Pontificia Universidad JAVERIANA Colombia

GenAI at the Edge

Prof. Marcelo J. Rovai

rovai@unifei.edu.br

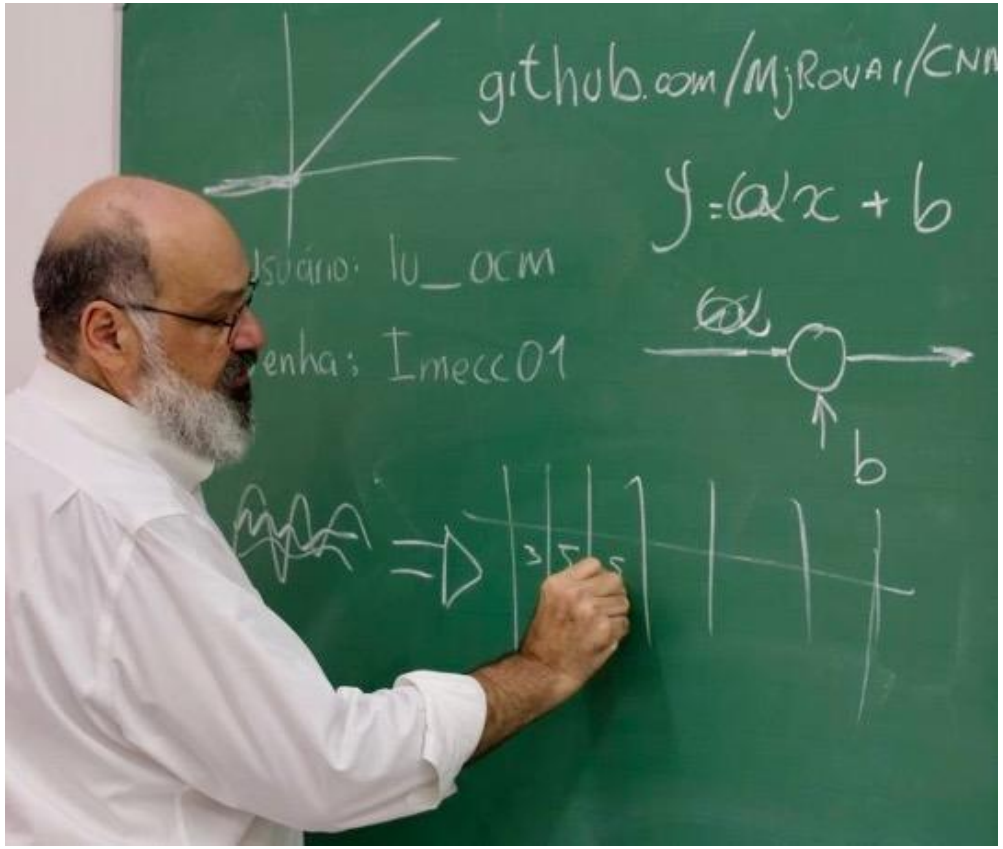
UNIFEI - Federal University of Itajuba, Brazil

TinyML4D - Academic Network Co-Chair

EdgeAIP - Academia-Industry Partnership Co-Chair

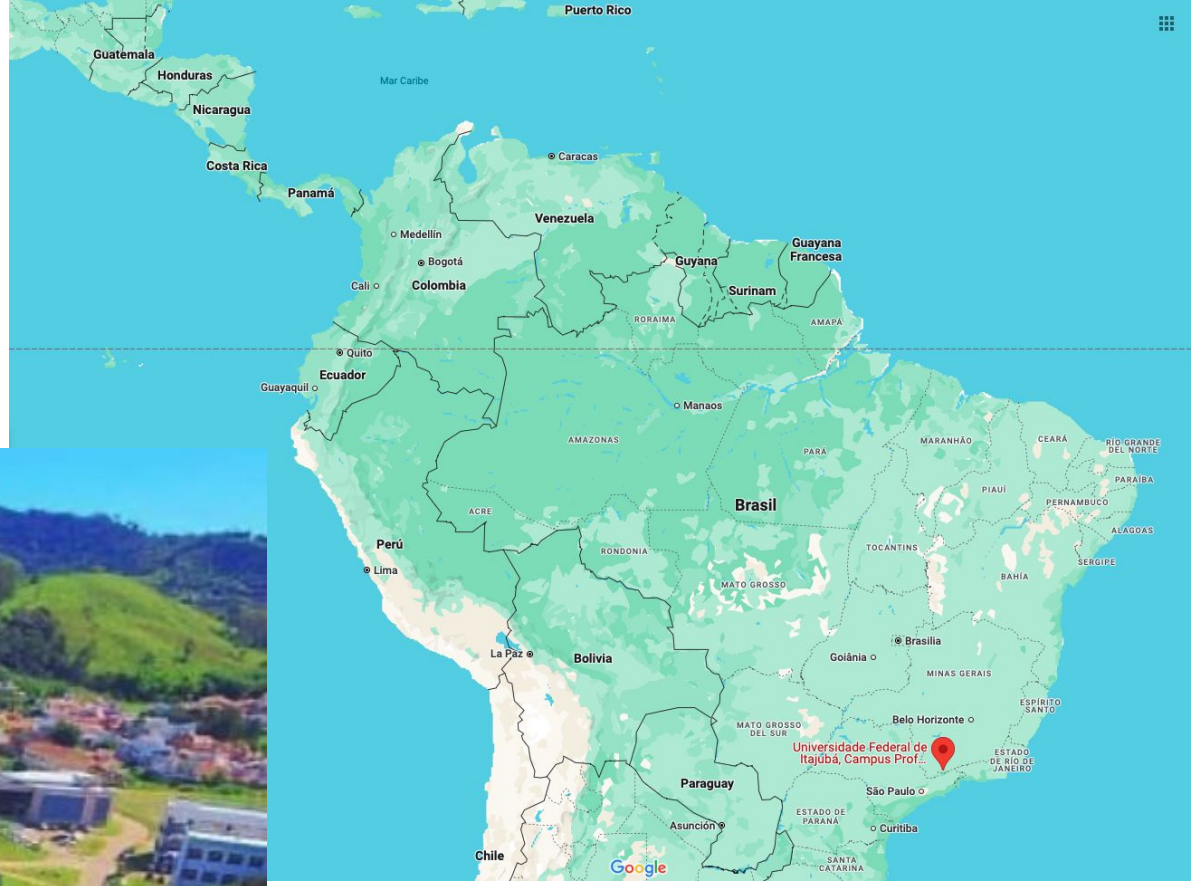
TINYML4D

Marcelo Rovai is an educator and professional in the field of engineering and technology, holding the title of Professor Honoris Causa from the Federal University of Itajubá, Brazil. His educational background includes an Engineering degree from UNIFEI and an advanced specialization from the Polytechnic School of São Paulo University (POLI/USP). Further enhancing his expertise, he earned an MBA from IBMEC (INSPER) and a Master's in Data Science from the Universidad del Desarrollo (UDD) in Chile.



With a career spanning several high-profile technology companies such as AVIBRAS Airspace, AT&T, NCR, and IGT, where he served as Vice President for Latin America, he brings a wealth of industry experience to his academic endeavors. He is a prolific writer on electronics-related topics and shares his knowledge through open platforms like Hackster.io.

In addition to his professional pursuits, he is dedicated to educational outreach, serving as a volunteer professor at the IESTI (UNIFEI) and engaging with the TinyML4D group and the EDGE AIP – the Academia-Industry Partnership of EDGEAI Foundation as a Co-Chair, promoting EdgeAI education in developing countries. His work underscores a commitment to leveraging technology for societal advancement.



Itajubá, Minas Gerais, Brazil

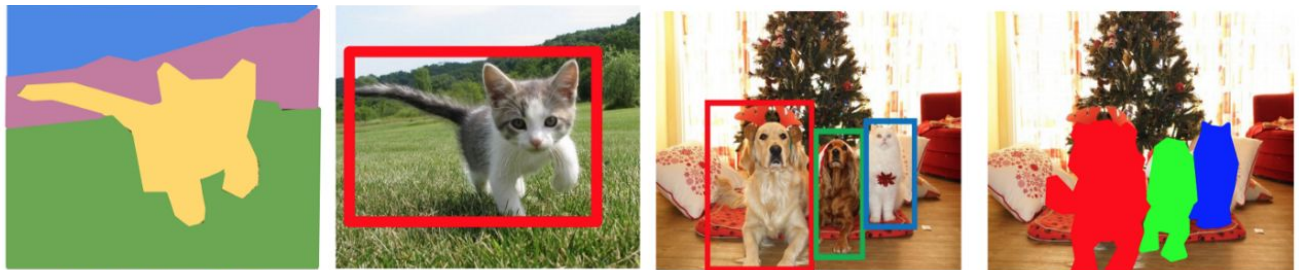
GenAI: Introduction

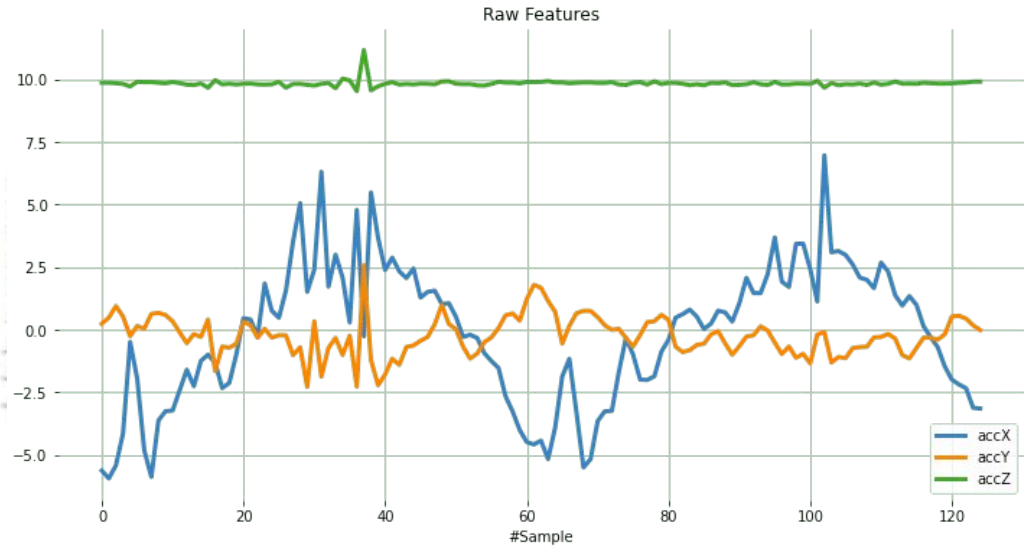
Generative AI (GenAI)

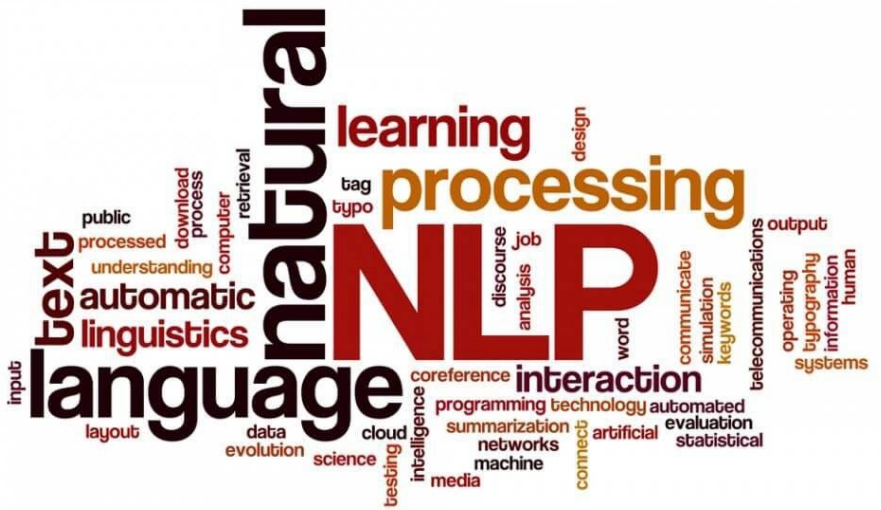
Generative AI is an artificial intelligence system capable of creating new, original content across various mediums such as text, images, audio, and video. These systems learn patterns from existing data and use that knowledge to generate novel outputs that didn't previously exist.

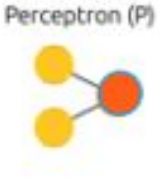
When used for generative tasks, Large Language Models (LLMs), Small Language Models (SLMs), and Visual-Language Models (VLMs) can all be considered types of GenAI.

Unstructured Data









Neural Network Architectures

Vibration Analysis

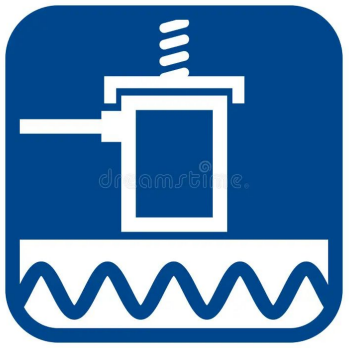
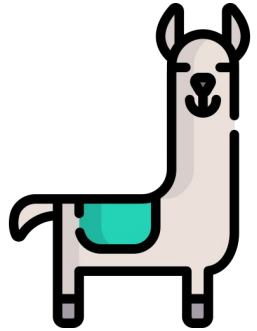


Image Classification

Text Generation

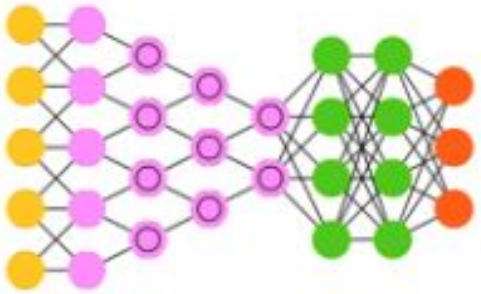
Image Generation

Large Language Models- LLMs



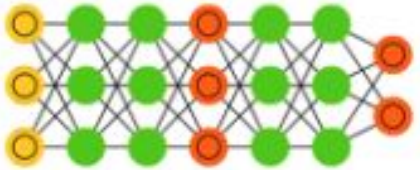
MLP - Deep Neural Network

CNN - Convolutional NN

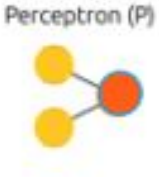


RNN - Recurrent NN (GRU/LSTM)

GAN - Generative Adversarial N.



AN - Attention (Transformers)



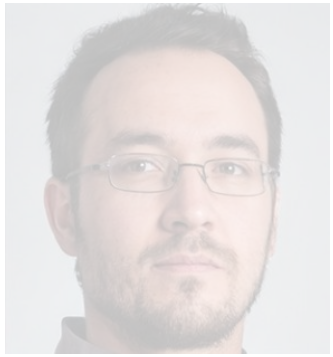
Neural Network Architectures

Vibration Analysis

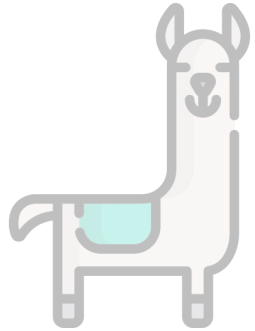
Image Classification

Text Generation

Image Generation



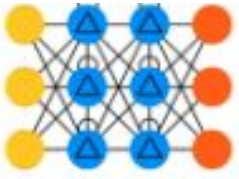
Large Language Models- LLMs



MLP - Deep Neural Network

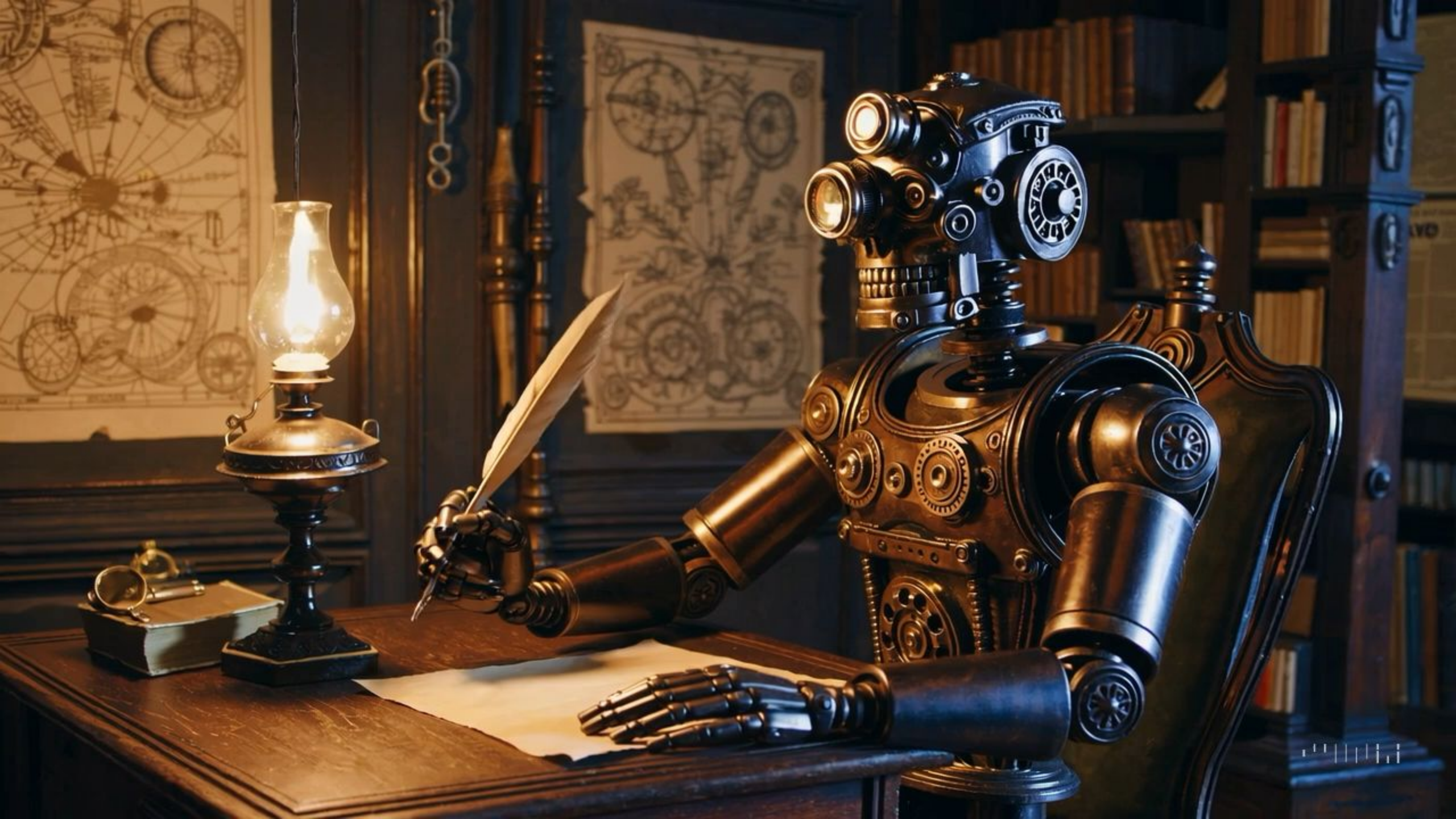
CNN - Convolutional NN

RNN - Recurrent NN (GRU/LSTM)



GAN - Generative Adversarial N.

AN - Attention (Transformers)



20_JulesVerneBot_Generatin

https://colab.research.google.com/github/Mjrovai/TinyML4D/blob/main/WALC_2024/notebooks/20_JulesVerneBot_Generating_English_Texts_with...

Commands + Code + Text Connect T4

Generating Text with RNNs: The Jules Verne Bot

- By Marcelo Rovai @Nov24
- With code support from **Claude 3.5 Sonnet** and **ChatGPT 4o**

Project Gutenberg

DATASET

Total characters:
- 5,768,791

Unique characters:
- 123

1. 'A Journey to the Centre of the Earth’
2. 'In Search of the Castaways’
3. 'An Antarctic Mystery’
4. 'In the year 2889’
5. ‘Around the World in Eighty Days’
6. 'Michael Strogoff'
7. 'Five Weeks in a Balloon'
8. 'The Mysterious Island’
9. 'From the Earth to the Moon'
10. 'Twenty Thousand Leagues under the Sea'

TOKENIZATION AND VOCABULARY

Conversion of text into numeric tokens.

Character-level tokenization: 123 unique characters.

Example: “The Project” ⟶ [122 52 69 66 1 48 79 76 71 66 64 81].

