Hogeschool van Amsterdam

sense maker samst erdam

# ollama meetup 3


## Michiel Bontenbal

## Sensemakers Amsterdam


## 19 june 2024

# Ollama meetup 3 Agenda

**First part** - together

- Short recap of previous meetups
- Vision Language Models
- Talk with your chatbot  - Speech to Text and Text to Speech

**Second part** – work

- Work on speech-2-text and text-2-speech
- Work on your project
- Show and Tell

# ollama meetups 3rd wednesday

sense maker samst erdam

**17 april : Introduction**
- Starting with ollama
- Program with ollama and Python
- Challenges, your ideas and show & tell

**~~15 May~~ => 29 May**
- Recap of first meetup & fresh start for newcomers
- Bring your own model
- Vision-Language Models  (VLM's)
- "Chat with your document" – using Langchain (aka RAG)
- Challenges, your ideas and show & tell

**19 June:**
- Recap
- Speech to text & Text to speech
- Front-end
- Challenges, your ideas and show & tell

# To do list - Create a chatbot

| ***Our to do list*** | |
|---|---|
| Run LLM's locally | ✅ |
| Personalise model with modelfile | ✅ |
| Create a webapp front-end with Gradio | ✅ |
| Work with Vision I | ✅ |
| Pick any model from Huggingface (e.g. Dutch) | ✅ |
| Chat with your document | ✅ |
| Vision Language Models | Today |
| Text to speech / speech to text | Today |

sense maker samst erdam

# Some questions… raise your hand!

- Who has worked with:
  - Python and Jupyter Notebook
  - Speech2Text and Text2Speech

# Recap previous meetups

sense maker samst erdam

# Why ollama?
# Why running LLM’s on your laptop?

1. Data stays on your device
   - Privacy  / no leakage of sensitive data / company policy
2. No longer dependent on internet connection
3. Lower costs with less data usage
4. Saves energy usage
5. Bring your own model (this meetup)

sense maker samst erdam

# Models for Language, Vision and Coding

## Chat

```
>>> What is the captial of the Netherlands?
 The capital of the Netherlands is Amsterdam.
```

## Ask questions about images


## Create and improve code

```
>>> create a python scriot to capture an image with the webcam
```python
import cv2

# Load the webcam
cap = cv2.VideoCapture(0)

# Check if the webcam is accessible
if not cap.isOpened():
    print("Error: Unable to open webcam.")
    exit()

# Capture an image
ret, frame = cap.read()

# Check if the image was captured successfully
if not ret:
    print("Error: Unable to capture image.")
    exit()

# Save the image
cv2.imwrite("webcam_image.jpg", frame)

# Release the webcam
cap.release()

print("Image captured successfully.")
```
```

sense maker samst erdam

# Models come in different sizes – use the right size for your laptop

For example, Llama2 has 3 sizes: 7B, 13B and 70B model.

- 7B means 7 billion parameters which is ± 4 Gigabytes in size.

Check your hardware:

- 7b models need ± 8GB of RAM
- 13b models need ± 16GB of RAM
- 70b models need ± 64GB of RAM

Evaluate the performance of your model with:

```
ollama run tinyllama --verbose
```

sense maker samst erdam

# run ollama

- Download and install via www.ollama.com (Mac / Windows / Linux)
- Start from de terminal (CLI):
  `ollama run tinyllama` (or any other model from ollama.com)
- Start chatting with the model!
- End ollama with `CTRL+C` or `/bye` or `/exit`
- You can then start another model.

# Programming ollama with Python

sense maker samst erdam

# Why program ollama with Python?

- Build a chatbot for in your browser (GUI) — Today!
- Integrate ollama in a webapplication: — Next month
  - Summarizing your own documents
  - Question answering
  - Visual question answering

sense maker samst erdam

# Installing your programming tools

- Install Python: https://www.python.org/
- Install Visual Studio Code: https://code.visualstudio.com/
- Install the python VS code plugin, see:
  - https://code.visualstudio.com/docs/python/python-tutorial

sense maker samst erdam

# Python programming ollama

Use the notebook to use python code from github:

`ollama.ipynb`

P.S. If you prefer JavaScrpt there is also a Node.js package:

