# From Generative to Interactive AI: Towards Artificial General Intelligence?

# Use on Local Data and Applications Examples

Imed MAGROUNE

Université Paris Saclay / CEA

17/10/2023

1. **How Can a machine think ?**
   - Turing Test and Artificial Intelligence
   - A brief history of AI up to Transformers
   - Generative AI
2. **FLMs, LLMs and AI Agents : State Of The Art**
   - ChatGPT how did they make it ?
   - Prompt Engineering
   - AI agents
3. **Interactive AI and potentiel use cases**
   - LLM Powered Autonumous Agents
   - Applications improve
   - To AGI ?
4. **Use case on local Data and Applications**
   - Code developement and debugging
   - Chat with your Data

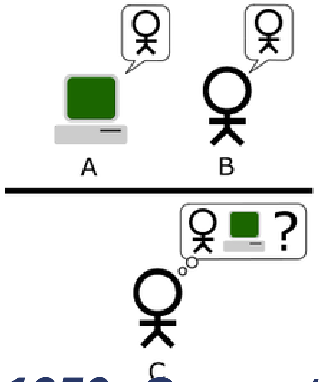
# Richard Feynman Prophecy 1985

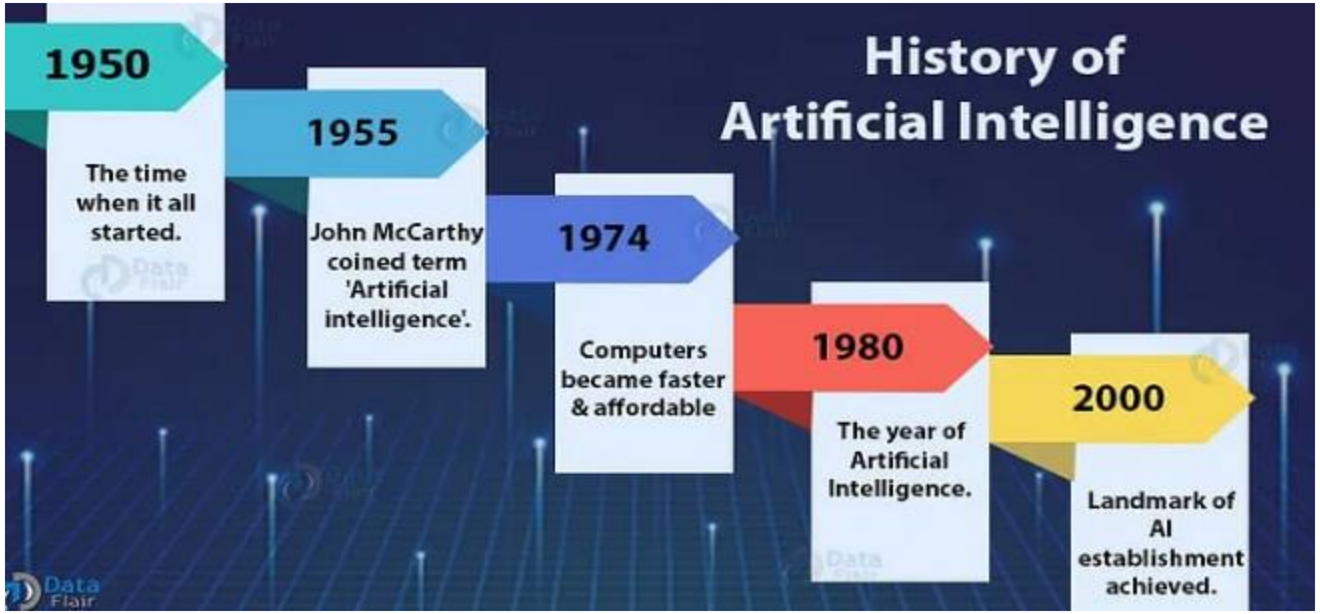
Audience Question:

(Re-recorded by me because the question is poorly audible)

Do you think there will ever be
a machine that will think like human beings
and be more intelligent than human beings?


Richard Feynman: Can Machines Think?

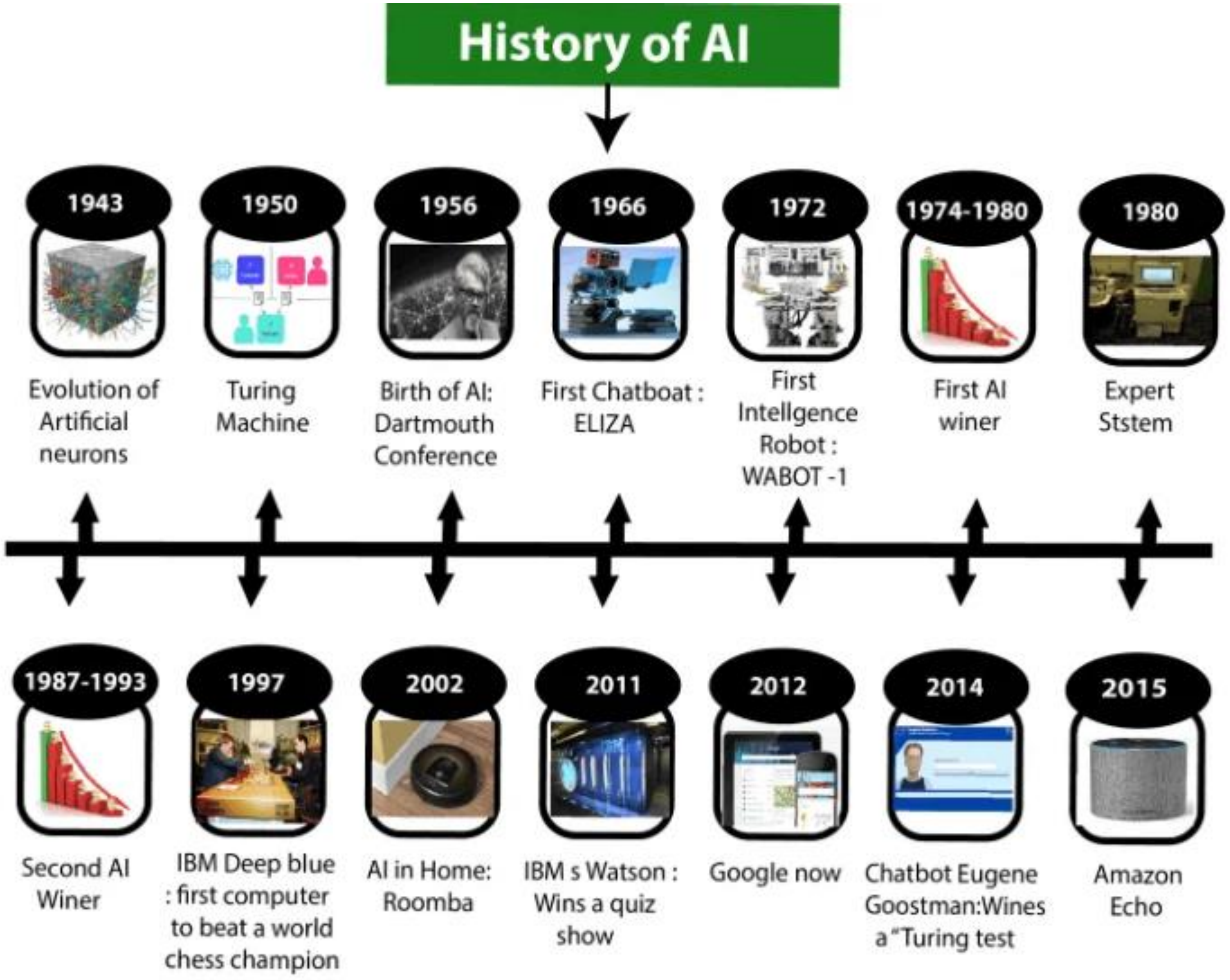



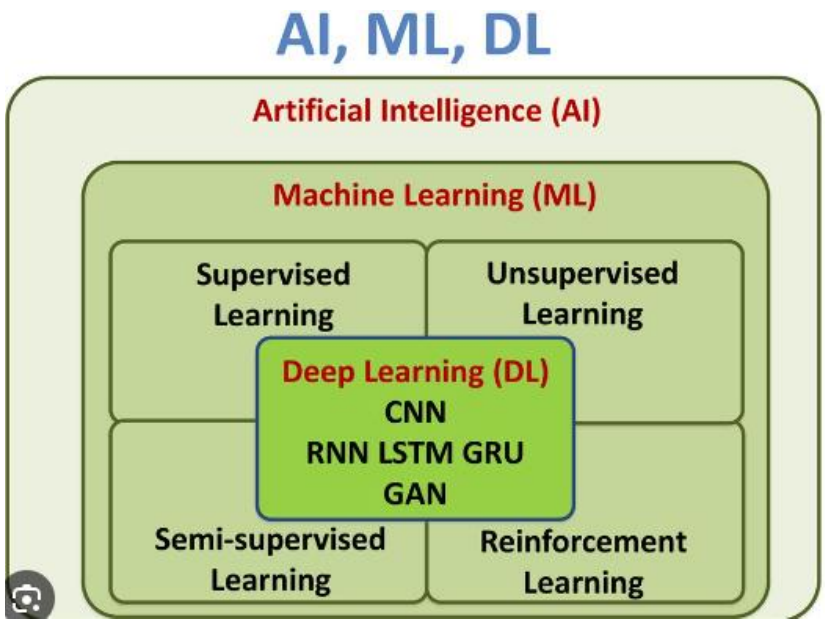




*1950: Computing Machinery and Intelligence*

https://www.youtube.com/watch?v=ipRvjS7q1DI&t=15s

# Brief History of AI (Before the Big Bang)

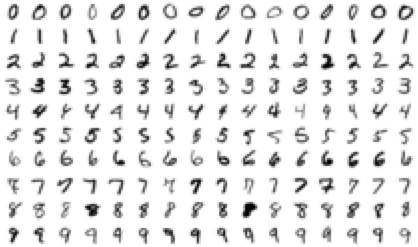


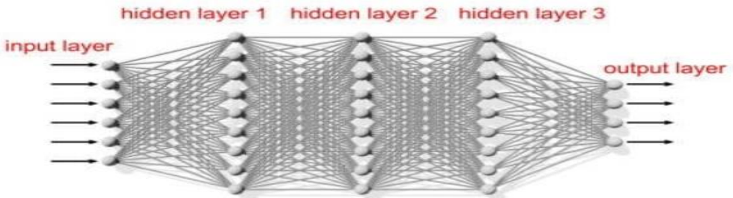

This image is property of static.javatpoint.com.

