Build with AI

Google Developer Groups

Google Developer Experts

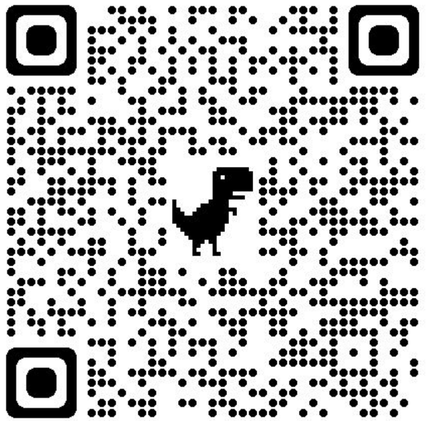
Multimodality with Gemini

Henry Ruiz
Research Scientist
Texas A&M University AgriLife Research
GDE ML
@devharuiz
https://haruiz.github.io/

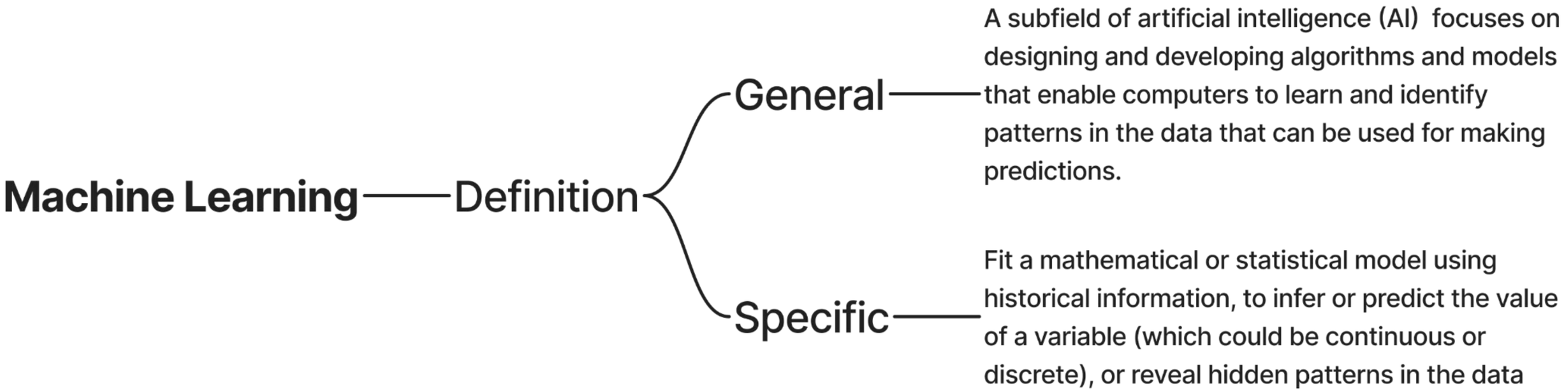
Gemini

Table of Contents

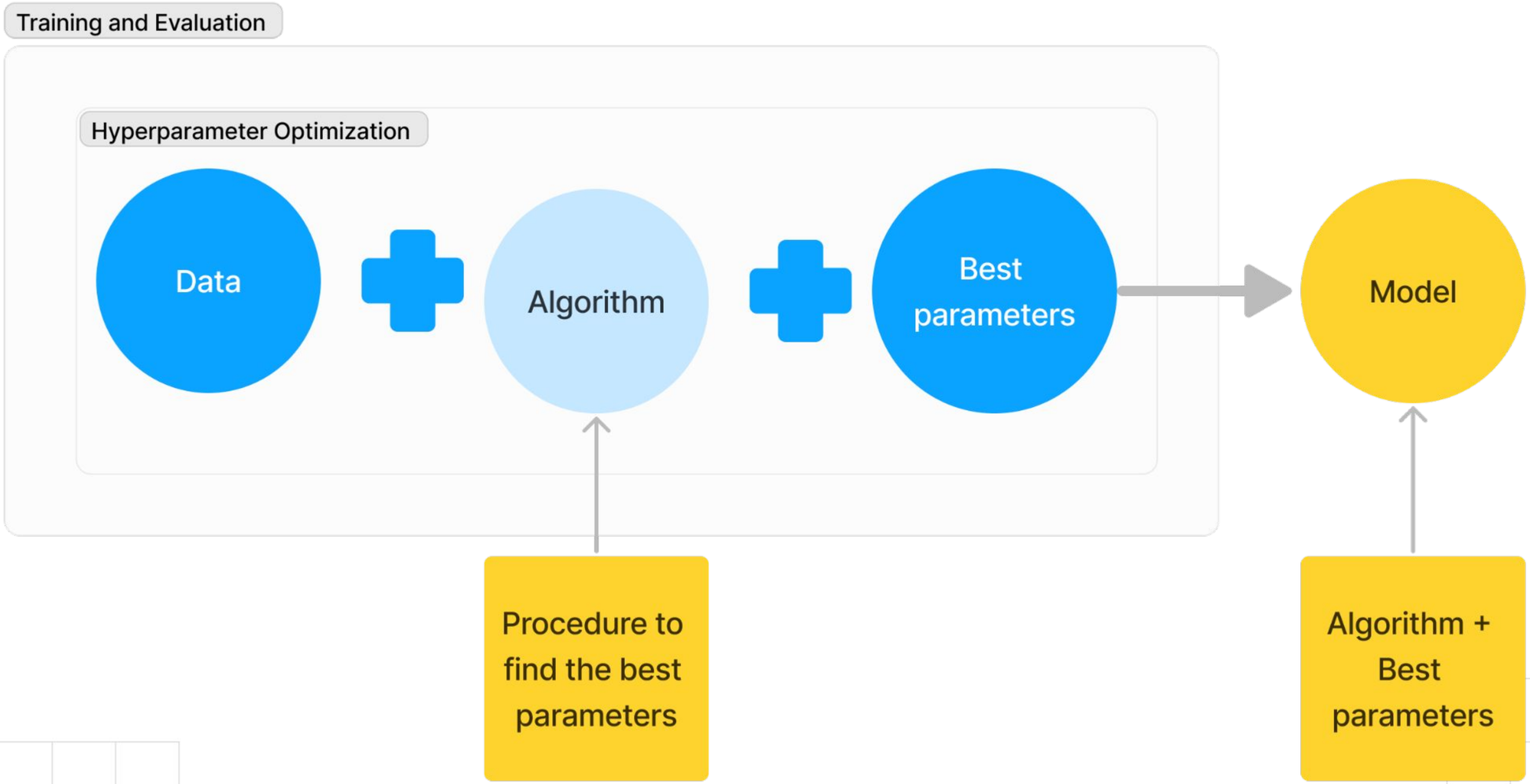
Access to the notebook



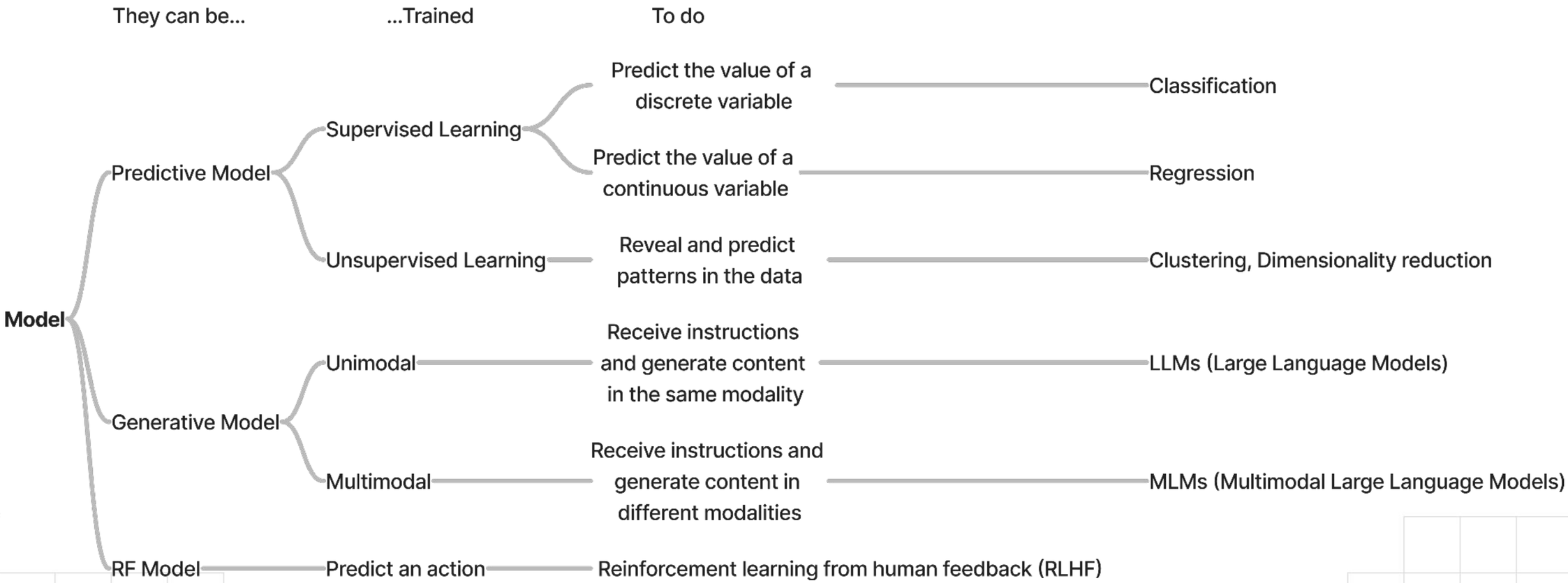
What is Machine Learning?