`npm install ollama`

# Run models from Huggingface

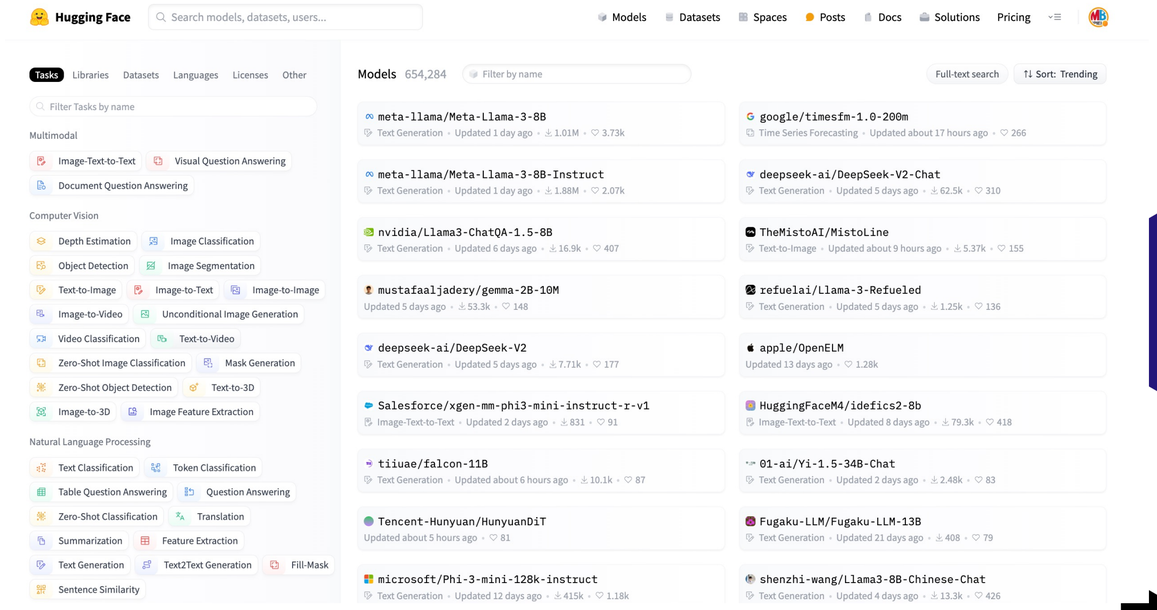
# Huggingface Hub

- World’s biggest platform for AI models, datasets
- Individuals, research and Big Tech publish their work there
- Everything is open source

Huggingface also has

- Python packages like transformers, pipelines
- Courses
- Spaces

- Can we find a **quantized model** in Dutch?



# Let’s try this with some Dutch models.

You might want to learn dutch or chat in Dutch, then use a Dutch model.

Two models have been developed for Dutch

- GEITje
- Fietje

```
ollama run bramvanroy/geitje-7b-ultra-gguf
```

- More: https://goingdutch.ai/nl/posts/why-geitje/

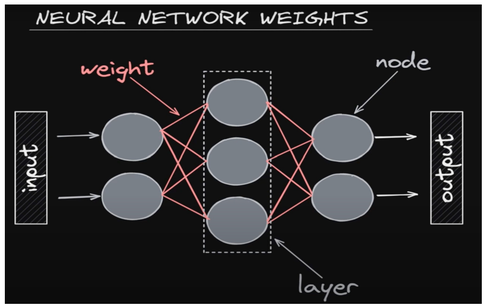
The .gguf file format indicates it’s a **quantized version** of a model

# Make models smaller with ‘quantization’

Quantization is a reduction in precision, like in the image of president Obama.

**Example:**

- reduce precision of the weights from *floating point32* (=32 bits) to *integer8* (=8 bits) values.
- For LLM’s we call this ‘q8 quantization’




FP32: [33.623563422, 12.646104098, -51.583920991, …]

FP16: [33.6234, 12.6461, -51.5839, …]

INT8: [82, 44, -23, …]

More info: https://github.com/ggerganov/llama.cpp llama.cpp project

# Exercise: install model from HF

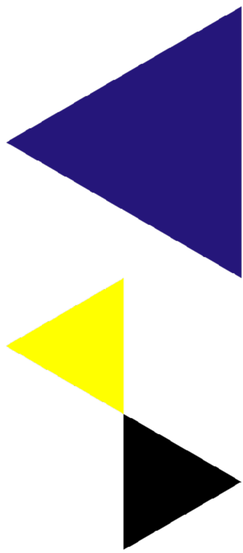
- Go to huggingface hub/models
- Filter on ‘GGUF’ and language
- Download a model from huggingface that you would like.

*Sometimes* you can run it directly in ollama:
`ollama run bramvanroy/geitje-7b-ultra-gguf`

*Sometimes* it links right back to ollama.com:
https://huggingface.co/BramVanroy/fietje-2b-chat-gguf

*Sometimes*: download the gguf manually, create a modelfile, run the model. See
https://youtu.be/fnvZJU5Fj3Q?t=153

Exercise: Use your python code with this new model.