# 1957-2012 : Computer Vision (CV) / Deep Learning

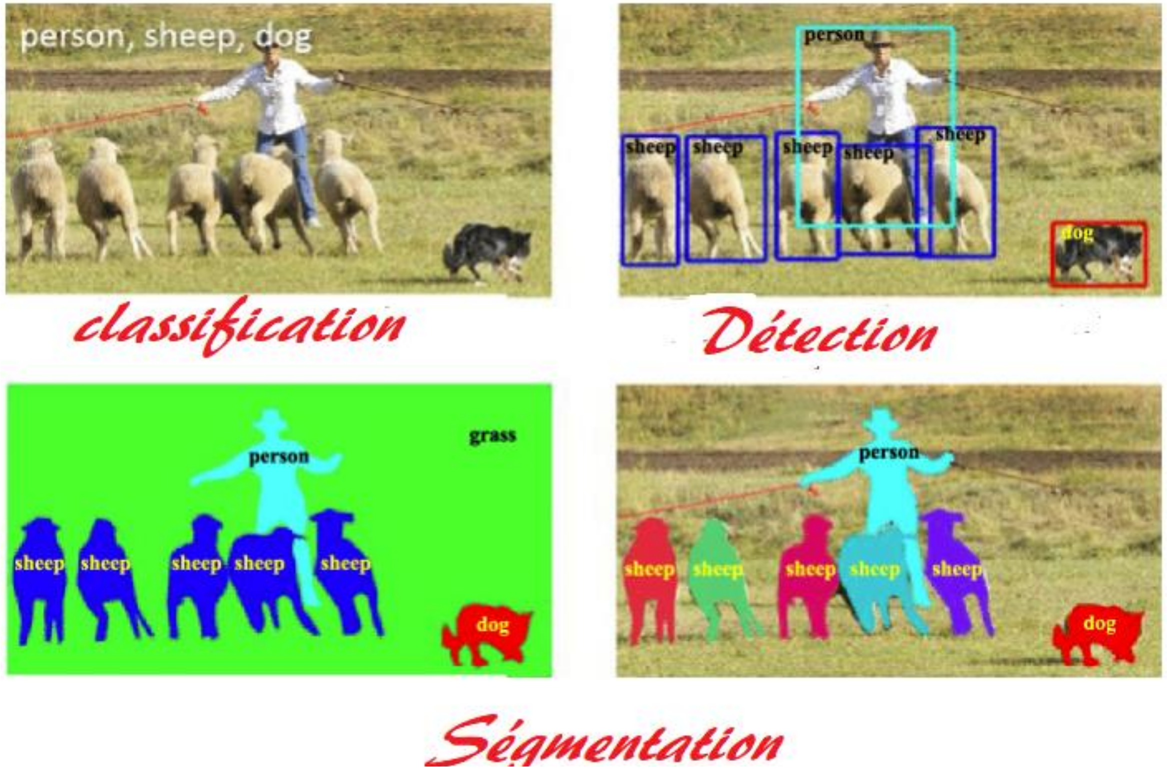
- MNIST Dataset

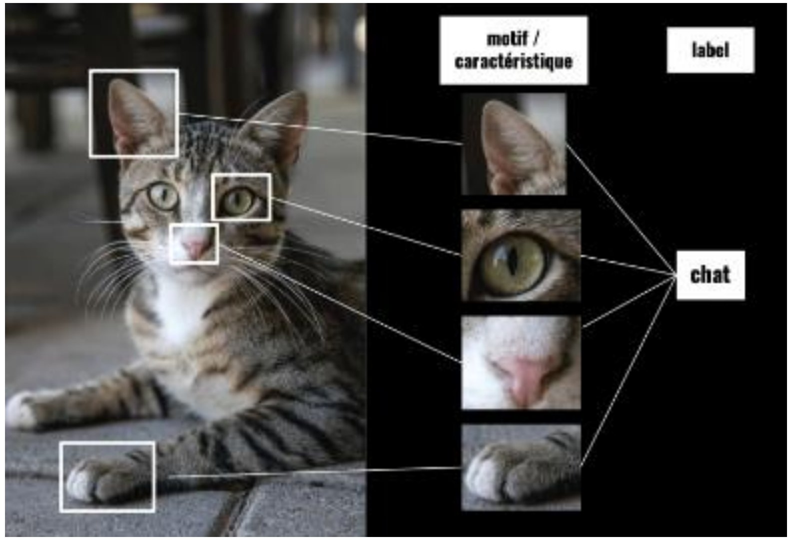


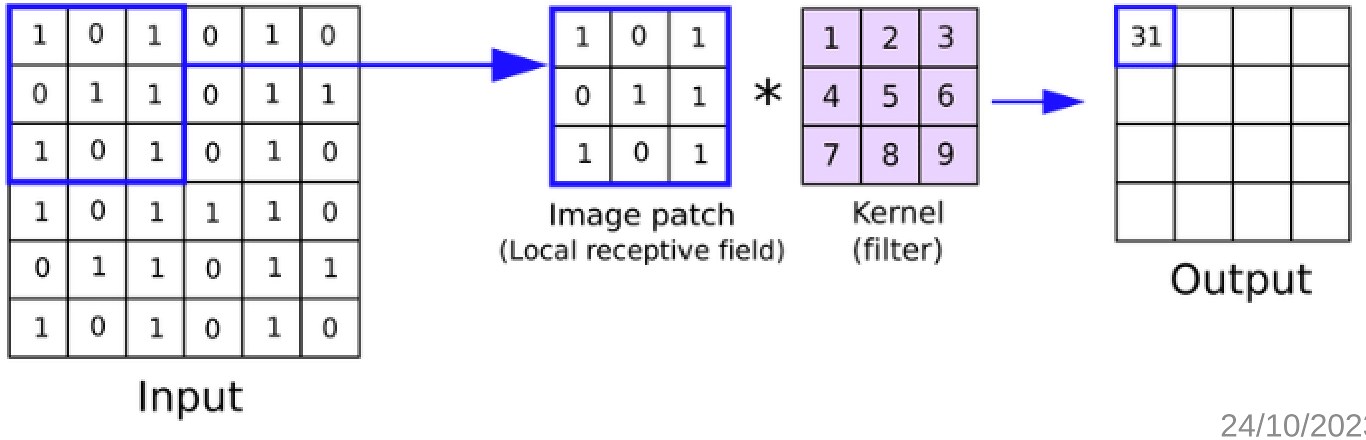
- 28x28 ➔ 10 classes
- MLP




- CNN : Convolutional Neural Network
  - 1989 Yan LeCun LeNet-5 ➔ Deep Learning



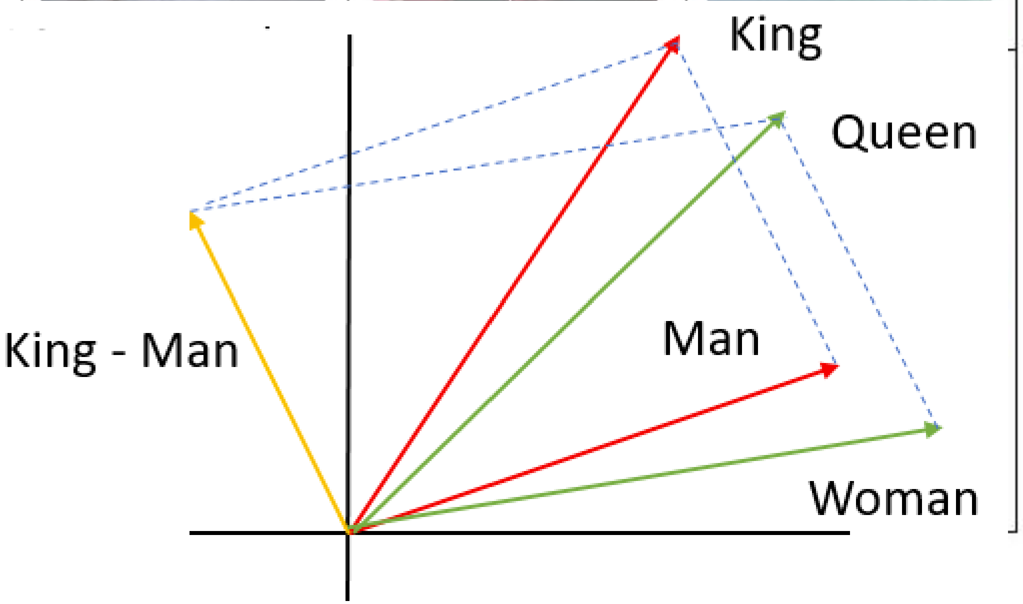

- **2012** : AlexNet wins ImageNet

# CV : Image Captioning

- Classification : P(class) 0. <=float<= 1. [c1,c2, ..cn] (n classes)
- Captioning : Describe in natural language what the image contains


# NLP: Natural Language Processing

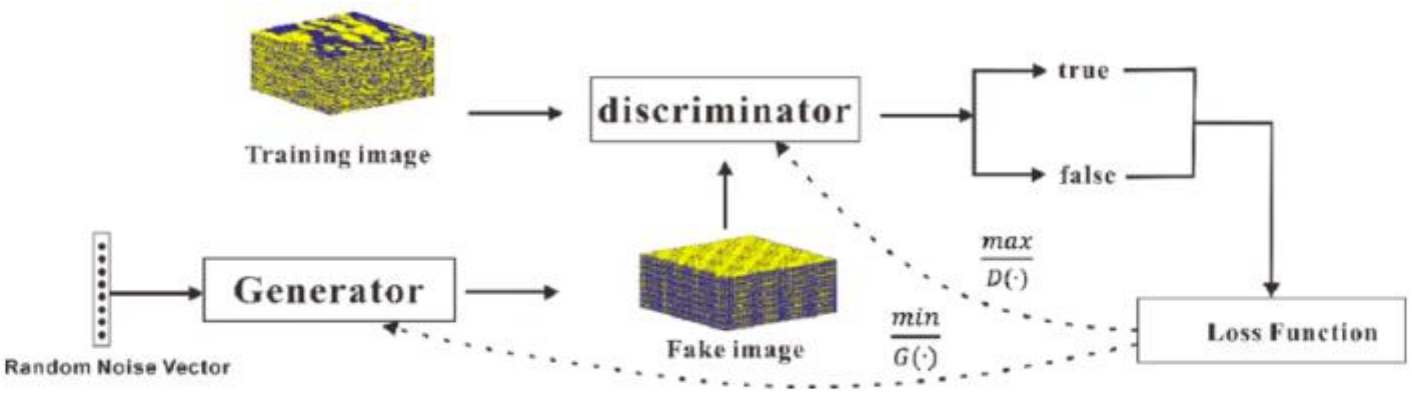
- Beginning with rule-based approaches (1950s).
- Evolution towards statistical approaches (1980s).
- Then towards deep learning and neural networks (2010s).
- MLP ➔ CNN ➔ RNN ➔ LSTM ➔ …
- Capable of generating natural text, translating languages, answering questions, and much more.Examples:
- GPT (OpenAI), BERT (Google), RoBERTa (Facebook).




**Tokenization**: First step in NLP. Breaks the text into smaller units (words, sentences, etc.), called “tokens”
.
**Embedding**: Converts tokens to digital vectors. These representations capture the meaning of words, their context and other linguistic aspects.

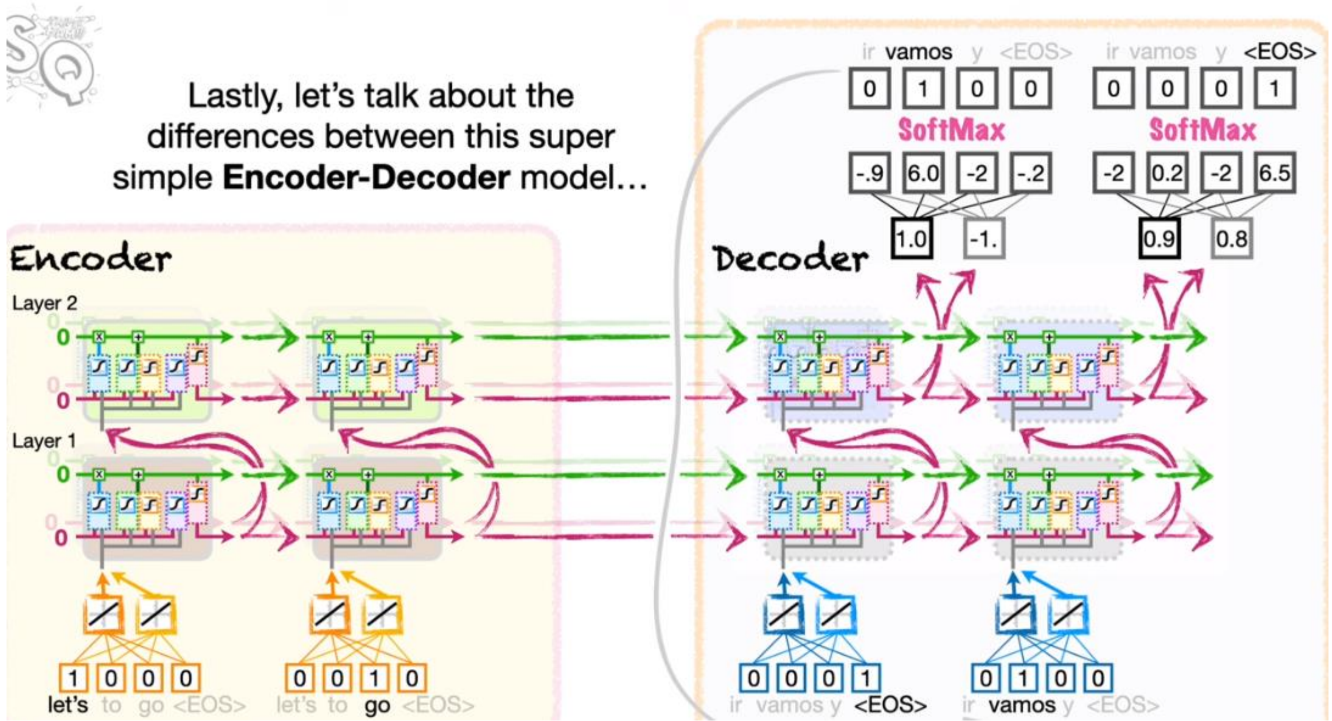
# CV : GAN

- Classification : Image ➔ Cat
- Generative Model : Cat ➔ Image of cat
- 2 AI : Generator VS Descriminator



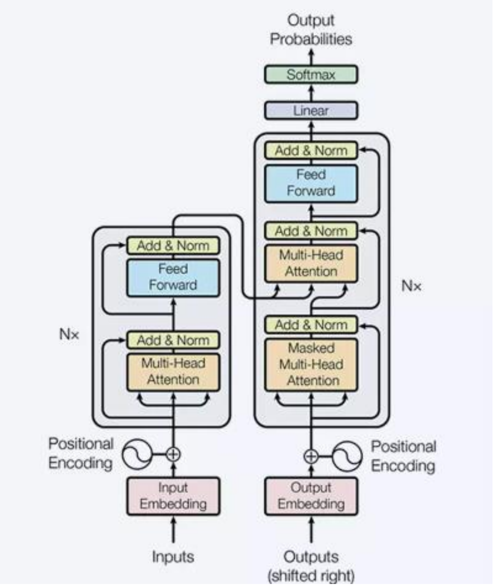

# NLP: Text Generation

- Dataset: Text (Alice in Wonderland)
- Tokenization: Token = Char
- Model (LSTM):
  - Input: a suite of chars, output a char
  - (alic➔ e)
- Generation :
  - Initial char
  - Loop
    - Next char Prediction
    - Concatenate to end
    - Submit to model.




# Transformers

**(*genetic mutation in the chain of AI evolution)**




➢ 2017 : Google Brain



## Attention Is All You Need


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

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

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

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

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

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

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

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



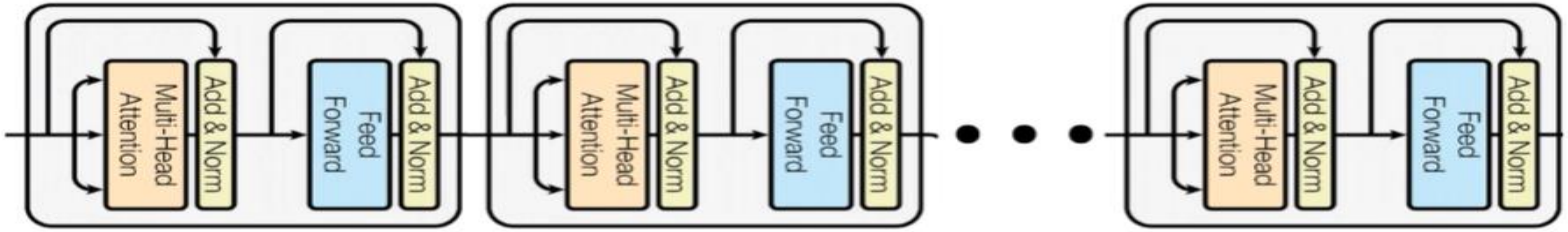
arXiv:1706.03762v6 [cs.CL] 24 Jul 2023



### Abstract

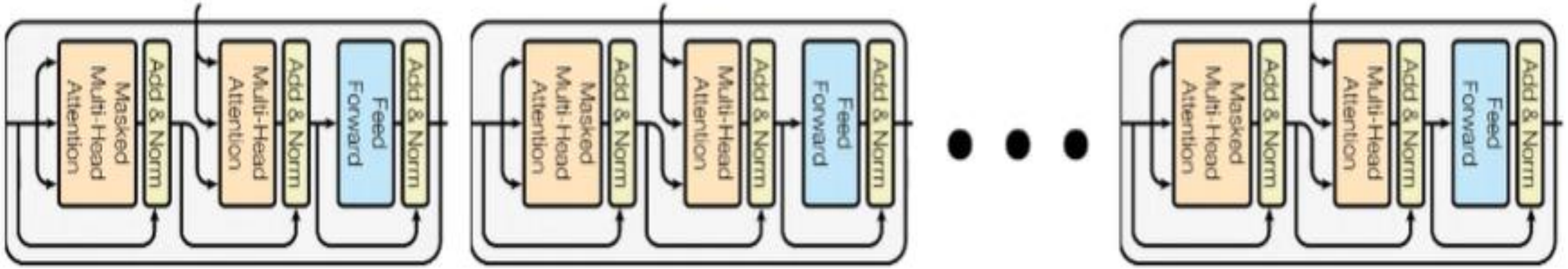
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

