the AI domain.

Features

Portable API support across AI providers for Chat, text-to-image, and Embedding models. Both synchronous and stream API options are supported. Dropping down to access model-specific features is also supported.

Chat Models

- Amazon Bedrock
  - Anthropic
  - Cohere's Command
  - AI21 Labs' Jurassic-2
  - Meta's LLama
  - Amazon's Titan
- Anthropic Claude
- Azure Open AI
- Google Vertex AI
  - PaLM2
  - Gemini
- Groq
- HuggingFace - access thousands of models, including those from Meta such as Llama
- MistralAI
- MiniMax
- Moonshot AI
- Ollama - run AI models on your local machine
- OpenAI
- QianFan
- ZhiPu AI
- Watsonx.AI

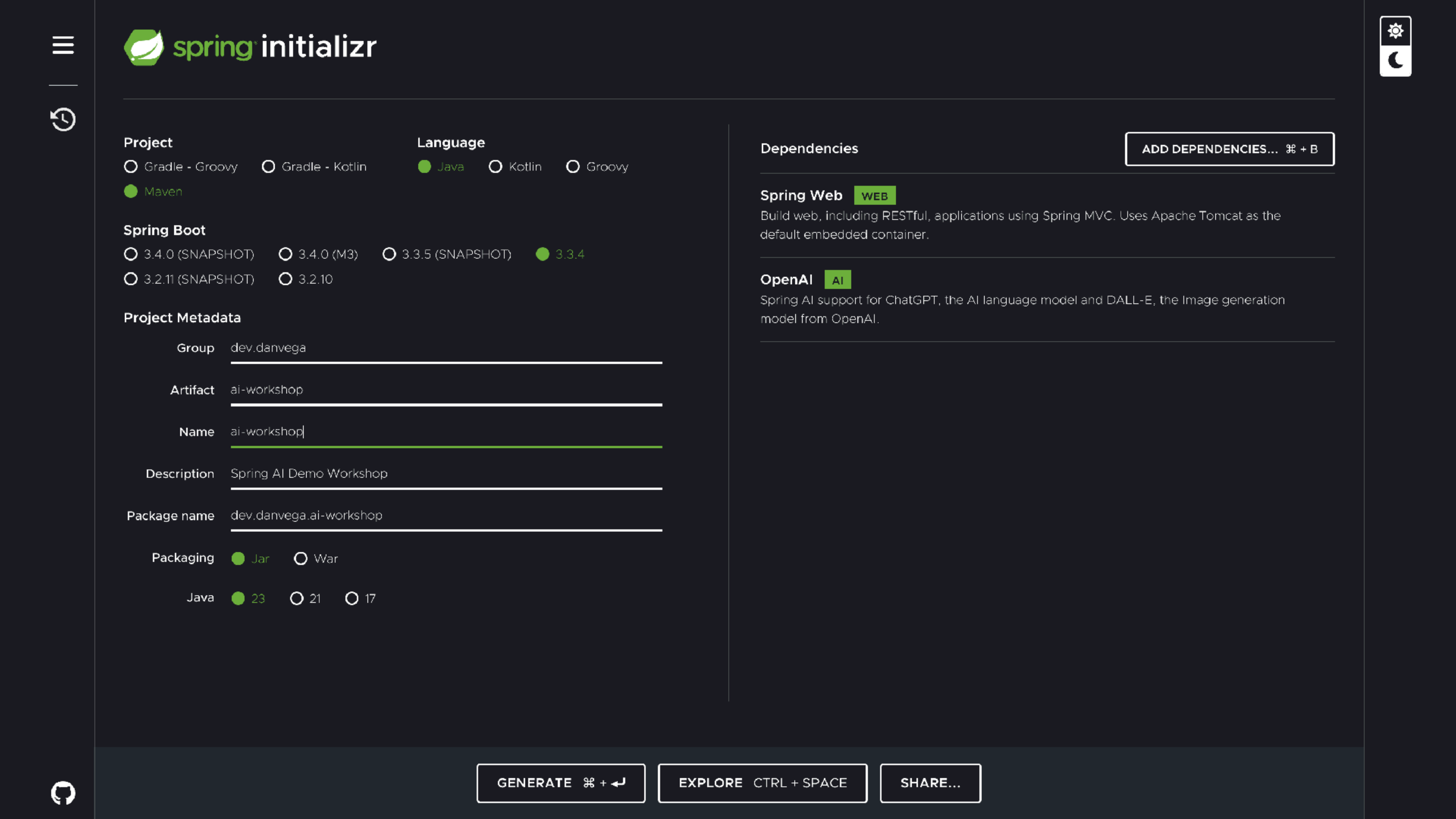
Text-to-image Models

SPRING AI API

The Spring AI API covers a wide range of functionality

- Chat Model: Give it some text, get some text back
- Text to Image: Give it some text, get an image back
- Transcription: Give it some audio, get some text back
- Embedding: Portable API across Vector Store Providers
- Functions: AI model to invoke your POJO (*java.util.Function*)

GETTING STARTED


spring initializr
Project
Gradle - Groovy
Gradle - Kotlin
Maven
Language
Java
Kotlin
Groovy
Spring Boot
3.4.0 (SNAPSHOT)
3.4.0 (M3)
3.3.5 (SNAPSHOT)
3.3.4
3.2.11 (SNAPSHOT)
3.2.10
Project Metadata
Group
dev.danvega
Artifact
ai-workshop
Name
ai-workshop
Description
Spring AI Demo Workshop
Package name
dev.danvega.ai-workshop
Packaging
Jar
War
Java
23
21
17
Dependencies
ADD DEPENDENCIES... ⌘ + B
Spring Web
WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.
OpenAI
AI
Spring AI support for ChatGPT, the AI language model and DALL-E, the Image generation model from OpenAI.
GENERATE ⌘ + ↵
EXPLORE CTRL + SPACE
SHARE...

PICK YOUR AI MODEL

We will be using Open AI

- **Open AI**
- Anthropic 3
- Google Gemini
- Groq
- Hugging Face
- Mistral AI
- Ollama

Playground
Assistants
Fine-tuning
API keys
Storage
Usage
Settings

API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically disable any API key that we've found has leaked publicly.