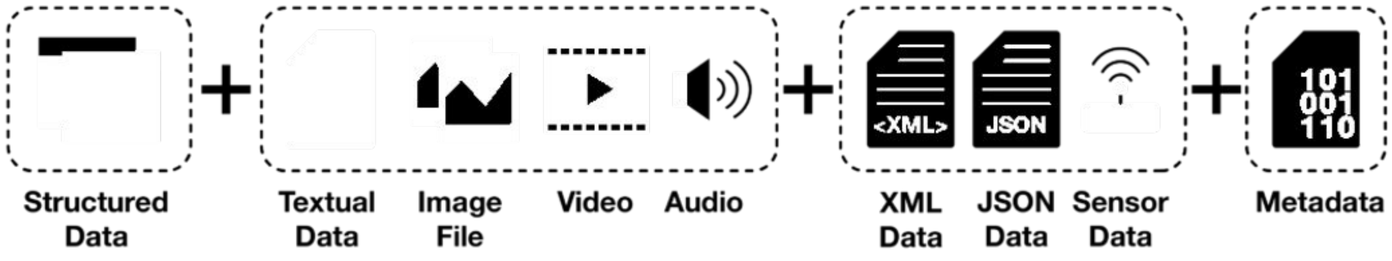
….The learning process involves

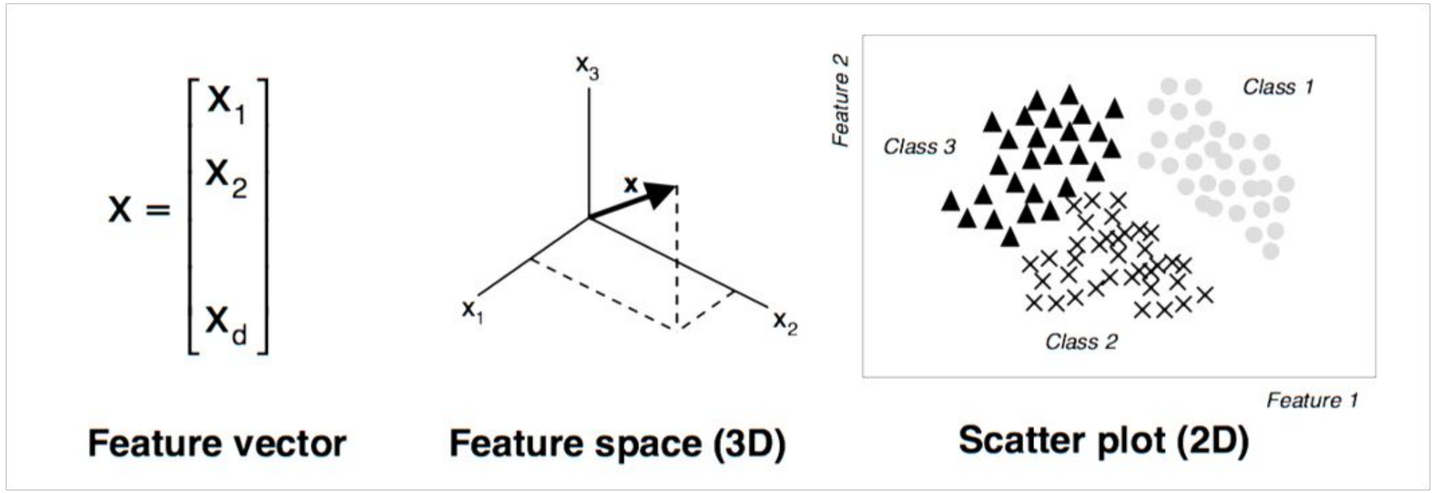


The Model can be trained for… (Most common tasks)



But, In the end, ML models only understand vectors and numbers

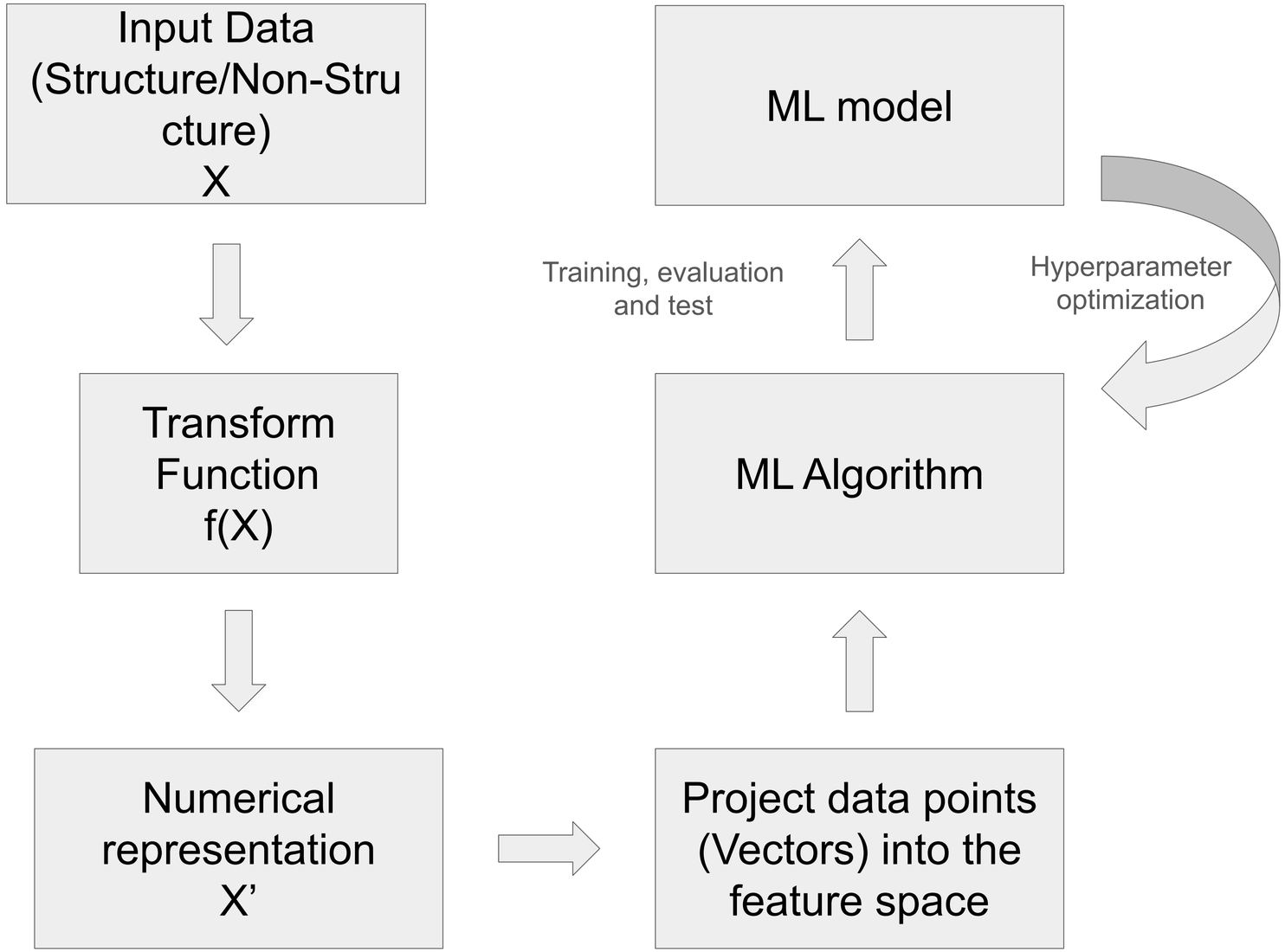




Before or during training the data which could be unstructured or structure must be transformed into a vectorial representation, embedding and projected into the features space.

In the feature, space is where the “magic” happens, and the algorithms are applied (as vectors can be mathematically operated).

Transforming my data into embeddings



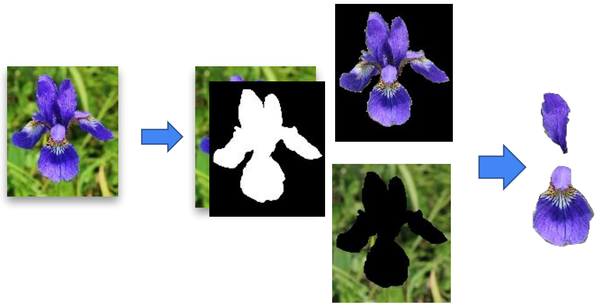
That numerical representation can be hand-crafted,
applying features engineering

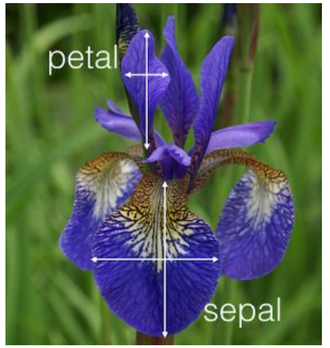
or learned during training.

Example: You have been asked to develop a mobile app in Flutter to classify flowers from pictures.