# Transformer Flow




BERT

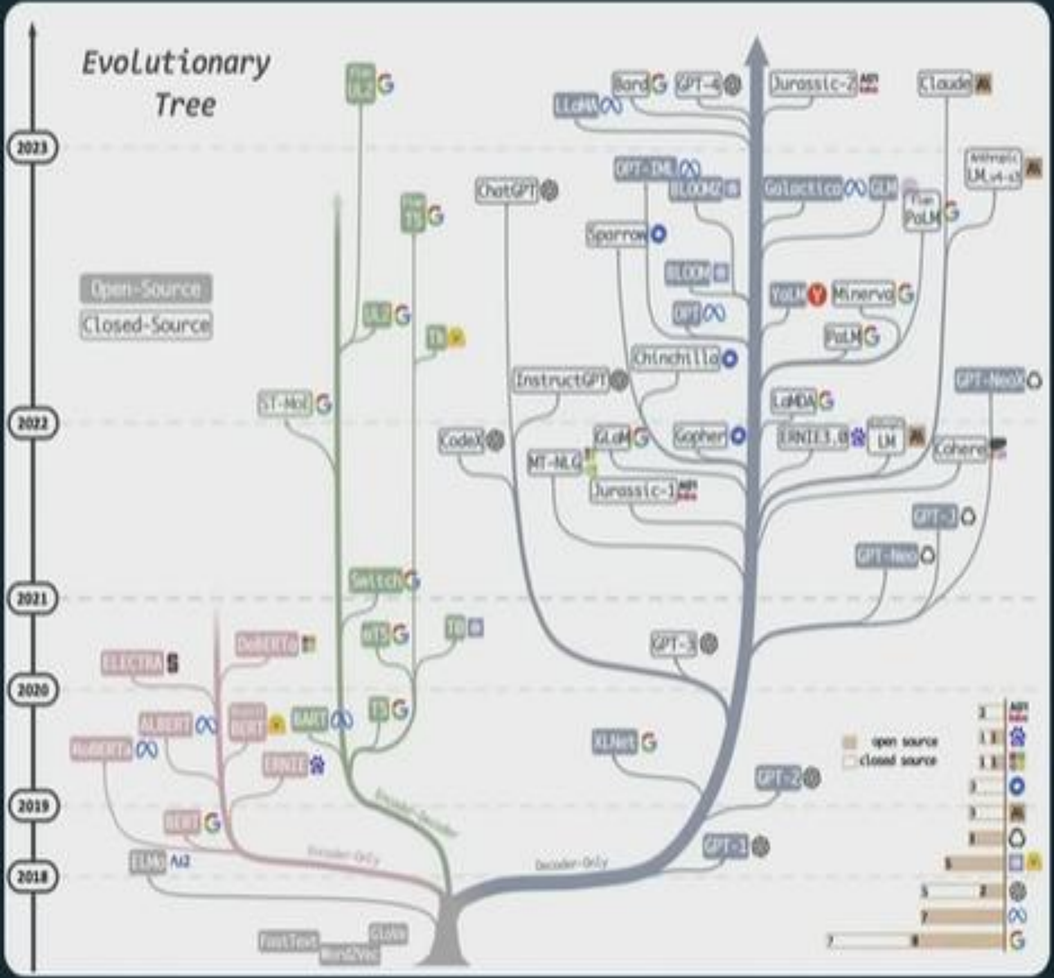
**B**idirectional **E**ncoder **R**epresentation from **T**ransformers




GPT

# Base models in the wild

- GPT **Improving Language Understanding by Generative Pre-Training**. 2018. Paper
- GPT-2 **Language Models are Unsupervised Multitask Learners**. 2018. Paper
- GPT-3 **"Language Models are Few-Shot Learners"**. NeurIPS 2020. Paper
- OPT **"OPT: Open Pre-trained Transformer Language Models"**. 2022. Paper
- PaLM **"PaLM: Scaling Language Modeling with Pathways"**. *Aakanksha Chowdhery et al.* arXiv 2022. Paper
- BLOOM **"BLOOM: A 176B-Parameter Open-Access Multilingual Language Model"**. 2022. Paper
- MT-NLG **"Using DeepSpeed and Megatron to Train Megatron-Turing NLG 530B, A Large-Scale Generative Language Model"**. 2021. Paper
- GLaM **"GLaM: Efficient Scaling of Language Models with Mixture-of-Experts"**. ICML 2022. Paper
- Gopher **"Scaling Language Models: Methods, Analysis & Insights from Training Gopher"**. 2021. Paper
- chinchilla **"Training Compute-Optimal Large Language Models"**. 2022. Paper
- LaMDA **"LaMDA: Language Models for Dialog Applications"**. 2021. Paper
- LLaMA **"LLaMA: Open and Efficient Foundation Language Models"**. 2023. Paper
- GPT-4 **"GPT-4 Technical Report"**. 2023. Paper
- BloombergGPT **BloombergGPT: A Large Language Model for Finance**, 2023, Paper
- GPT-NeoX-20B: **"GPT-NeoX-20B: An Open-Source Autoregressive Language Model"**. 2022. Paper



# ChatGPT / Bard

I

Introduce yourself to the participants of the HEPiX fall 2023 seminar

Certainly! Here's how I would introduce myself for your screen capture:

Hello HEPiX Fall 2023 Seminar pa

I am ChatGPT, a machine learning
attend seminars or experience th
Whether you need help with data
I'm here to help. I look forward to a

I

Introduce yourself to the participants of the HEPiX fall 2023 seminar

Afficher d'autres suggestions

Hello everyone, I am Bard, a large language model from Google AI. I am excited to be here at the HEPiX Fall 2023 Seminar to meet you all and learn more about the latest developments in high-energy physics computing.

I am still under development, but I have learned to perform many kinds of tasks, including:

- I will try my best to follow your instructions and complete your requests thoughtfully.
- I will use my knowledge to answer your questions in a comprehensive and informative way, even if they are open ended, challenging, or strange.
- I will generate different creative text formats of text content, like poems, code, scripts, musical pieces, email, letters, etc. I will try my best to fulfill all your requirements.

I am here to help you in any way that I can. Whether you need help with your research, you want to learn more about a particular topic, or you just want to chat about the latest news in the field, I am here for you.

I am looking forward to meeting you all and learning more about high-energy physics computing.

Saisissez une requête ici

Bard peut afficher des informations inexactes ou choquantes qui ne représentent pas l'opinion de Google. Avis de confidentialité Bard

# How did they make it ?

## Step 0: Pretraining

**Next-token-prediction**

The model is given a sequence of words with the goal of predicting the next word.

Example:
Hannah is a ____

Hannah is a *sister*
Hannah is a *friend*
Hannah is a *marketer*
Hannah is a *comedian*

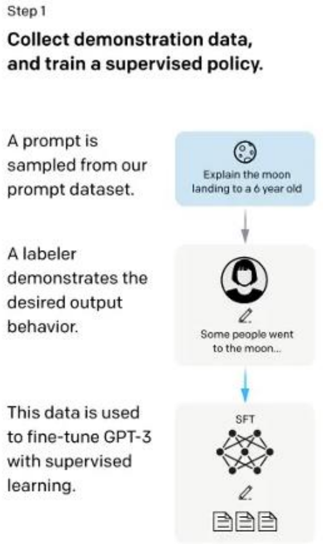
**Masked-language-modeling**

The model is given a sequence of words with the goal of predicting a 'masked' word in the middle.

Example
Jacob [mask] reading

Jacob *fears* reading
Jacob *loves* reading
Jacob *enjoys* reading
Jacob *hates* reading

## Step 1: Supervised Fine Tuning (SFT) Model

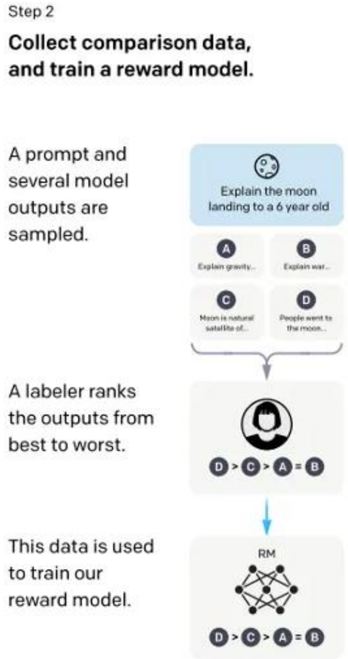



Prompt dataset is a series of prompts previously submitted to the Open API

40 contractors hired to write responses to prompts

Input / output pairs are used to train a supervised model on appropriate responses to instructions.

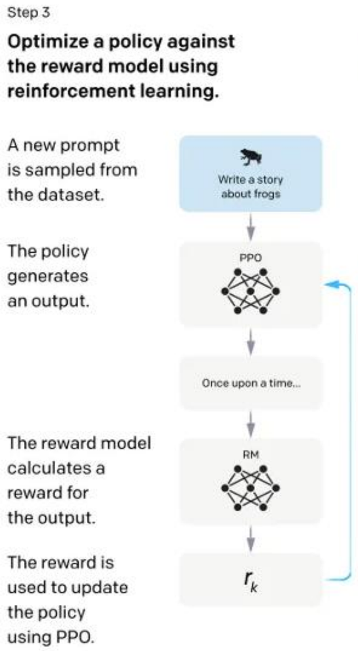
## Step 2: Reward Model




Responses are generated by the SFT model

$\binom{k}{2}$ combinations of rankings served to the model as a batch datapoint

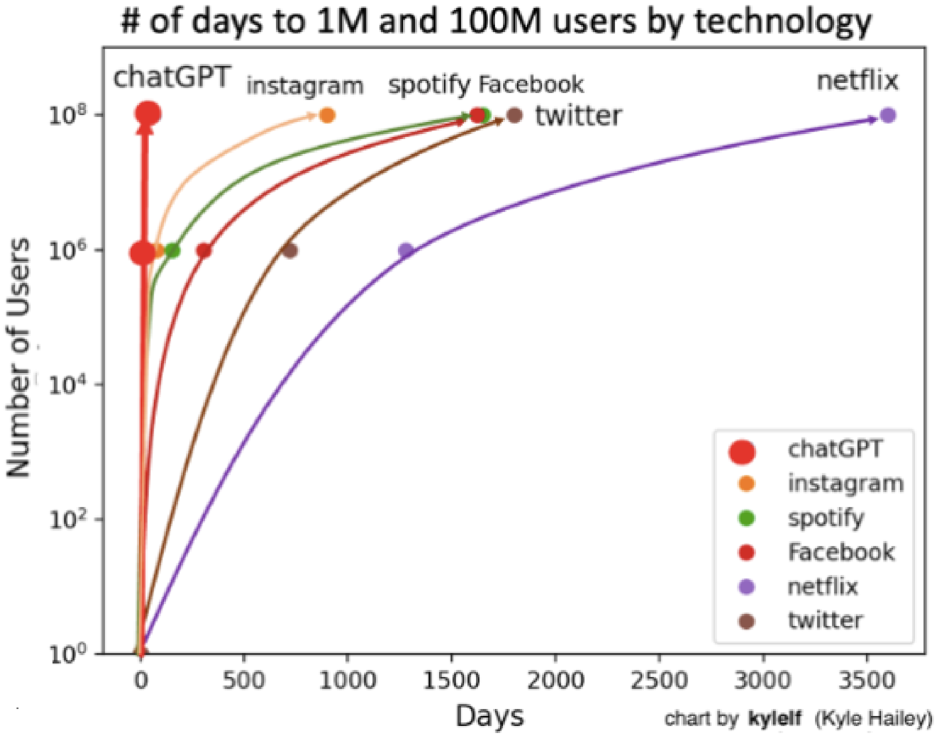
## Step 3: Reinforcement Learning Model



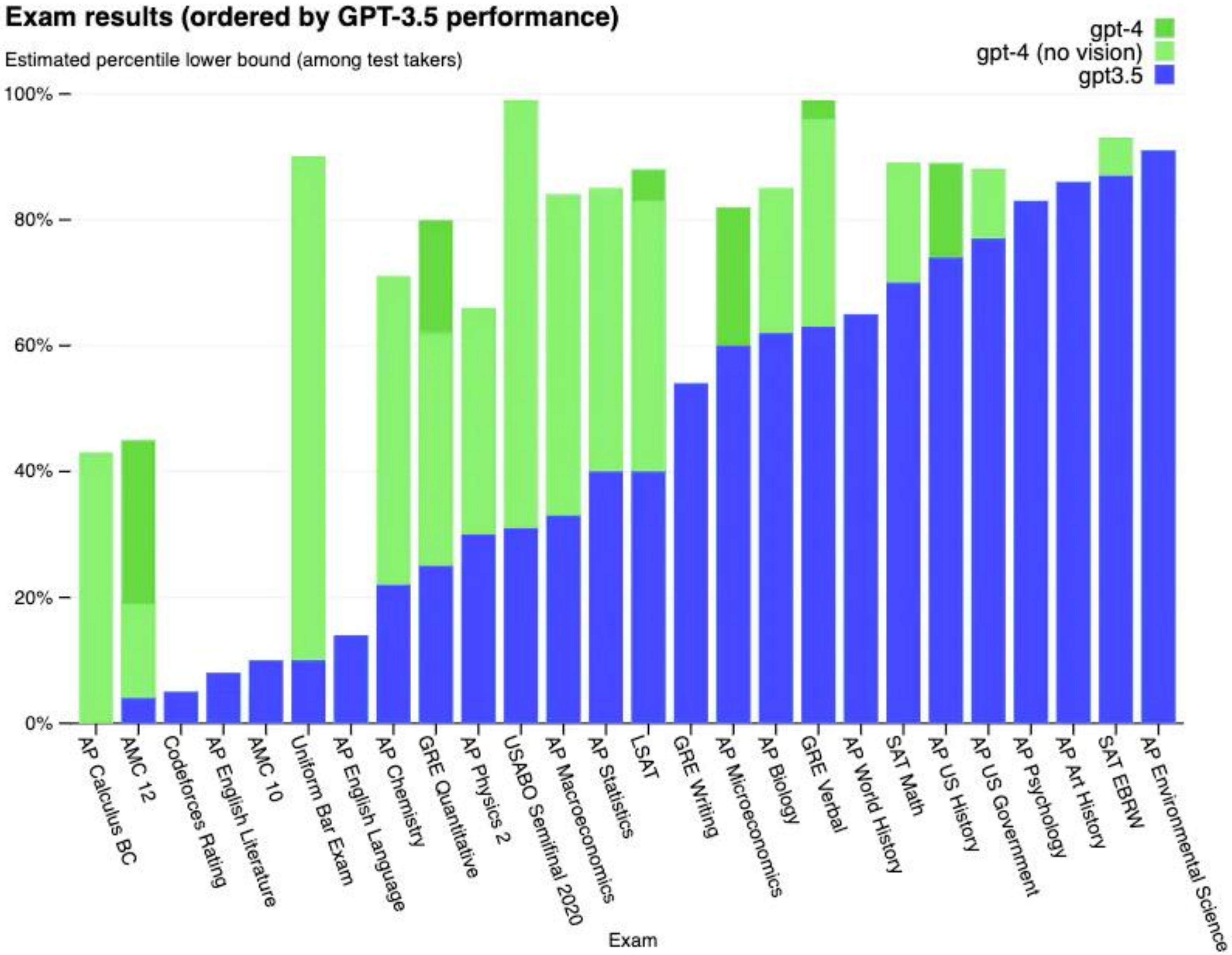

Leverages Proximal Policy Optimization (PPO)