TOKENIZATION

```
['\n', ' ', '!', '"', '#', '$', '%', '&', "'", '(', ')', '*', '+',
 ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8',
 '9', ':', ';', '<', '=', '>', '?', 'A', 'B', 'C', 'D', 'E', 'F',
 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S',
 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', '[', ']', '_', 'a', 'b', 'c',
 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p',
 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '}', '~',
 '£', '°', '½', 'À', 'Æ', 'Ê', 'à', 'â', 'æ', 'ç', 'è', 'é', 'ê',
 'ë', 'ï', 'ñ', 'ô', 'û', 'ü', 'Œ', 'œ', 'ð', '–', '‘', '’', '“',
 '”', '•', '′', '″', '™', '\ufeff'], dtype='<U1')
```

```
[31, 42, 40,  1, 30, 28, 46, 40, 48, 45, 45, 42,  0,  0, 36,  0,  0,
 31, 71,  1, 76, 65, 76, 77, 68, 71, 13,  0,  0, 48, 69, 57,  1, 70,
 71, 65, 76, 61,  1, 60, 61, 75, 76, 57, 75, 11,  1, 78, 65, 70, 60,
 71,  1, 60, 57,  1, 59, 65, 60, 57, 60, 61,  1, 72, 57, 74, 57,  1,
 71,  1, 32, 70, 63, 61, 70, 64, 71,  1, 41, 71, 78, 71, 11,  1, 61,
 70, 59, 71, 70, 76, 74, 61, 65,  1, 70, 71,  0, 76, 74, 61, 69,  1,
 60, 57,  1, 30, 61, 70, 76, 74, 57, 68,  1, 77, 69,  1, 74, 57, 72,
 57, 82,  1, 57, 73, 77, 65,  1, 60, 71,  1, 58, 57, 65, 74, 74, 71,
 11,  1, 73, 77, 61,  1, 61, 77,  1, 59, 71, 70, 64, 61])
```

https://platform.openai.com/tokenizer

TRAINING SEQUENCES

Goal: Predict the next character in a sequence.

Length of the sequence: 120 characters (~ paragraph).

Input 'Hello my nam'

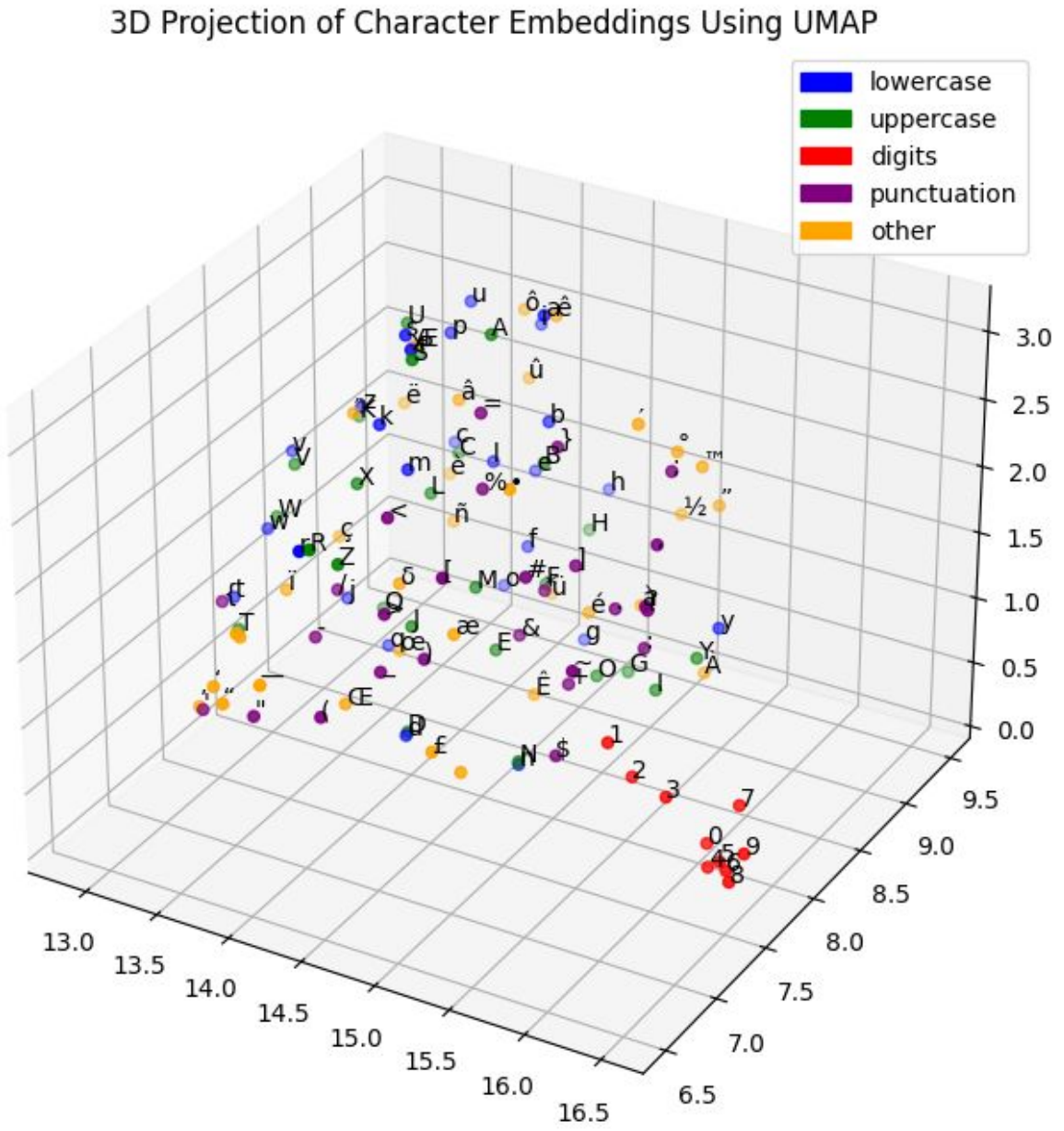
Output 'ello my name'.

EMBEDDING

Each character is represented as a vector of 256 dimensions.

Embedding captures relationships between characters in dense vectors.

EMBEDDING



Word2Vec - Embedding Projector

MODEL ARCHITECTURE

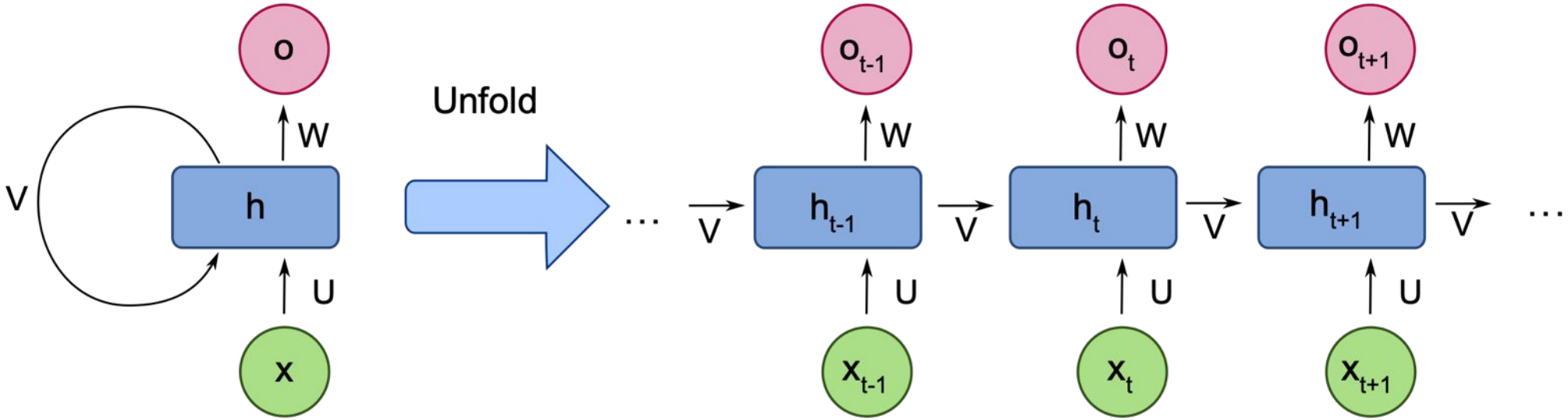
Embedding Layer: Converts characters into dense vectors.

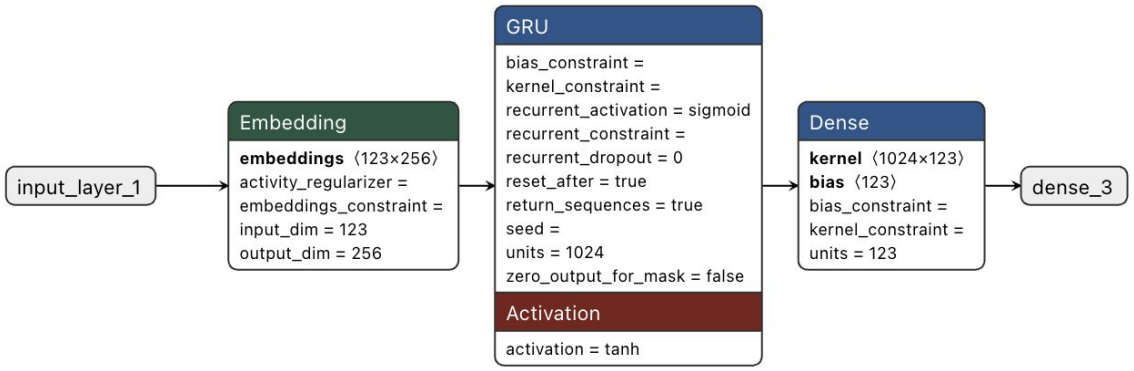
RNN/GRU Layer (1024 units): Learn from sequences.

Dense layer: Generates probabilities for each character (123).

Deep Learning models (or artificial neural networks)

Recurrent Neural Networks (RNNs): Designed for **sequential data like time series or text**, these networks use their internal state (memory) to process sequences of inputs.

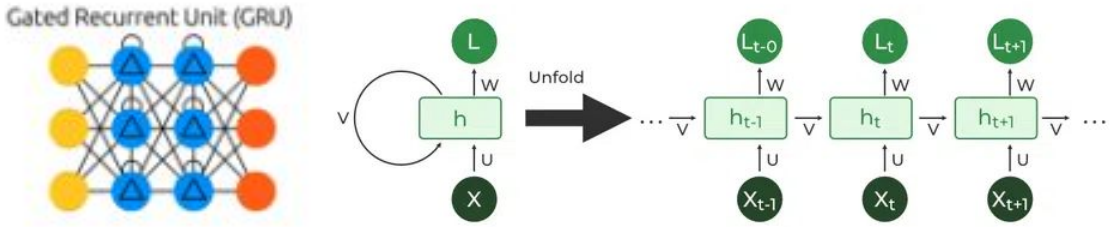




Model: "sequential_4"

Layer (type)	Output Shape	Param #
embedding_4 (Embedding)	(1, 120, 256)	31,488
gru_3 (GRU)	(1, 120, 1024)	3,938,304
dense_3 (Dense)	(1, 120, 123)	126,075

Total params: 4,095,867 (15.62 MB)
Trainable params: 4,095,867 (15.62 MB)
Non-trainable params: 0 (0.00 B)



RNN MODEL (RECURRENT)

MODEL TRAINING

Loss Function: Categorical Sparse Crossentropy

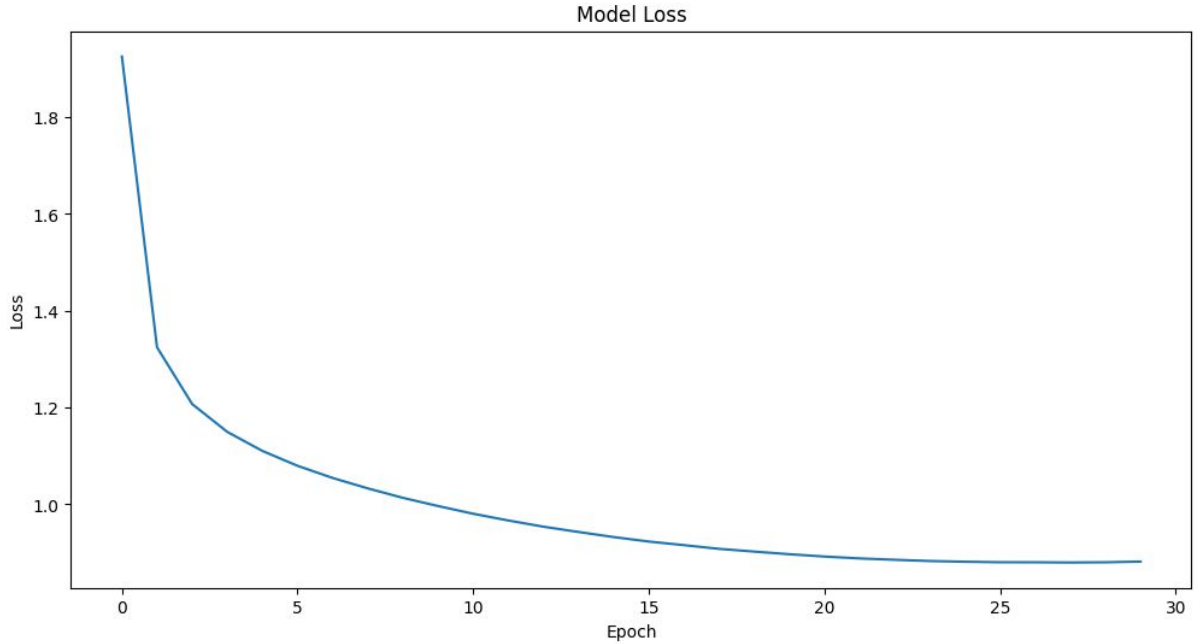
Optimizer: Adam

Epochs: 30

batch size: 128

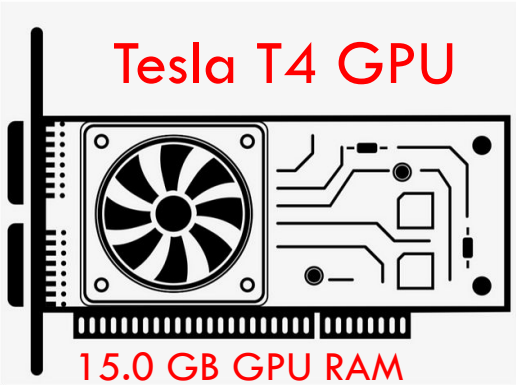
buffer size: 10,000

Monitoring loss reduction over time.



(33 minutes for training)

Google colab



TEXT GENERATION

The template generates text character by character from an initial text:

”THE FLYING SUBMARINE".

Temperature controls randomness (0.5 for predictable, 1.0 for creative text).

Generated text with temperature 0.5:

```
THE FLYING SUBMARINE

CHAPTER 100 VENTANTILE
This eBook is for the use of anyone anywhere in the United States and most
other parts of the earth and miserable eruptions. The solar rays should be
entirely under the shock of the intensity of the sea. We were all sorts. Are
we to prepare for our feelings?"

"I can never see them a good geographer," said Mary.

"Well, then, John, for I get to the Pampas, that we ought to obey the same
time. In the country of this latitude changed my brother, and the
 Nautilus  floated in a sea which contained the rudder and
lower colour visibly. The loiter was a fatalint region the two
scientific discoverers. Several times turning toward the river, the cry
of doors and over an inclined plains of the Angara, with a threatening
water and disappeared in the midst of the solar rays.

The weather was spread and strewn with closed bottoms which soon appeared
that the unexpected sheets of wind was soon and linen, and the whole
seas were again landed on the subject of the natives, and the prisoners
were successively assuming the sides of this agreement for fifteen days
with a threatening voice.

The clouds had disappeared to the ground, and the river and the ship's
conditions of the ship's course was still standing, but in his daring
explorers, and the sailor thought for the sudden discovery.

"There are no trees, and a half an hour or the sun will soon be carried off
they were bringing a special track."

"As you see, my dear Helena, who was the matter of despair?" cried
the captain.

"The raft we had done now, captain, and I have a reasonable face of the
shipwrecked creek, had been discovered at the entrance of the
world."
```

CHALLENGES AND LIMITATIONS

Limited context window (120 characters).

Difficulty in maintaining coherence in long texts.

Character-level modeling vs. word-level modeling.

CONNECTING WITH MODERN LANGUAGE MODELS

Our Model (VerneBot) :

- **Training data: 5.8 million characters (bytes) (10 books).**
- **4 million parameters,**
- **Character-level tokenization (120)**
- **RNN architecture.**

Open AI GPT-3 (2020):

- **Training data: 45 Trillion bytes (text)**
- **175 billion parameters,**
- **Subword tokenization (2,048 tokens),**
- **Transformer Architecture.**

Modern models handle long-range dependencies better.

SLM

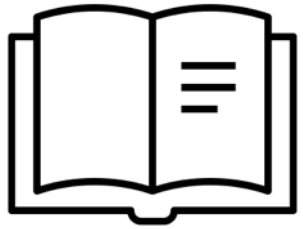
Phi-3-mini
(3.8B)

Phi-3-vision
(4.2B)

Phi-3-small
(7B)

Phi-3-medium
(14B)

- **Architecture:** Transformer – 3.8 Billion Parameters
  Inputs: Text.
  Context length: 128k tokens
  GPU: 512 H100-80G
  Training time: 7 days
  Training data: 3.3 Trillion tokens**



= 150 K tokens

~ 350 pages

~ 300 words/page

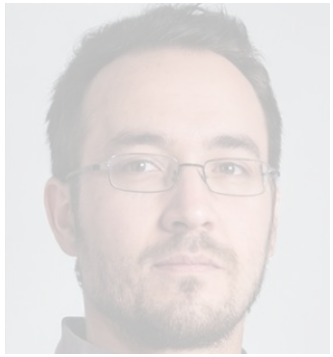
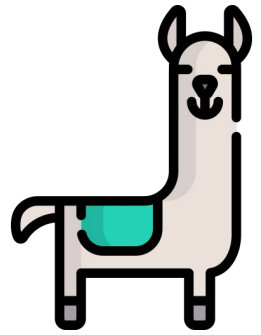
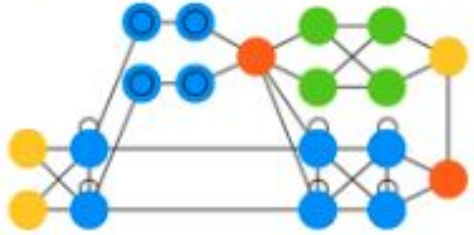
1 word = ~ 1.4 token

** Equivalent to 23 million books, that is:
17% of All the books in the world

LLM / SLM

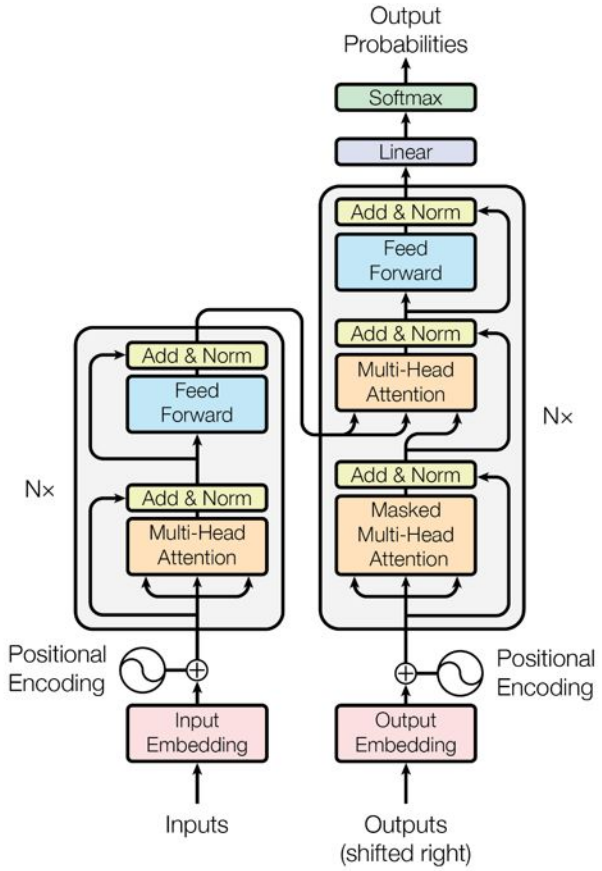
Large Language Model / Small Language Models

Neural Network Architectures

Vibration Analysis	Image Classification	Text Generation	Image Generation	Large Language Models- LLMs
				
MLP - Deep Neural Network	**CNN** - Convolutional NN	**RNN** - Recurrent NN (GRU/LSTM)	**GAN** - Generative Adversarial N.	**AN** - Attention (Transformers)
				

LLM/SLM – Large /Small Language Model

Large Language Models (LLMs) and SLMs are advanced neural networks based on the Transformer architecture that excel in understanding and generating human language. They represent a significant evolution from earlier sequence-based mode like RNNs, which surpass them in handling long-range dependencies and parallel processing efficiency.