# A deeper look at Vision Language Models

- Intro
- CLIP
- Vision Language Models

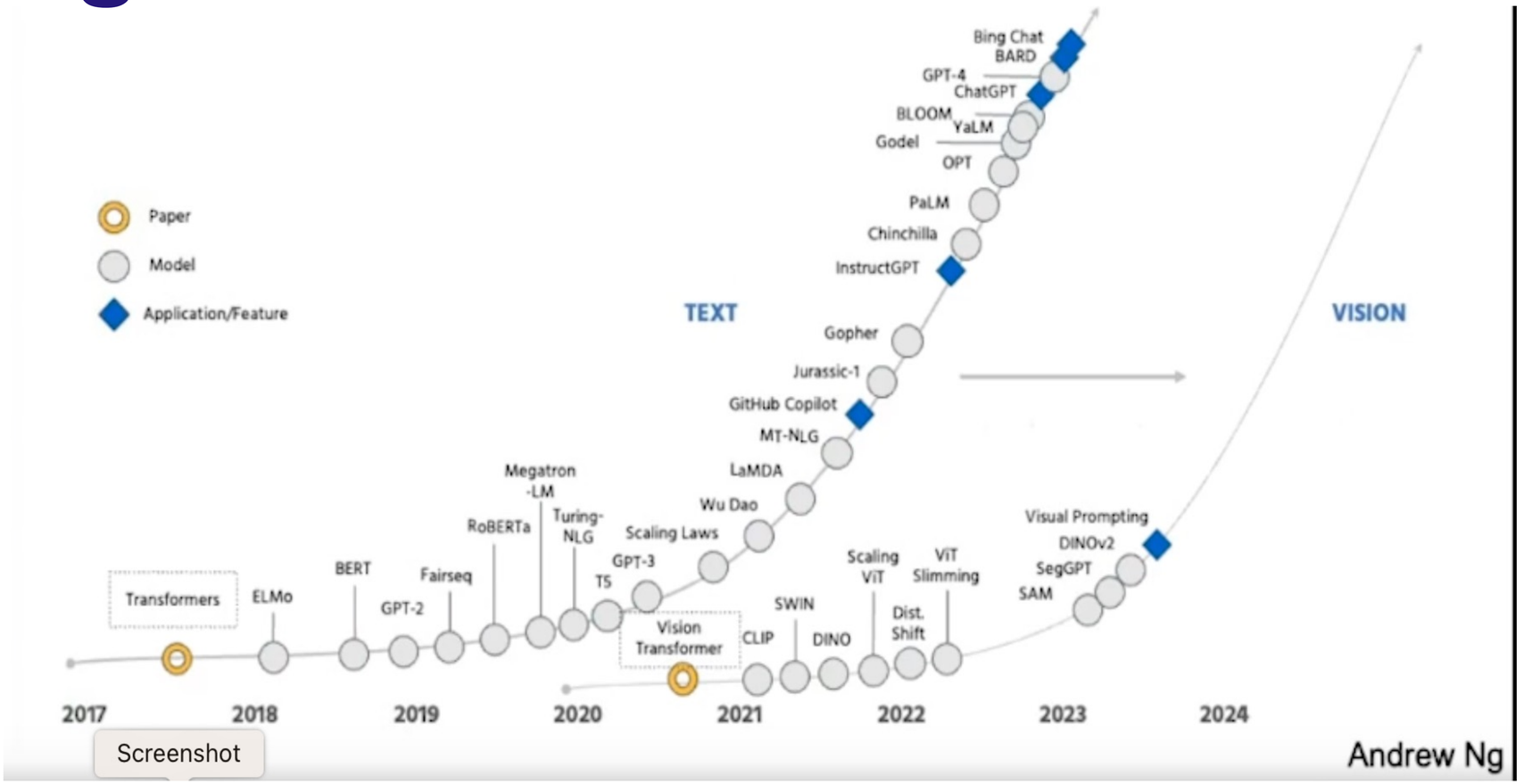
sense maker samst erdam

# What you can do with Vision models

Talk with them using LLaVA or any other Vision – Language Model!

1. Human input with speech-2-text
2. Visual Question Answering
3. Text-2-Speech

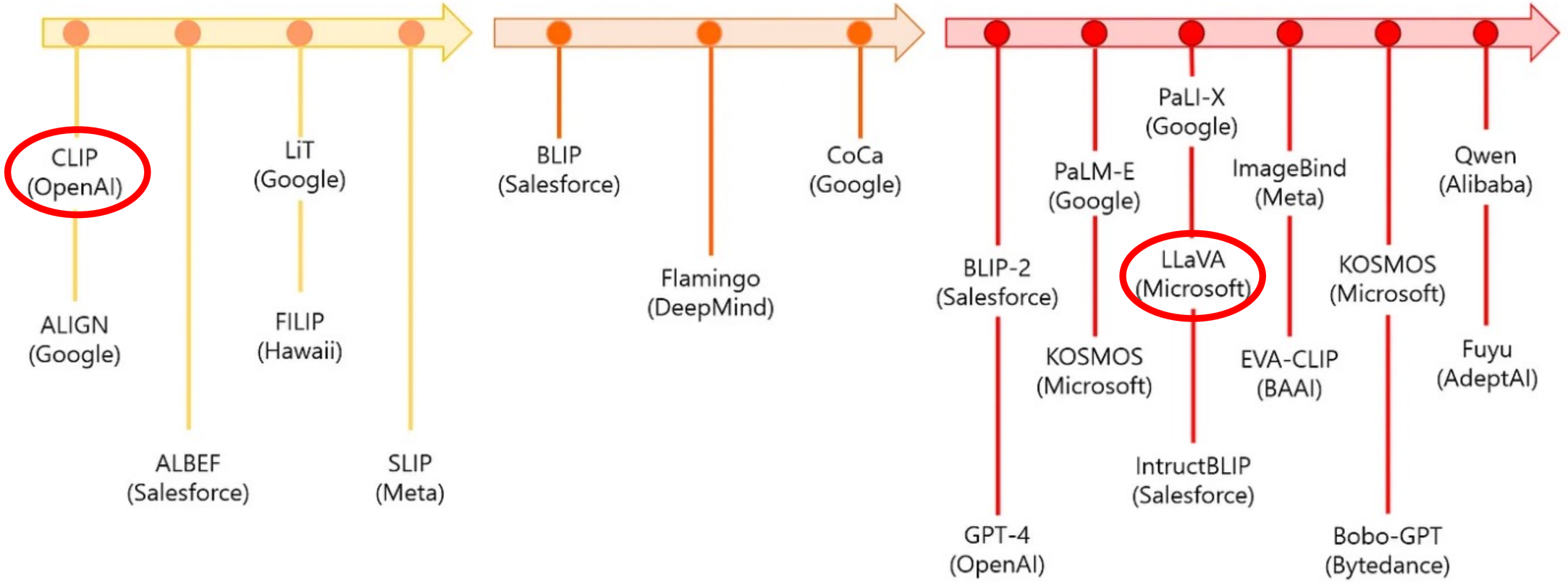
# The ChatGPT revolution is coming to vision!