Enable tracking to see usage per API key on the Usage page.

NAME	SECRET KEY	TRACKING	CREATED	LAST USED	PERMISSIONS	
Secret key	sk-...aWYE	+ Enable	Feb 6, 2023	Never	All	
spring-ai	sk-...NKve	Enabled	Mar 12, 2024	Mar 13, 2024	All	

+ Create new secret key

Default organization

If you belong to multiple organizations, this setting controls which organization is used by default when making requests with the API keys above.

Personal

Note: You can also specify which organization to use for each API request. See Authentication to learn more.

TOKENS

GPT-4o

GPT-4o is our most advanced multimodal model that's faster and cheaper than GPT-4 Turbo with stronger vision capabilities. The model has 128K context and an October 2023 knowledge cutoff.

Learn about GPT-4o ↗

Model	Pricing	Pricing with Batch API*
gpt-4o	$5.00 / 1M input tokens	$2.50 / 1M input tokens
	$15.00 / 1M output tokens	$7.50 / 1M output tokens
gpt-4o-2024-08-06	$2.50 / 1M input tokens	$1.25 / 1M input tokens
	$10.00 / 1M output tokens	$5.00 / 1M output tokens
gpt-4o-2024-05-13	$5.00 / 1M input tokens	$2.50 / 1M input tokens
	$15.00 / 1M output tokens	$7.50 / 1M output tokens

Vision pricing calculator

Set model

gpt-4o-2024-08-06

Set width: 150 px by Set height: 150 px = $0.000638

Low resolution

*Batch API pricing requires requests to be submitted as a batch. Responses will be returned within 24 hours for a 50% discount. Learn more about Batch API ↗

Model	Pricing	Pricing with Batch API*
gpt-4o	$5.00 / 1M input tokens	$2.50 / 1M input tokens
	$15.00 / 1M output tokens	$7.50 / 1M output tokens
gpt-4o-2024-08-06	$2.50 / 1M input tokens	$1.25 / 1M input tokens
	$10.00 / 1M output tokens	$5.00 / 1M output tokens

https://platform.openai.com/tokenizer

Tokenizer

Learn about language model tokenization

OpenAI's large language models (sometimes referred to as GPT's) process text using **tokens**, which are common sequences of characters found in a set of text. The models learn to understand the statistical relationships between these tokens, and excel at producing the next token in a sequence of tokens.

