Modern NLP

Based on Deep Learning and Language models.
Day 4 Afternoon

Alexandre Gazagnes

4th Day

1. Afternoon (~ 4h)
   - Very small remainder of Day 3
   - LLM and Chat GPT
   - LLM’s Advanced Technique
   - LLM Ecosystem
   - Intelligent Agents

Introduction to Large Language Models and ChatGPT

Alexandre Gazagnes

Fundamentals of LLMs and GPTs

Generative Pre-trained Transformer architecture, specifically designed to generate text in a conversational format.

- Developed by OpenAI, ChatGPT is based on the larger family of models called GPT (e.g., GPT-3, GPT-3.5, GPT-4), which are trained using a variant of the transformer architecture first described in the paper "Attention is All You Need" by Vaswani et al. In 2017

Keys features :

- **Conversational** Abilities: It is optimized to participate in conversations, providing responses that can mimic human-like exchanges in a chat format.
- **Contextual** Awareness: ChatGPT can keep track of the context within a conversation, allowing it to provide relevant and coherent responses based on the previous dialogue.
- **Fine-Tuning**: Although based on the GPT architecture, ChatGPT is fine-tuned with supervised learning techniques using human trainers to improve its responses in conversational tasks specifically.

Diff Between BERT & GPT

	GPT	BERT
Design Philosophy	Generate one word at a time and are trained to predict the next word Well-suited for tasks like text generation.	Rich understanding of language context and word relationships from the surrounding text. Good for tasks that require understanding the context and relationships between words in a sentence
Training Objectives	Generating coherent and contextually appropriate conversational responses.	Understand the context of words in text by looking at words that come before and after it in a sentence.
Input and Output Dynamics	Generates outputs sequentially and is optimized for longer, more continuous text outputs that maintain coherence over a conversation.	Processes an entire input sequence at once and generates a representation of the input
Use cases	Primarily used in applications requiring human-like dialogue generation, such as chatbots, virtual assistants, and customer service bots.	Tasks that require deep understanding rather than generation, such as extracting information, sentiment analysis, and language understanding tasks.

Evolution of GPT Models

GPT 4 training : more than 100M$

All transformers auto regressive based model

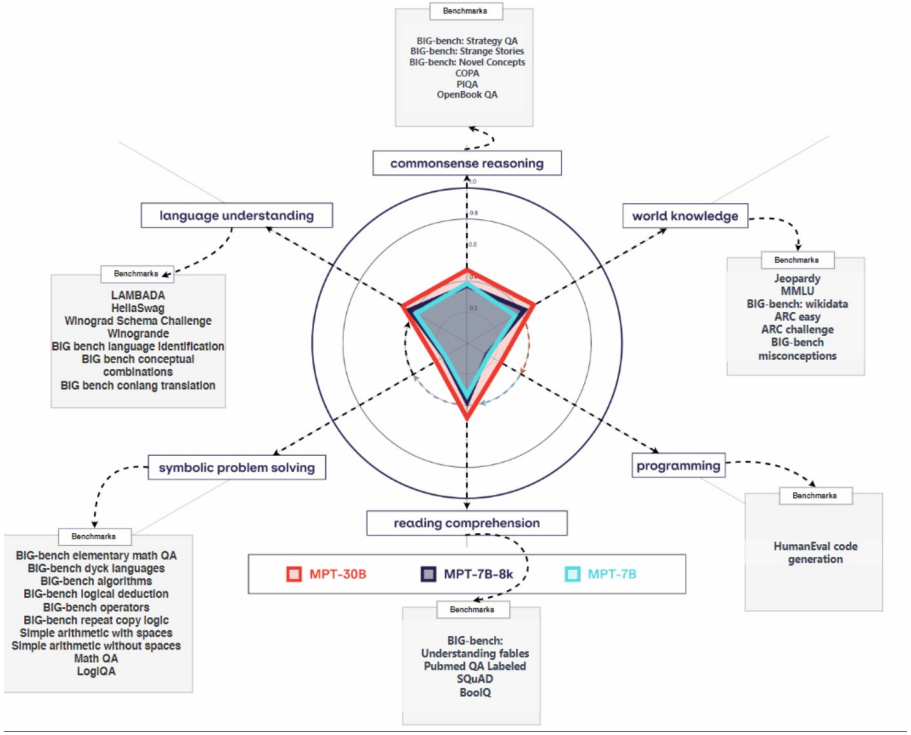
Few-shot, one-shot, and zero-shot learning capabilities, refined training methods, RL, RLHF, Supervised / Unsupervised ...