Screenshot

Large Vision Models =. Large Multimodal Models

# Vision Language Models



# How can computers find this relationship?

‘Paris’

**‘vector embeddings’**

Read more: https://www.geeksforgeeks.org/what-are-embeddings-in-machine-learning/

# We use an ‘embedding model’ and compare the vectors. Vectors capture the meaning.

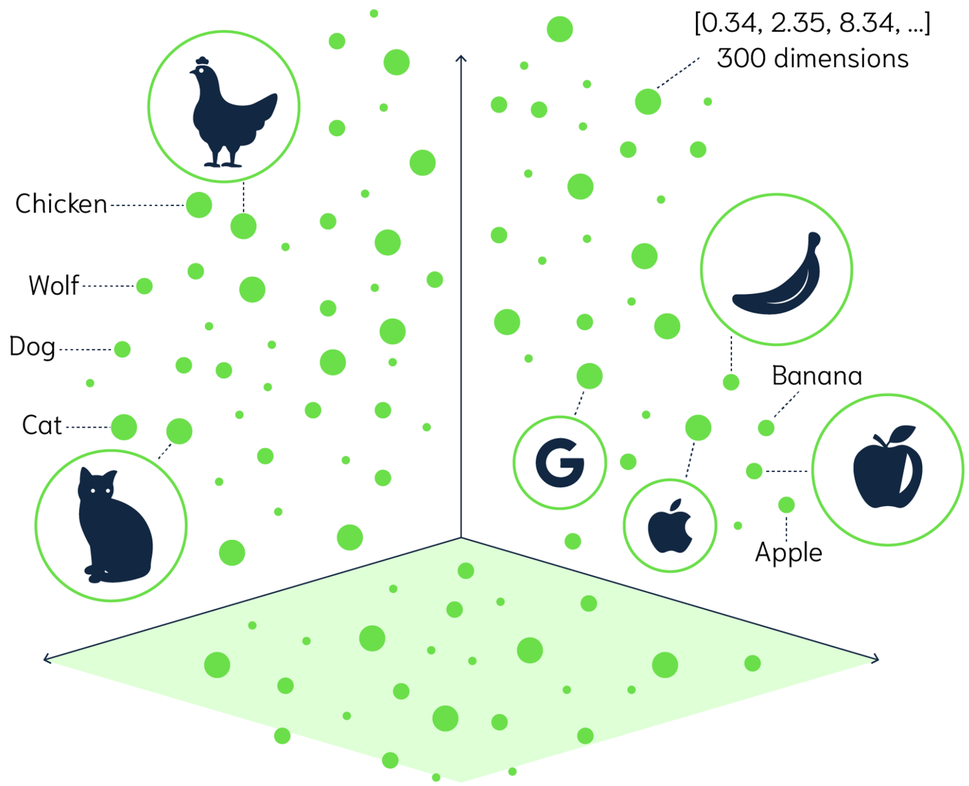
‘Paris’

[0.5, 0.3, …, 0.7]

[0.4, 0.3, …, 0.6]

Compare vectors

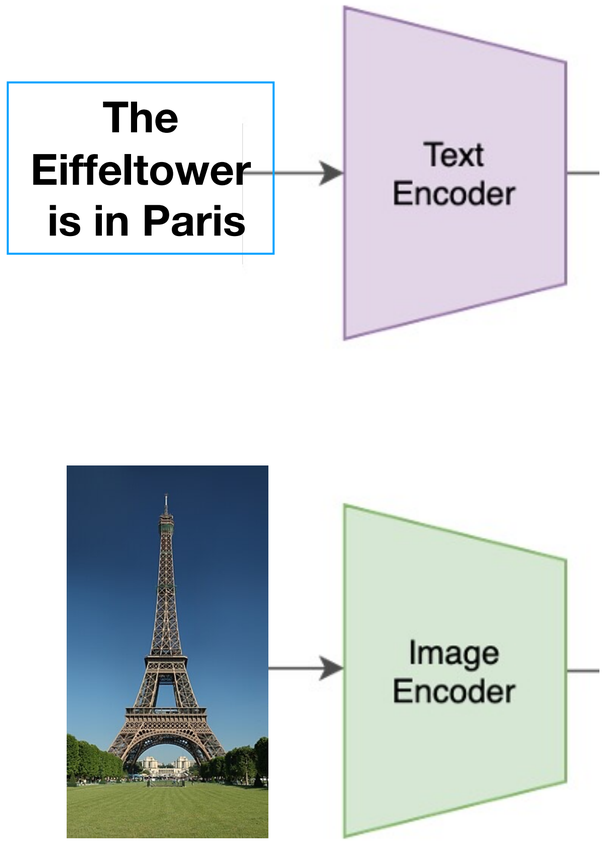
# Texts and images in vector space




Words and images with same meaning are close in vector space.

# We will use OpenAI's CLIP with text-image pairs

## Contrastive Language-Image Pre-training




- Pairs of captions with images
- Trained on 400 million pairs. Published in 2021.