Before deep learning (using traditional ML models and feature engineering)

Features Engineering Approach (Features are known)





	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa

Traditional ML relies on your Feature engineering skills to transform your data into numbers/embeddings

segmentation

Representation and Description (Feature Engineering)

Pre-processing

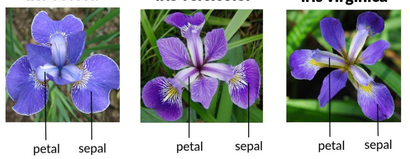
Knowledge Base (Features Store)

Recognition and interpretation (ML Modeling)

Results

Proficiency issue

Image Acquisition

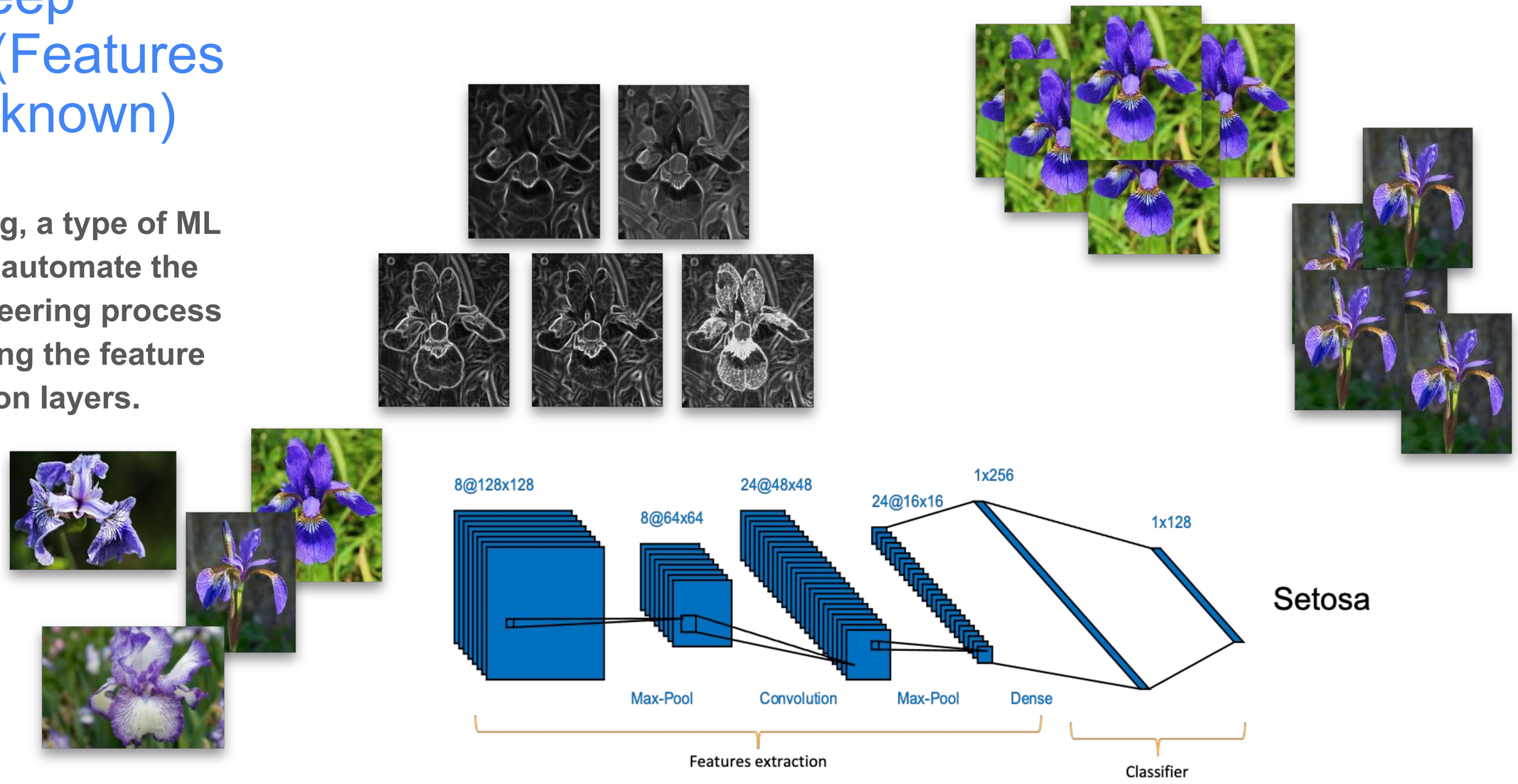


Hand Crafted feature extraction using image processing

Using Deep Learning algorithms (Automating the feature engineering)

Deep learning(Features are unknown)

Deep Learning, a type of ML algorithms, automate the feature engineering process by introducing the feature extraction layers.



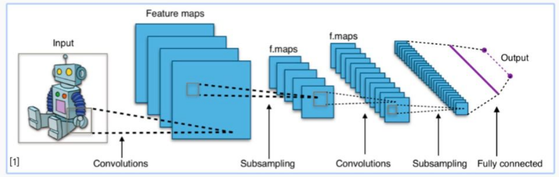
Automating Feature Extraction through Deep Learning: A Breakthrough in Machine Learning Advances

Before each task had its own DNN architecture

Therefore, each deep architecture, depending of the task, had is own building blocks (feature extraction layers, lost function etc.)

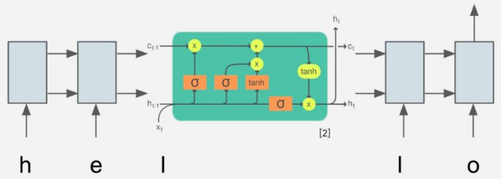
Computer Vision

Convolutional NNs (+ResNets)



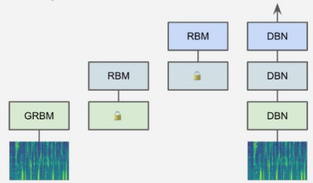
Natural Lang. Proc.

Recurrent NNs (+LSTMs)



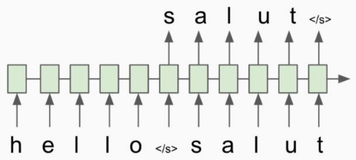
Speech

Deep Belief Nets (+non-DL)



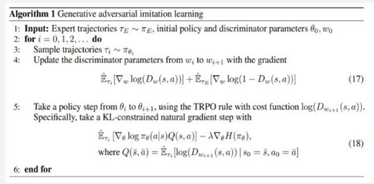
Translation

Seq2Seq



RL

BC/GAIL



Algorithm 1 Generative adversarial imitation learning

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0
2: **for** $i = 0, 1, 2, \ldots$ **do**
3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4: Update the discriminator parameters from w_i to w_{i+1} with the gradient

$$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \quad (17)$$

5: Take a policy step from θ_i to θ_{i+1}, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$. Specifically, take a KL-constrained natural gradient step with

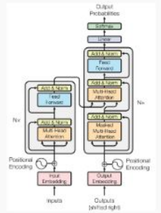
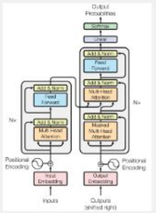
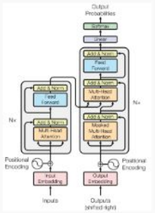
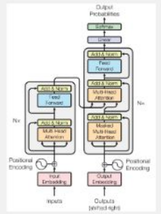
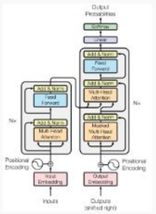
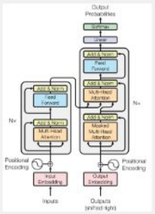
$$\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta),$$
$$\text{where } Q(\bar{s},\bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \,|\, s_0 = \bar{s}, a_0 = \bar{a}] \quad (18)$$

6: **end for**

[1] CNN image CC-BY-SA by Aphex34 for Wikipedia https://commons.wikimedia.org/wiki/File:Typical_cnn.png
[2] RNN image CC-BY-SA by GChe for Wikipedia https://commons.wikimedia.org/wiki/File:The_LSTM_Cell.svg

Now: Transformers is all you need

The transformer architecture, introduced in the paper "Attention is All You Need" by Vaswani et al. in 2017, has revolutionized the field of natural language processing (NLP) and beyond

Computer Vision	Natural Lang. Proc.	Reinf. Learning
		
Speech	**Translation**	**Graphs/Science**
		

Transformer image source: "Attention Is All You Need" paper

https://t.co/aYfnVKPDjT

https://www.asimovinstitute.org/neural-network-zoo/

https://jalammar.github.io/illustrated-transformer/

Experts

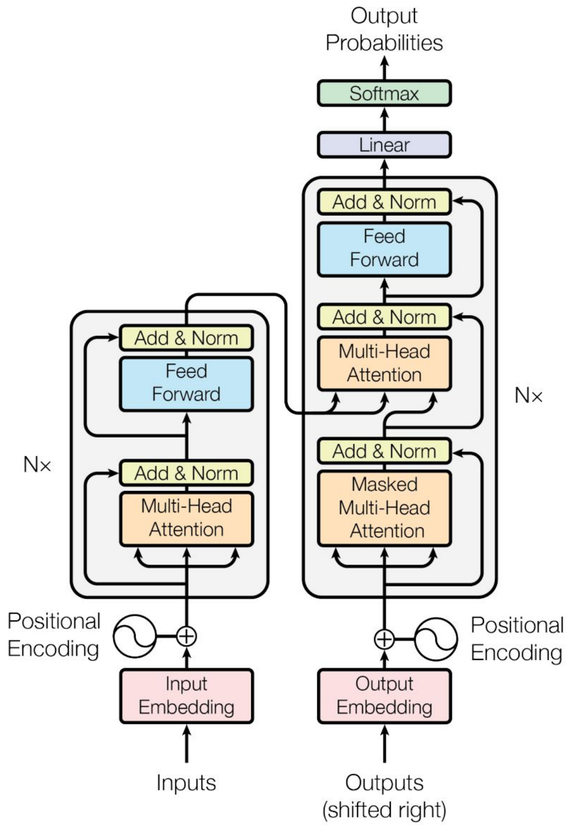
Google Developers

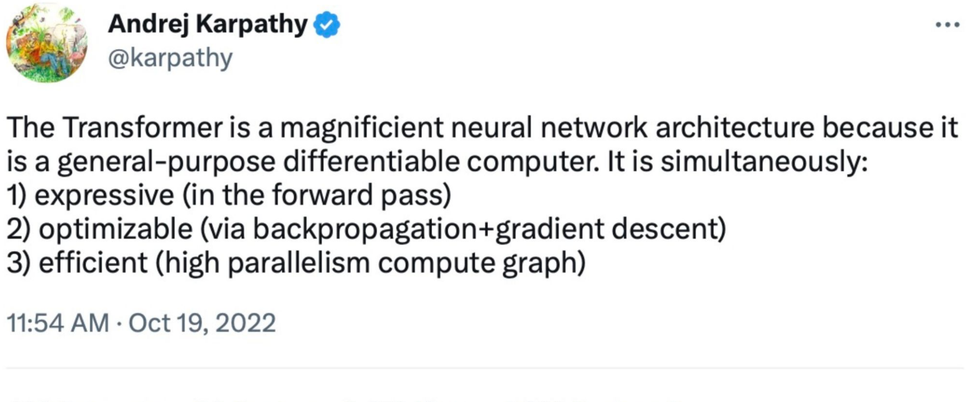
Transformers novelty

Attention Mechanism: Attention mechanisms allows the model to focus on relevant parts of the input sequence during processing.