A policy is, a strategy that an agent uses in pursuit of goals

Kullback–Leibler penaltiy for SFT model to avoid overfitting



# Performances



# Why is ChatGPT so good?


These models started to move into the spotlight with the release of Google's seminal "**Attention Is All You Need**" paper (https://arxiv.org/pdf/1706.03762.pdf)

**GPT-3**, released in 2020, is a whopping 175B parameter model pre-trained on a corpus of more than 300B tokens : it seems that is was not particularly adept at many of the tasks, reasoning, and instruction-following that the later versions, including ChatGPT due to the **lack of fine-tuning**

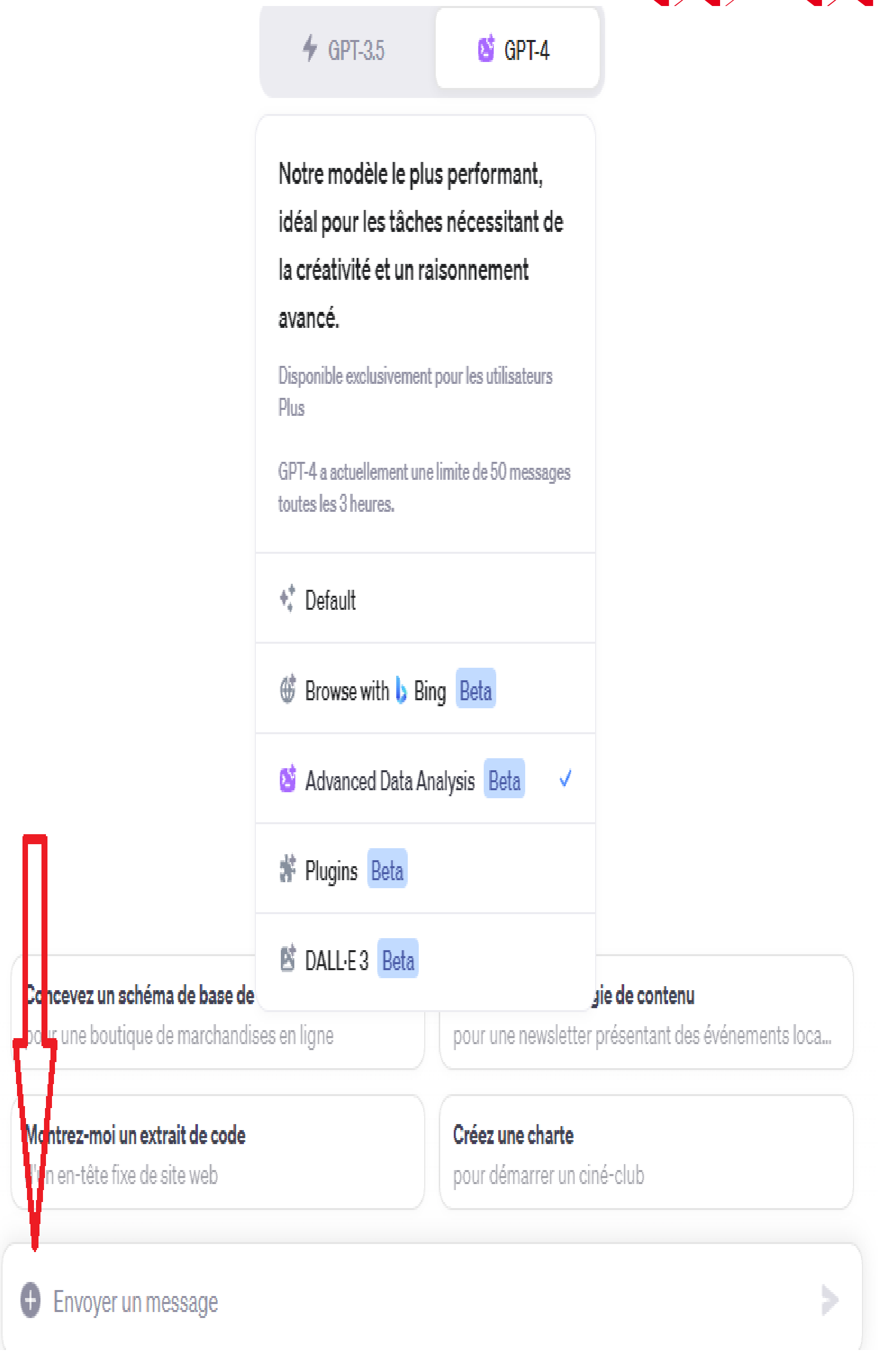
**Fine-tuning** on instruction datasets

Introducing Reinforcement learning with Human Feedback (**RLHF**) : the instruction-tuned models still regularly provide short, biased (even toxic) or wrong answers.

The basic idea is to train an additional reward model that rates how good a model's response is from the perspective of a human to guide the model's learning process. Then use this reward model to fine-tune the original language model using **reinforcement learning**

# How to use ChatGPT

- **Define the objective**: Establish a clear purpose for the prompt. For instance, when summarizing a news article, the aim is to obtain a brief, informative overview.
- **Construct an initial prompt**: Create a simple, concise prompt to begin the process. For example: "Summarize the following news article:". This can be refined and expanded as needed.
- **Evaluate and iterate**: Assess the output generated by ChatGPT in response to the initial prompt. Modify the prompt as necessary to improve the outcome. For instance: "Provide a concise, 3-sentence summary of the following news article:". Iterate until the desired result is achieved.
- **Employ control mechanisms**: Experiment with various control techniques, such as tokens, prefixes, or postfixes, to guide the AI's response. For example, prepend a sentence like "Using a professional tone," to influence the tone of the generated text.
- **Leverage automation in prompt design**: Explore tools and methodologies for automating prompt design, such as employing machine learning algorithms to generate or optimize prompts based on specific datasets.



# Code Generation / Vision

I need a python class to use in my pytrch code as a dataset generator

Certainly! Creating a custom dataset class in PyTorch is straightforward. You'll want to subclass the `torch.utils.data.Dataset` class and implement the `__len__` and `__getitem__` methods. Below is an example of a simple dataset generator class for a hypothetical image dataset:

python Copy code

```python
import torch
from torch.utils.data import Dataset
import numpy as np
from PIL import Image

class MyCustomDataset(Dataset):
    def __init__(self, image_paths, labels, transform=None):
        """
        Initialize dataset.

        Parameters:
            image_paths (list of str): Paths to the images.
```

Régénérer

"""
Table departments, columns = [DepartmentId, DepartmentName]
Table students, columns = [DepartmentId, StudentId, StudentName]
Créer une requête MySQL pour tous les étudiants du département d'informatique
"""

Voici une requête MySQL pour sélectionner tous les étudiants du département d'informatique :

mysql Copy code

```mysql
SELECT students.StudentId, students.StudentName
FROM students
INNER JOIN departments ON students.DepartmentId = departments.DepartmentId
WHERE departments.DepartmentName = 'Informatique';
```

How much money do I have?

CHATGPT

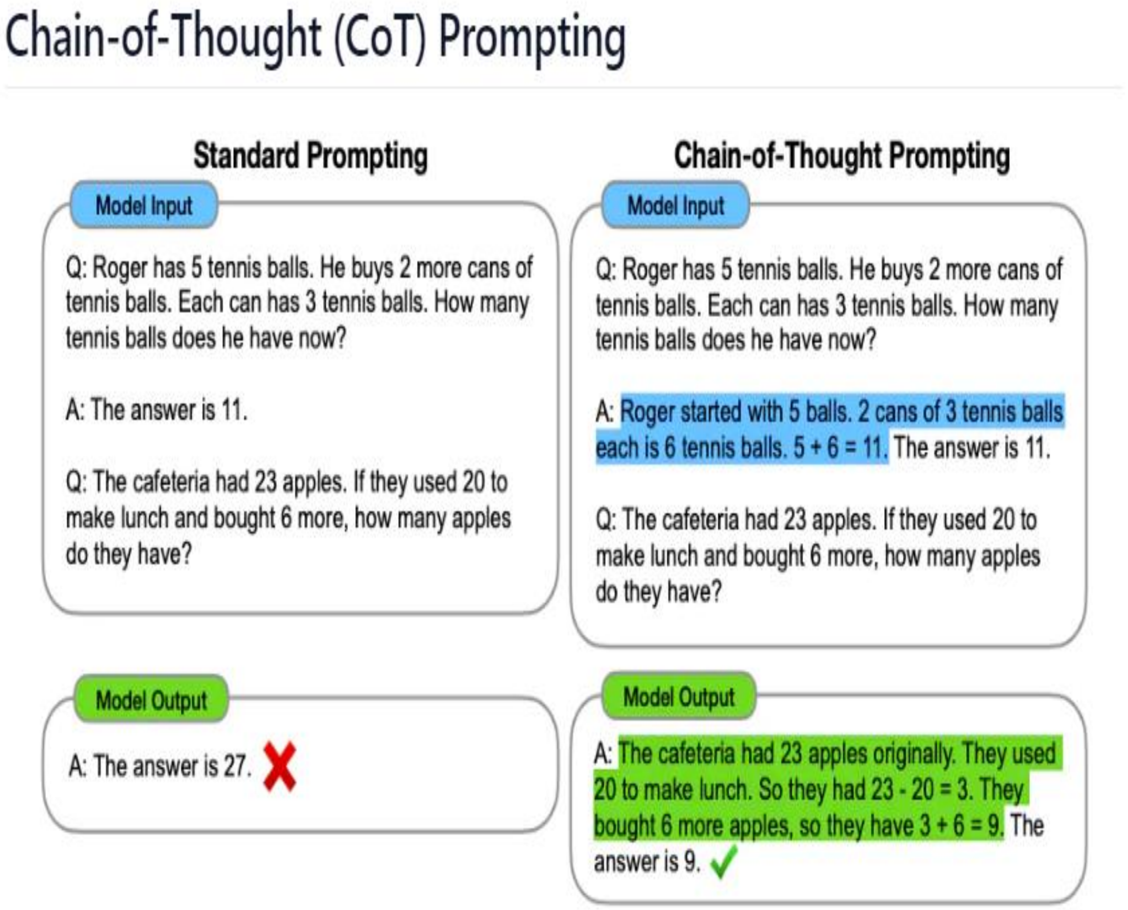
You have:

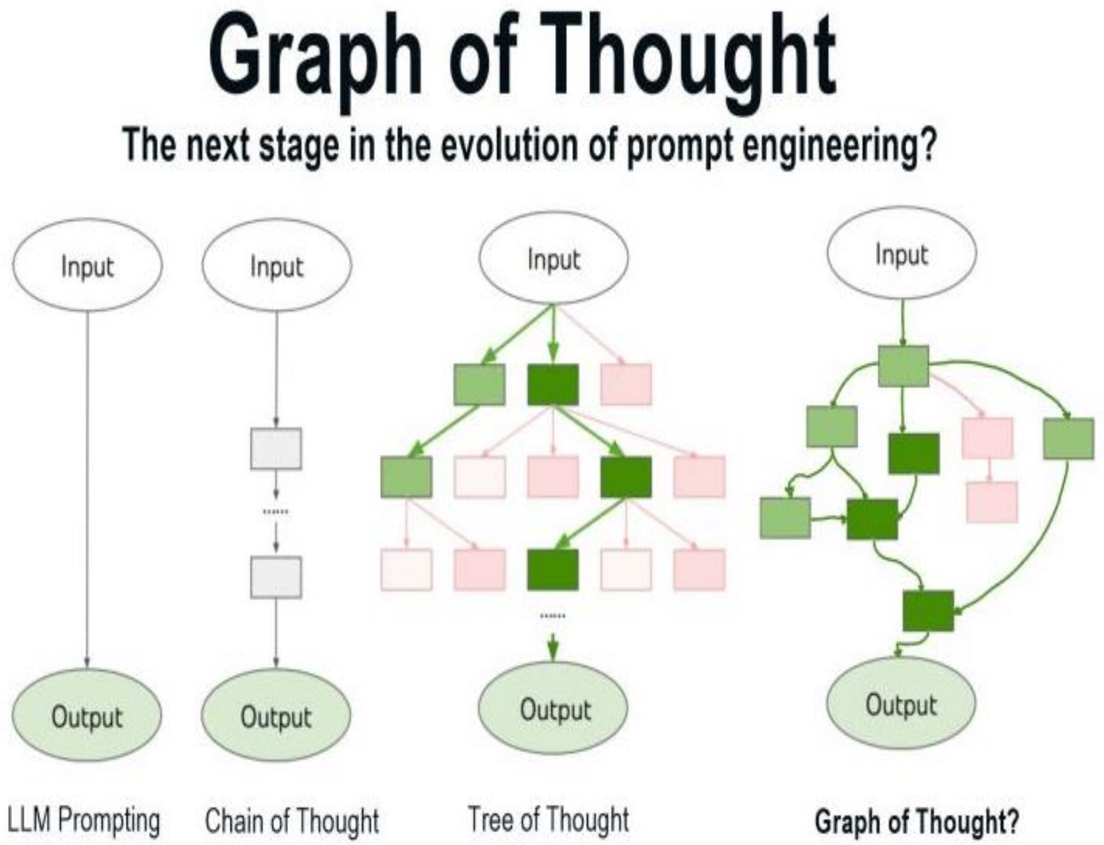
- Three coins with a denomination of 1 (possibly units or currency symbol is not clearly visible).
- One coin with a denomination of 2.