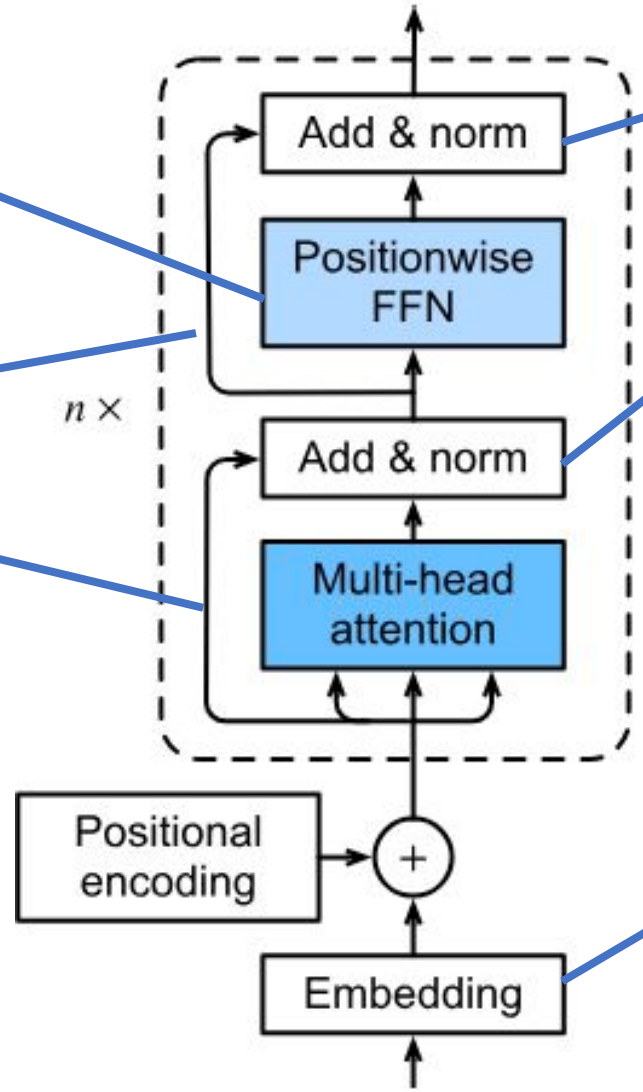
The Illustrated Transformer

Transformers

Dense Layer

Residuals Conexions
(CNN)

Normalization
(CNN)