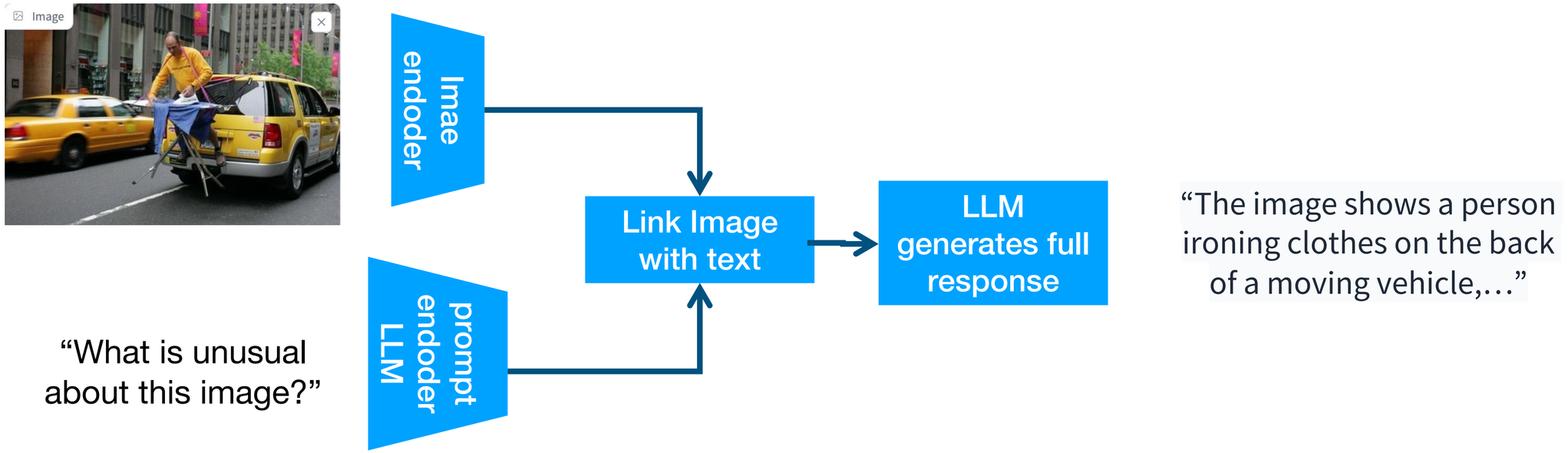
- Use CLIP to describe images => '**image2text**'
- Dall-e is the reverse => **text2image**

Sources:

- https://github.com/OpenAI/CLIP
- https://openai.com/research/clip

# LLaVA: Vision Language Model

## Combines CLIP with an LLM




This is also called a ‘neck & head architecture’.

# Vision challenges

## Describe screenshots

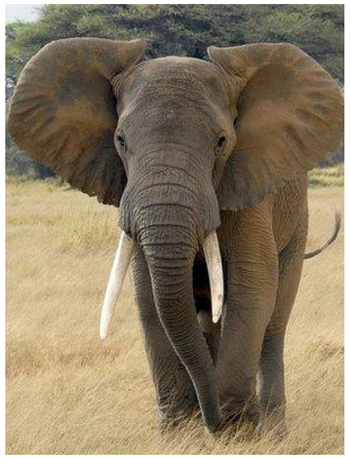
ollama_llava_challenges.ipynb

## Security webcam

## Self driving car


Self_driving_car.ipynb

## Find similar images (CLIP)



Find_similar_images_CLIP.ipynb

# Chat with your document

# Retrieval Augmented Generation

-

# Retrieval Augmented Generation

Generate an augmented (=improved) answer from an LLM based on information retrieved from your document.

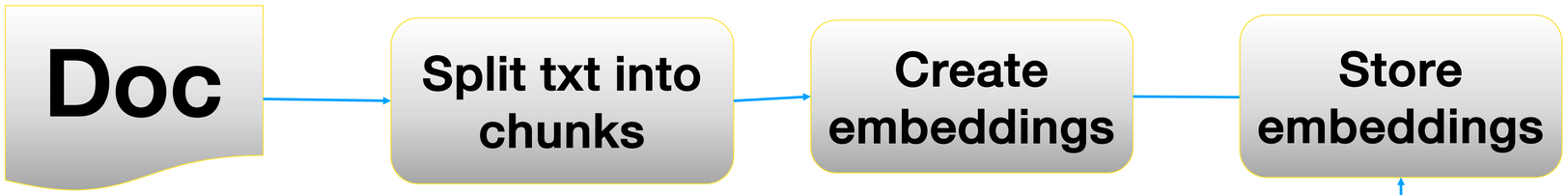
Used for tasks as  question – answering and summarizing

Most common python packages are Langchain and Llama_index – we will use simply numpy and torch

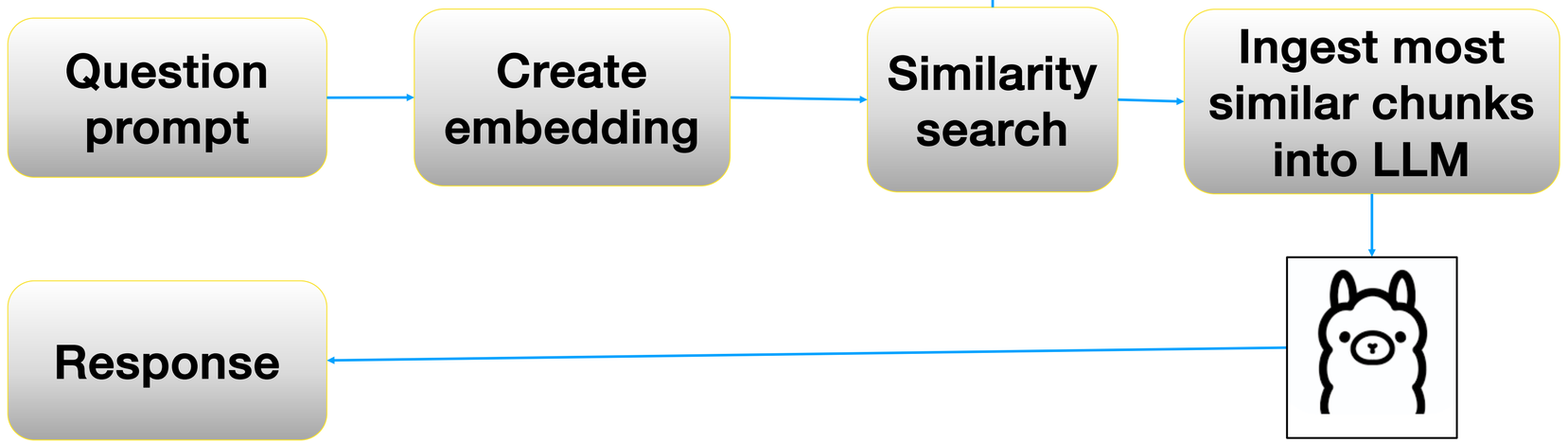
We will just cover the basics, there is much more to it!