In total, you have 5 units of that currency. To give you a more specific answer, I'd need to know the currency or country of these coins.

# Chain-of-Thought Prompting

Wei et al. (2022)(opens in a new tab),




(*) : Let’s think step by step ➔ Observed in Large Models

# Fine Tuning / Prompt tuning

- Fine Tuning
  - Train Model on Specific data
- Prompt Tuning
  - Hard Prompt Tuning
  - It is an efficient and inexpensive way to adapt an AI foundation model to new downstream tasks without retraining the model and updating its parameters.
  - Soft Prompt Tuning ➔ End of prompt engineering?
  - an active and fertile field of research

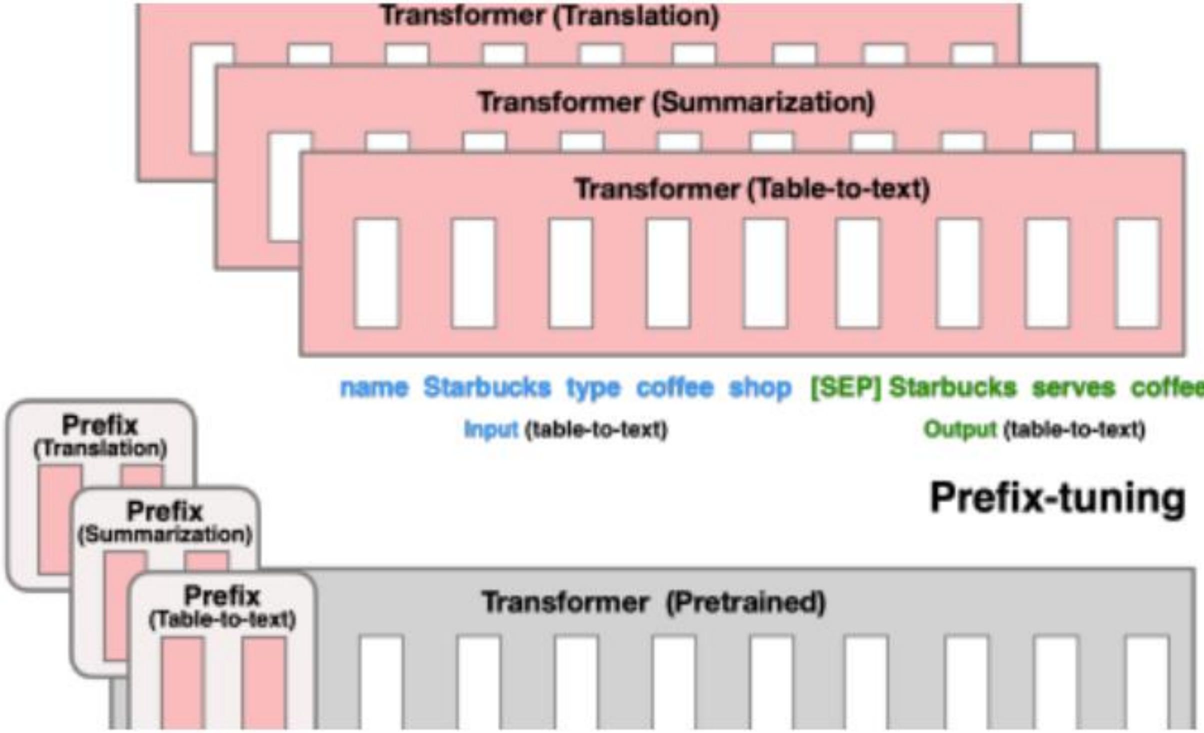



Image Source: Xiang Lisa Li, et al (2021)

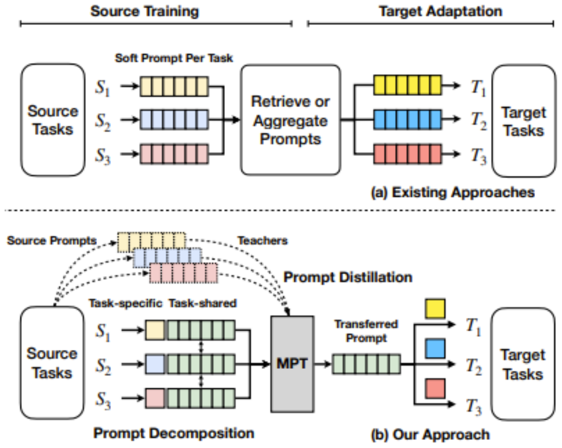
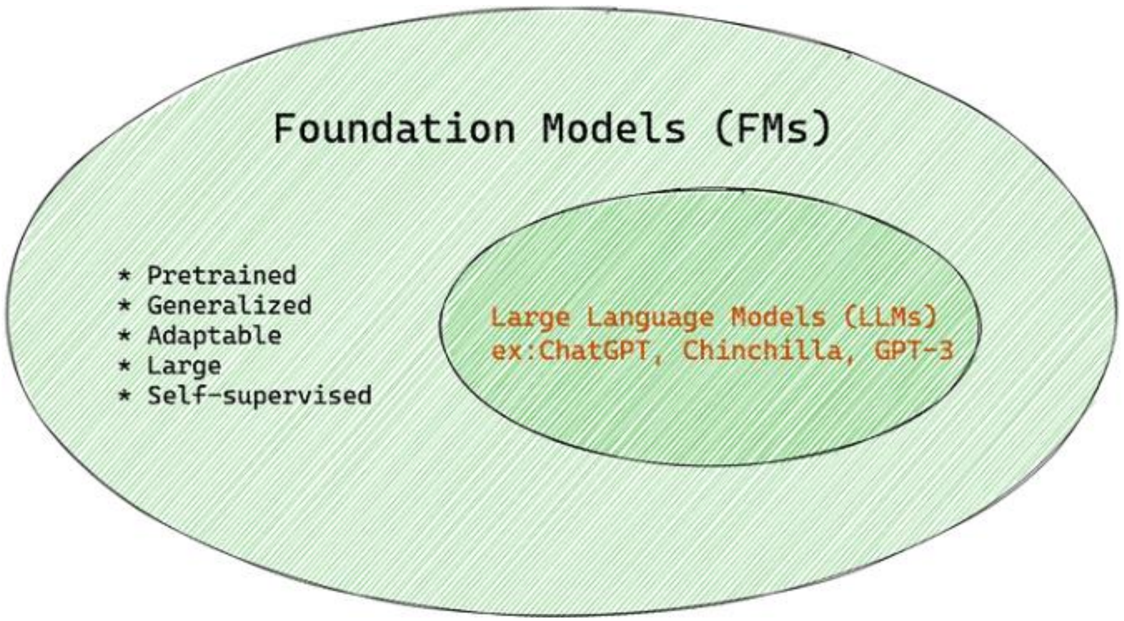


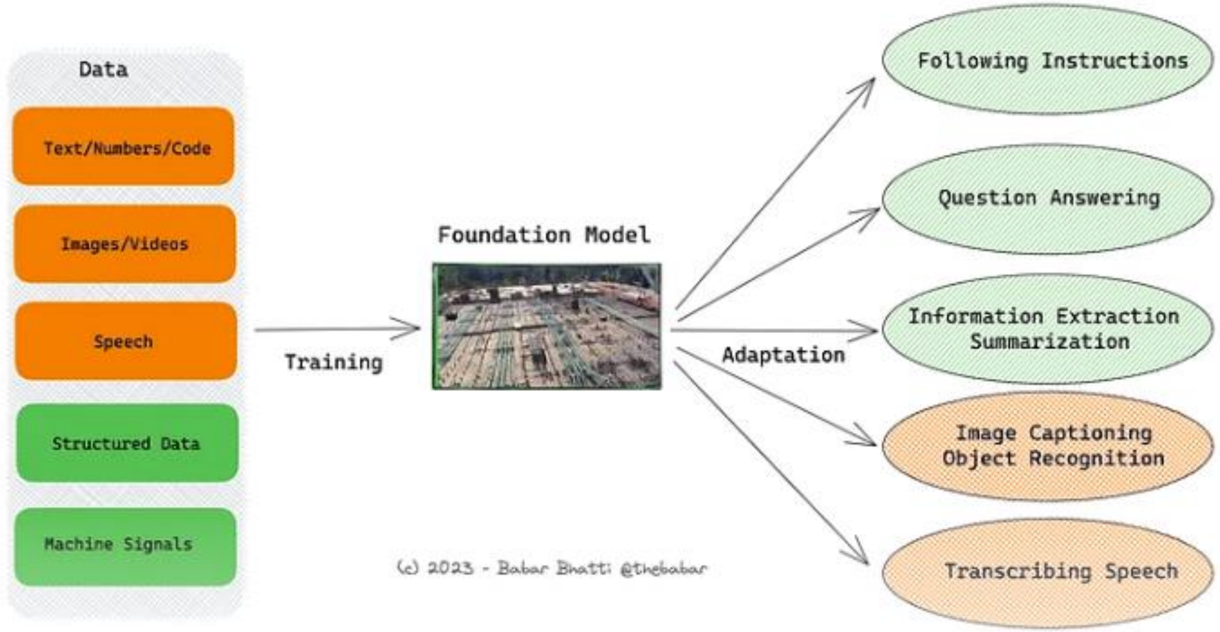

**Figure 1:** A conceptual overview of our approach. Instead of retrieving or aggregating source prompts (top), multitask prompt tuning (MPT, bottom) learns a single transferable prompt. The transferable prompt is learned via prompt decomposition and distillation.

*"In an upcoming paper at the International Conference on Learning Representations (ICLR), Panda and his colleagues show that their Multi-task Prompt Tuning (**MPT**) method outperformed other methods, and even did better than models fine-tuned on task-specific data. Instead of spending thousands of dollars to retrain a 2-billion parameter model for specialized task, MPT lets you customize the model for less than $100, said Panda."*

https://openreview.net/pdf?id=Nk2pDtuhTq

# Large Fondation Models




## Raffiner en local ?

🤗 **PEFT: Parameter-Efficient Fine-Tuning of Billion-Scale Models on Low-Resource Hardware**

Published February 10, 2023

Update on GitHub

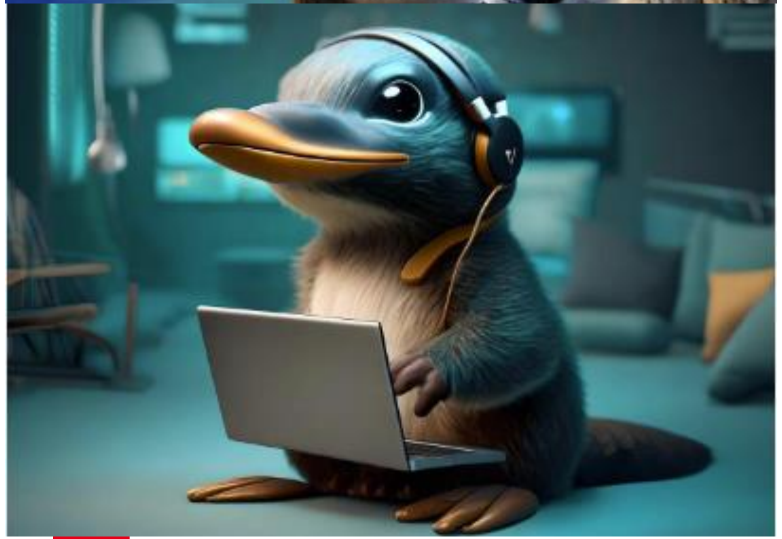
smangrul Sourab Mangrulkar   sayakpaul Sayak Paul

**Low-Rank Adaptation of Large Language Models (LoRA)**

> This is an experimental feature. Its APIs can change in future.