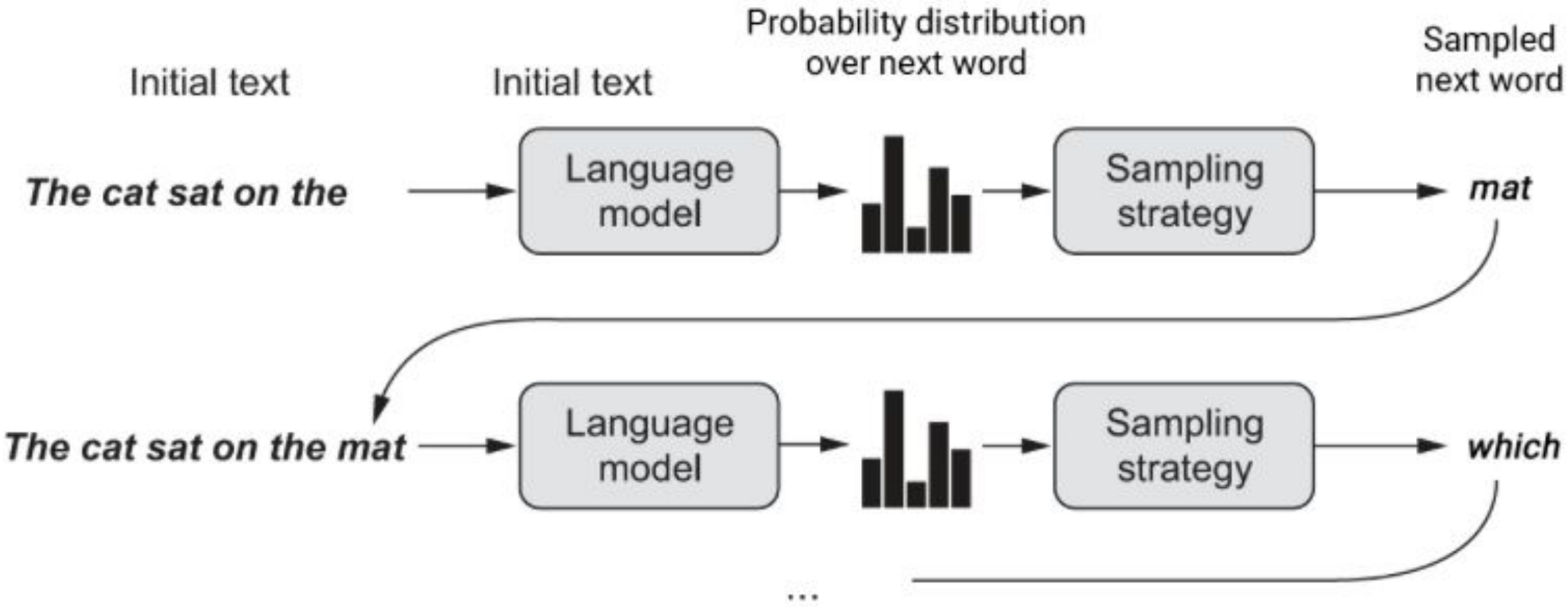
NLP based on RNN

Transformer language model visualization

https://bbycroft.net/llm

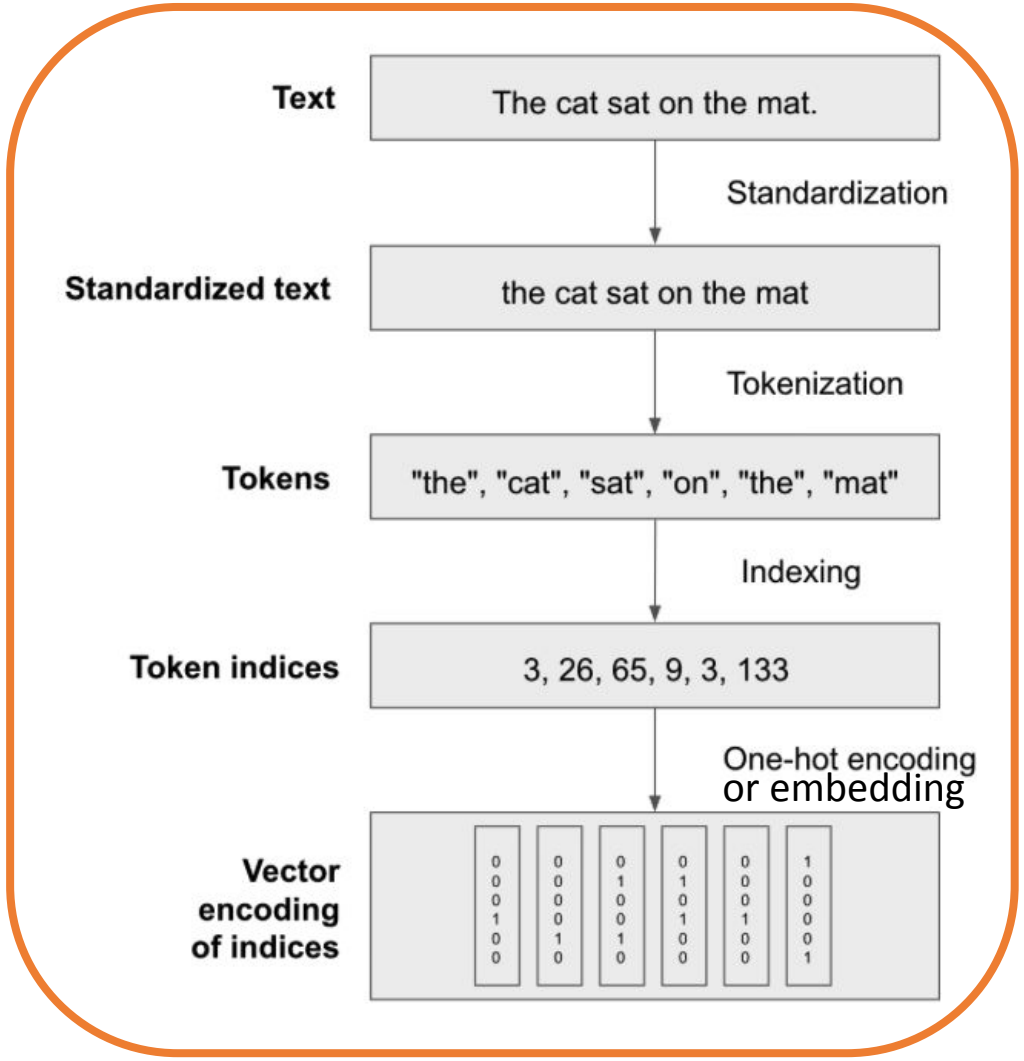
https://d2l.ai/_images/transformer.svg

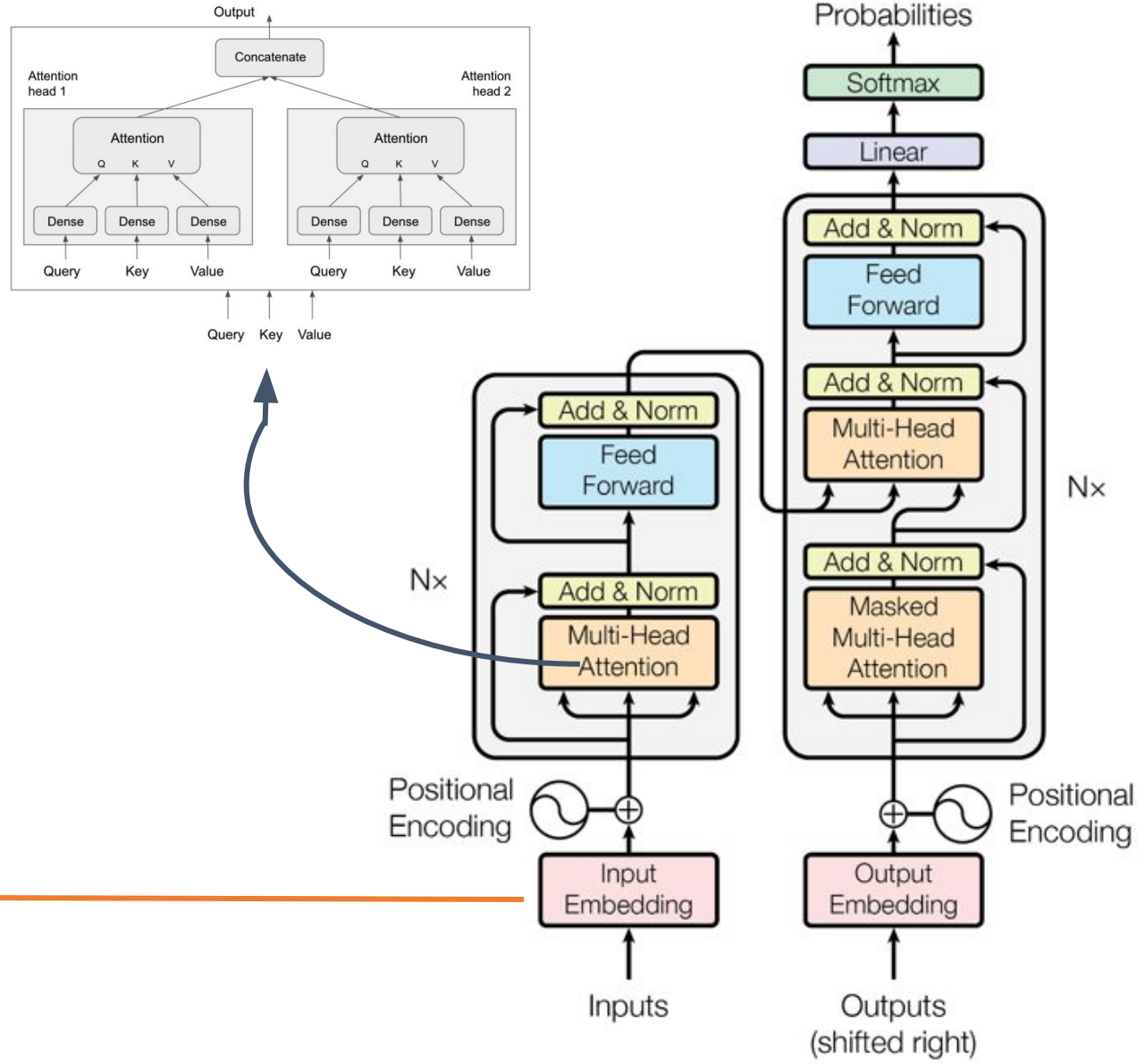
LLM/SLM – Large /Small Language Model



https://platform.openai.com/tokenizer

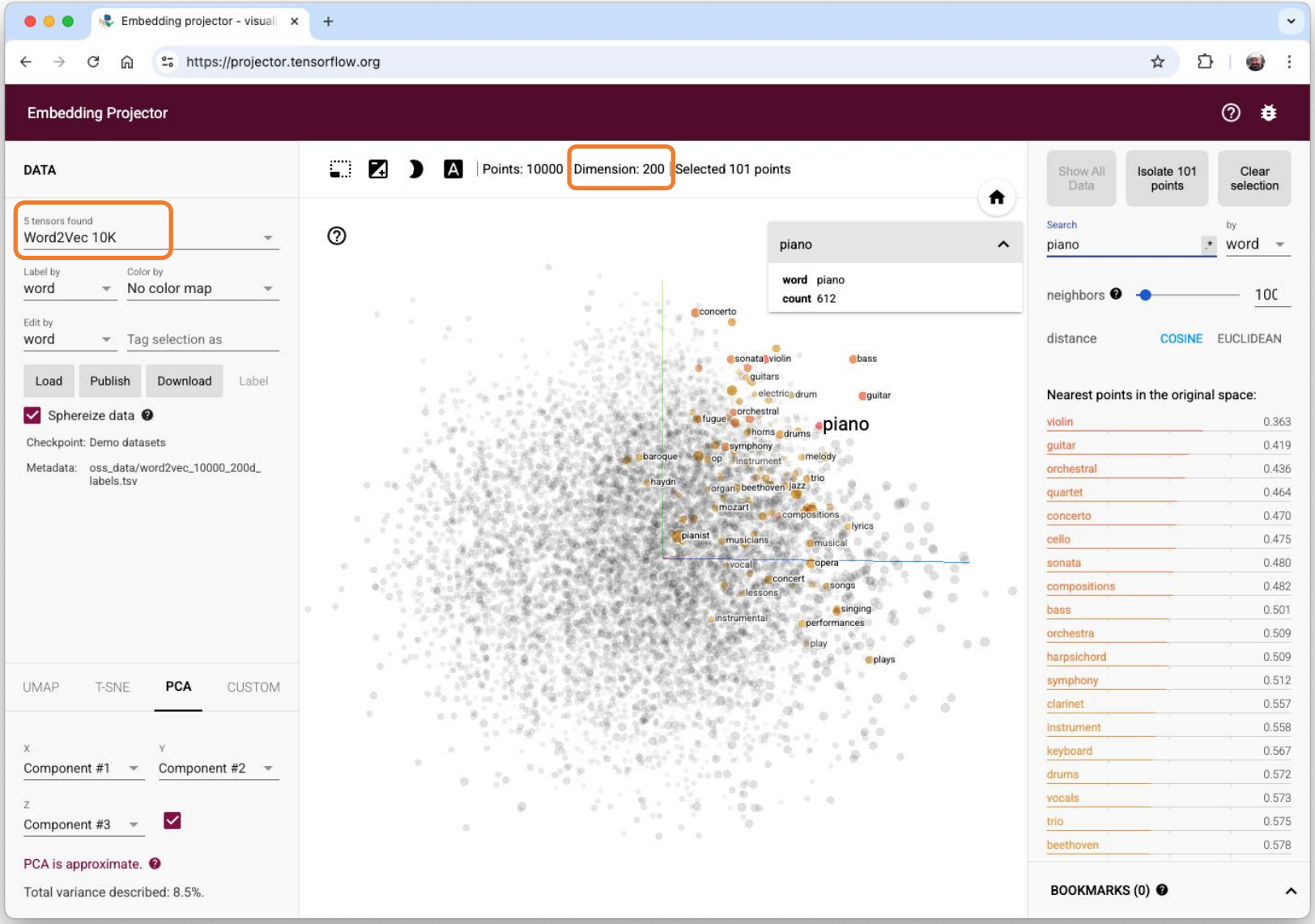
Embedding



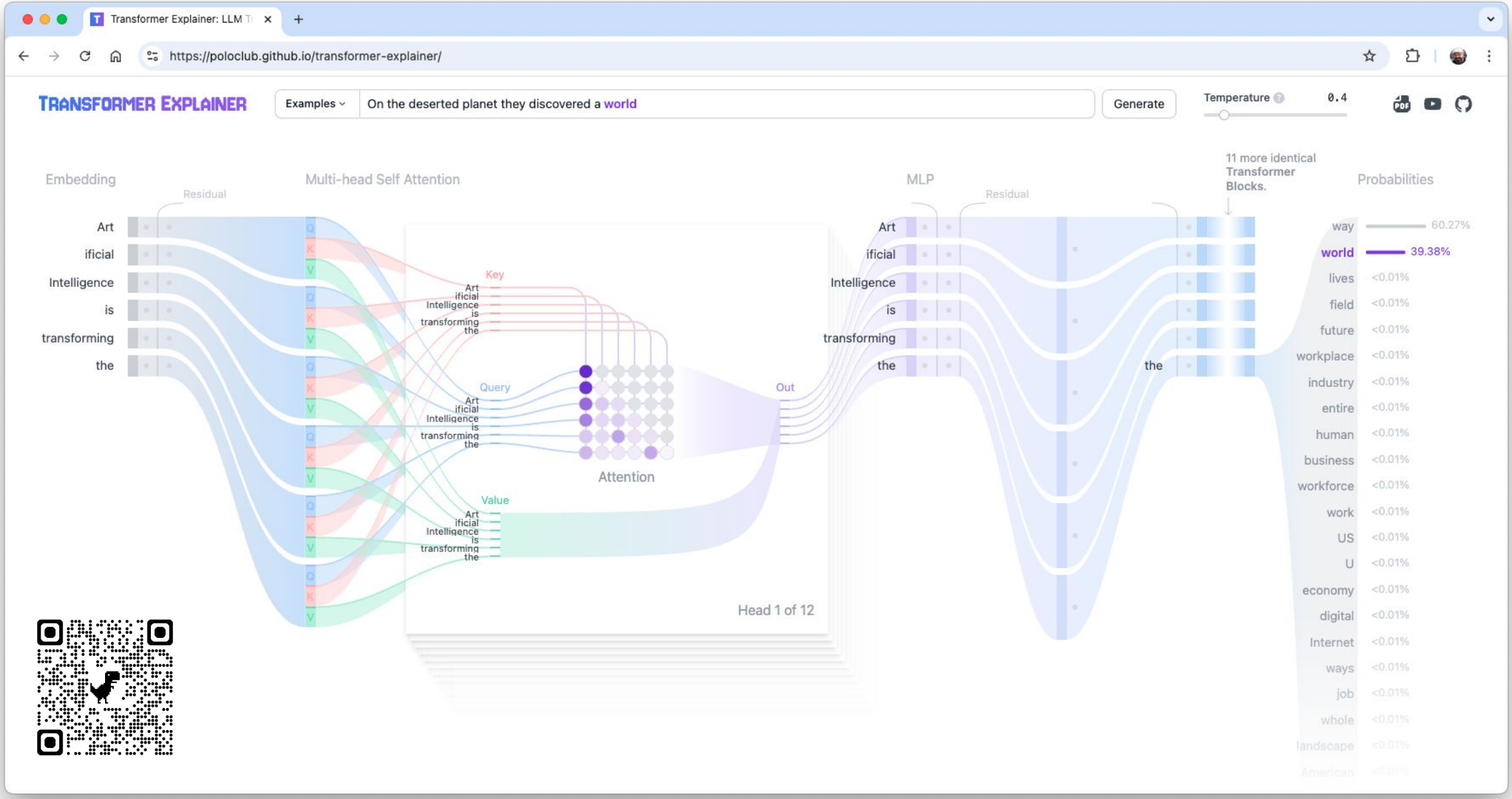


Encoder Decoder

Embedding



https://projector.tensorflow.org/



https://poloclub.github.io/transformer-explainer/

LLM TRAINING

Pretraining

Base model

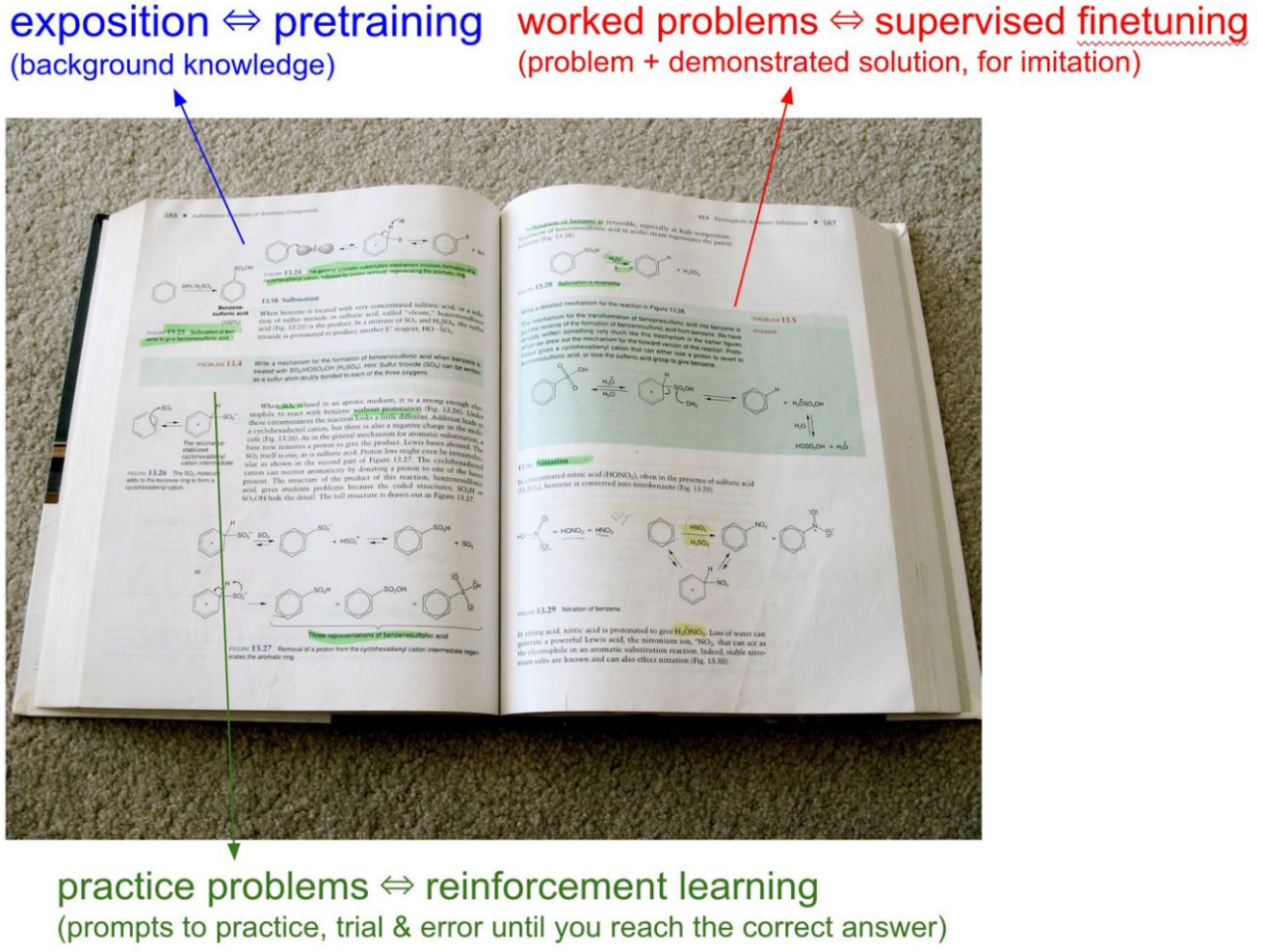
Post-Training
Supervised
Finetuning

SFT model

Post-Training
Reinforcement
Learning

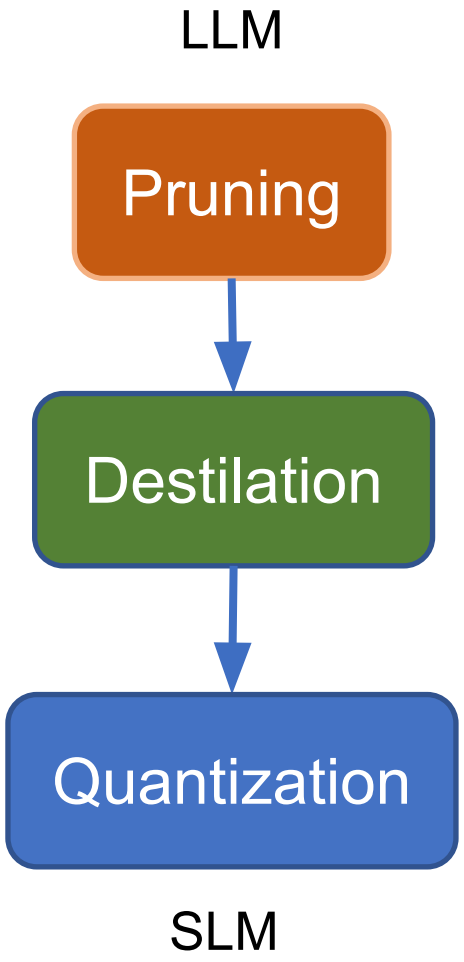
RL model

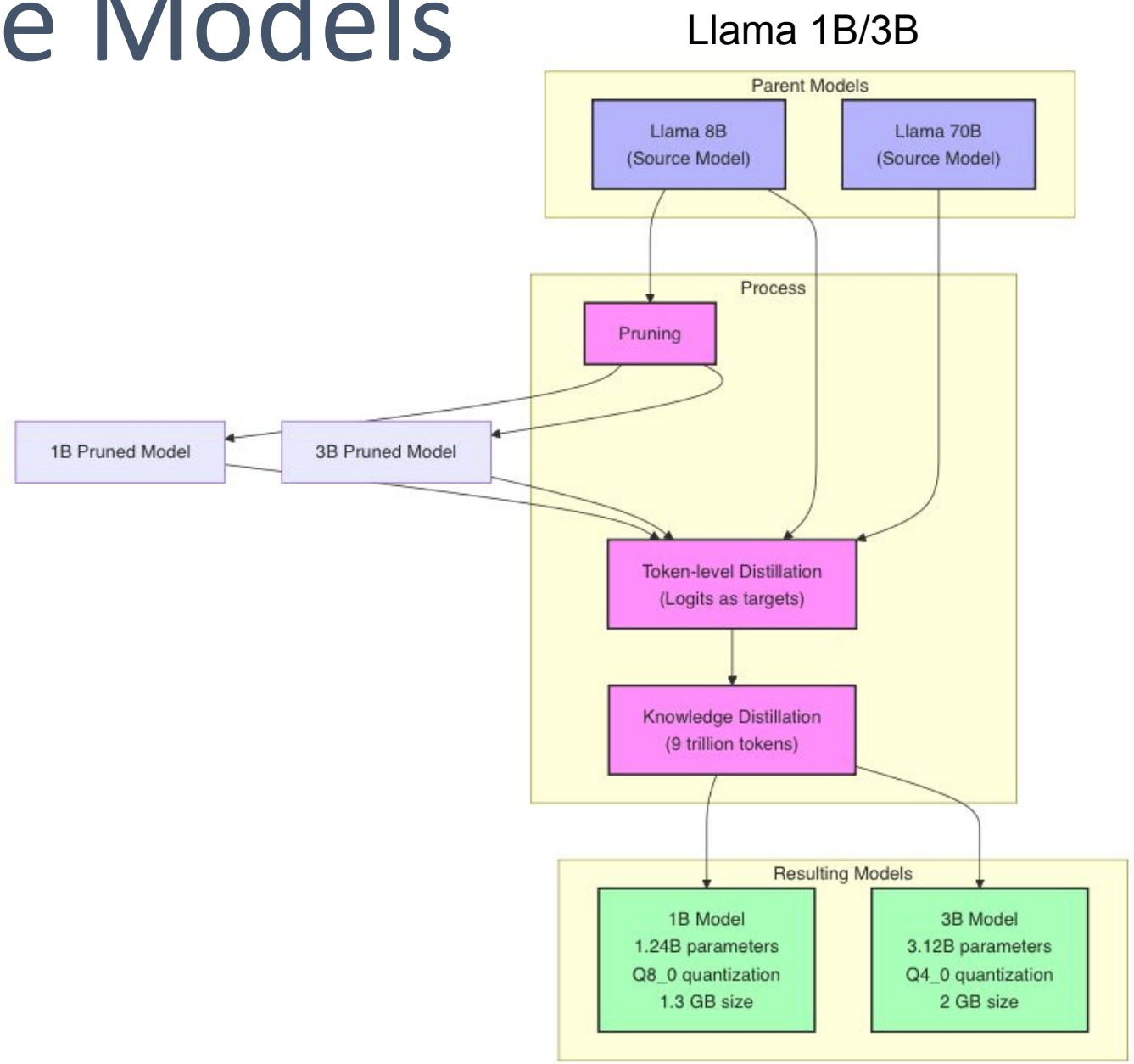
ChatGPT



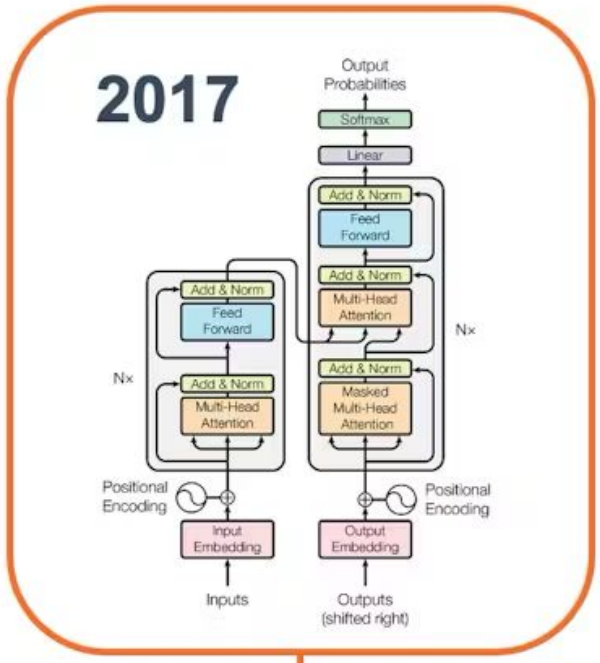
Deep Dive into LLMs like ChatGPT

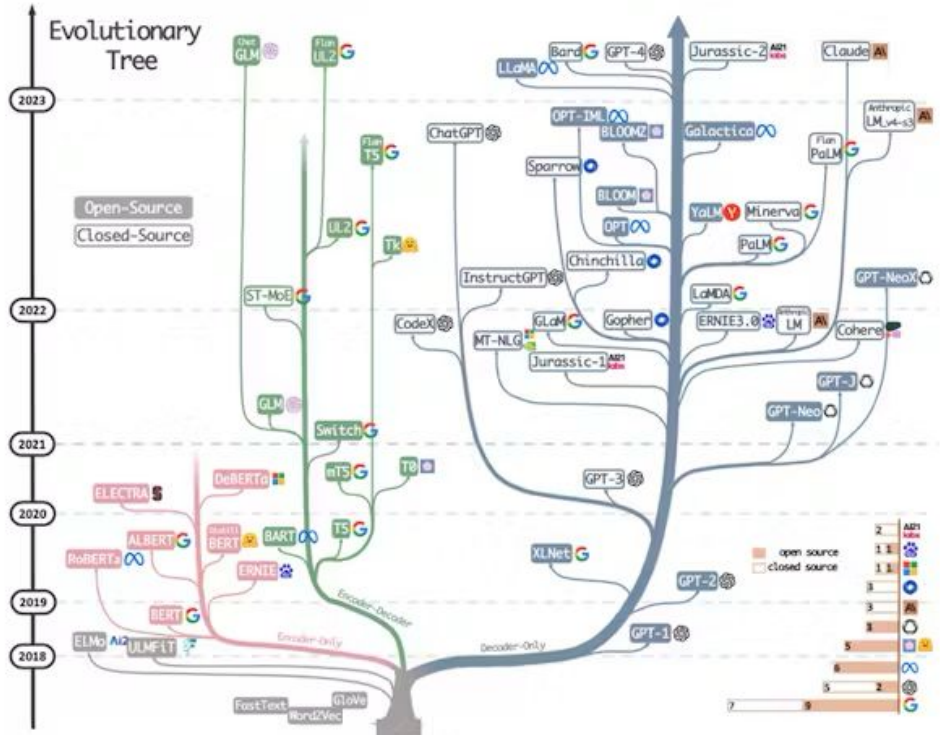
Small Language Models

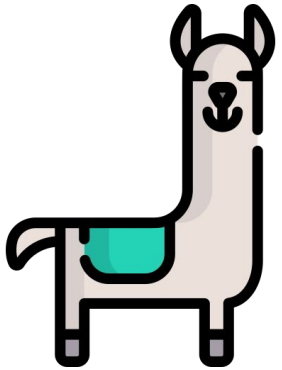




Transformers to LLMs and SLMs







Small Models

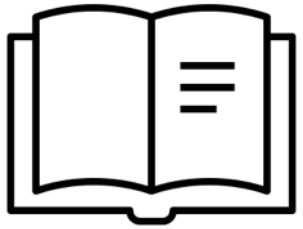
LlaMa

Gemma

```
marcelo_rovai — mjrovai@raspi-5: ~ — ssh mjrovai@192.168.4.209 — 74×12
(ollama) mjrovai@raspi-5:~ $ ollama run llama3.2:3b --verbose
>>> What is the capital of France?
The capital of France is Paris.

total duration:       1.808927736s
load duration:        39.854862ms
prompt eval count:    32 token(s)
prompt eval duration: 221.506ms
prompt eval rate:     144.47 tokens/s
eval count:           8 token(s)
eval duration:        1.506376s
eval rate:            5.31 tokens/s
```

```
marcelo_rovai — mjrovai@raspi-5: ~ — ssh mjrovai@192.168.4.209 — 67×13
(ollama) mjrovai@raspi-5:~ $ ollama run gemma2:2b --verbose
>>> What is the capital of France?
The capital of France is **Paris**. 🗼


total duration:       4.373339337s
load duration:        48.129697ms
prompt eval count:    16 token(s)
prompt eval duration: 1.968114s
prompt eval rate:     8.13 tokens/s
eval count:           13 token(s)
eval duration:        2.313284s
eval rate:            5.62 tokens/s
```

llava-phi-3 is a LLaVA model (**L**arge **La**nguage and **V**ision **A**ssistant) fine-tuned from Microsoft Phi-3 mini



= 147K tokens

~ 350 pages

~ 300 words/page

1 word = ~ 1.4 token

A **4-bit** quantized 3.8 billion parameter * language model trained on 3.3 trillion tokens**, whose overall performance, as measured by both academic benchmarks and internal testing, rivals that of models such as Mixtral 8x7B and GPT-3.5

* 2.4 GB

** 22.5 Million books - 17% of all books written in the world

llava-phi-3 (2.9 GB)

Ollama

```
mjrovai@rpi-5: ~/Documents/OLLAMA
Help
ute.

/Documents/OLLAMA $
/Documents/OLLAMA $ python calc_distance_image.py /
home/mjrovai/Documents/OLLAMA/image_test_1.jpg

 The image shows Paris, with lat:48.86 and long: 2.35, located in
France and about 11,630 kilometers away from Santiago, Chile.

 [INFO] ==> The code (running llava-phi3), took 232.60845186299412
 seconds to execute.

mjrovai@rpi-5:~/Documents/OLLAMA $
```

```
mjrovai@rpi-5: ~
File  Edit  Tabs  Help
>>> Awswer with one short sentence, what is the capital of France and its distanc
... e in Km from Santiago, Chile
 The capital of France is Paris and it is around 12,674 kilometers away
from Santiago, Chile.

total duration:       13.860074968s
load duration:        1.537039ms
prompt eval count:    27 token(s)
prompt eval duration: 5.925386s
prompt eval rate:     4.56 tokens/s
eval count:           26 token(s)
eval duration:        7.539223s
eval rate:            3.45 tokens/s
>>> Send a message (/? for help)
```

(13 seconds)

```
mjrovai@rpi-5: ~/Documents/OLLAMA
Help

/Documents/OLLAMA $
/Documents/OLLAMA $ python calc_distance_image.py
/home/mjrovai/Documents/OLLAMA/image_test_3.jpg

 The image shows Machu Picchu, with lat:-13.16 and long: -72.54,
 located in Peru and about 2,250 kilometers away from Santiago,
Chile.

 [INFO] ==> The code (running llava-phi3), took 267.579568572007
7 seconds to execute.

mjrovai@rpi-5:~/Documents/OLLAMA $
```

(4 minutes)

LLMs: Optimization Techniques

Techniques for Enhancing SLM at the Edge

Fundamentals: Optimizing Prompting Strategies

- Chain-of-Thought Prompting
- Few-Shot Learning
- Task Decomposition

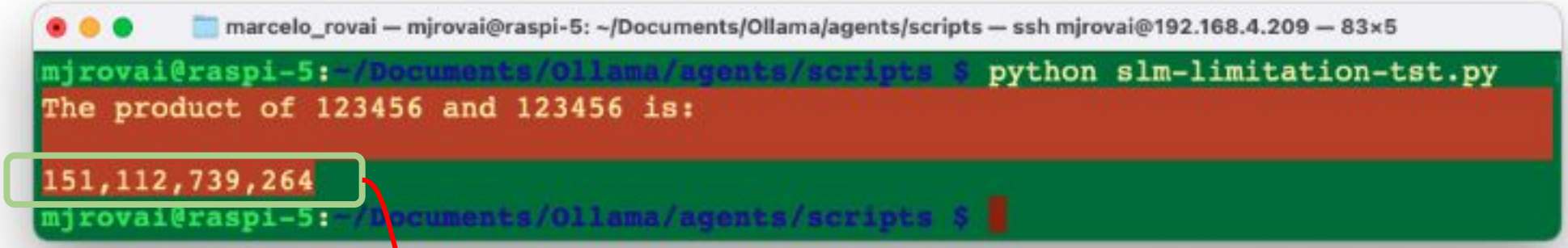
Intermediate: Building Intelligence Systems

- Building Agents with SLMs
- General Knowledge Router
- Function Calling
- Response Validation