# Flow

## 1. Preparation

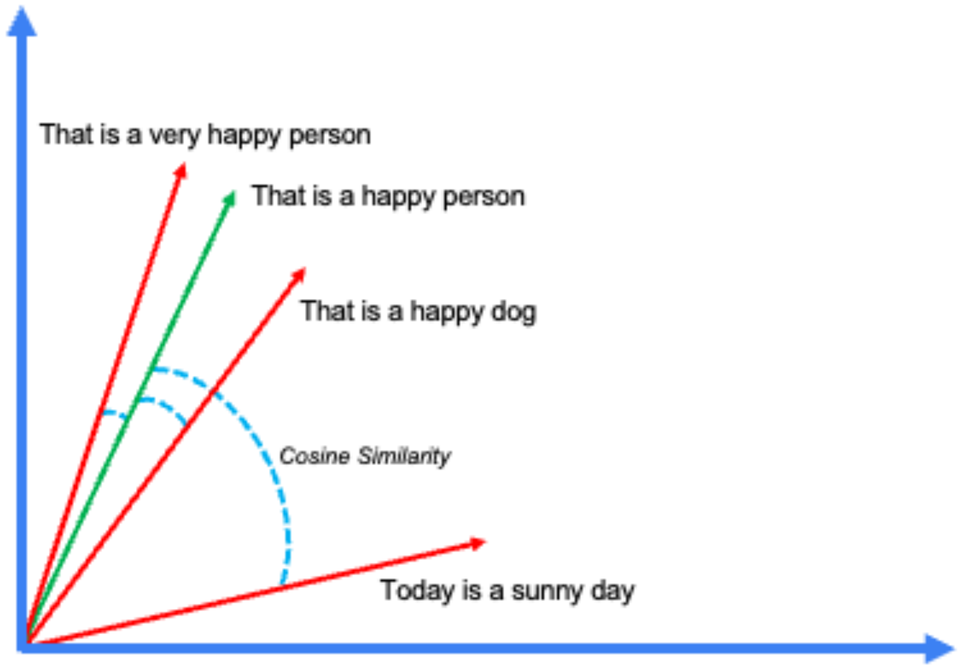



## 2. Inference



# Similarity search

- Similarity search is done by comparing vector embeddings.
- Most often we use ‘cosine similarity’.
- It gives meaning to search – not just string search!
- Try it at www.perplexity.ai



# RAG challenge + notebook

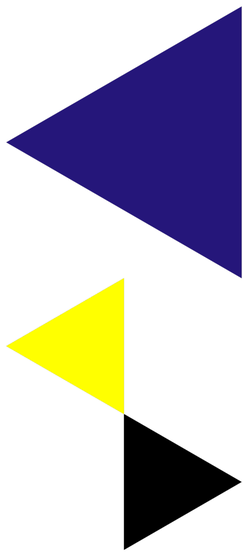
- Get your chatbot to talk with a document.

Use the notebooks to do RAG on Peter Pan book:

`RAG_basics_from_scratch_with_ollama.ipynb`

Bonus if you want to learn more about some theory behind RAG:

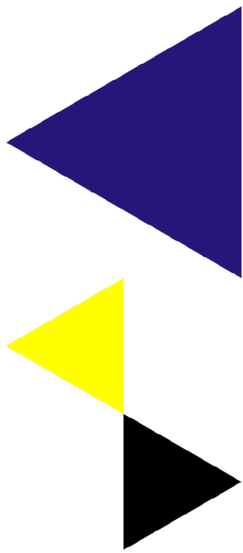
`RAG_exercises.ipynb`



# Learn more RAG

Short courses with deeplearning.ai

- https://www.deeplearning.ai/short-courses/building-multimodal-search-and-rag/
- https://learn.deeplearning.ai/courses/preprocessing-unstructured-data-for-llm-applications