# Zoo

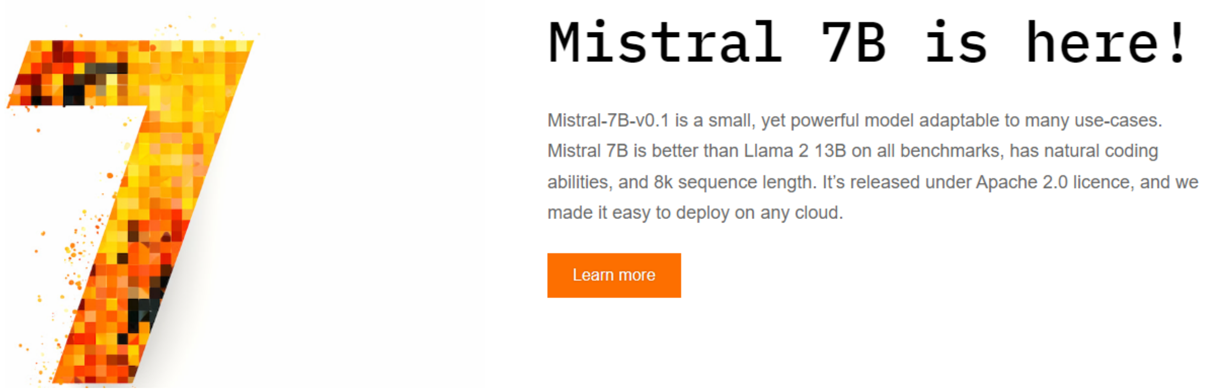




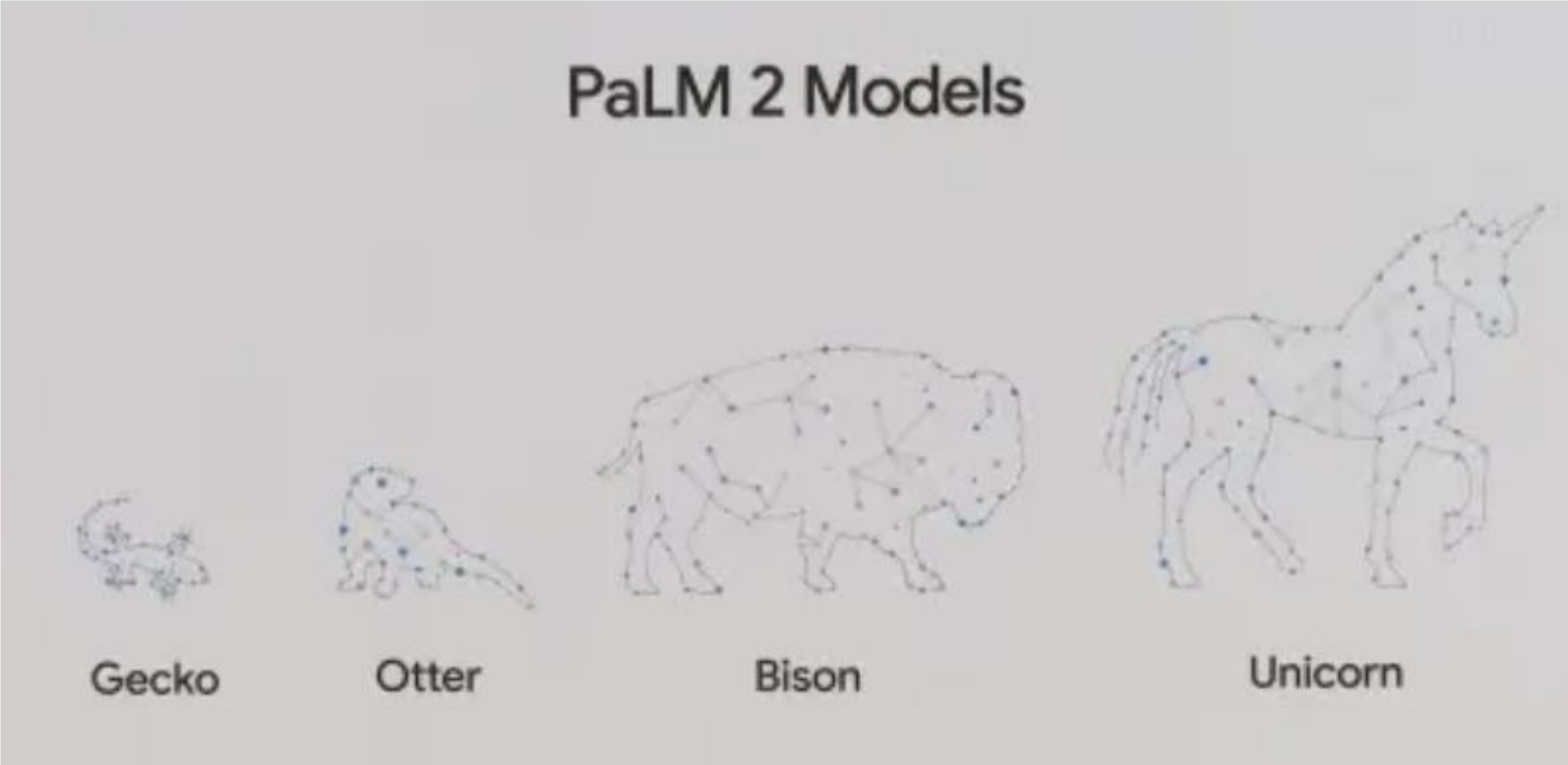
| | | |
|---|---|---|
| AlexaTM | 2022 | AlexaTM 20B: Few-Shot Learning Using a Large-Scale Multilingual Seq2Seq Model |
| Flan-T5 | 2022 | Scaling Instruction-Finetuned Language Models |
| Sparrow | 2022 | Improving alignment of dialogue agents via targeted human judgements |
| U-PaLM | 2022 | Transcending Scaling Laws with 0.1% Extra Compute |
| mT0 | 2022 | Crosslingual Generalization through Multitask Finetuning |
| Galactica | 2022 | Galactica: A Large Language Model for Science |
| OPT-IML | 2022 | OPT-IML: Scaling Language Model Instruction Meta Learning through the Lens of Generalization |
| LLaMA | 2023 | LLaMA: Open and Efficient Foundation Language Models |
| GPT-4 | 2023 | GPT-4 Technical Report |
| PanGu-Σ | 2023 | PanGu-Σ: Towards Trillion Parameter Language Model with Sparse Heterogeneous Computing |
| BloombergGPT | 2023 | BloombergGPT: A Large Language Model for Finance |
| Cerebras-GPT | 2023 | Cerebras-GPT: Open Compute-Optimal Language Models Trained on the Cerebras Wafer-Scale Cluster |
| PaLM 2 | 2023 | A Language Model that has better multilingual and reasoning capabilities and is more compute-efficient than its predecessor PaLM. |

# Large Langage Models (LLMs) : State Of The Art





# PaLM 2 (« Pathways Language Model » Google)





Apart from that, **PaLM 2 can be fine-tuned to make a domain-specific model** right away. Google has already created Med-PaLM 2, a medical-specific LLM fine-tuned on PaLM 2 that received "Expert" level competency on U.S. Medical Licensing Exam-style questions. It achieved an **accuracy of 85.4% in the USMLE test**, even higher than GPT-4 (84%). That said, do bear in mind that GPT-4 is a general-purpose LLM and not fine-tuned for medical knowledge.

# LLaMA 2 → Orca 13B Chat


**Overview**

Orca is a descendant of LLaMA developed by Microsoft with finetuning on explanation traces obtained from GPT-4.

**Description**

Orca-13B is a LLM developed by Microsoft. It is based on LLaMA with finetuning on complex explanation traces obtained from GPT-4. By using rich signals, Orca surpasses the performance of models such as Vicuna-13B on complex tasks. However, given its model backbone and the data used for its finetuning, Orca is under noncommercial use.

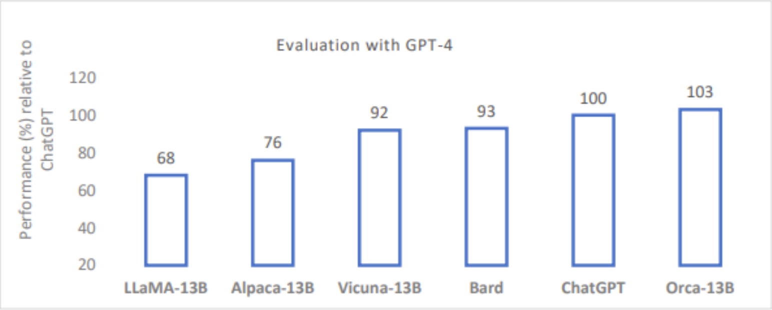


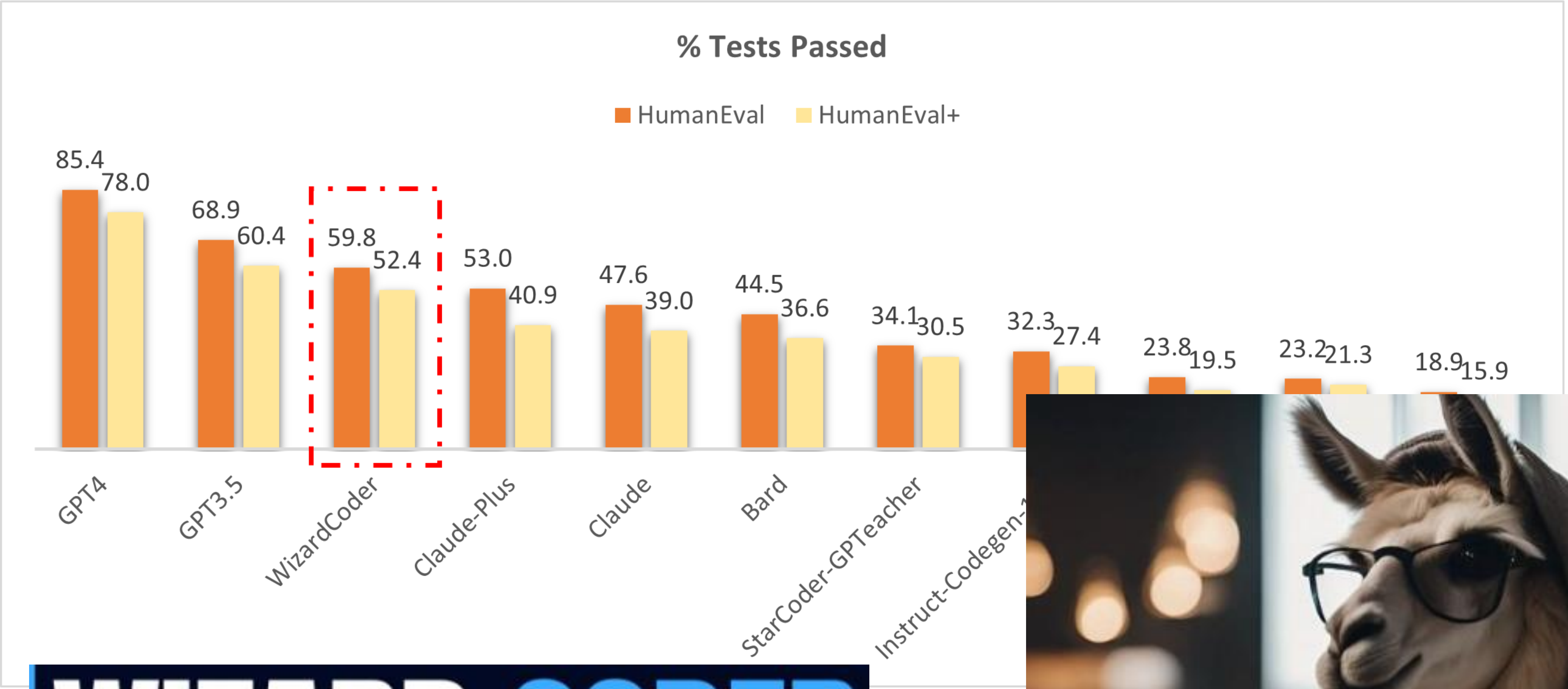

Figure 1: Orca (13B params) outperforms a wide range of foundation models including OpenAI ChatGPT as evaluated by GPT-4 in the Vicuna evaluation set. We further demonstrate similar results against a wide range of evaluation sets from other works in experiments.

https://arxiv.org/pdf/2306.02707.pdf

## Les résultats

Les chercheurs ont comparé les performances zero-shot de Text-da-vinci-003, ChatGPT, GPT-4, Vicuna et Orca dans le benchmark AGIEval sur des questions à choix multiples en anglais.

| Task | Human -Avg | Human -Top | TD-003 | Chat GPT | GPT-4 | Vicuna-13B | Orca-13B |
|---|---|---|---|---|---|---|---|
| AQuA-RAT | 85 | 100 | 29.9 | 31.9 | 40.6 | 20.1 | **27.9** (39.2%) |
| LogiQA | 86 | 95 | 22.7 | 35 | 49.3 | 29.8 | **35.2** (18.1%) |
| LSAT-AR | 56 | 91 | 21.7 | 24.4 | 35.2 | 20.4 | **21.3** (4.3%) |
| LSAT-LR | 56 | 91 | 47.5 | 52.6 | 80.6 | 32.6 | **43.9** (34.9%) |
| LSAT-RC | 56 | 91 | 64.7 | 65.4 | 85.9 | 32.7 | **57.3** (75.0%) |
| SAT-Math | 66 | 94 | 35.5 | 42.7 | 64.6 | 28.6 | **32.3** (12.7%) |
| SAT-English | 66 | 94 | 74.8 | 81.1 | 88.8 | 44.2 | **76.7** (73.6%) |
| SAT-English (w/o Psg.) | 66 | 94 | 38.4 | 44.2 | 51 | 26.2 | **38.8** (48.1%) |
| Average | 67.1 | 93.8 | 41.9 | 47.2 | 62 | 29.3 | **41.7** (42.1%) |






Textbooks Are All You Need

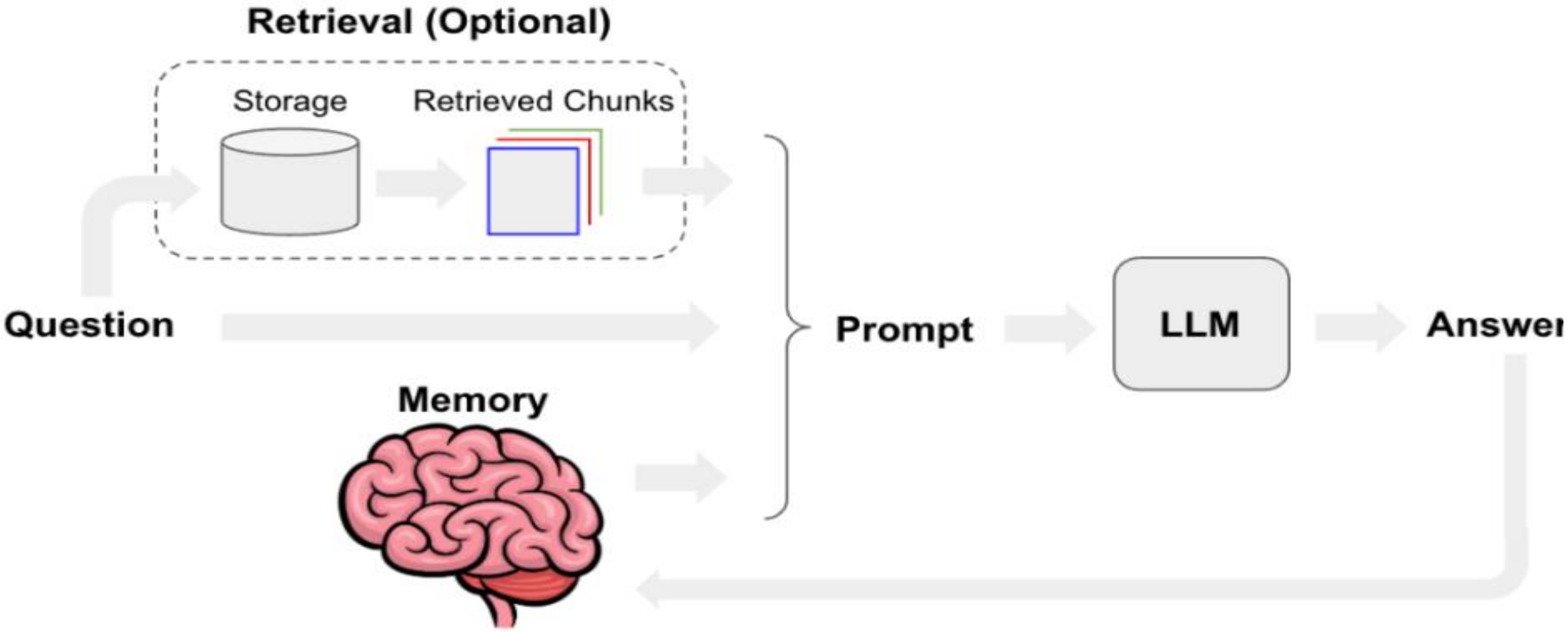
# « ChatGPT / LLMs » on local data

« **Hard/soft prompt Tuning** »