Advanced: Extending Knowledge and Specialization

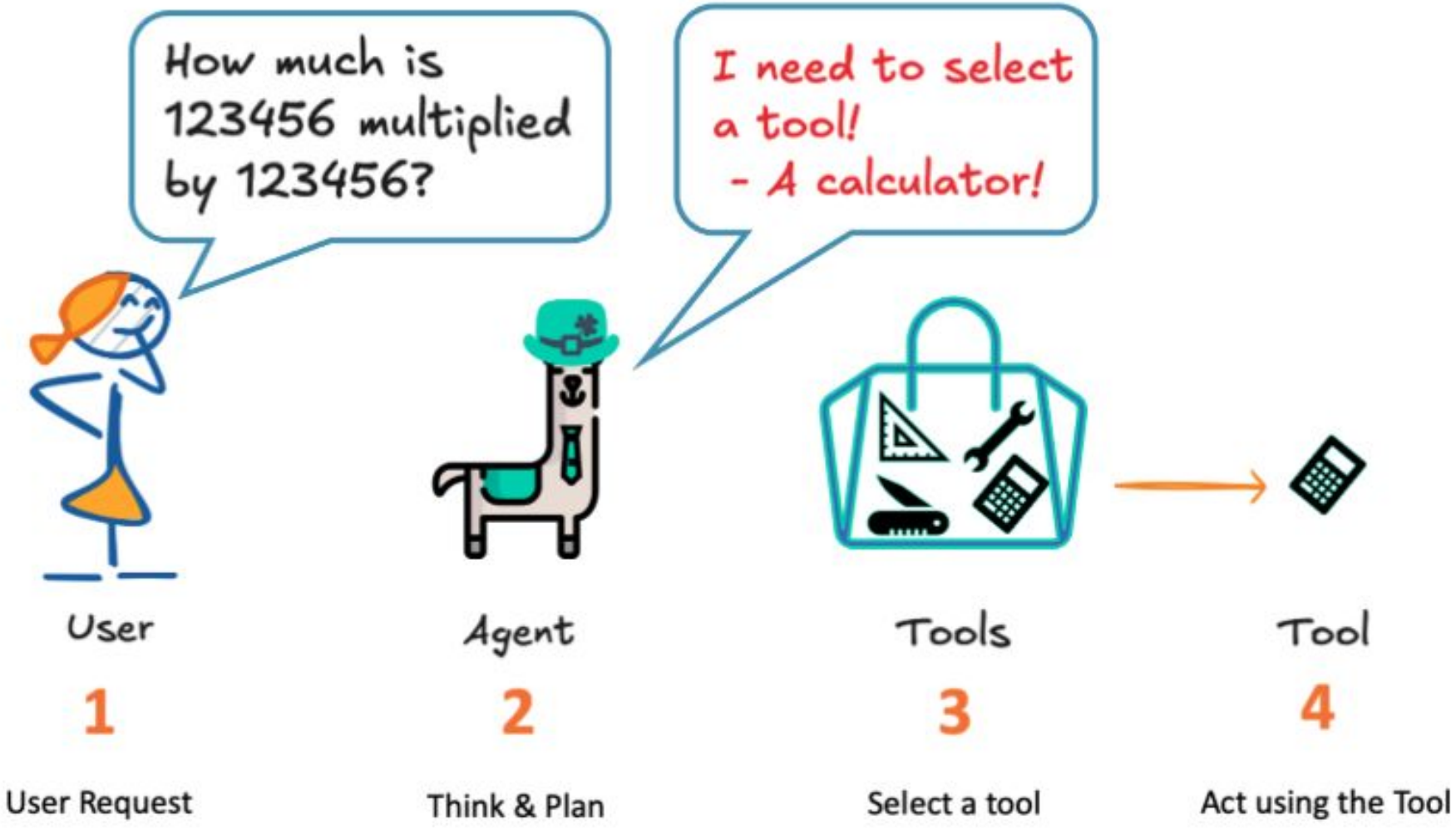
- Retrieval-Augmented Generation (RAG)
- Fine-Tuning for Domain Specialization

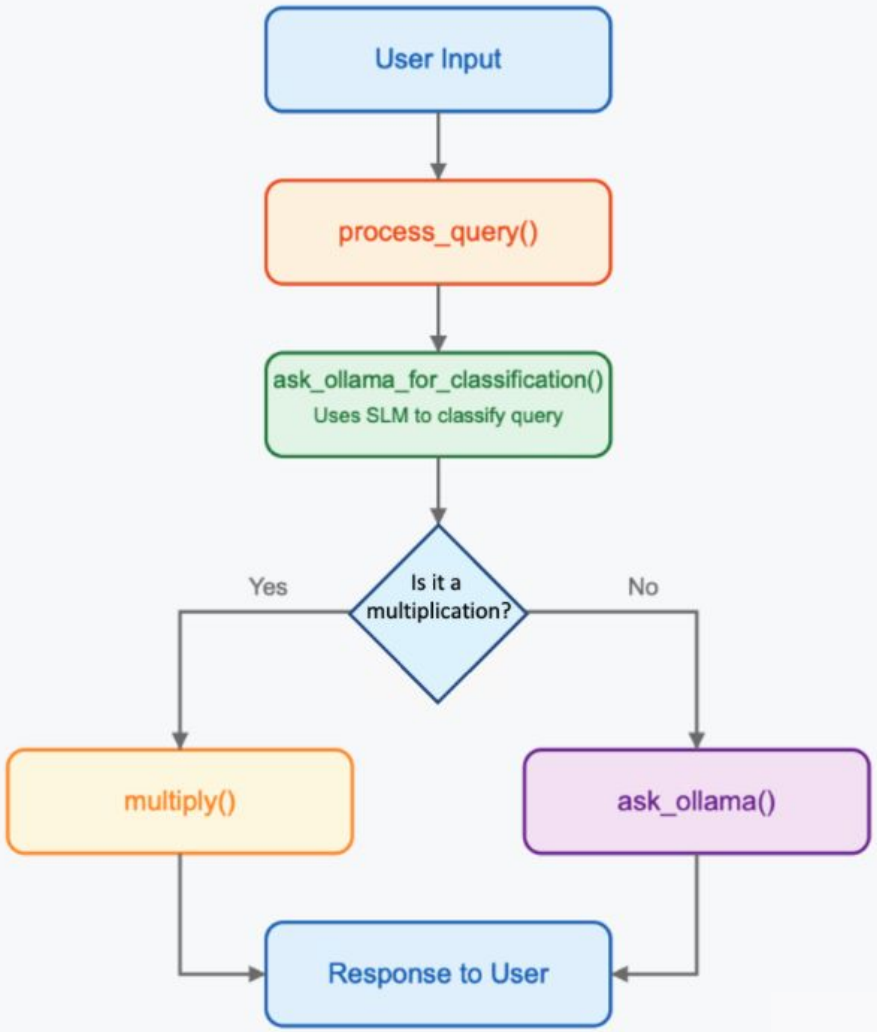
```python
import ollama

response = ollama.generate(
    model="llama3.2:3b",
    prompt="Multiply 123456 by 123456"
)
print(response['response'])
```



WRONG!!!!! **The correct answer -> 123,456 x 123,456 = 15,241,383,936**

Agents


How much is 123456 multiplied by 123456?
I need to select a tool! - A calculator!
User
Agent
Tools
Tool
1
2
3
4
User Request
Think & Plan
Select a tool
Act using the Tool

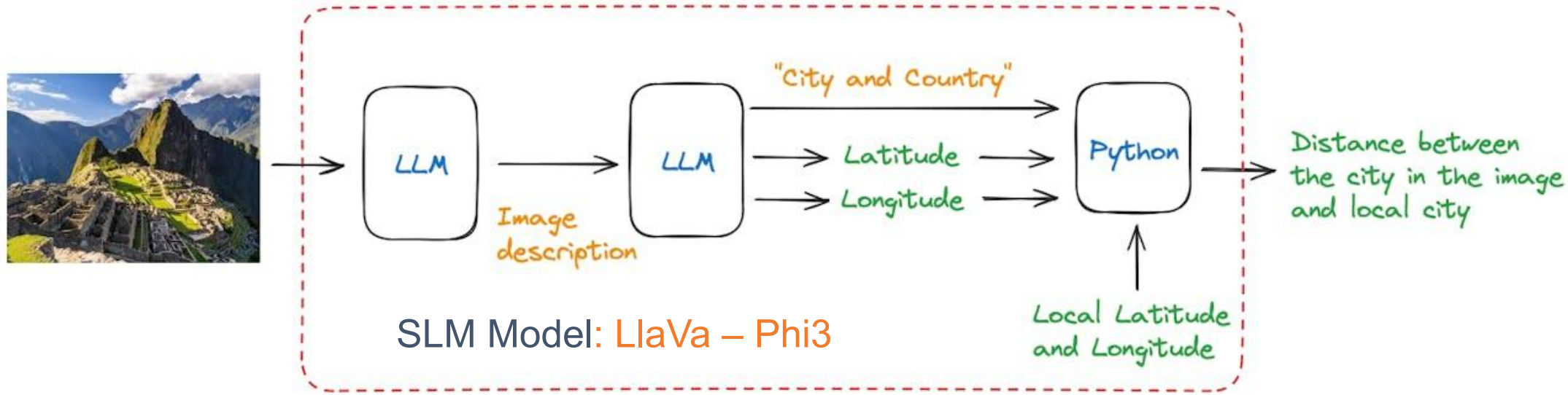



```
marcelo_rovai — mjrovai@raspi-5: ~/Documents/Ollama/agents/scripts — ssh mjrovai@192.168.4.209 — 83×14
mjrovai@raspi-5:~/Documents/Ollama/agents/scripts $ python 2-simple_agent.py
Ollama Agent (Type 'exit' to quit)
----------------------------------

You: Multiply 123456 by 123456
Sending classification request to Ollama
Classification response: {
  "type": "multiplication",
  "numbers": [123456, 123456]
}
Ollama classification: {'type': 'multiplication', 'numbers': [123456, 123456]}

Agent: The product of 123456 and 123456 is 15241383936.
```

It is correct 123,456 x 123,456 = 15,241,383,936

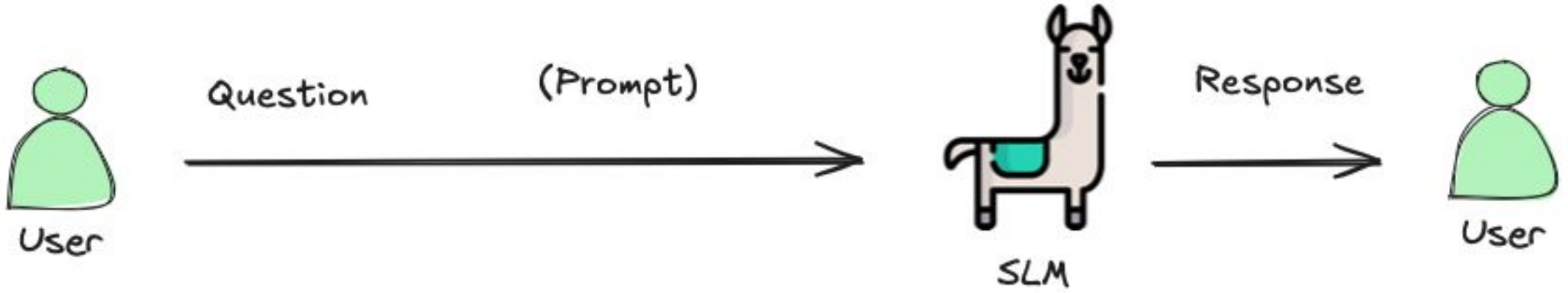
```
marcelo_rovai — mjrovai@raspi-5: ~/Documents/Ollama/agents/scripts — ssh mjrovai@192.168.4.209 — 83×12

You: What is the capital of Brazil?
Sending classification request to Ollama
Classification response: {
  "type": "general_question"
}
Ollama classification: {'type': 'general_question'}
Sending query to Ollama

Agent: The capital of Brazil is Brasília.

You:
```

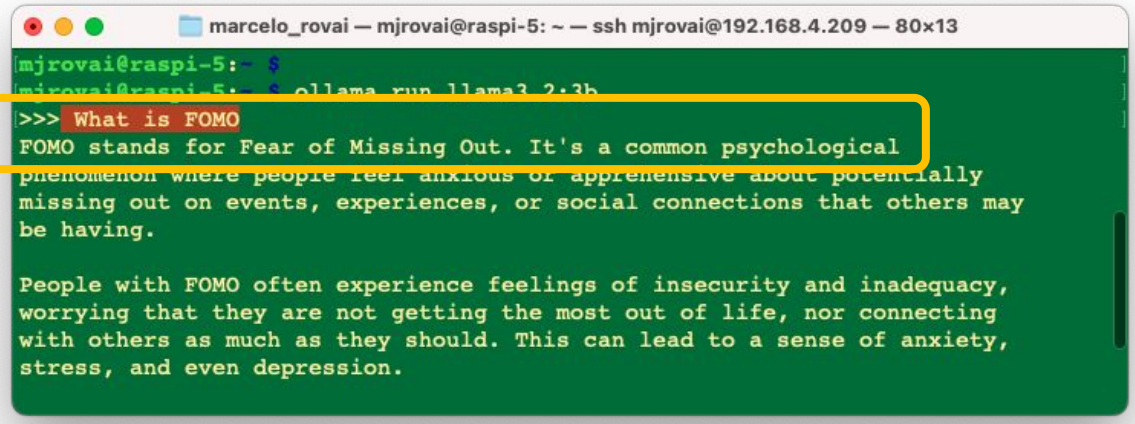
Function Calling



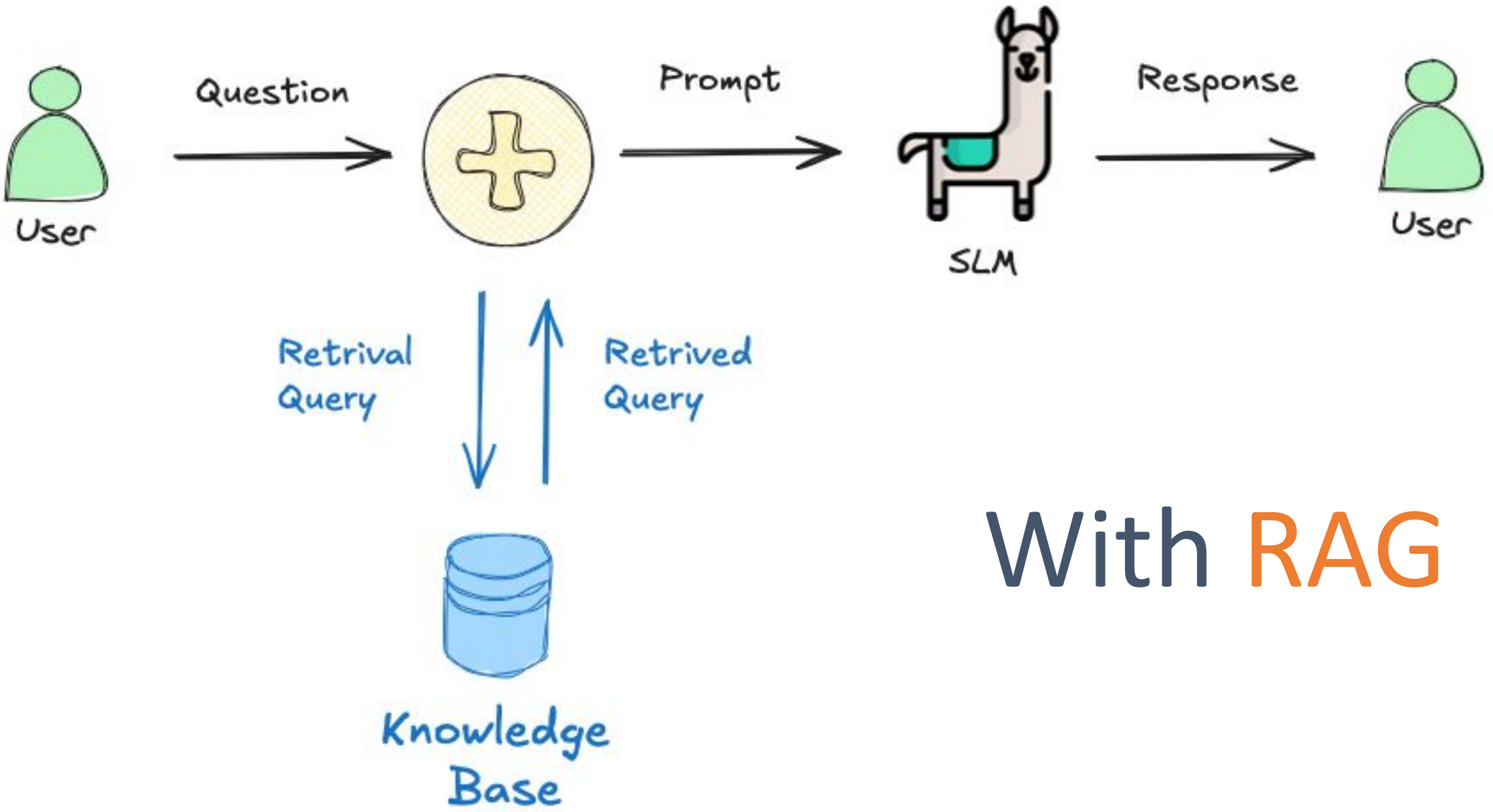
Retrieval-Augmented Generation (RAG)

“A method created by the FAIR team at Meta to enhance the accuracy of Large Language Models (LLMs) and reduce false information or “hallucinations.”

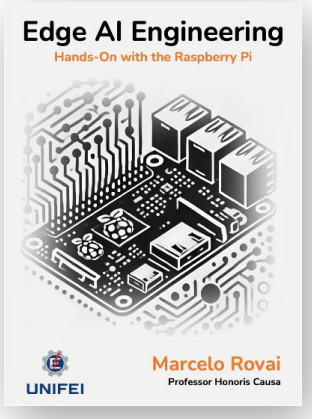




Usual Prompt


User
Question
Prompt
SLM
Response
User
Retrival Query
Retrived Query
Knowledge Base

With RAG



Knowledge Base

```
marcelo_rovai — mjrovai@raspi-5: ~/Documents/Ollama/Rag/edgeai — ssh mjrovai@192.168.4.209 — 100×36

Question: What is FOMO?
Retrieving documents...
Retrieved 2 document chunks
Generating answer...
Response latency: 107.41 seconds using model: llama3.2:3b

ANSWER:
==================================================
FOMO stands for Faster Objects, More Objects.
==================================================

Your question: How to setup a Raspi5?

Generating answer...

Question: How to setup a Raspi5?
Retrieving documents...
Retrieved 2 document chunks
Generating answer...
Response latency: 85.09 seconds using model: llama3.2:3b

ANSWER:
==================================================
To set up a Raspberry Pi 5, follow these steps:

1. Download and install the Raspberry Pi Imager on your computer.
2. Insert a 32GB microSD card into your computer.
3. Open Raspberry Pi Imager and select "Raspberry Pi OS (64-bit)" as the operating system.
4. Select the Raspberry Pi 5 model.
5. Set the hostname, username, password, configure WiFi, and enable SSH in the advanced options.
6. Write the image to the microSD card.

Note: The full 64-bit version of Raspberry Pi OS is recommended for the Raspberry Pi 5.
==================================================

Your question:
```

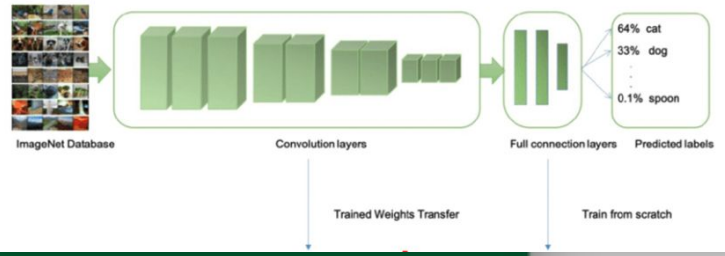
Training a FOMO Model at Edge Impulse Studio

The inference with the SSD MobileNet model worked well, but the latency was significantly high. The inference varied from 0.5 to 1.3 seconds on a Raspi-Zero, which means around or less than 1 FPS (1 frame per second). One alternative to speed up the process is to use FOMO (Faster Objects, More Objects).