Parallelization: Input sequences can be process in parallel due to their architecture design, leading to faster training times compared to sequential models that process inputs sequentially.

Scalability: Transformers are more scalable to longer sequences compared to traditional recurrent models. They can process inputs of arbitrary length without increasing computational complexity significantly, making them suitable for tasks requiring long context understanding, such as machine translation and document summarization.





Experts

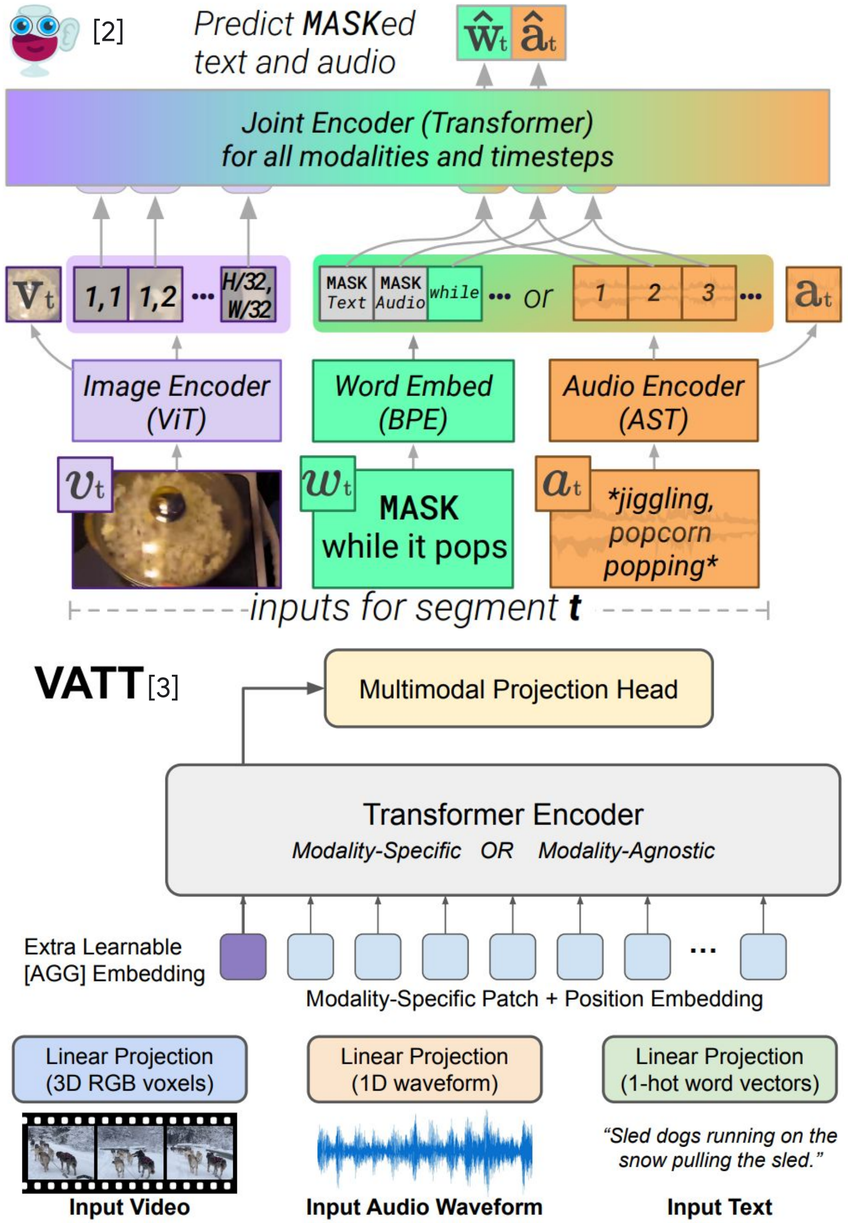
Anything you can tokenize, you can feed to Transformer

ca 2021 and onwards

Tokenize different modalities each in their own way (some kind of "patching"), and send them all jointly into a Transformer...
Seems to just work...
Currently an explosion of works doing this!

Tokens are vector representations of text, and embeddings are tokens with semantic context.

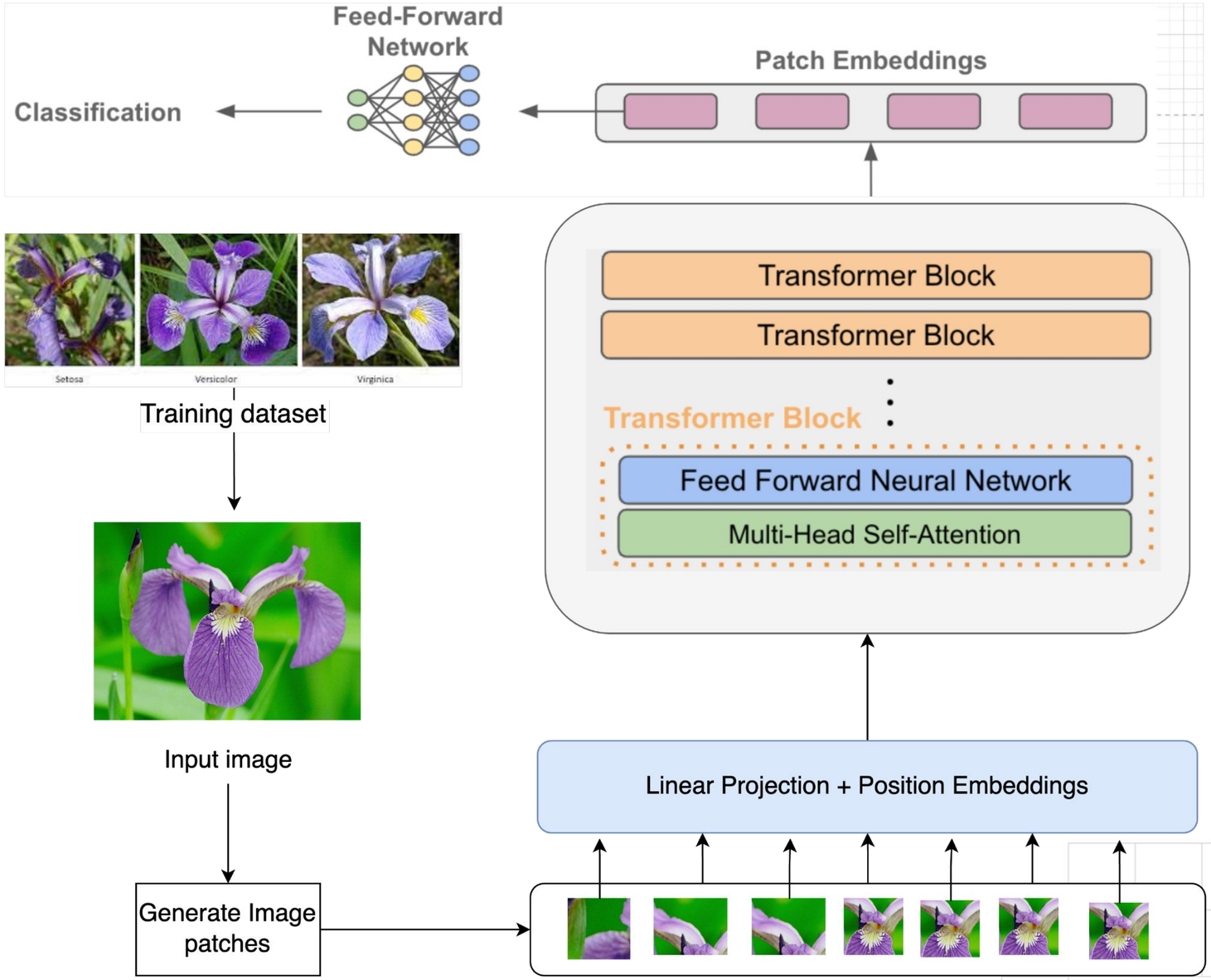
[1]



Images from:
[1] LIMoE by B Mustafa, C Riquelme, J Puigcerver, R Jenatton, N Houlsby
[2] MERLOT Reserve by R Zellers, J Lu, X Lu, Y Yu, Y Zhao, M Salehi, A Kusupati, J Hessel, A Farhadi, Y Choi
[3] VATT by H Akbari, L Yuan, R Qian, W-H Chuang, S-F Chang, Y Cui, B Gong

Transforming images into sequences

1. Convert an image into a sequence of flattened image patches
2. Pass the image patch sequence through the transformer model
3. Transform the input image into the **embeddings representation**
4. Take the first element of the transformer's output sequence and pass it through a classification module



Transformers

Transformers, In summary, **pushed forward the field of large language models and generative AI models** by addressing key limitations of traditional recurrent architectures, enabling more efficient processing of long-range dependencies, scalability to large datasets, and superior performance on various NLP tasks through pre-training and fine-tuning strategies.