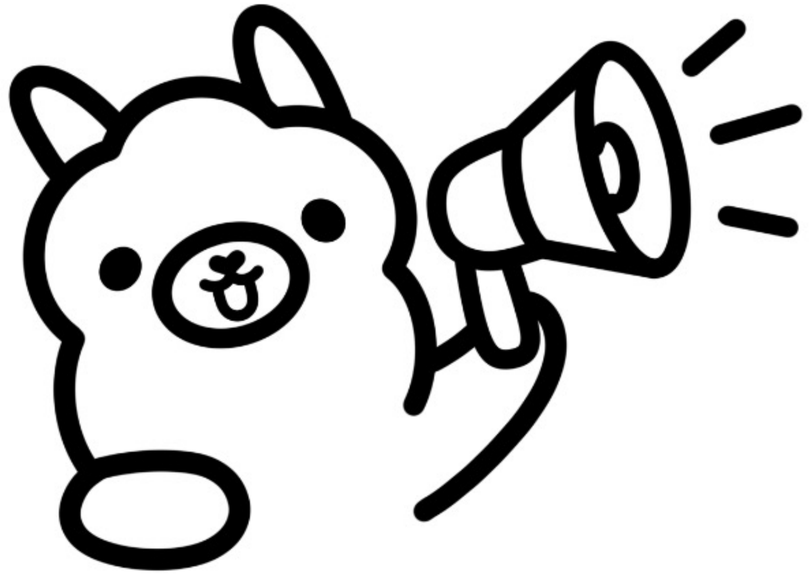


# To do list - Create a chatbot

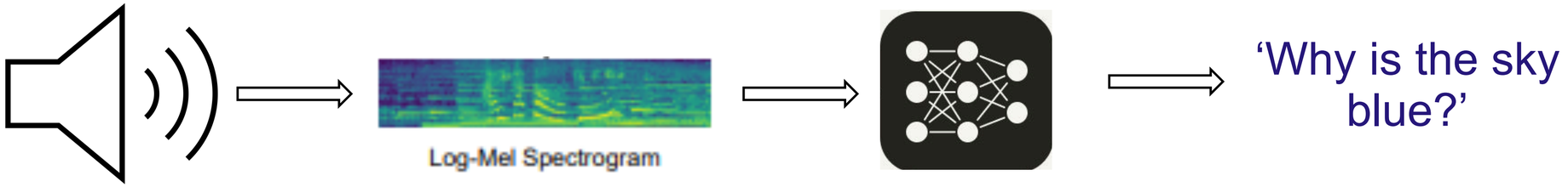
| **Our to do list** | |
|---|---|
| Run LLM’s locally | ✅ |
| Personalise model with modelfile | ✅ |
| Create a front-end with Gradio | ✅ |
| Pick any model from Huggingface (e.g. Dutch) | ✅ |
| Work with Vision | ✅ |
| Chat with your document | ✅ |
| Text to speech / speech to text | 19 June |
| Front-end with Gradio | 19 June |

# Talking with ollama

## Speech to Text
## &
## Text to Speech



# How Speech to Text works

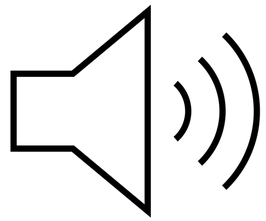



As our AI model we will use **Whisper** by OpenAI

It’s a Transformer (encoder-decoder) based model.

# Text to Speech - offline

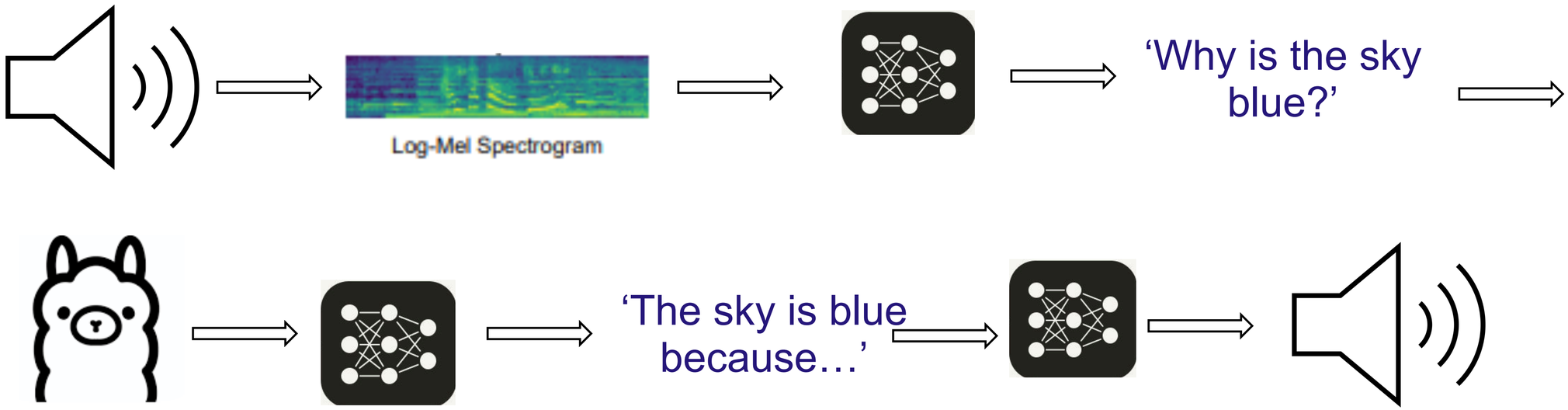
‘The sky is blue because…’ ⟹ **Text to Speech = hardware specific**

**MacOS – Windows - Linux** ⟹ 

**We will use a python library called Pyttsx3 to program it.**

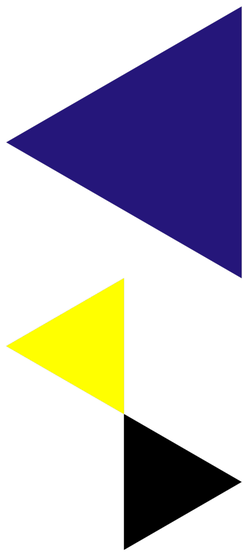
**There are thus differences between Mac / Windows / Linux**