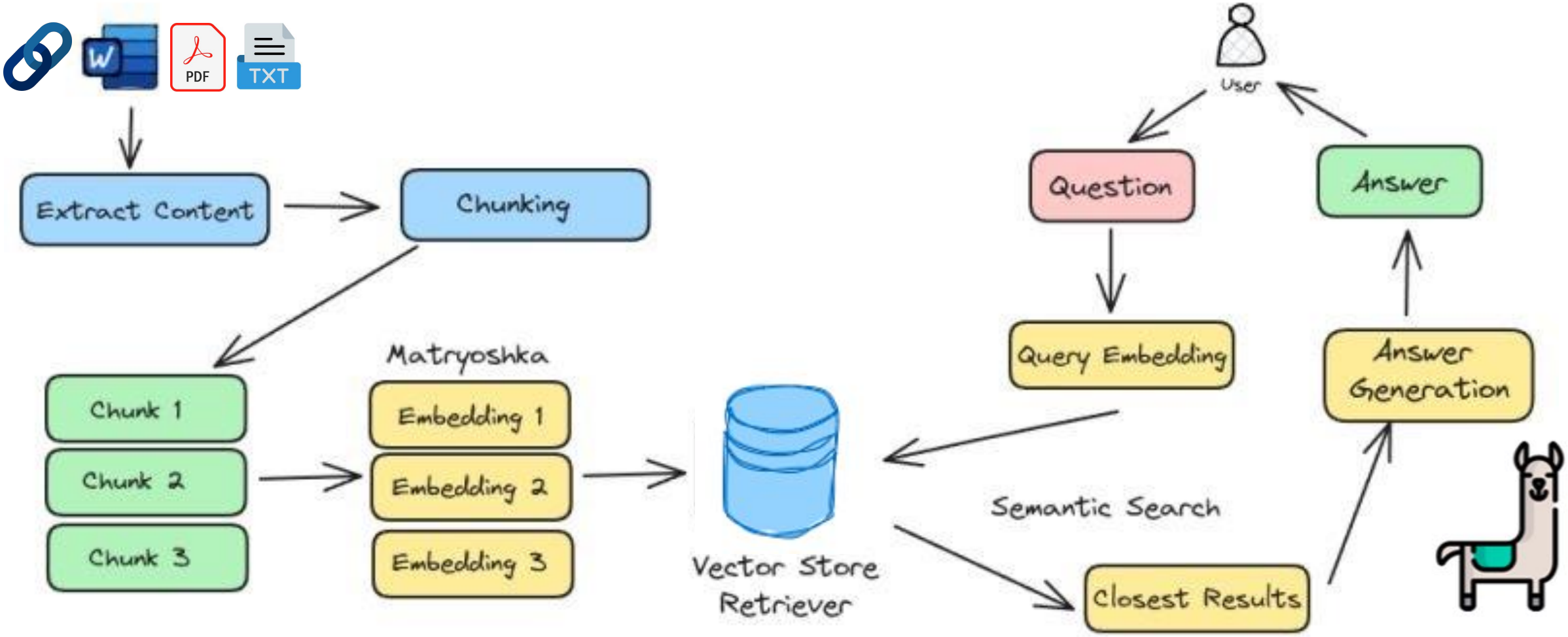
This novel machine learning algorithm lets us count multiple objects and find their location in an image in real-time using up to 30x less processing power and memory than MobileNet SSD or YOLO. The main reason this is possible is that while other models calculate the object's size by drawing a square around it (bounding box), FOMO ignores the size of the image, providing only the information about where the object is located in the image through its centroid coordinates.

How FOMO works?

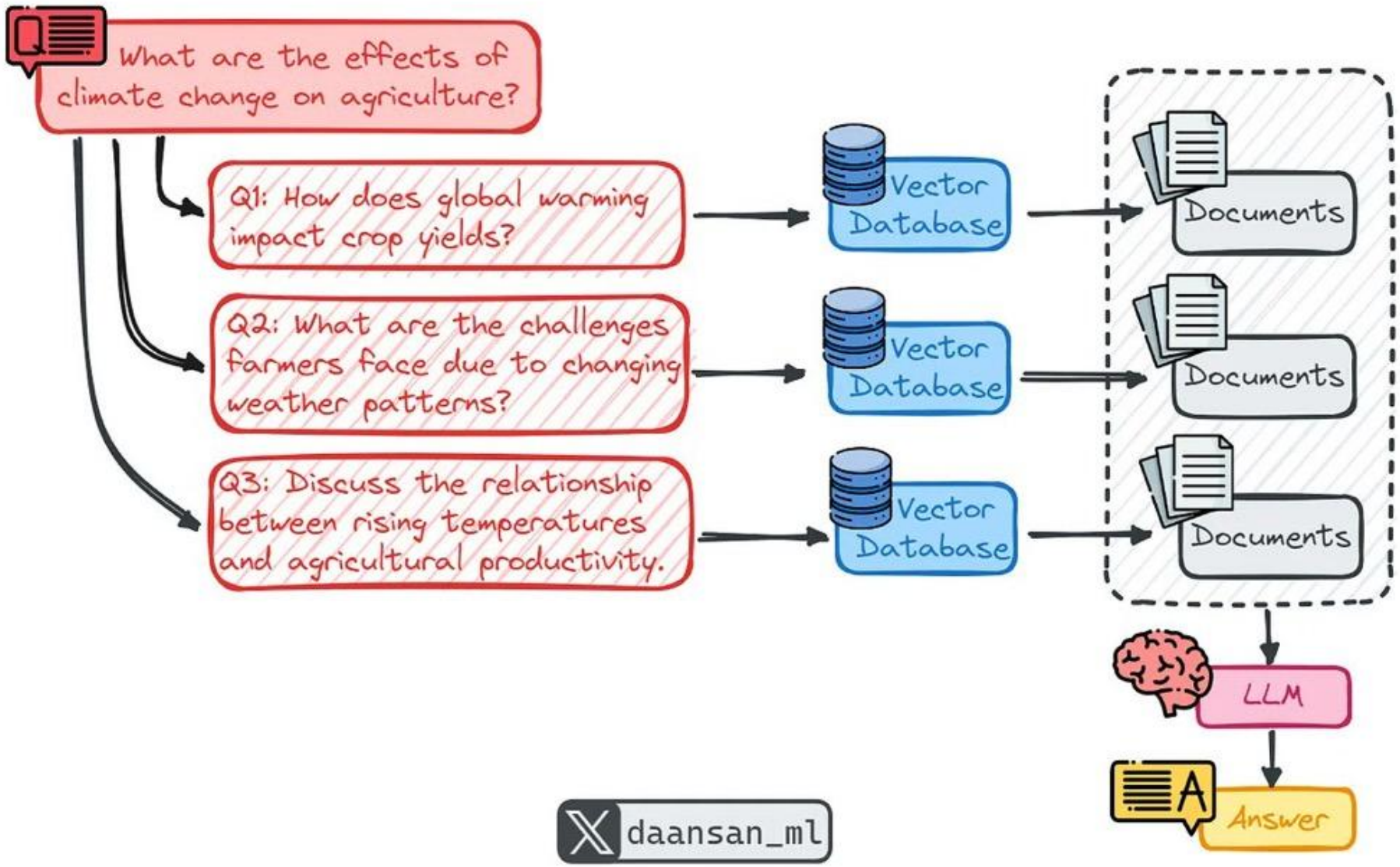
In a typical object detection pipeline, the first stage is extracting features from the input image. **FOMO leverages MobileNetV2 to perform this task**. MobileNetV2 processes the input image to produce a feature map that captures essential characteristics, such as textures, shapes, and object edges, in a computationally efficient way.



RAG: Simple Query



Advanced RAG: Multi Query



VLM

Vision Language Models

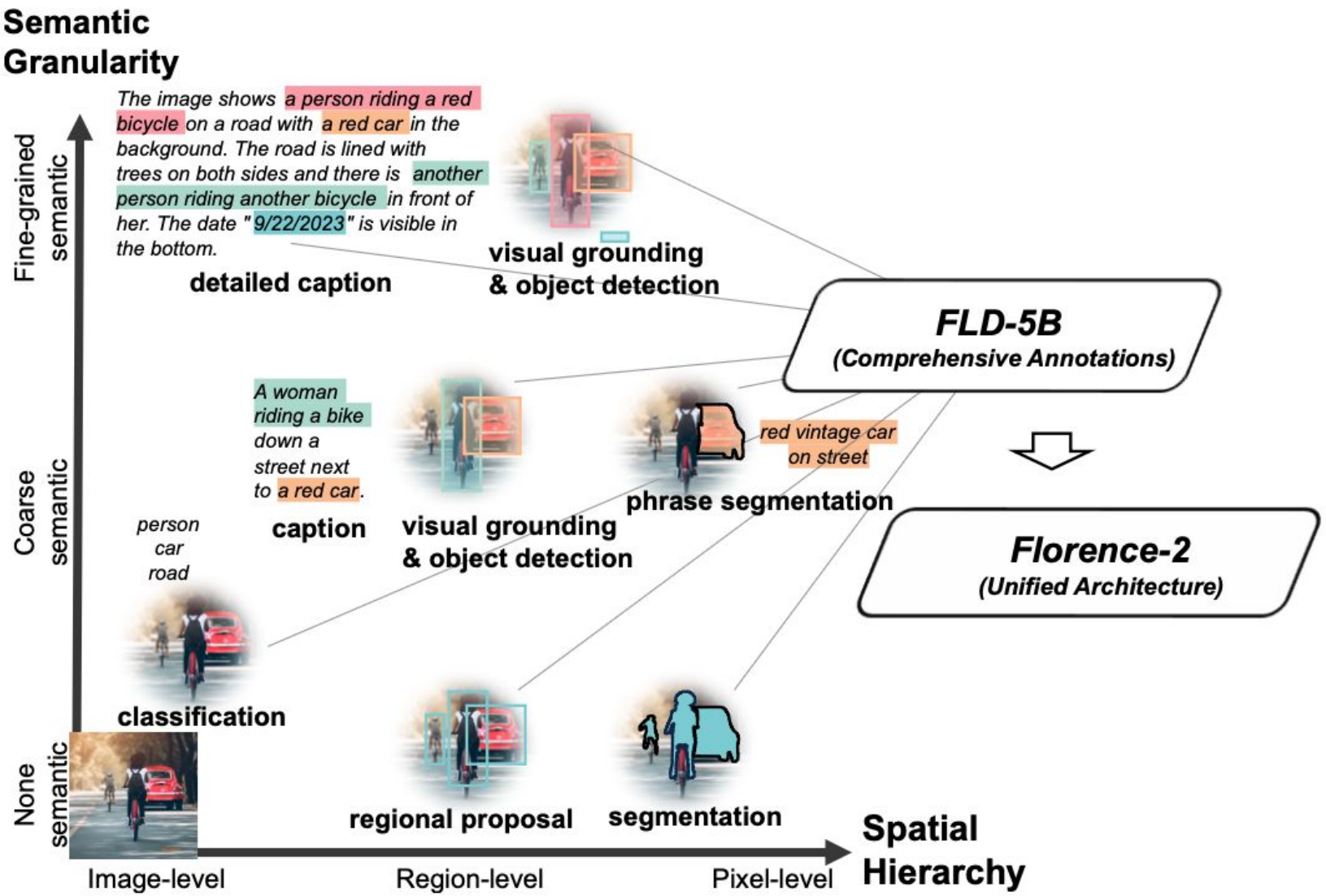
Vision Language Models (VLMs) are artificial intelligence systems integrating computer vision and natural language processing capabilities. This fusion enables them to process, understand, and generate both visual (images, videos) and textual data, allowing for a wide range of multimodal tasks that require joint reasoning across these domains.

Florence-2 stands out for its zero-shot performance, compactness, and ability to handle multiple vision-language tasks without extensive fine-tuning. It is particularly well-suited for scenarios where rapid deployment and efficiency are crucial, such as edge devices or when a unified model is preferred.

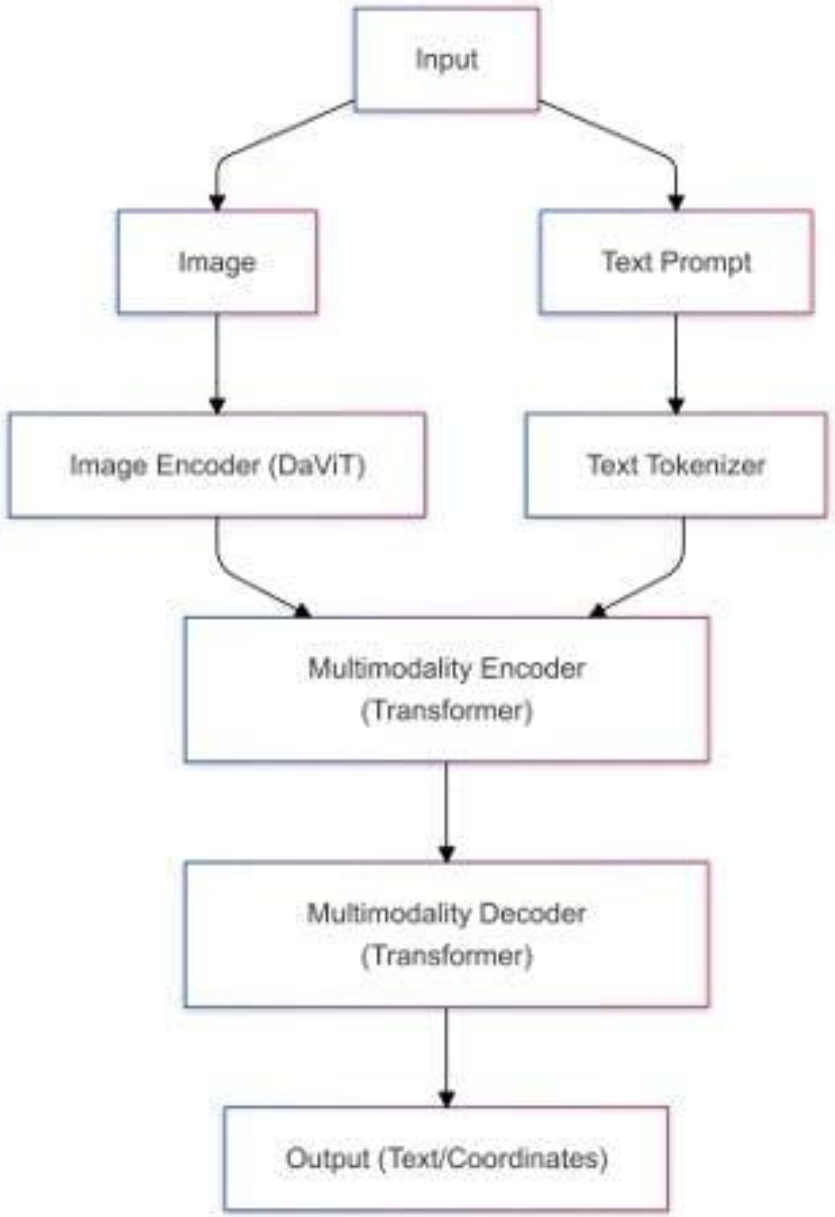
PaliGemma (especially PaliGemma 2) is designed for flexibility and scalability, excelling when fine-tuned for specific tasks or domains. It supports a broader range of languages and higher-resolution images, making it a strong choice for complex, custom, or multilingual applications.

Florence-2

Advancing a Unified Representation for a Variety of Vision Tasks



Paper: https://arxiv.org/abs/2311.06242


Input
Image
Text Prompt
Image Encoder (DaViT)
Text Tokenizer
Multimodality Encoder
(Transformer)
Multimodality Decoder
(Transformer)
Output (Text/Coordinates)

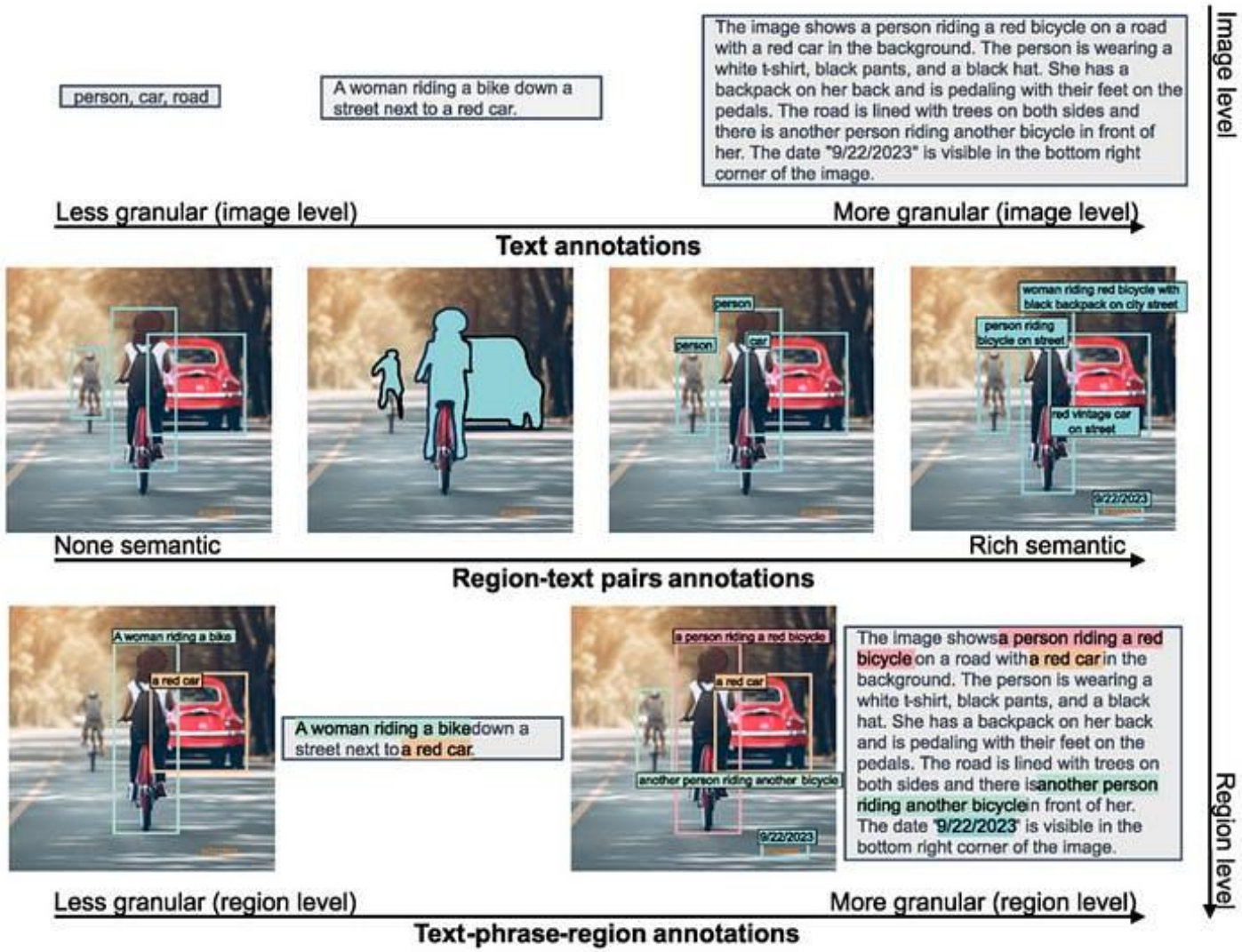

person, car, road
A woman riding a bike down a street next to a red car.
The image shows a person riding a red bicycle on a road with a red car in the background. The person is wearing a white t-shirt, black pants, and a black hat. She has a backpack on her back and is pedaling with their feet on the pedals. The road is lined with trees on both sides and there is another person riding another bicycle in front of her. The date "9/22/2023" is visible in the bottom right corner of the image.
Image level
Less granular (image level)
More granular (image level)
Text annotations
person
car
woman riding red bicycle with black backpack on city street
person riding bicycle on street
red vintage car on street
9/22/2023
None semantic
Rich semantic
Region-text pairs annotations
A woman riding a bike
a red car
A woman riding a bike down a street next to a red car.
a person riding a red bicycle
a red car
another person riding another bicycle
9/22/2023
The image shows a person riding a red bicycle on a road with a red car in the background. The person is wearing a white t-shirt, black pants, and a black hat. She has a backpack on her back and is pedaling with their feet on the pedals. The road is lined with trees on both sides and there is another person riding another bicycle in front of her. The date "9/22/2023" is visible in the bottom right corner of the image.
Region level
Less granular (region level)
More granular (region level)
Text-phrase-region annotations

Image Encoder

What does the image describe?

Locate the objects in the image.

Locate the phrases in the caption: **A woman riding a bike**.

What does the **region (0.41, 0.15, 0.63, 0.73)** describe?

What is the polygon mask of region **(0.41, 0.15, 0.63, 0.73)**?

Multi-task prompts

visual embeddings

text + location embeddings

Transformer Encoders

Transformer Decoders

text + location tokens

The image shows a person riding a red bicycle on a road with a red car in the background. The person is wearing a white t-shirt, black pants, and a black hat. She has a backpack on her back and is pedaling with their feet on the pedals. The road is lined with trees on both sides and there is another person riding another bicycle in front of her. The date "9/22/2023" is visible in the bottom right corner of the image.

person (0.41, 0.15, 0.63, 0.73) … car (0.58, 0.26, 0.89, 0.61)

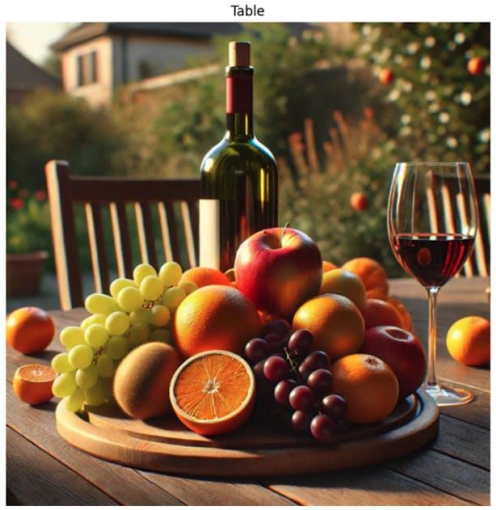
A women riding a bike (0.41, 0.15, 0.63, 0.73

person riding red bicycle on road

(0.48, 0.19, 0.48, 0.18, 0.49, 0.17, ...)