Transformer is the backbone architecture for many state-of-the-art models, such as Gemini, GPT-4, DALL-E-2, Codex, and Gopher.

It was first proposed to solve the limitations of traditional models such as RNNs in handling variable-length sequences and context awareness.

Generative AI (GAI)

Generative AI is a **type of artificial intelligence (AI) that can create new content**, such as text, images, and music.

It does this by **learning from existing data** and then using that data to generate new outputs. The results after training is a probabilistic model that is able to generate new data.

The artificially generated data is expected to have the same statistical properties and structure as real-world data or our training dataset distribution.

How do these type of ML models work?

In technical terms, the concept of AI-guided content generation (AIGC) **involves utilizing advanced algorithms in generative AI to produce content that aligns with human instructions.**

This process typically entails two key stages: first,

1. **extracting the intended information from the given instructions**
2. **Transform that information into context in form of embedding that can be passed to the foundation model**
3. **generating content based on the identified intentions.**

By employing GAI algorithms, **AIGC enables the model to learn and be guided by human input**, resulting in the creation of content that fulfills the specified criteria.

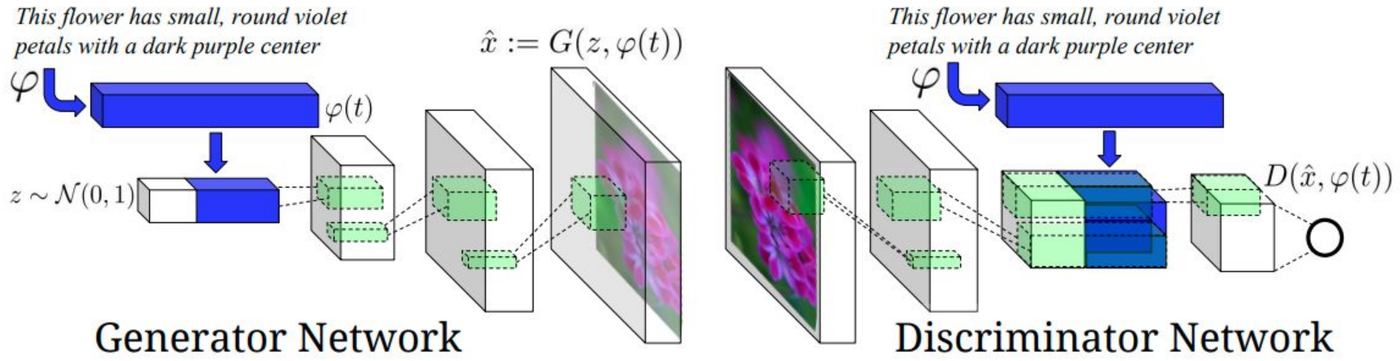


Figure 2. Our text-conditional convolutional GAN architecture. Text encoding $\varphi(t)$ is used by both generator and discriminator. It is projected to a lower-dimensions and depth concatenated with image feature maps for further stages of convolutional processing.

Generative Adversarial Text to Image Synthesis

A Comprehensive Survey of AI-Generated Content (AIGC): A History of Generative AI from GAN to ChatGPT

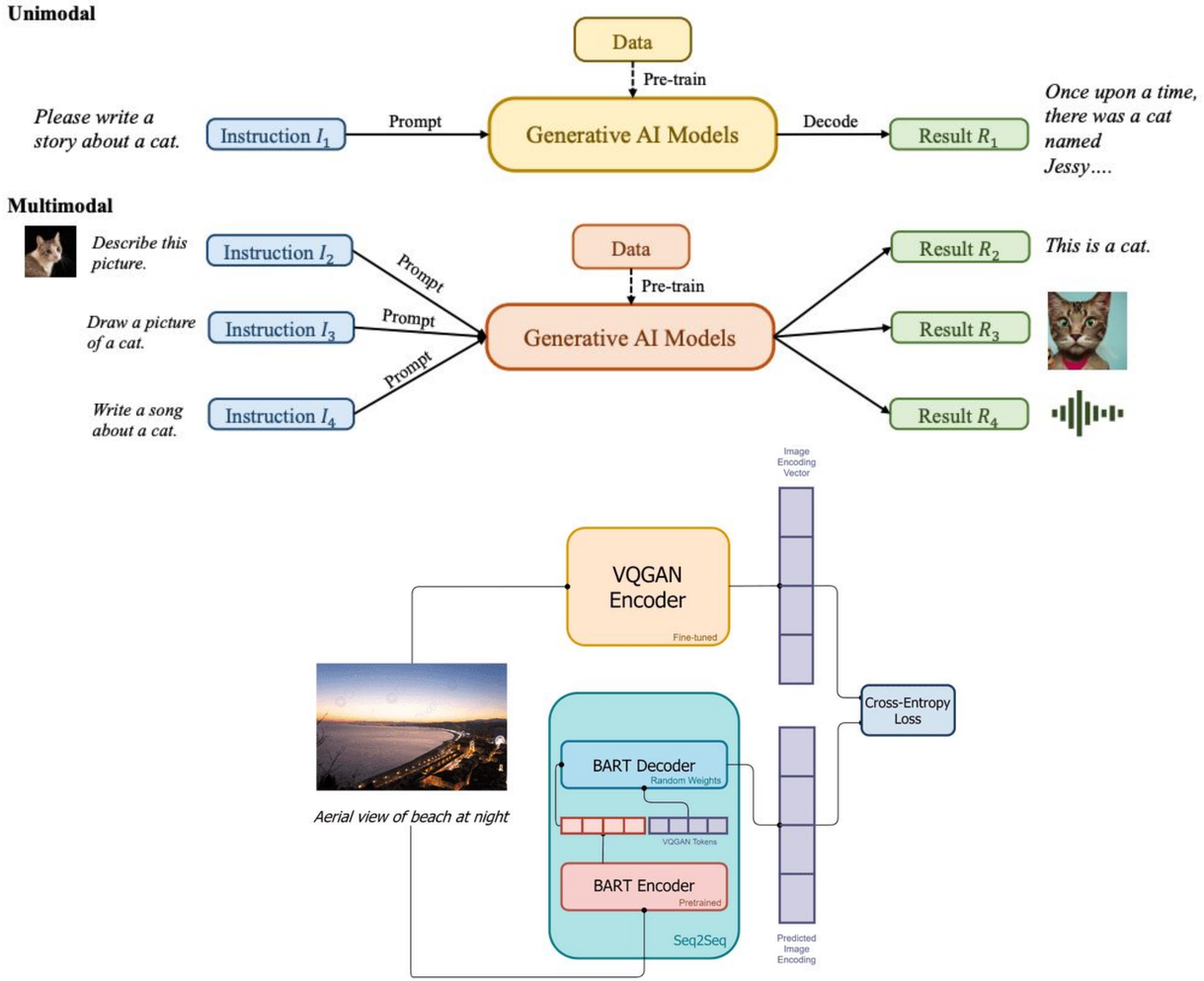
Unimodal vs. Multimodal

Generative models can be categorized into two types: unimodal models and multimodal models.

Unimodal models receive instructions and generate content in the same modality. For example, a unimodal text-to-text model would receive instructions in the form of text and generate text as output.

Multimodal models receive instructions and generate content in different modalities. For example, a multimodal image-to-text model would receive instructions in the form of an image and generate text as output.

Multimodal models are more complex than unimodal models, but they can also generate more creative and engaging content.



[A Comprehensive Survey of AI-Generated Content (AIGC): A History of Generative AI from GAN to ChatGPT]

[Vision Language models: towards multi-modal deep learning]

What are LLMs

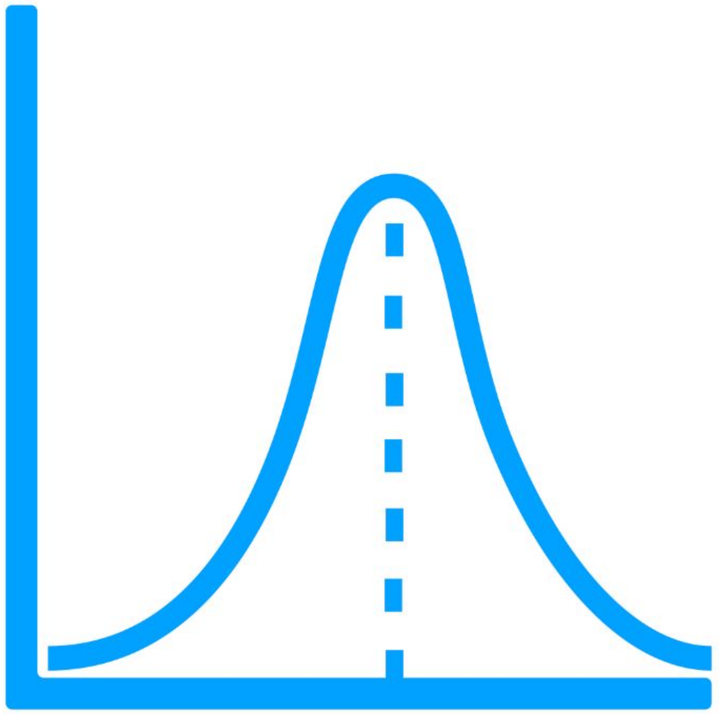
High Level:

A type of Generative AI architecture, that usually means a >1B parameter **transformer-based model** trained to predict the next token or sequence of tokens in a sequence

Low Level:

2 files in disk: weights and model

LMs are “just” statistical models of text

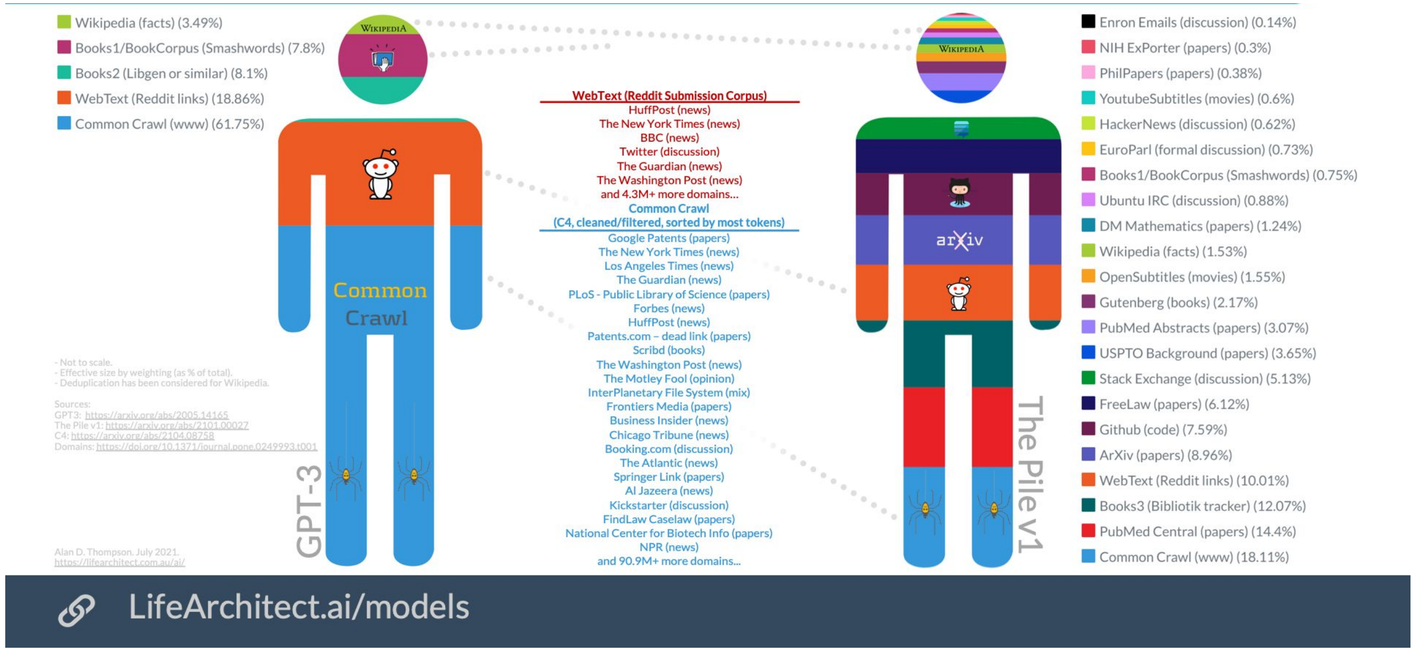


=

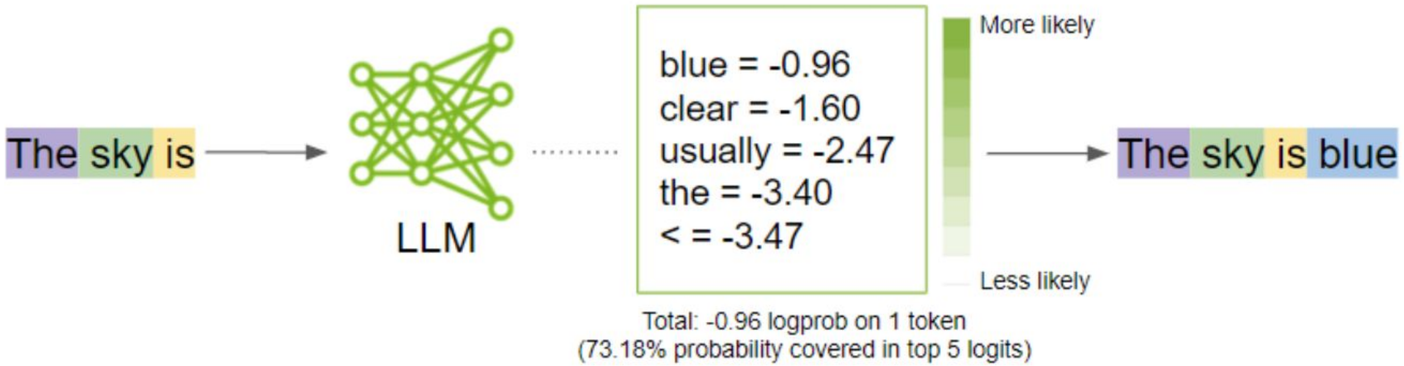
Gemini

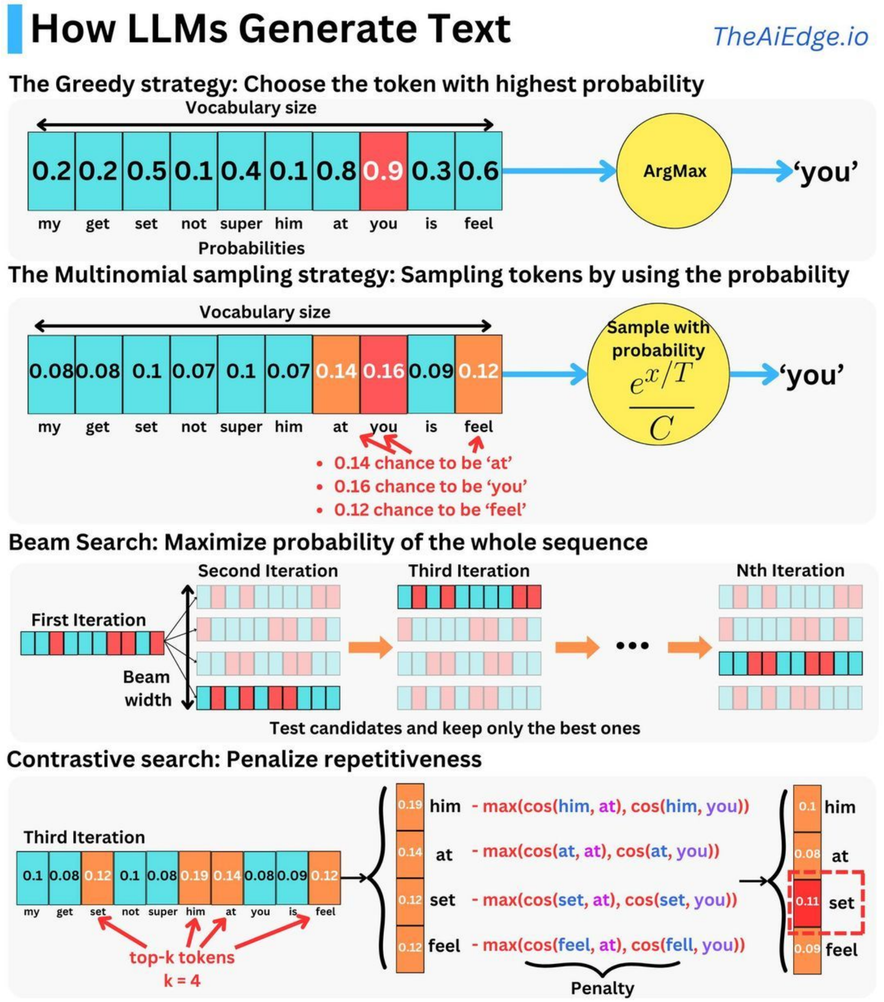
They assign a **probability** to every suffix of a prompt, token-by-token.

How does LIMs work?



LifeArchitect.ai/models





Credits to: https://newsletter.theaiedge.io/about

LLM APP pipeline

Interface

Application Programming Interface and LLMs orchestration tools

- PALM2/GPT4 API
- Hugging Face, Langchain, LLamaIndex

Conversational Interface

No Internet

- ChatGPT

Internet

- Bard Chat
- Bing chat

Input

- Labeled data
- Unlabeled data
- Training Code

LLM

Foundation Model (Multimodal/Unimodal) (Gemini, PaLM, LaMDA)

This is the key!!!