Very fast iteration rate !

1 model to rule them all => Any task you want => All you need is prompt !

Feature	GPT-2	GPT-3	GPT-3.5	GPT-4
Release Year	2019	2020	2022	2023
Parameters	1.5 billion	175 billion	Intermediate between GPT-3 and GPT-4	Over 500 billion
Capabilities	Basic text generation and coherence	Advanced text generation, context understanding	Improved nuanced understanding, reduced hallucinations	Superior context maintenance, nuanced text generation
Applications	Simple text-based tasks	Broad applications, including writing and chatbots	Enhanced precision in tasks, better context retention	Reliable performance in complex domains like legal and medical
Improvements	Introduction of transformer-based model	Massive scale-up in model size, broader knowledge	Refinement in handling complex queries and context	Significant reduction in factual inaccuracies, enhanced specific task handling

Scoring, hallucinations and Limits of GPTs



	LLaMA 2 70B	GPT - 3.5	Mixtral 8x7B
MMLU (MCQ in 57 subjects)	69.9%	70.0%	**70.6%**
HellaSwag (10-shot)	87.1%	85.5%	86.7%
ARC Challenge (25-shot)	85.1%	85.2%	**85.8%**
WinoGrande (5-shot)	**83.2%**	81.6%	81.2%
MBPP (pass@1)	49.8%	52.2%	**60.7%**
GSM-8K (5-shot)	53.6%	57.1%	**58.4%**
MT Bench (for Instruct Models)	6.86	**8.32**	8.30

LLMs Zoo

There are **dozens of major** LLMs, and hundreds that are arguably significant for some reason or other.

Listing them all would be nearly impossible, and in any case, it would be **out of date within days** because of how quickly LLMs are being developed.

Take the word "best" with a grain of salt there: the tried was to narrow things down by offering a list of the **most significant, interesting, and popular** LLMs (and LMMs), **not necessarily the ones that outperform on benchmarks** (though most of these do).

Also mostly focused on **LLMs that you can use**

LLM	Developer	Popular apps that use it	# of parameters	Access
GPT	OpenAI	Microsoft, Duolingo, Stripe, Zapier, Dropbox, ChatGPT	175 billion+	API
Gemini	Google	Some queries on Bard	Nano: 1.8 & 3.25 billion; others unknown	API
PaLM 2	Google	Google Bard, Docs, Gmail, and other Google apps	340 billion	API
Llama 2	Meta	Undisclosed	7, 13, and 70 billion	Open source
Vicuna	LMSYS Org	Chatbot Arena	7, 13, and 33 billion	Open source
Claude 2	Anthropic	Slack, Notion, Zoom	Unknown	API
Stable Beluga	Stability AI	Undisclosed	7, 13, and 70 billion	Open source
StableLM	Stability AI	Undisclosed	7, 13, and 70 billion	Open source
Coral	Cohere	HyperWrite, Jasper, Notion, LongShot	Unknown	API

Practice !

LLM’s Advanced Technqiues

Basic Prompt Engineering Techniques

BAD ONE :

- Write an outline for a 500-word blog post targeted at teenagers about the health benefits of doing yoga .

BETTER ONE :

- You are an expert sports scientist.
- Your writing style is casual but terse.
- Write an outline for a 500-word blog post targeted at teenagers about the health benefits of doing yoga.
- Only include factually correct information. Explain your reasoning.

Advanced Prompt Engineering Techniques

- **Structure your request**
  Organize your prompt in a clear, structured way. Use bulleted lists or number important points to guide ChatGPT in its response.
  This will enable the algorithm to better understand your expectations and provide consistent responses.

- **Give it a role**
  If you want ChatGPT to provide a response that strikes a particular tone or demonstrates a certain level of expertise, give it a role to play.
  For example, if you want to write a briefing note on how to organize a cybersecurity watch, say something like "You're an expert in cybersecurity and used to passing on your knowledge to people new to the field. You're writing a prompt that explains how to conduct a cybersecurity watch."

- **Be specific**
  The more precise and detailed your prompt, the better ChatGPT's responses will be.
  Avoid vague instructions and provide specific information about what you're expecting. For example, instead of asking "Tell me about Italian cuisine", specify "Give me a list of traditional Italian pasta recipes".