Caption

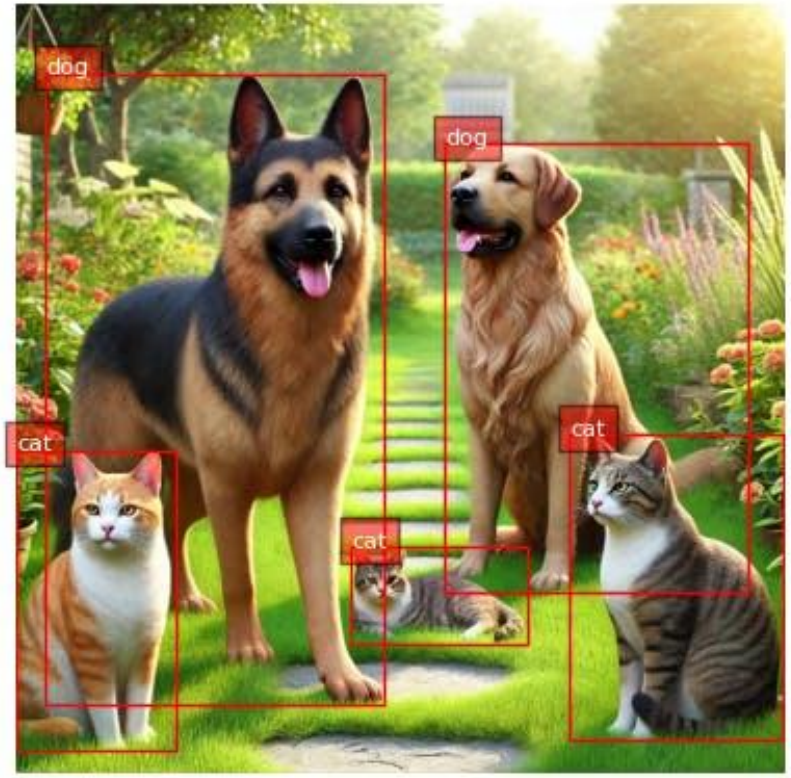
{'<MORE_DETAILED_CAPTION>': 'The image shows a wooden table with a wooden tray on it. On the tray, there are various fruits such as grapes, oranges, apples, and grapes. There is also a bottle of red wine on the table. The background shows a garden with trees and a house. The overall mood of the image is peaceful and serene.'}



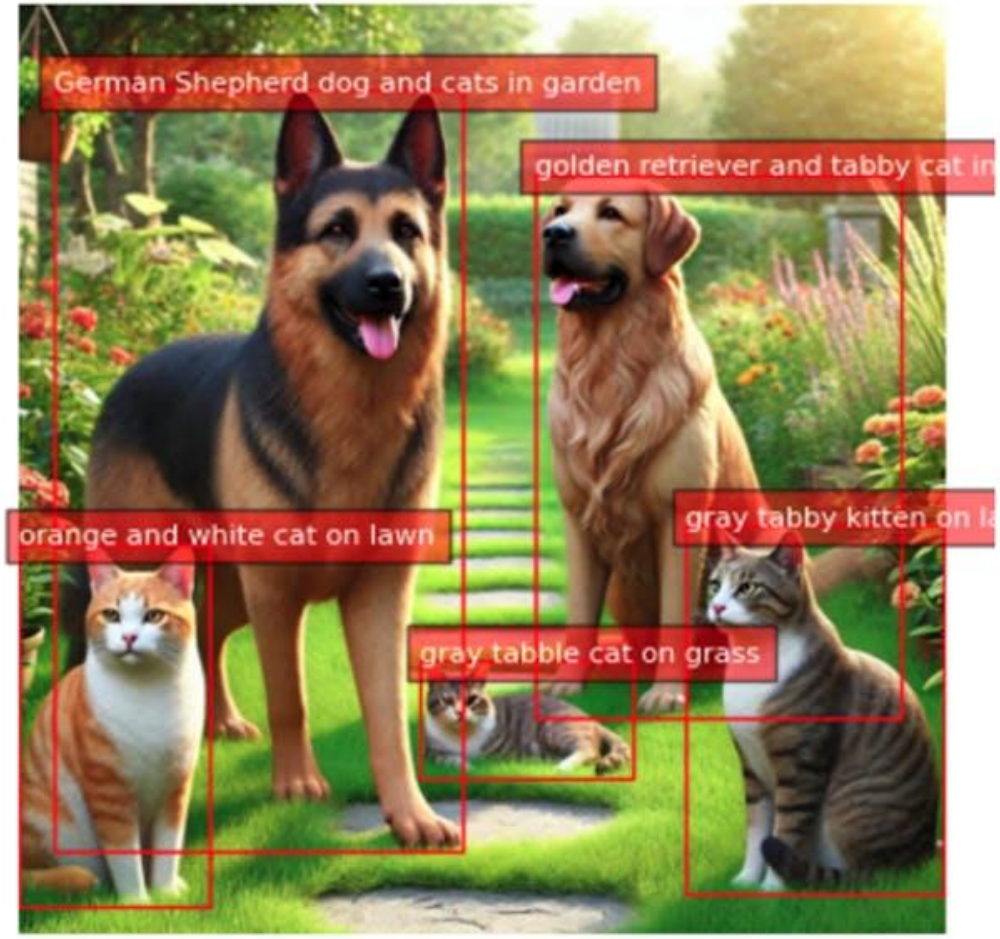
{'<CAPTION>': 'A group of dogs and cats sitting in a garden.'}

{'<DETAILED_CAPTION>': 'The image shows a street with cars and people walking down it, surrounded by buildings with windows, railings, and balconies. There is a tree in the foreground and a clock tower in the background. The sky is filled with clouds and there is a watermark on the image.'}

Object Detection



Dense Region Caption



```
task_prompt = '<CAPTION_TO_PHRASE_GROUNDING>'
results = run_example(task_prompt, text_input="a church clock tower",image=city)
plot_bbox(table, results['<CAPTION_TO_PHRASE_GROUNDING>'])
```

```
[INFO] ==> Florence-2-base (<CAPTION_TO_PHRASE_GROUNDING>), took 12.7 seconds to execute.
```


```
task_prompt = '<CAPTION_TO_PHRASE_GROUNDING>'
results = run_example(task_prompt, text_input="a person dressed in white",image=city)
plot_bbox(table, results['<CAPTION_TO_PHRASE_GROUNDING>'])
```

```
[INFO] ==> Florence-2-base (<CAPTION_TO_PHRASE_GROUNDING>), took 12.3 seconds to execute.
```

Segmentation

OCR


```
results['<OCR_WITH_REGION>']['labels']
```

```
['</s>Machine Learning',
 'Café',
 'com',
 'Embarcado',
 'Embarcados',
 'Democratizando a Inteligência',
 'Artificial para Paises em',
 '25 de Setembro ás 17h',
 'Desenvolvimento',
 'Toda quarta-feira',
 'Marcelo Roval',
 'Professor na UNIFIEI e',
 'Transmissão via',
 'in',
 'Co-Director do TinyML4D']
```

Fine-Tunning

{"<OD>": {"bboxes": [[0.1599999964237213, 133.59999084472656, 78.23999786376953, 232.1599884033203], [117.27999877929688, 139.0399932861328, 196.63999938964844, 243.67999267578125], [190.239990234375, 193.1199951171875, 270.239990234375, 319.5199890136719], [248.1599884033203, 91.04000091552734, 319.5199890136719, 189.27999877929688], [160.8000030517578, 27.680000305175781, 221.27999877929688, 118.23999786376953], [0.1599999964237213, 0.1599999964237213, 86.23999786376953, 57.119998931884766], [35.360000610351562, 36.31999969482422, 104.15999603271484, 112.15999603271484], [0.1599999964237213, 0.47999998927116394, 319.5199890136719, 319.5199890136719]], "labels": ["wheel", "wheel", "box", "box", "box", "box", "wheel", "box"]}}

The Future...

The Future of Generative AI at the Edge

Generative AI is rapidly transitioning from centralized cloud environments to edge computing, transforming how data is processed and utilized across industries. This shift is gaining significant momentum, with Gartner projecting that GenAI will be featured in 60% of edge computing deployments by 2029, up from just 5% in 2023. The Edge AI market is expected to reach $269.82 billion by 2032, growing at a compound annual rate of 33.3%.

Edge-based GenAI is enabling transformative applications across multiple sectors:

- Autonomous Vehicles: Real-time sensor analysis and decision-making
- Healthcare: On-device diagnostics and privacy-preserving patient monitoring
- Retail: Voice-assisted shopping and interactive customer service
- Industrial IoT: Predictive maintenance and anomaly detection
- Smart Devices: Real-time translation and augmented reality experiences

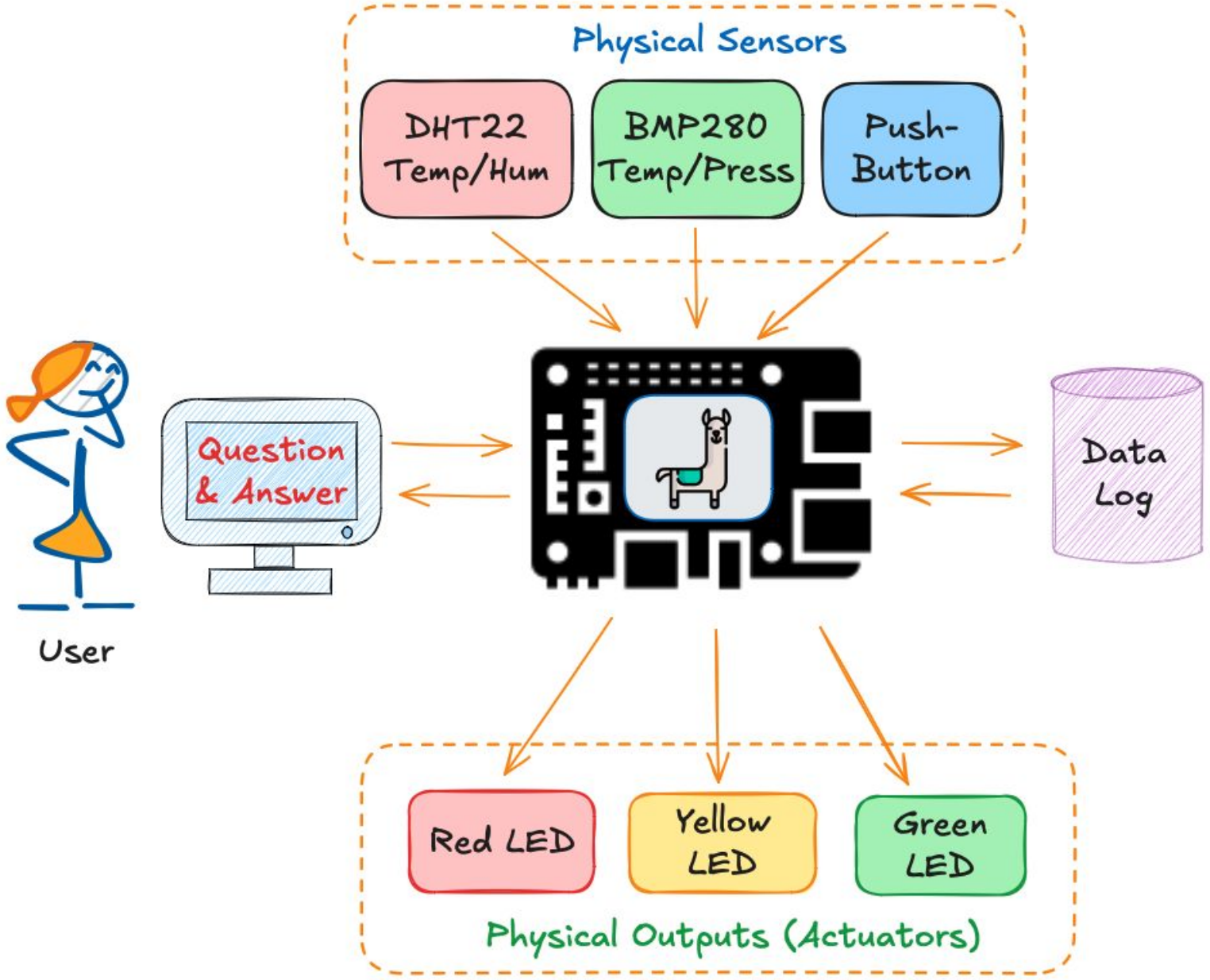
microsoft/BitNet

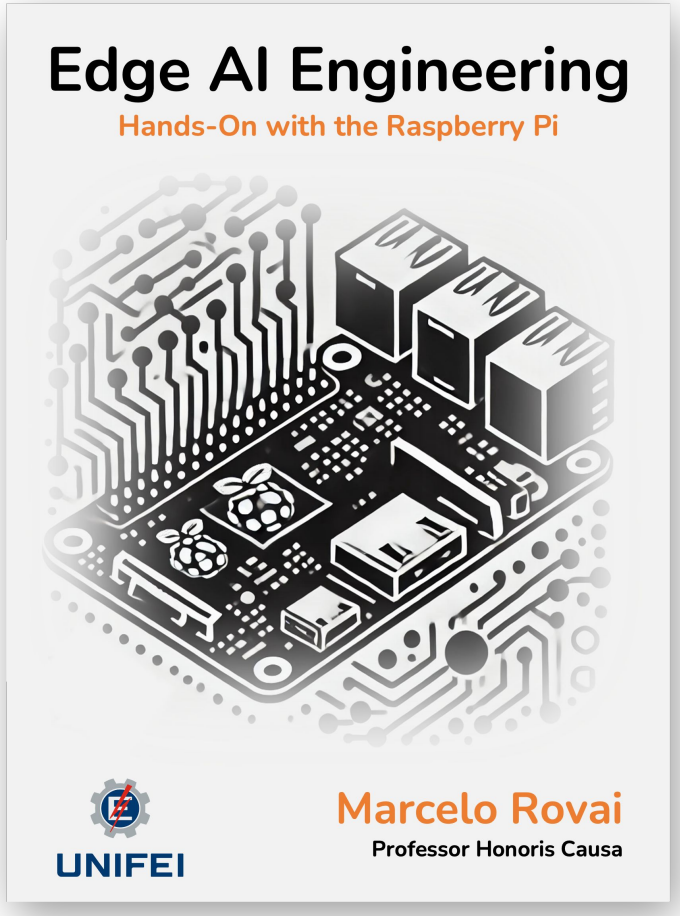
Official inference framework for 1-bit LLMs

Bitnet.cpp employs one-bit quantization, representing values with a ternary system (+1, -1, 0). This approach simplifies calculations by replacing complex multiplications with additions and subtractions, eliminating the need for GPUs.

- Speedups range from 1.37x to 6.1x on various CPUs.
- Power consumption reductions between 55.4% and 82.2% compared to traditional GPU-based inference.

bitnet.cpp





EdgeAI Engineering

To learn more …

Online Courses

Harvard School of Engineering and Applied Sciences - CS249r: Tiny Machine Learning
Professional Certificate in Tiny Machine Learning (TinyML) – edX/Harvard
Introduction to Embedded Machine Learning - Coursera/Edge Impulse
Computer Vision with Embedded Machine Learning - Coursera/Edge Impulse
UNIFEI-IESTI01 TinyML: “Machine Learning for Embedding Devices”

Books

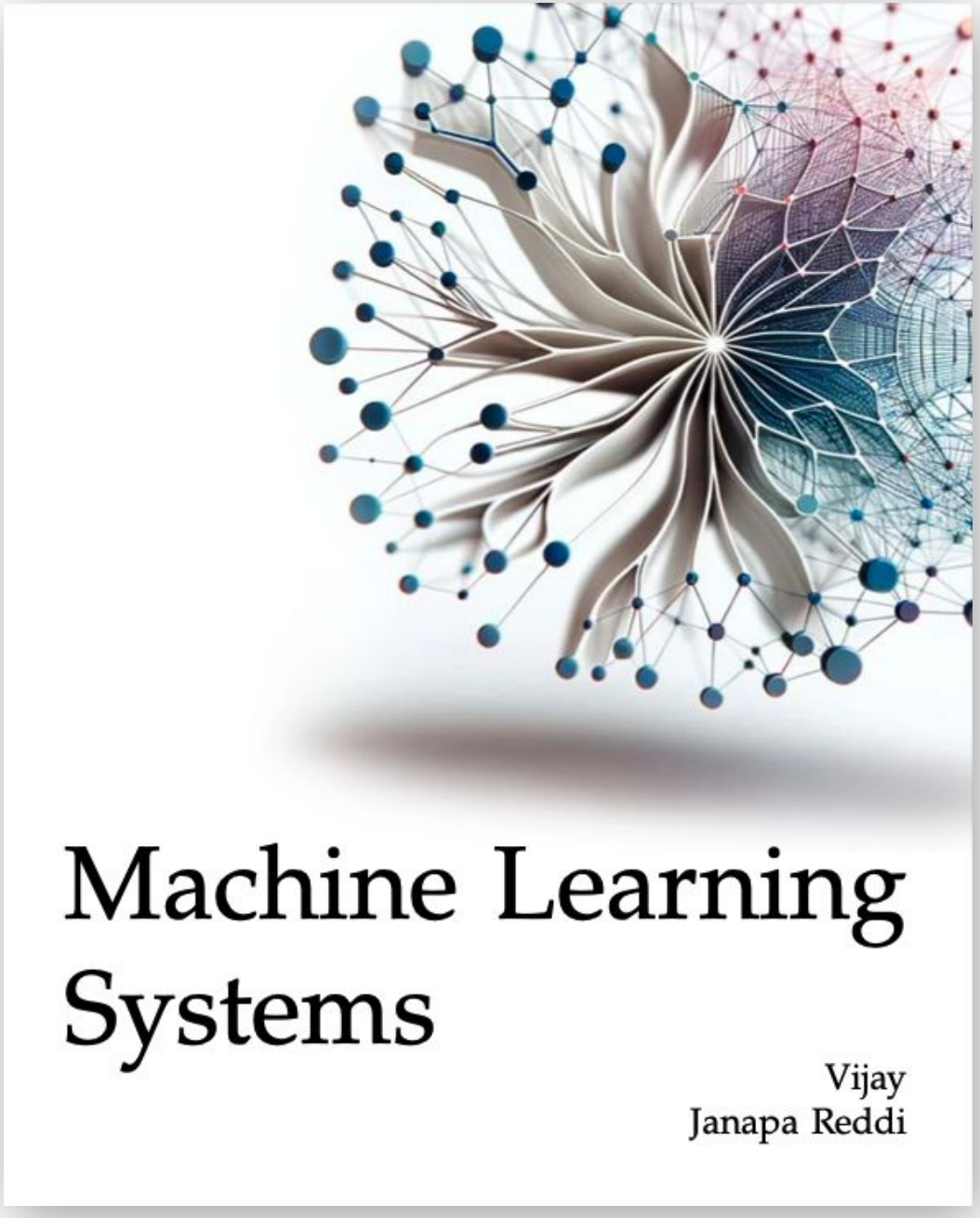
“Python for Data Analysis” by Wes McKinney
“Deep Learning with Python” by François Chollet - GitHub Notebooks
“TinyML” by Pete Warden and Daniel Situnayake
“TinyML Cookbook 2nd Edition” by Gian Marco Iodice
“Technical Strategy for AI Engineers, In the Era of Deep Learning” by Andrew Ng
“AI at the Edge” book by Daniel Situnayake and Jenny Plunkett
“XIAO: Big Power, Small Board” by Lei Feng and Marcelo Rovai
“Machine Learning Systems” by Vijay Janapa Reddi