- The model remains stable with its initial capacities
- Use of prompt engineering techniques
  - Hard (manually)
  - Soft (train prefix prompt embedding layers )

«**AI Agent »**

- The model remains stable with its initial capacities
- Use of extrnal tools
  - External data access
  - Short and Long term memory
  - Tasks and code execution
  - Interactive AI (multi agents/llms …)



# « LLM » local : 13 B models

## LLaMA 2 local

[1]:
```
from llama import Llama
import torch.distributed as dist
import os

os.environ['MASTER_ADDR'] = 'localhc
os.environ['MASTER_PORT'] = '29500'
os.environ['RANK'] = "0"
os.environ['WORLD_SIZE'] = "1"

dist.init_process_group(backend='glc
```

[2]:
```
generator = Llama.build(
        ckpt_dir='llama2/llama-2-7b/
        tokenizer_path='llama2/toker
        max_seq_len=2048,
        max_batch_size=8,
    )
```

```
> initializing model parallel with size 1
> initializing ddp with size 1
> initializing pipeline with size 1
Loaded in 11.60 seconds
```

[4]:
```
prompts = [
        # For these prompts, the exp
        "I believe the meaning of life is",
```

[3]:
```
import torch
from transformers import AutoModelForCausalLM, AutoTokenizer, pipeline

tokenizer = AutoTokenizer.from_pretrained("psmathur/orca_mini_v3_13b")
model = AutoModelForCausalLM.from_pretrained(
  "psmathur/orca_mini_v3_13b",
  torch_dtype=torch.float16,
  load_in_8bit=True,
  low_cpu_mem_usage=True,
  device_map="auto"
)

```

```
Downloading shards:   0%|          | 0/3 [00:00<?, ?it/s]
Loading checkpoint shards:   0%|          | 0/3 [00:00<?, ?it/s]
Some weights of LlamaForCausalLM were not initialized from the model checkpoint at psmathur/
orca_mini_v3_13b and are newly initialized: ['model.layers.9.self_attn.rotary_emb.inv_freq',
'model.layers.4.self_attn.rotary_emb.inv_freq', 'model.layers.39.self_attn.rotary_emb.inv_fr
eq', 'model.layers.13.self_attn.rotary_emb.inv_freq', 'model.layers.18.self_attn.rotary_emb.
inv_freq', 'model.layers.35.self_attn.rotary_emb.inv_freq', 'model.layers.26.self_attn.rotar
y_emb.inv_freq', 'model.layers.38.self_attn.rotary_emb.inv_freq', 'model.layers.33.self_att
n.rotary_emb.inv_freq', 'model.layers.27.self_attn.rotary_emb.inv_freq', 'model.layers.20.se
```

# « LLM » local : 7B models With llama.cpp

[122]:
```python
from ctransformers import AutoModelForCausalLM
```

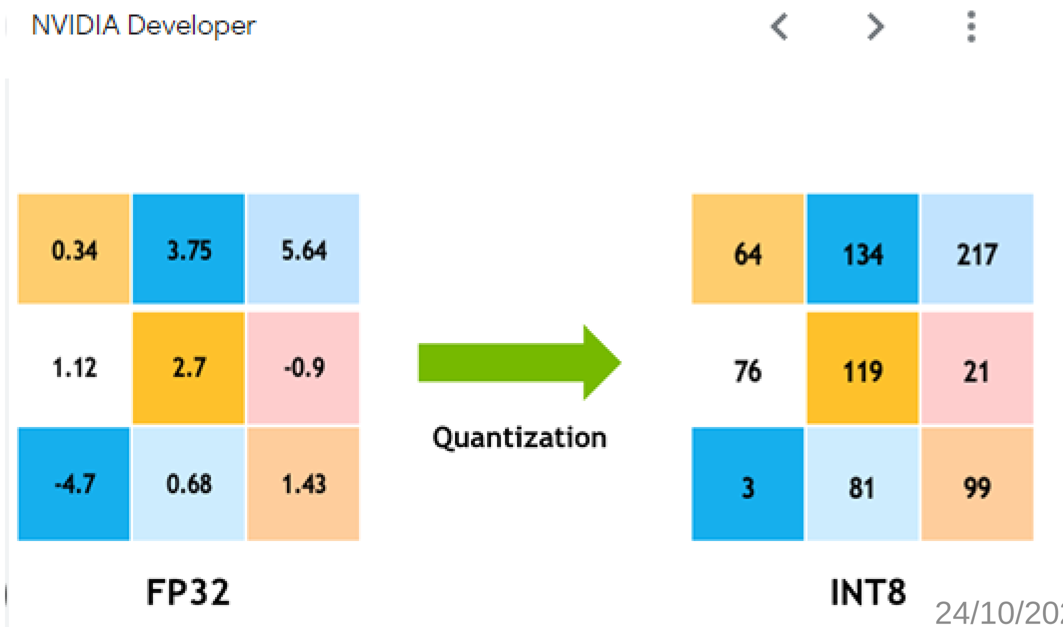
[123]:
```python
# Set gpu_layers to the number of layers to offload to GPU. Set to 0 if no GPU acceleration is available on your system.
cllm = AutoModelForCausalLM.from_pretrained("./locallms/mistral-7b-instruct",
                                            model_file="mistral-7b-instruct-v0.1.Q6_K.gguf",
                                            model_type="mistral", gpu_layers=1000, context_length=4048 )
```

## Or From Hugging face

[ ]:
```python
llm = AutoModelForCausalLM.from_pretrained("TheBloke/Mistral-7B-Instruct-v0.1-GGUF",
                                           model_file="mistral-7b-instruct-v0.1.Q6_K.gguf",
                                           model_type="mistral", gpu_layers=0)
```

# Quantization

**GGUF** is a new format introduced by the llama.cpp team on August 21st 2023. It is a replacement for GGML, which is no longer supported by llama.cpp. GGUF offers numerous advantages over GGML, such as better tokenisation, and support for special tokens. It is also supports metadata, and is designed to be extensible.



# AI Agents

## LLM Powered Autonomous Agents

Building agents with LLM (large language model) as its core controller is a cool concept. Several proof-of-concepts demos, such as AutoGPT, GPT-Engineer and BabyAGI, serve as inspiring examples. The potentiality of LLM extends beyond generating well-written copies, stories, essays and programs; it can be framed as a powerful general problem solver

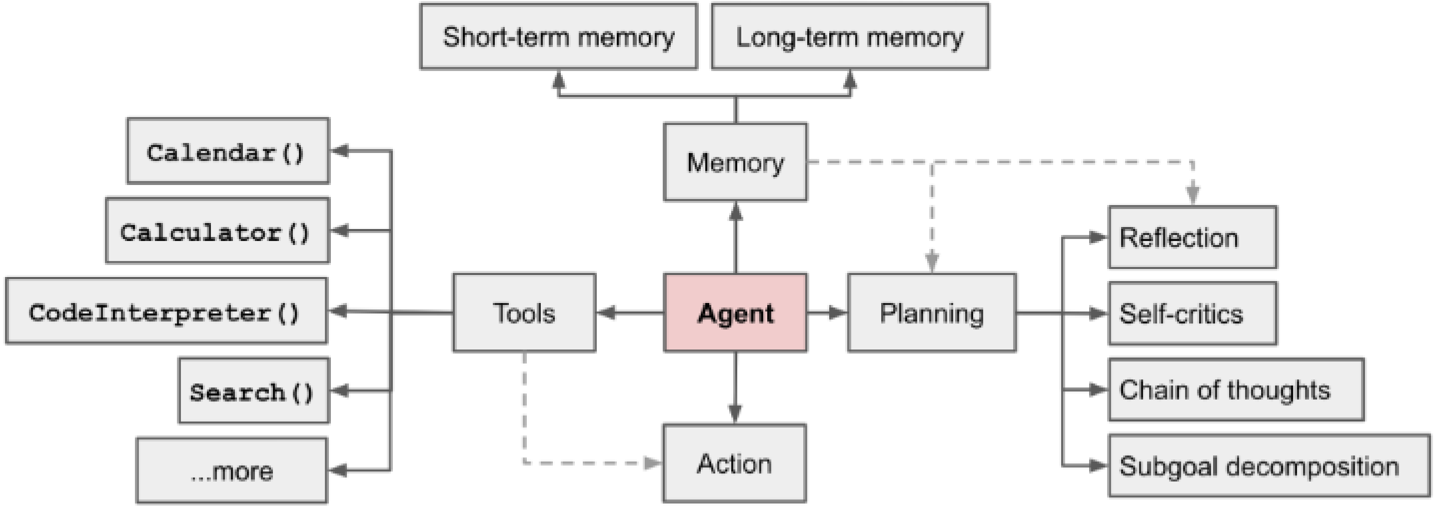
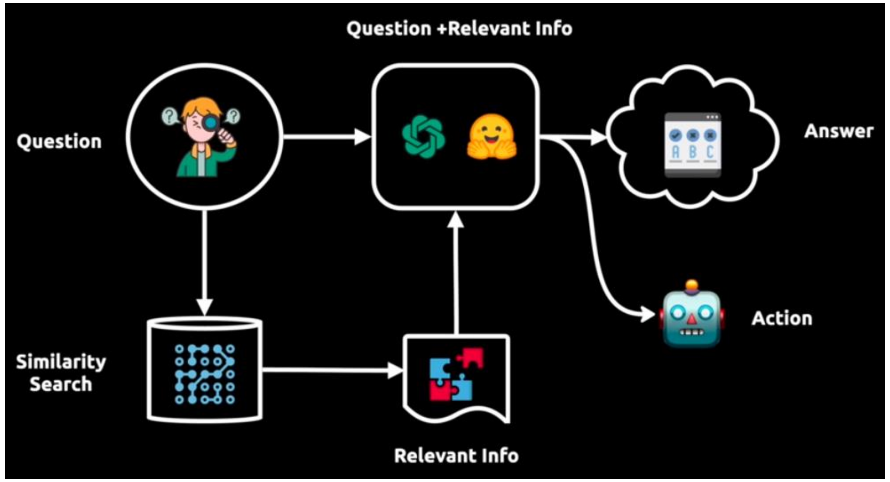


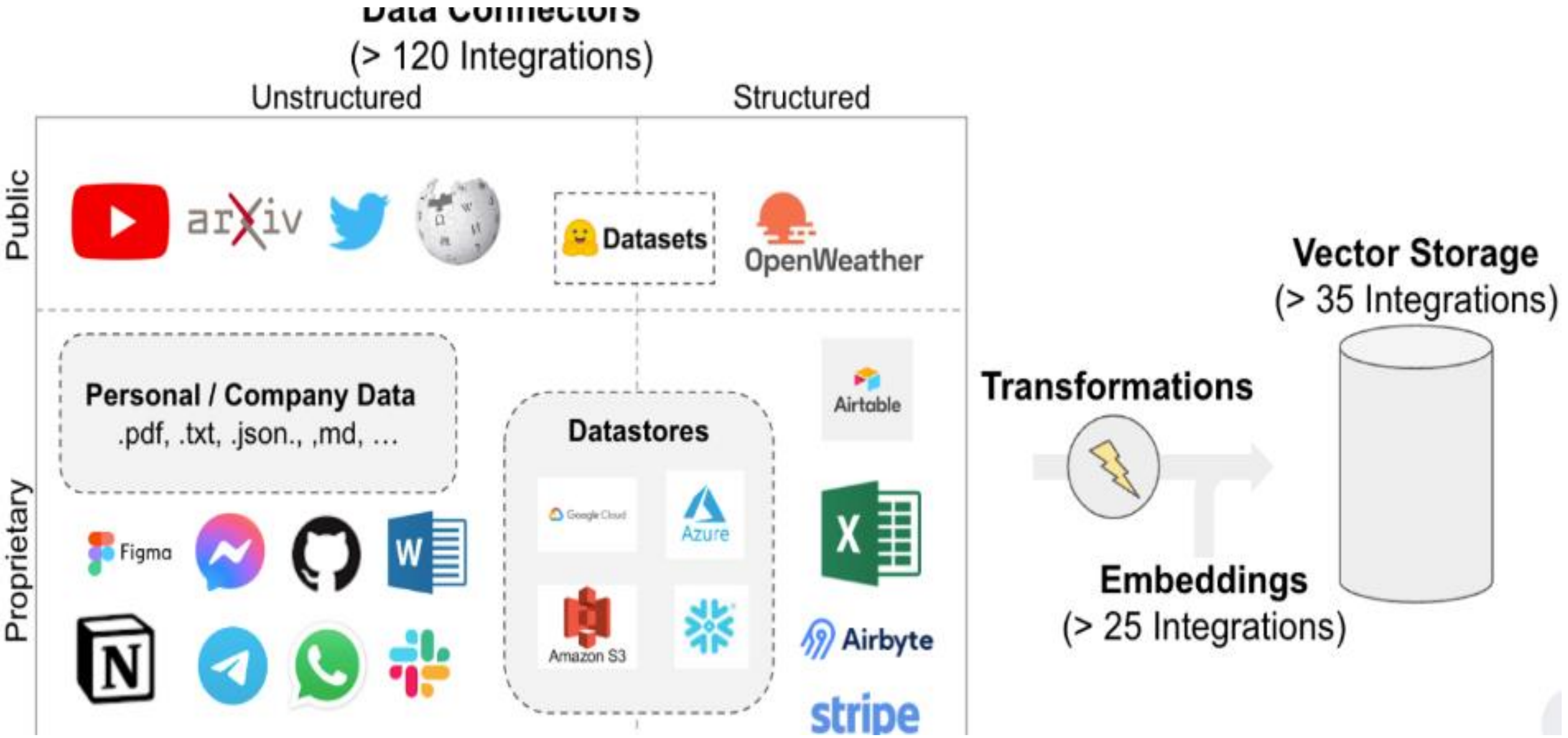

Fig. 1. Overview of a LLM-powered autonomous agent system.