Output

Generate new content → Text generation

Generate new content → Code generation

Generate new content → Image generation

Google is the industry pioneer in AI. Foundations models evolution

2015

Google DeepMind AlphaGo defeats Go champion

2016

Google's DeepMind helps detect eye disease

2017

Google invents Transformer kickstarting LLM revolution

2018

Google's groundbreaking large language model, BERT

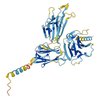
T5

2019

Text-to-Text Transfer Transformer LLM 10B P Model Open Sourced

2020

Google LaMDA Model Trained to converse



2022

AlphaFold predicts structures of all known proteins

2023

A conversational AI Service powered by PaLM2

2024

Family of multimodal LLMs & products

3,000
Researchers

7,000
Publications

Responsible AI

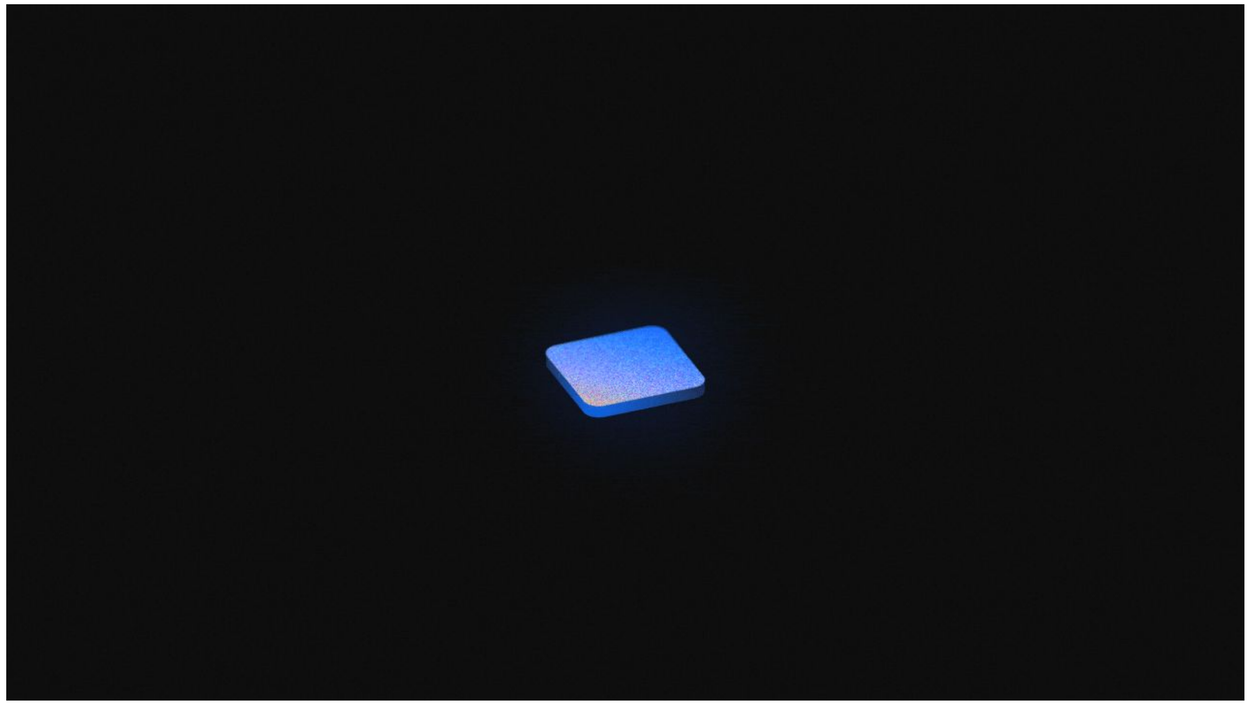
- Built & Tested for Safety
- Upholds high scientific standards
- Socially Beneficial
- Privacy in design
- Accountable to People
- Avoid creating unfair bias

Welcome to the Gemini era

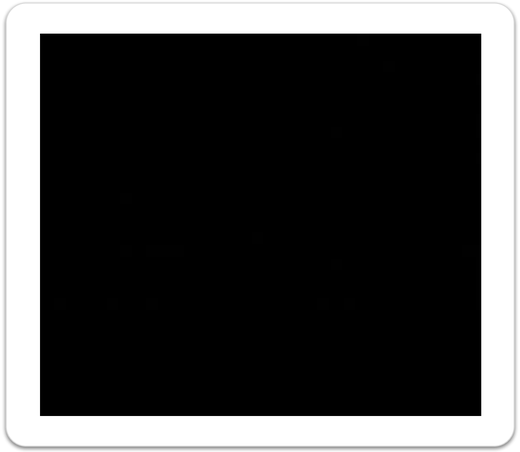
The next chapter of Generative AI innovation

Gemini

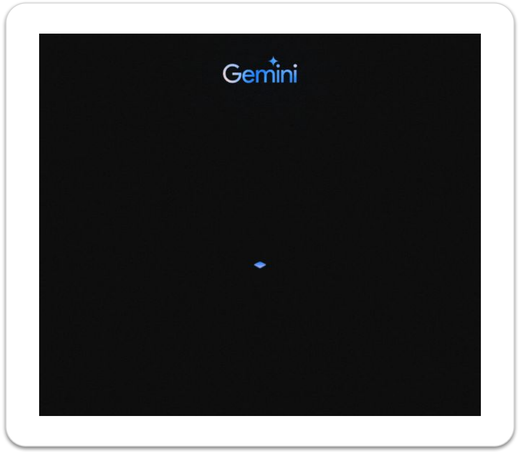
Gemini is the most capable and general model we've ever built, and is the result of a large-scale collaborative effort by teams across Google, including Google DeepMind and Google Research.



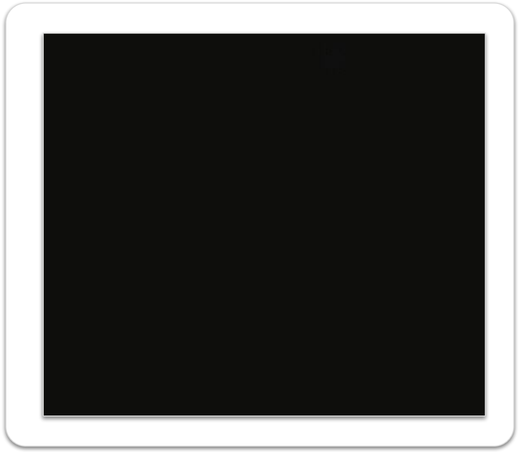
Gemini marks the next phase on our journey to making AI more helpful for everyone and provides



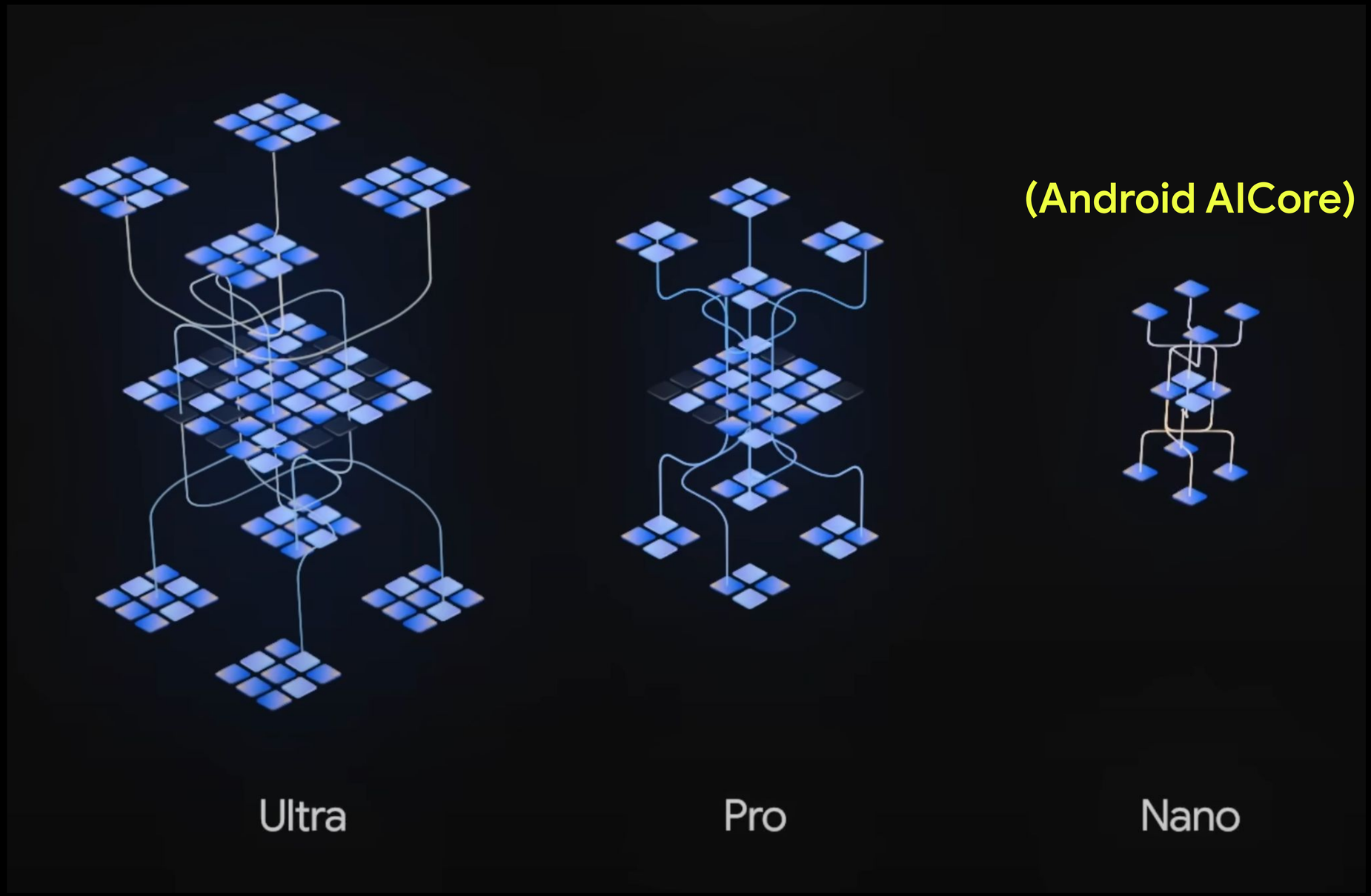
State-of-the-art, natively multimodal reasoning capabilities