You can use the tool below to understand how a piece of text might be tokenized by a language model, and the total count of tokens in that piece of text.

It's important to note that the exact tokenization process varies between models. Newer models like GPT-3.5 and GPT-4 use a different tokenizer than previous models, and will produce different tokens for the same input text.

GPT-3.5 & GPT-4 | GPT-3 (Legacy)

```
Hello, My name is Dan Vega, Java Champion, Spring Developer Advocate,
Husband and #GirlDad based outside of Cleveland OH. I created this
website as a place to document my journey as I learn new things and share
them with you. I have a real passion for teaching and I hope that one of
blog posts, videos or courses helps you solve a problem or learn
something new.
```

Clear | Show example

Tokens 78 **Characters** 363

Hello, My name is Dan Vega, Java Champion, Spring Developer Advocate, Husband and #GirlDad based outside of Cleveland OH. I created this website as a place to document my journey as I learn new things and share them with you. I have a real passion for teaching and I hope that one of blog posts, videos or courses helps you solve a problem or learn something new.

Text | Token IDs

A helpful rule of thumb is that one token generally corresponds to ~4 characters of text for common English text. This translates to roughly ¾ of a word (so 100 tokens ~= 75 words).

```
spring.application.name=hello-spring-ai
spring.ai.openai.api-key=YOUR_KEY_HERE
spring.ai.openai.chat.options.model=gpt-4o
```

Configuration Properties

The prefix `spring.ai.openai.chat` is the property prefix that lets you configure the chat client implementation for OpenAI.

Property	Description	Default
spring.ai.openai.chat.enabled	Enable OpenAI chat client.	true
spring.ai.openai.chat.base-url	Optional overrides the spring.ai.openai.base-url to provide chat specific url	-
spring.ai.openai.chat.api-key	Optional overrides the spring.ai.openai.api-key to provide chat specific api-key	-
spring.ai.openai.chat.options.model	This is the OpenAI Chat model to use	`gpt-3.5-turbo` (the `gpt-3.5-turbo`, `gpt-4`, and `gpt-4-32k` point to the latest model versions)
spring.ai.openai.chat.options.temperature	The sampling temperature to use that controls the apparent creativity of generated completions. Higher values will make output more random while lower values will make results more focused and deterministic. It is not recommended to modify temperature and top_p for the same completions request as the interaction of these two settings is difficult to predict.	0.8
spring.ai.openai.chat.options.frequencyPenalty	Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat the same line verbatim.	0.0f
spring.ai.openai.chat.options.logitBias	Modify the likelihood of specified tokens appearing in the completion.	-
spring.ai.openai.chat.options.maxTokens	The maximum number of tokens to generate in the chat completion. The total length of input tokens and generated tokens is limited by the model's context length.	-
spring.ai.openai.chat.options.n	How many chat completion choices to generate for each input message. Note that you will be charged based on the number of generated tokens across all of the choices. Keep n as 1 to minimize costs.	1
spring.ai.openai.chat.options.presencePenalty	Number between -2.0 and 2.0. Positive values penalize new tokens based on whether they appear in the text so far, increasing the model's likelihood to talk about new topics.	-
spring.ai.openai.chat.options.responseFormat	An object specifying the format that the model must output. Setting to `{ "type": "json_object" }` enables JSON mode, which guarantees the message the model generates is valid JSON.	-

CHECK OUT MY DEMO

THANK YOU

dan.vega@broadcom.com

@therealdanvega

https://www.danvega.dev

https://www.bytesizedai.dev