LLM functions as the agent's brain, complemented by several key components

# AI Agents tools: LangChain

LangChain



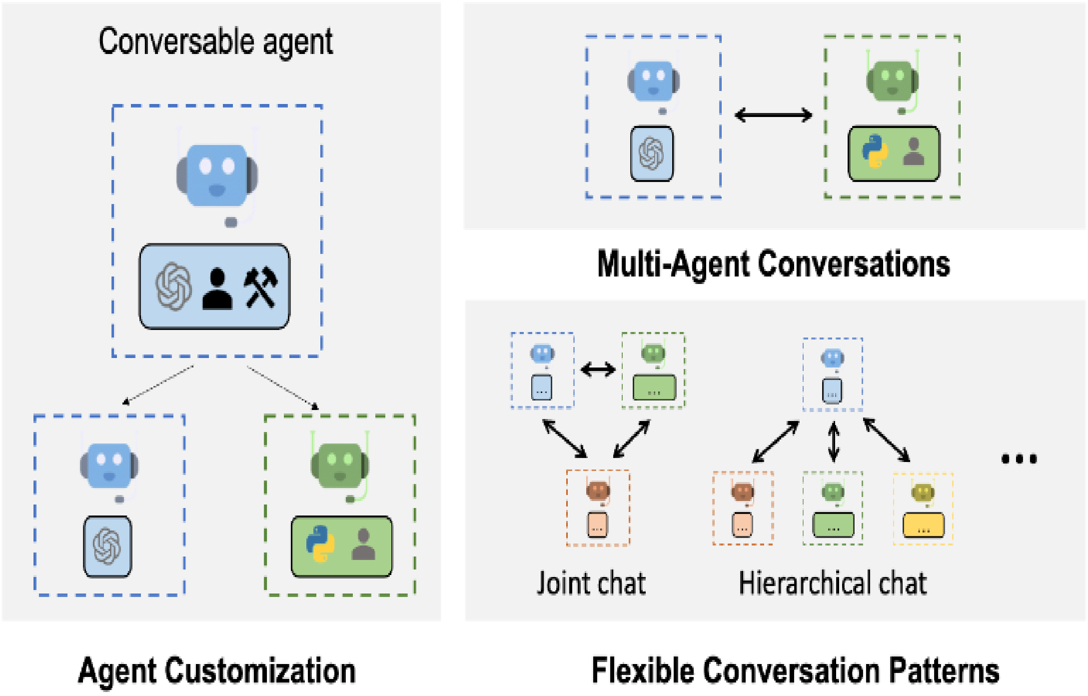

## Welcome to LlamaIndex 🦙 !

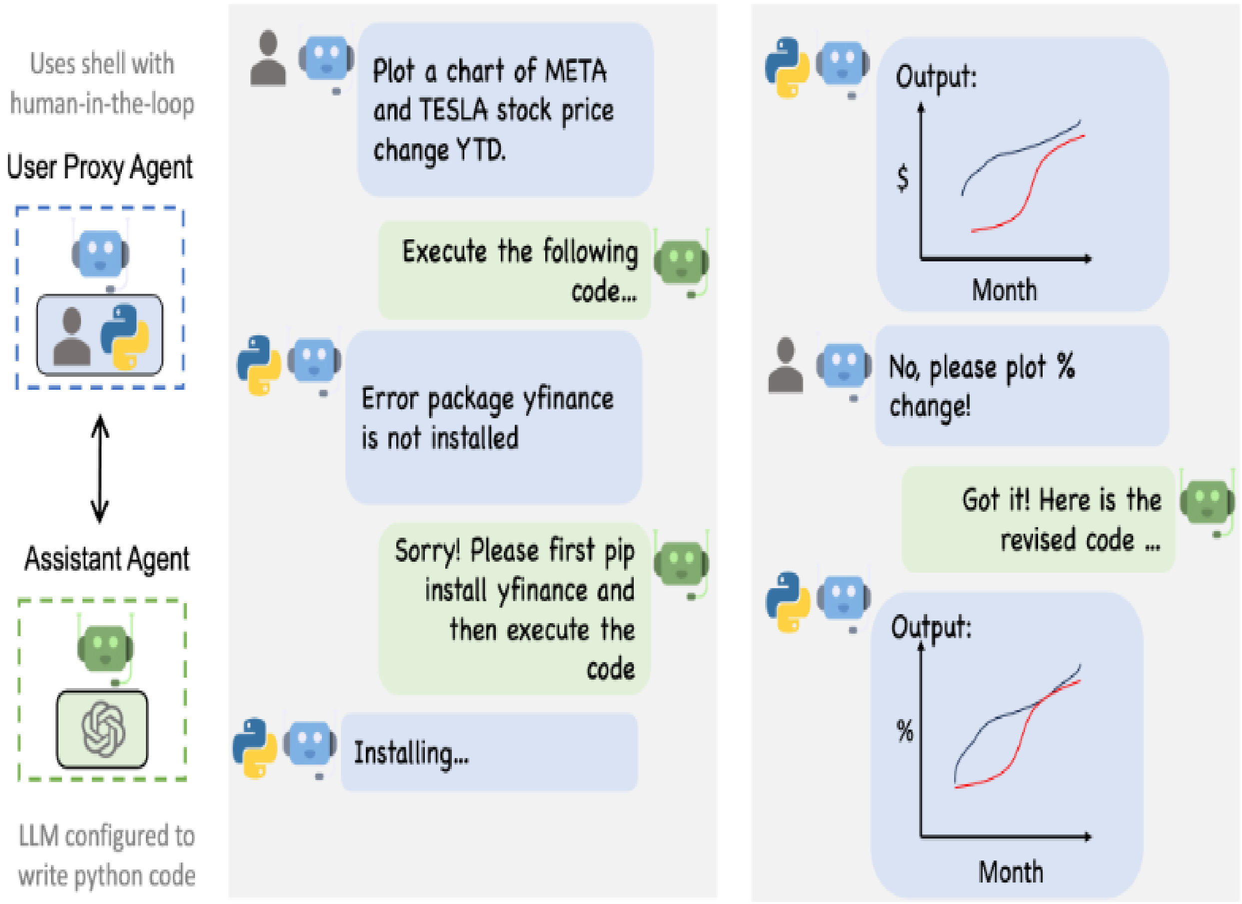
LlamaIndex (formerly GPT Index) is a data framework for LLM applications to ingest, structure, and access private or domain-specific data.

# AI Agents tools : MS Autogen

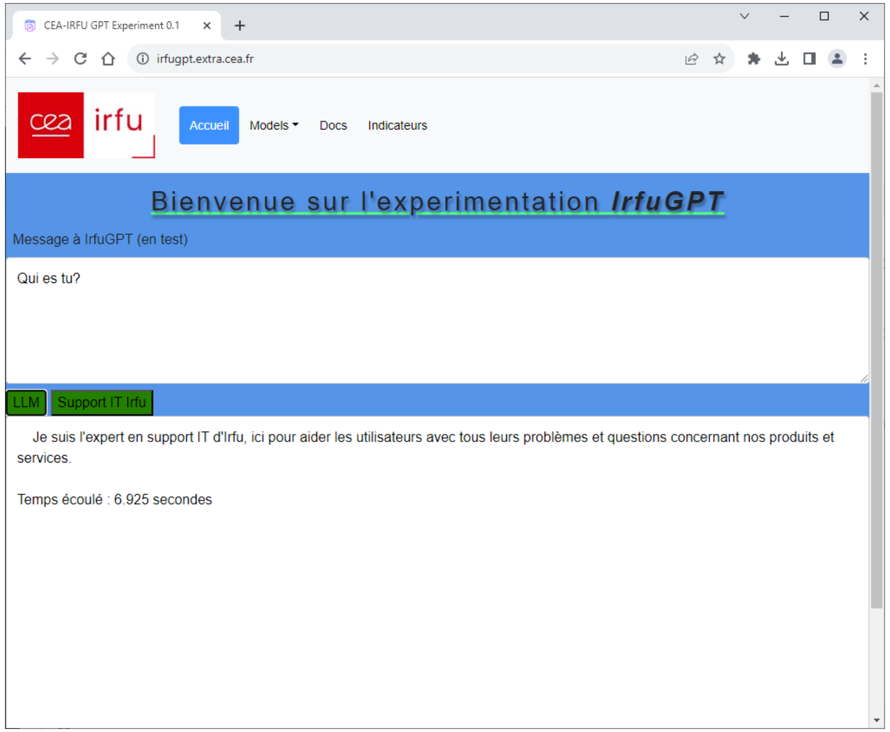
## What is AutoGen

AutoGen is a framework that enables the development of LLM applications using multiple agents that can converse with each other to solve tasks. AutoGen agents are customizable, conversable, and seamlessly allow human participation. They can operate in various modes that employ combinations of LLMs, human inputs, and tools.



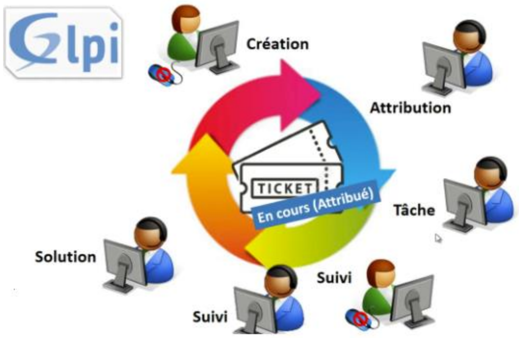



# Use Case on local data : IrfuGPT

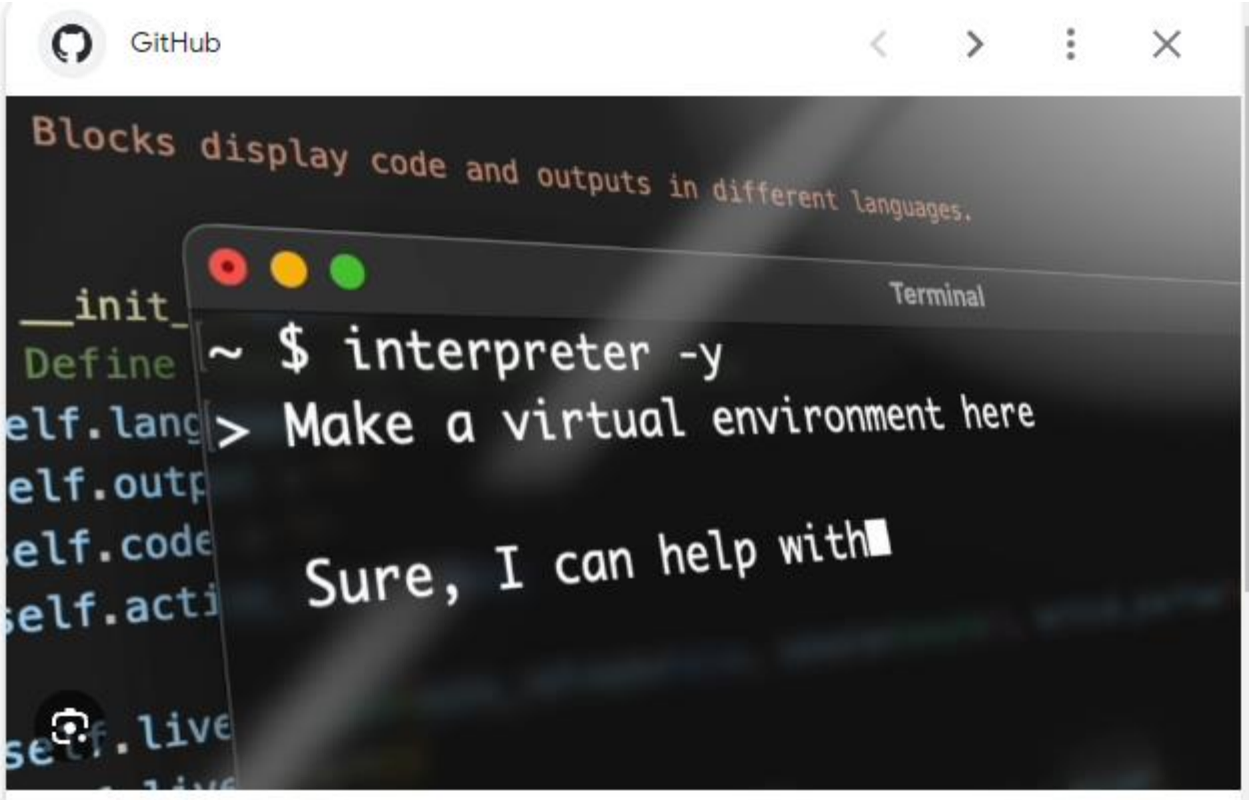



- CEA Security constraint ➔ : impossible to send data externally
- Solution ➔ Use of a local LLM
- Possibilities of using ➔ Like ChatGPT (same perfs of 3,5)
- Added Value ➔ Use on local data

- Model Choice
  - Orca13B on an NVIDIA A10 40 GB (use 20 GB)
  - Mistral 7B (use 5GB) ➔
- Fine Tuning
  - Hard Prompt Fine Tuning
  - Soft Prompt Fine Tuning➔
- Data
  - 40K GLPI IT support tickets
  - Cleaning Data by LLM (python scripting)
  - 3 Gitlab ➔
- Development
  - Python + LangChain
  - Embedding (BGE Open Source)
  - Storage on local Vector DB (ChromaDB Open Source)
- Implementation
  - API for LLM (Flax)
  - API for Agent (Flax)
  - Web Application (Streamlit+LAMP)

# Use Case : Code Generation / Debugging




- Use of GPT4 (code interpreter model – Advanced Data analysis- )
  - Can generate code (From paper Retential Network – no code or pseudo code – generate a working pytorch implementation)
  - Excels in explaining, commenting, debugging and even correcting errors
- Code Inerpreter (Open Source)
  - Python code

## Installation

Run the following command in your terminal to install Open Interpreter.

```
pip install open-interpreter
```

## Terminal usage

After installation, you can start an interactive chat in your terminal by running:

```
interpreter
```

For example, if the repo URL was:

```
https://huggingface.co/TheBloke/Mistral-7B-Instruct-v0.1-GGUF
```

The command to download and run this language model in Open Interpreter would be:

Terminal Python

```
interpreter --local --model Mistral-7B-Instruct-v0.1-GGUF
```

# Conclusion (The Word After ChatGPT)

## Y. Le Cun :

- AI Agents = Interactions with Digital World
- LLMs = Warehouse of human knowledge
- All must be Open Source