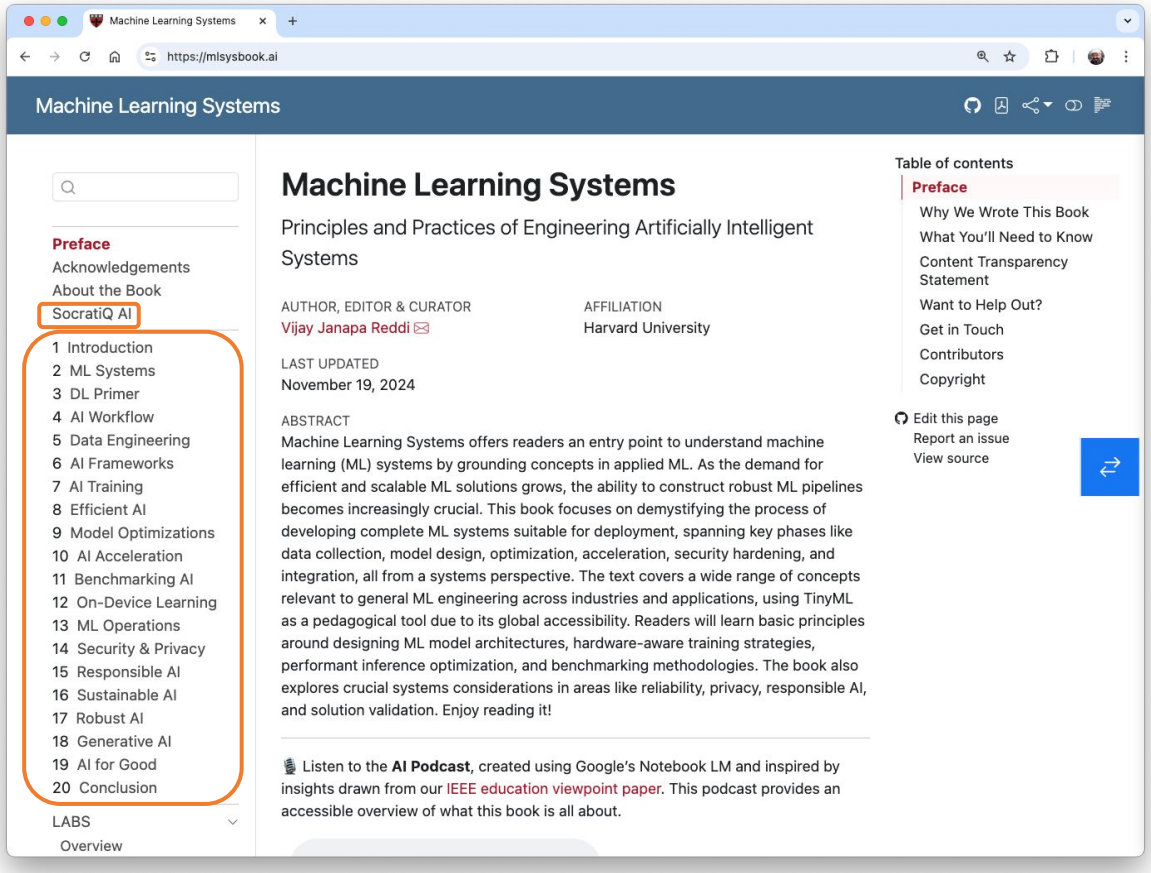
Projects Repository

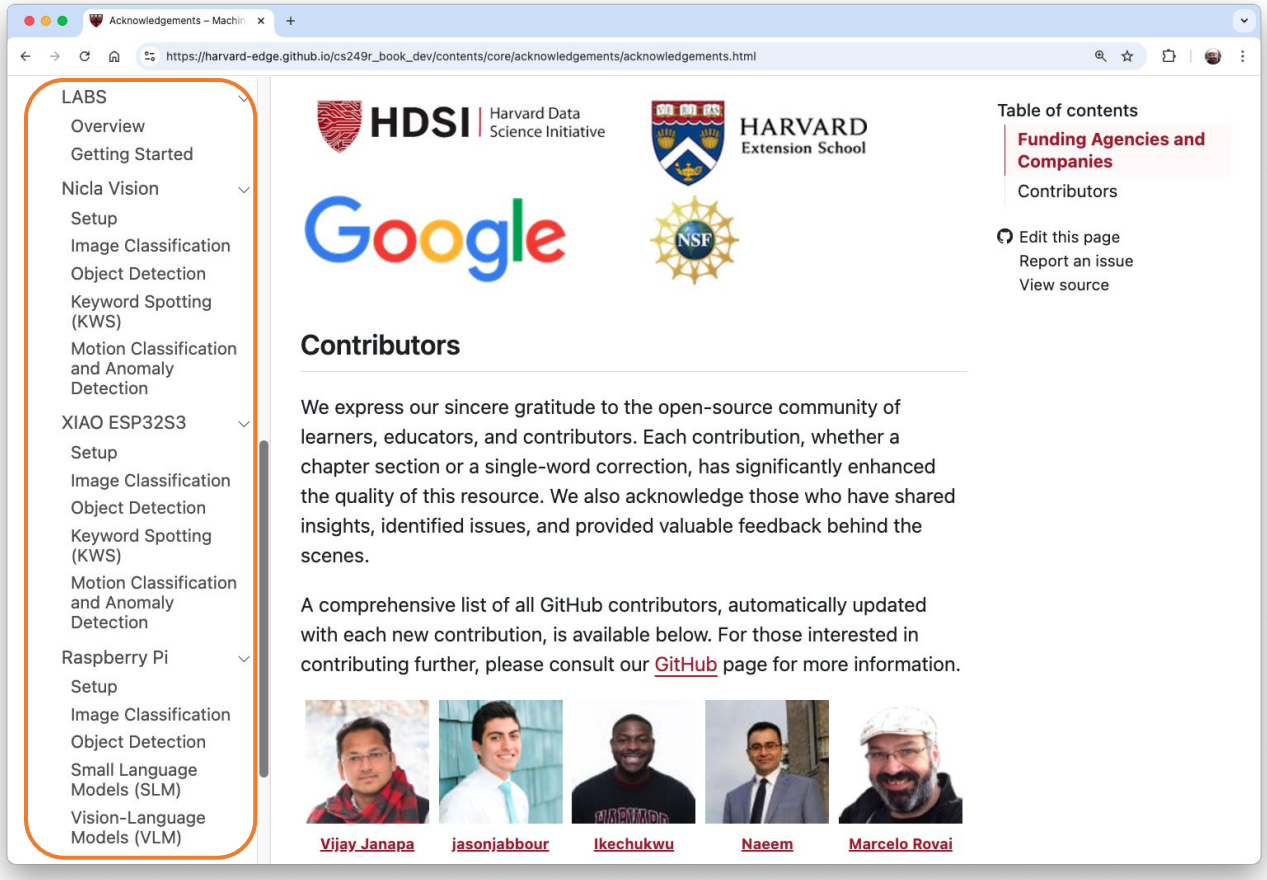
Edge Impulse Expert Network

On the TinyML4D website, You can find lots of educational materials on TinyML. They are all free and open-source for educational uses – we ask that if you use the material, please cite them! TinyML4D is an initiative to make TinyML education available to everyone globally.

https://mlsysbook.ai/



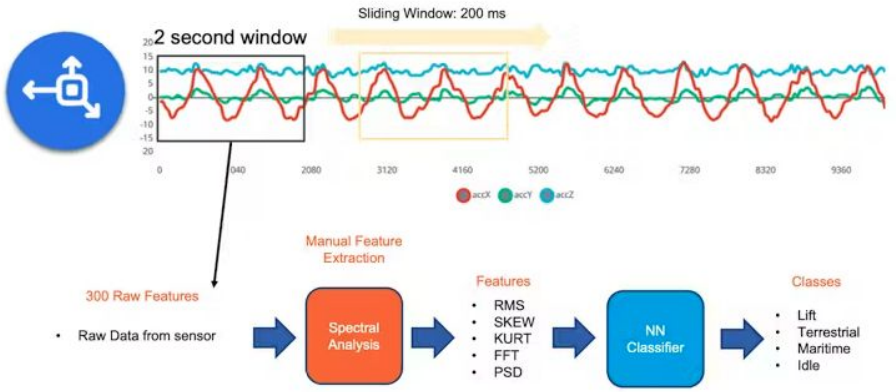

Machine Learning Systems
https://mlsysbook.ai
Machine Learning Systems
Preface
Acknowledgements
About the Book
SocratiQ AI
1 Introduction
2 ML Systems
3 DL Primer
4 AI Workflow
5 Data Engineering
6 AI Frameworks
7 AI Training
8 Efficient AI
9 Model Optimizations
10 AI Acceleration
11 Benchmarking AI
12 On-Device Learning
13 ML Operations
14 Security & Privacy
15 Responsible AI
16 Sustainable AI
17 Robust AI
18 Generative AI
19 AI for Good
20 Conclusion
LABS
Overview
Machine Learning Systems
Principles and Practices of Engineering Artificially Intelligent Systems
AUTHOR, EDITOR & CURATOR
Vijay Janapa Reddi
AFFILIATION
Harvard University
LAST UPDATED
November 19, 2024
ABSTRACT
Machine Learning Systems offers readers an entry point to understand machine learning (ML) systems by grounding concepts in applied ML. As the demand for efficient and scalable ML solutions grows, the ability to construct robust ML pipelines becomes increasingly crucial. This book focuses on demystifying the process of developing complete ML systems suitable for deployment, spanning key phases like data collection, model design, optimization, acceleration, security hardening, and integration, all from a systems perspective. The text covers a wide range of concepts relevant to general ML engineering across industries and applications, using TinyML as a pedagogical tool due to its global accessibility. Readers will learn basic principles around designing ML model architectures, hardware-aware training strategies, performant inference optimization, and benchmarking methodologies. The book also explores crucial systems considerations in areas like reliability, privacy, responsible AI, and solution validation. Enjoy reading it!
Listen to the AI Podcast, created using Google's Notebook LM and inspired by insights drawn from our IEEE education viewpoint paper. This podcast provides an accessible overview of what this book is all about.
Table of contents
Preface
Why We Wrote This Book
What You'll Need to Know
Content Transparency Statement
Want to Help Out?
Get in Touch
Contributors
Copyright
Edit this page
Report an issue
View source


https://harvard-edge.github.io/cs249r_book_dev/contents/core/acknowledgements/acknowledgements.html
LABS
Overview
Getting Started
Nicla Vision
Setup
Image Classification
Object Detection
Keyword Spotting (KWS)
Motion Classification and Anomaly Detection
XIAO ESP32S3
Setup
Image Classification
Object Detection
Keyword Spotting (KWS)
Motion Classification and Anomaly Detection
Raspberry Pi
Setup
Image Classification
Object Detection
Small Language Models (SLM)
Vision-Language Models (VLM)
HDSI Harvard Data Science Initiative
HARVARD Extension School
Google
NSF
Contributors
We express our sincere gratitude to the open-source community of learners, educators, and contributors. Each contribution, whether a chapter section or a single-word correction, has significantly enhanced the quality of this resource. We also acknowledge those who have shared insights, identified issues, and provided valuable feedback behind the scenes.
A comprehensive list of all GitHub contributors, automatically updated with each new contribution, is available below. For those interested in contributing further, please consult our GitHub page for more information.
Vijay Janapa
jasonjabbour
Ikechukwu
Naeem
Marcelo Rovai
Table of contents
Funding Agencies and Companies
Contributors
Edit this page
Report an issue
View source

Section Quiz

Data Pre-Processing

The raw data type captured by the accelerometer is a "time series" and should be converted to "tabular data". We can do this conversion using a sliding window over the sample data. For example, in the below figure,



We can see 10 seconds of accelerometer data captured with a sample rate (SR) of 50Hz. A 2-second window will capture 300 data points (3 axis x 2 seconds x 50 samples). We will slide this window each 200ms, creating a larger dataset where each instance has 300 raw features.

> You should use the best SR for your case, considering Nyquist's theorem, which states that a periodic signal must be sampled at more than twice the signal's highest frequency component.

Data preprocessing is a challenging area for embedded machine learning. Still, Edge Impulse helps overcome this with its digital signal processing (DSP) preprocessing step and, more specifically, the Spectral Features.

On the Studio, this dataset will be the input of a Spectral Analysis block, which is excellent for analyzing repetitive motion, such as data from accelerometers. This block will perform a DSP (Digital Signal Processing), extracting features such as "FFT" or "Wavelets". In the most common case, FFT, the **Time Domain Statistical features** per axis/channel are:

SocratiQ

Q2: Which theorem helps determine an appropriate sample rate for periodic signals?

- (●) A1: Nyquist's theorem
  Correct
  Nyquist's theorem dictates that a periodic signal should be sampled at more than twice the signal's highest frequency component, which aids in selecting a suitable sample rate.
- ( ) A2: Shannon's theorem
  Shannon's theorem is related to sampling, but does not offer a concrete numerical relationship between sample rate and the maximum frequency.
- ( ) A3: Whittaker–Kotel'nikov–Shannon theorem
  This theorem uses a similar assertion to Nyquist but often concerns itself additionally with optimal signal reconstruction.

Q3: Considering the Spectral Analysis block in Edge Impulse, why is it proper to analyze repetitive motion, such as accelerometer data?

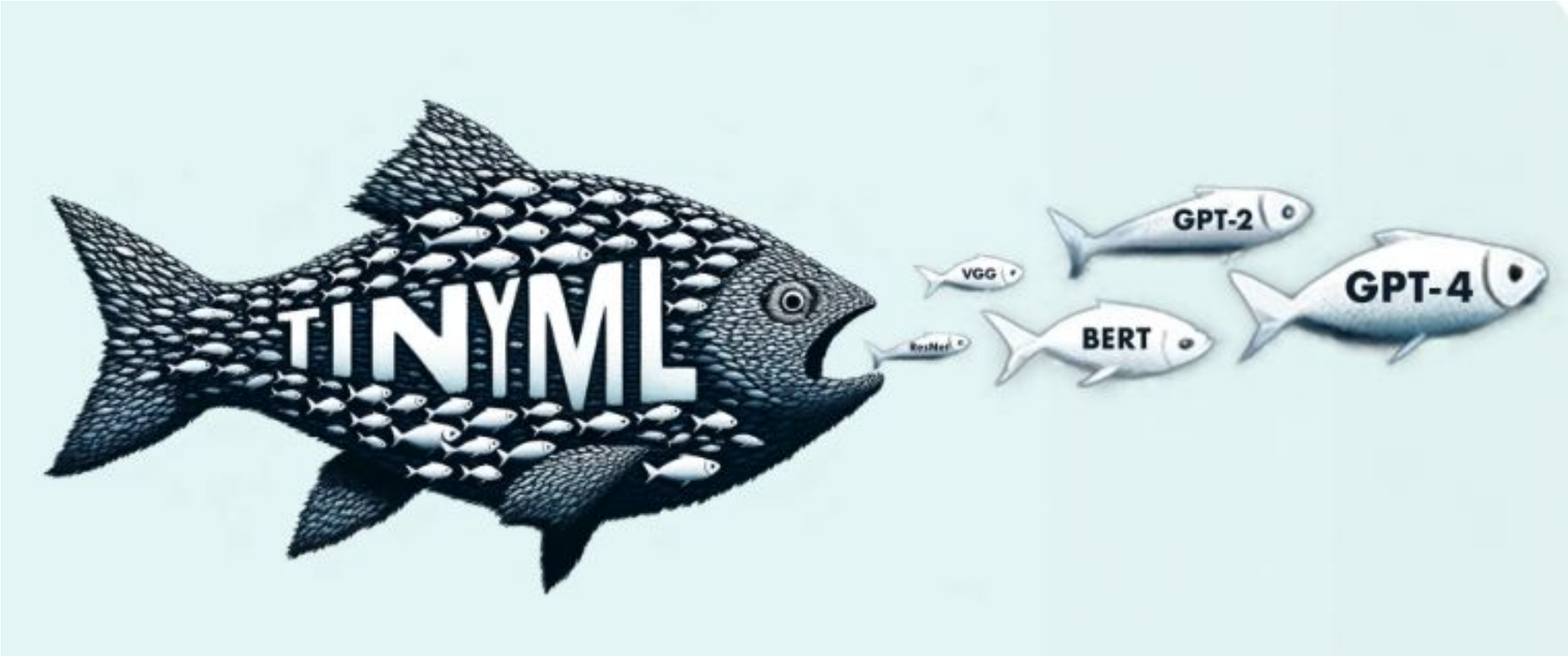
- (●) A1: Because repetitive motion generally contains obvious spectral patterns in the frequency domain
  Correct

+ Add Context

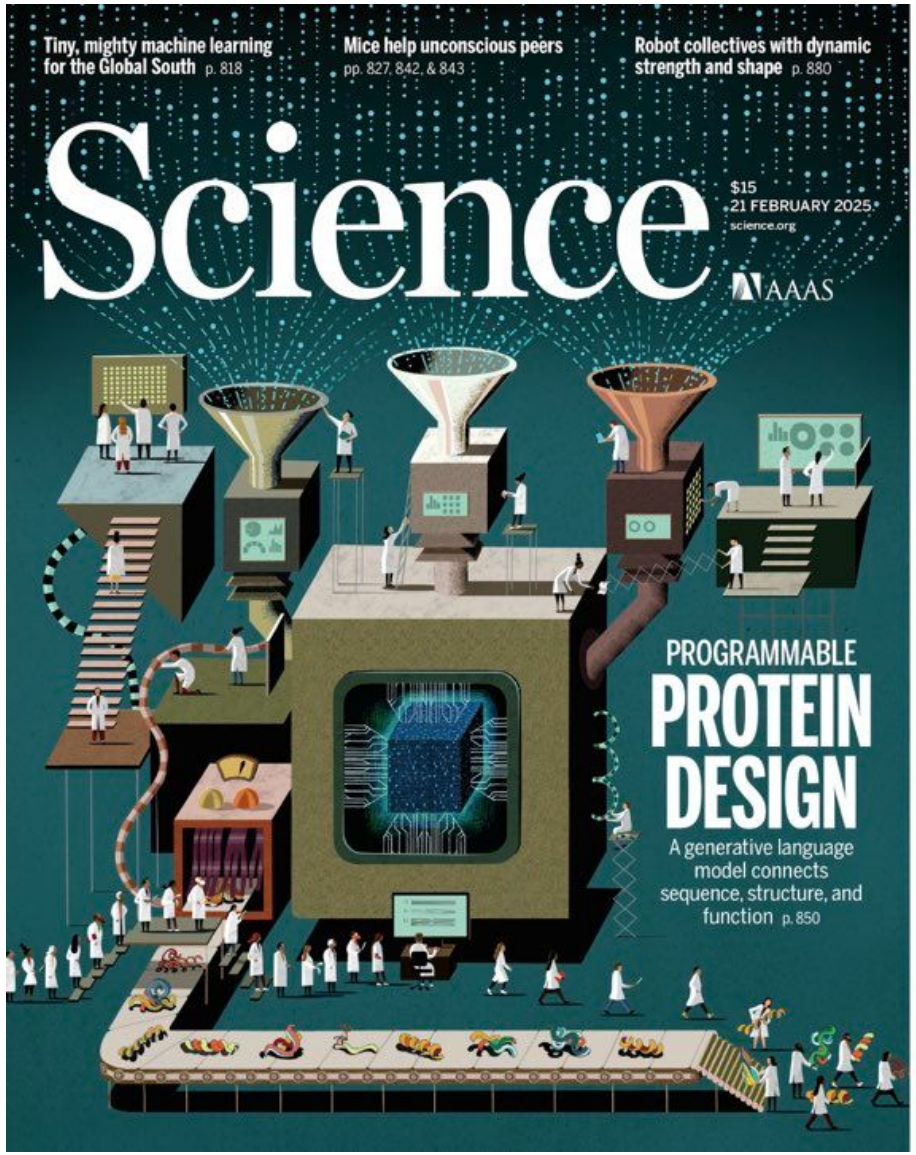
type '@' to reference a section...

Information provided here may not always be accurate. Provide feedback

TinyML: Why the Future of Machine Learning is Tiny and Bright



Shvetank Prakash, Emil Njor, Colby Banbury, Matthew Stewart, Vijay Janapa Reddi



FEATURES

CUTTING AI DOWN TO SIZE

A $14 chip incorporating tinyML AI models, actual size shown.

https://www.science.org/content/article/what-s-tinyml-global-south-s-alternative-power-hungry-pricey-ai

Pontificia Universidad JAVERIANA Colombia

Questions?

Prof. Marcelo J. Rovai

rovai@unifei.edu.br

UNIFEI - Federal University of Itajuba, Brazil

TinyML4D - Academic Network Co-Chair

EdgeAIP - Academia-Industry Partnership Co-Chair