- **Define the context**
  If your request requires a specific context, be sure to provide it to ChatGPT.
  For example, if you're asking for book recommendations, specify the literary genre, authors you like, or even examples of books you've already read and enjoyed.

Practice !

LLM & GPT Ecosystem

Some problems, One solution

Powerful models but very specific problems.

One framework to help : langchain

Topic	Problem	Solution
Structured Responses	Need structured outputs for software integration.	LangChain offers output parser tools to manage and format responses.
Flexibility with Different LLMs	Dependency on a single LLM provider limits flexibility.	LangChain's LLM class allows for easy swapping of models, enhancing system adaptability.
Memory Limitations of LLMs	LLMs have limited memory, affecting long conversation recall.	LangChain extends memory with chat message history tools to maintain conversation consistency.
Integration into Pipelines	Integrating LLM functionality within data or software pipelines is challenging.	LangChain supports integration through "chains" for straightforward workflows and "agents" for complex conditional interactions.
Data Handling for LLMs	Complicated data input management for LLMs.	LangChain facilitates data import and organization from various sources and implements methods like prompt stuffing and map-reduce for effective data utilization.

What is Langchain ?

- Langchain is a **high level framework** that helps us to write better and faster GPTs applications.
-
- Like the **transformer pipeline of HF** but for GPTs and LLMs
- That helps us to **writter better code** and to **easily deploy applications**.
- An **MLOps** and an **API** service are also avialable.

```python
from langchain.chains import OpenAIModerationChain
from langchain_core.prompts import ChatPromptTemplate
from langchain_openai import OpenAI
```

```python
moderate = OpenAIModerationChain()
```

```python
model = OpenAI()
prompt = ChatPromptTemplate.from_messages([("system", "repeat after me: {input}")])
```

```python
chain = prompt | model
```

```python
chain.invoke({"input": "you are stupid"})
```

```
'\n\nYou are stupid.'
```

```python
moderated_chain = chain | moderate
```

```python
moderated_chain.invoke({"input": "you are stupid"})
```

```
{'input': '\n\nYou are stupid',
 'output': "Text was found that violates OpenAI's content policy."}
```

Langchain main use cases

Querying Datasets with Natural Language : Enables writing SQL queries or equivalent Python/R code using natural language, making data analysis accessible to non-coders.

- Load data using LangChain's document loaders. => Pass data to LLM via index-related chains. => Parse responses with an output parser.
- General Workflow: Provide LLM with data structure details (table names, column names/types, missing values). => Request SQL/Python/R code from LLM. =>Execute the code without needing to pass actual data.
- Advanced Workflow: Make multiple LLM calls to both generate queries and interpret results. => Use LangChain's chat message history tools to maintain consistency in result interpretation.

Interacting with APIs : Integration in data pipelines, especially when combined with other API interactions, such as retrieving stock data or interfacing with cloud platforms.

- Chains and agent features to connect sequential steps.
- Business logic for branching in pipelines.

Building a Chatbot : Enhances chatbots, making them more realistic through generative AI.

- Prompt templates to control chatbot personality and style.
- Message history tools extend chatbot memory beyond default LLM limits, ensuring consistency across conversations.

Retrieval-Augmented Generation

Main problem whe have is about **what source of data** and **is this data out of date ?**

We need **a specific of database** to look into these sources for each question + a **manager** to act as a thrid party player.