Highly optimized for training and inference



Built with responsibility and safety at the core


(Android AICore)
Ultra
Pro
Nano

Google's Foundation Models on Vertex AI

Across a variety of model sizes to address use cases

GA

Gemini 1.0 Pro

Multimodal reasoning across a wide range of tasks

NEW

Gemini 1.5 Pro

Multimodal reasoning for longer prompts, 1 million context window

Limited Private GA

Gemini 1.0 Ultra

Largest and most capable model for highly complex tasks

NEW

Gemma 2B and 7B

Family of lightweight, state-of-the-art open models

PaLM for Text / Chat

Custom language tasks and multi-turn conversations

Imagen 2.0 for Text to Image

Create and edit images from simple prompts

Chirp for Speech to Text

Build voice enabled applications

Codey for Code Generation

Improve coding and debugging

Embeddings API for Text and Image

Extract semantic information from unstructured data

NEW

Claude on Vertex AI

Claude 2, Instant 1.2, and more

MISTRAL AI_ Meta

Open Models on Vertex AI

Mixtral 8x7B, Image Bind, DITO and more

NEW

Hugging Face Models

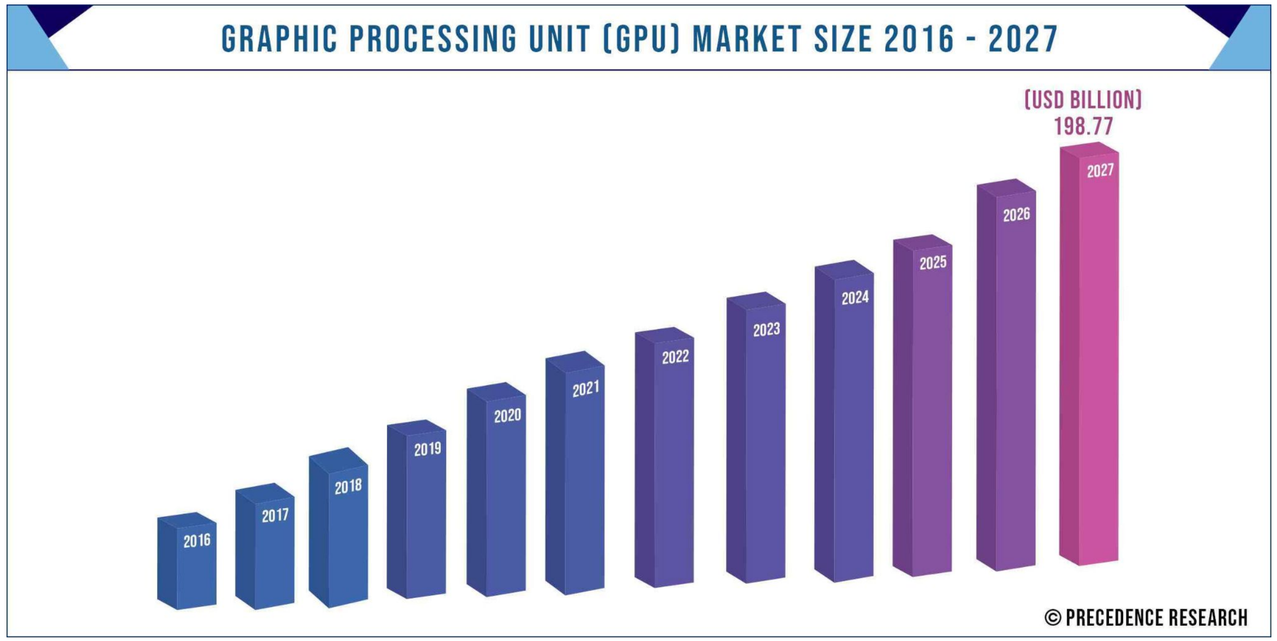
Few click deployment to Vertex AI

Gemma Open Models

ai.google.dev/gemma

Eight Things to Know about Large Language Models

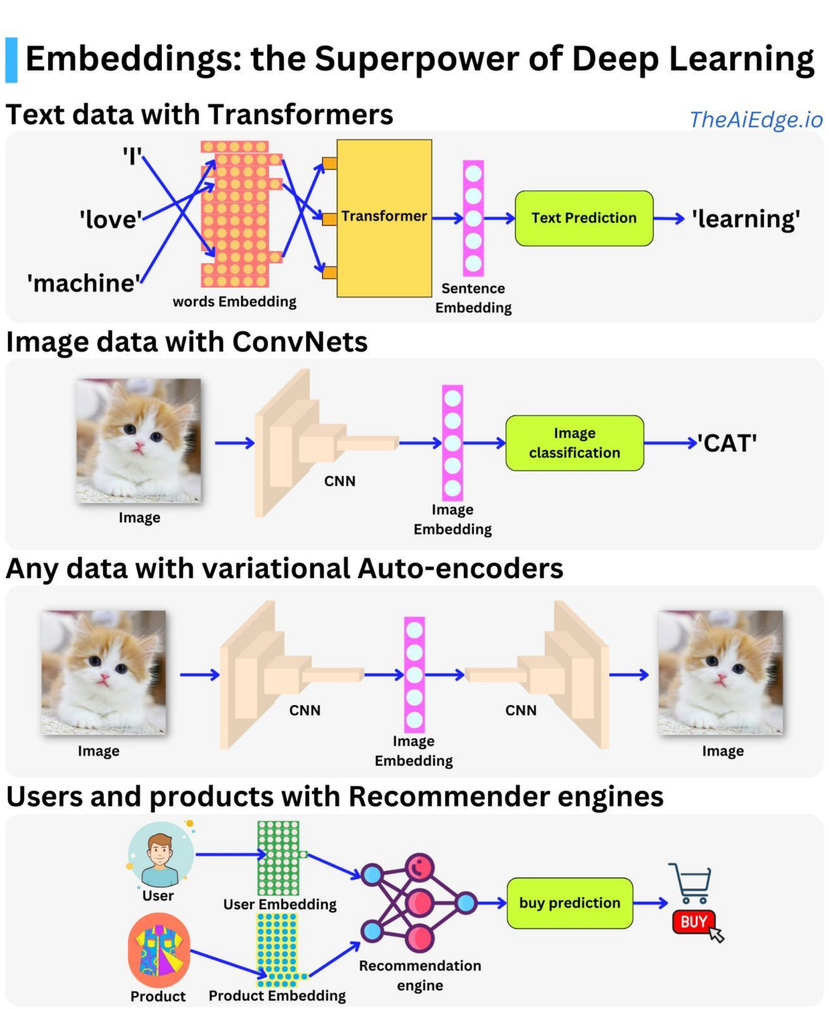
1. LLMs predictably get more capable with increasing investment, even without targeted innovation.
2. Many important LLM behaviors emerge unpredictably as a byproduct of increasing investment.
3. LLMs often appear to learn and use representations of the outside world.
4. There are no reliable techniques for steering the behavior of LLMs.
5. Experts are not yet able to interpret the inner workings of LLMs.
6. Human performance on a task isn't an upper bound on LLM performance.
7. LLMs need not express the values of their creators nor the values encoded in web text.
8. Brief interactions with LLMs are often misleading

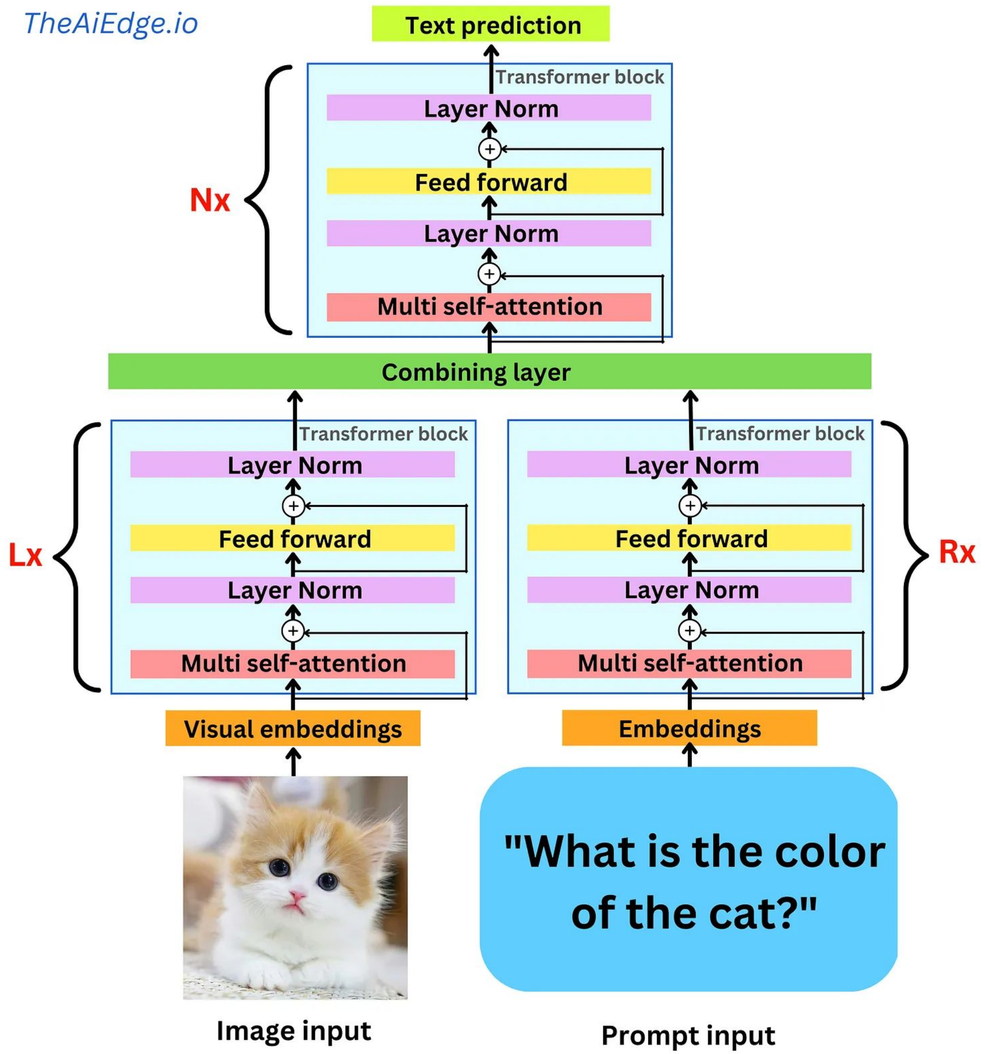


https://www.precedenceresearch.com/graphic-processing-unit-market

https://arxiv.org/abs/2304.00612

Modern Deep Learning





https://newsletter.theaiedge.io/

Demo time