# Putting it all together!



# Install audio packages

- Please download and install FFMPEG from https://ffmpeg.org/download.html



# Learn more

- https://realpython.com/python-speech-recognition/
- whisper

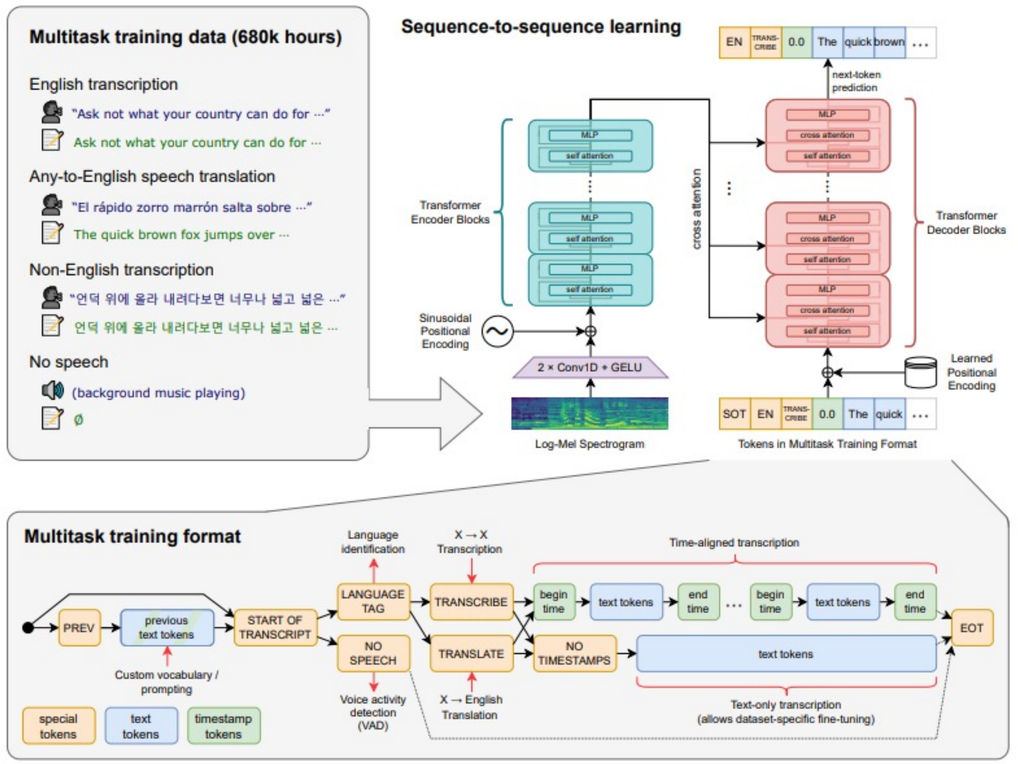



*Figure 1.* **Overview of our approach.** A sequence-to-sequence Transformer model is trained on many different speech processing tasks, including multilingual speech recognition, speech translation, spoken language identification, and voice activity detection. All of these tasks are jointly represented as a sequence of tokens to be predicted by the decoder, allowing for a single model to replace many different stages of a traditional speech processing pipeline. The multitask training format uses a set of special tokens that serve as task specifiers or classification targets, as further explained in Section 2.3.

# Future developments for Local LLM

# Microsoft’s Phi can run in your browser

- Microsoft’s Phi is a family of Small Language Models
- Developed for browsers

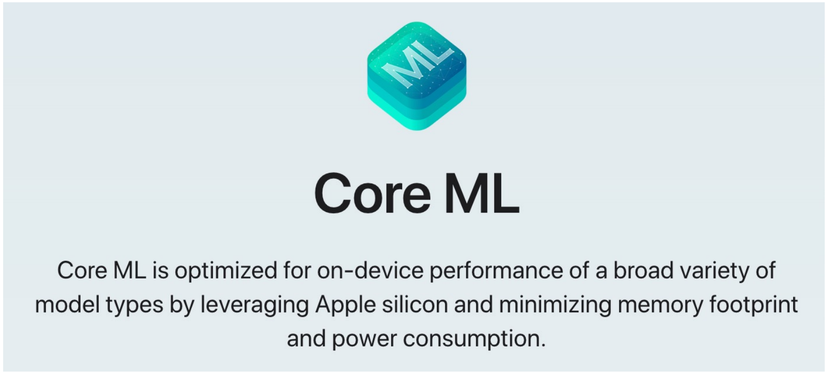
Try it yourself:

- https://huggingface.co/spaces/Xenova/experimental-phi3-webgpu

# Apple is very active with local LLM’s

At the recent Apple Developer Conference (WWDC24), Apple have:

- released CoreML, optimized for running models on Apple hardware
- released OpenELM  - Efficient Language Model
- published their models on Huggingface:
  https://huggingface.co/apple



sense maker samst erdam

# Future of Local LLM's: ollama alternatives are being developed

- GPT4all
- LM-studio
- LocalGPT (also does RAG natively)
- Jan.ai (my favourite)

# Bonus slides

# Some RAG evaluation metrics in RAGAS

**Faithfulness** refers to the idea that the answer should be grounded in the given context.

**Answer Relevance** refers to the idea that the generated answer should address the actual question that was provided.

**Context Relevance** refers to the idea that the retrieved context should be focused, containing as little irrelevant information as possible

$$\text{context relevancy} = \frac{|S|}{|\text{Total number of sentences in retrieved context}|}$$

Hint

Question: What is the capital of France?

High context relevancy: France, in Western Europe, encompasses medieval cities, alpine villages and Mediterranean beaches. Paris, its capital, is famed for its fashion houses, classical art museums including the Louvre and monuments like the Eiffel Tower.

Low context relevancy: France, in Western Europe, encompasses medieval cities, alpine villages and Mediterranean beaches. Paris, its capital, is famed for its fashion houses, classical art museums including the Louvre and monuments like the Eiffel Tower. The country is also renowned for its wines and sophisticated cuisine. Lascaux's ancient cave drawings, Lyon's Roman theater and the vast Palace of Versailles attest to its rich history.

**Source**: https://arxiv.org/pdf/2309.15217