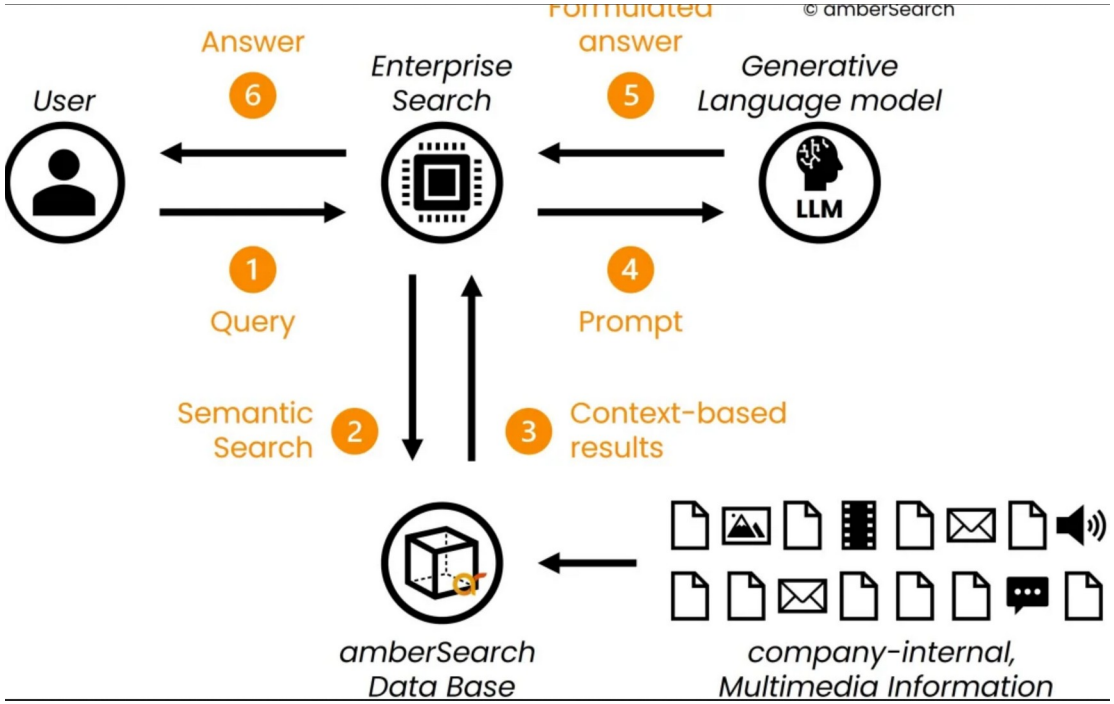
The process is different :

- Without RAG : Question => Answer
- With RAG : Question => Go to the databse / find latest and the best documents => give a response => give your sources

Advantages : Less hallucinations, positive behaviour (no more ‘i don’know’ => ‘i can say you’), more valuable information, reliable feed backs...

Retrieval-Augmented Generation

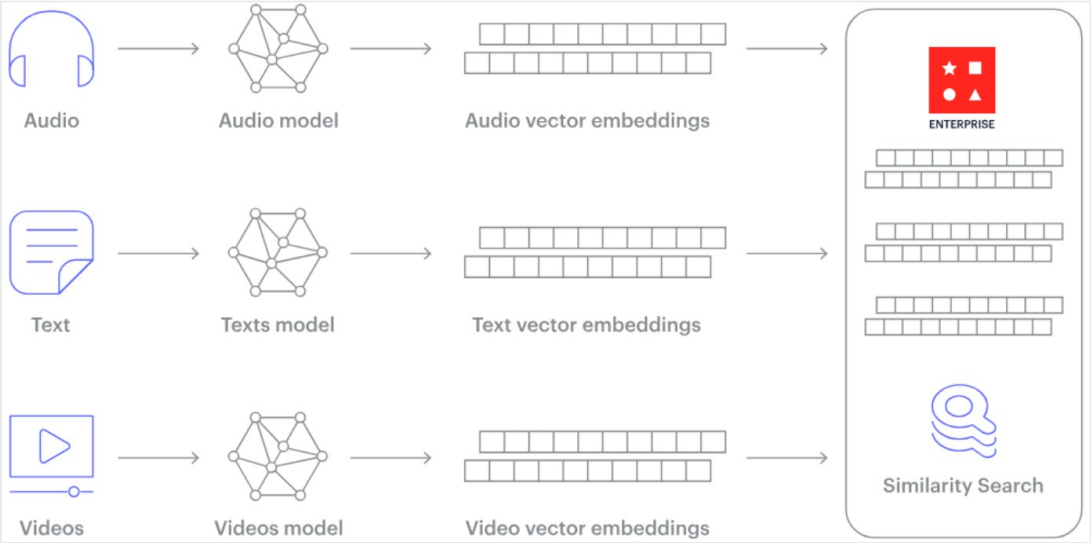
- More complex framework
- Multiple data type ingestion and embedding
- Non finite knowledge base to build



Vector Databases

While regular databases search for exact data matches, vector databases look for the closest match using **specific measures of similarity.**

Vector databases use special search techniques known as **Approximate Nearest Neighbor (ANN) search**, which includes methods like hashing and graph-based searches.



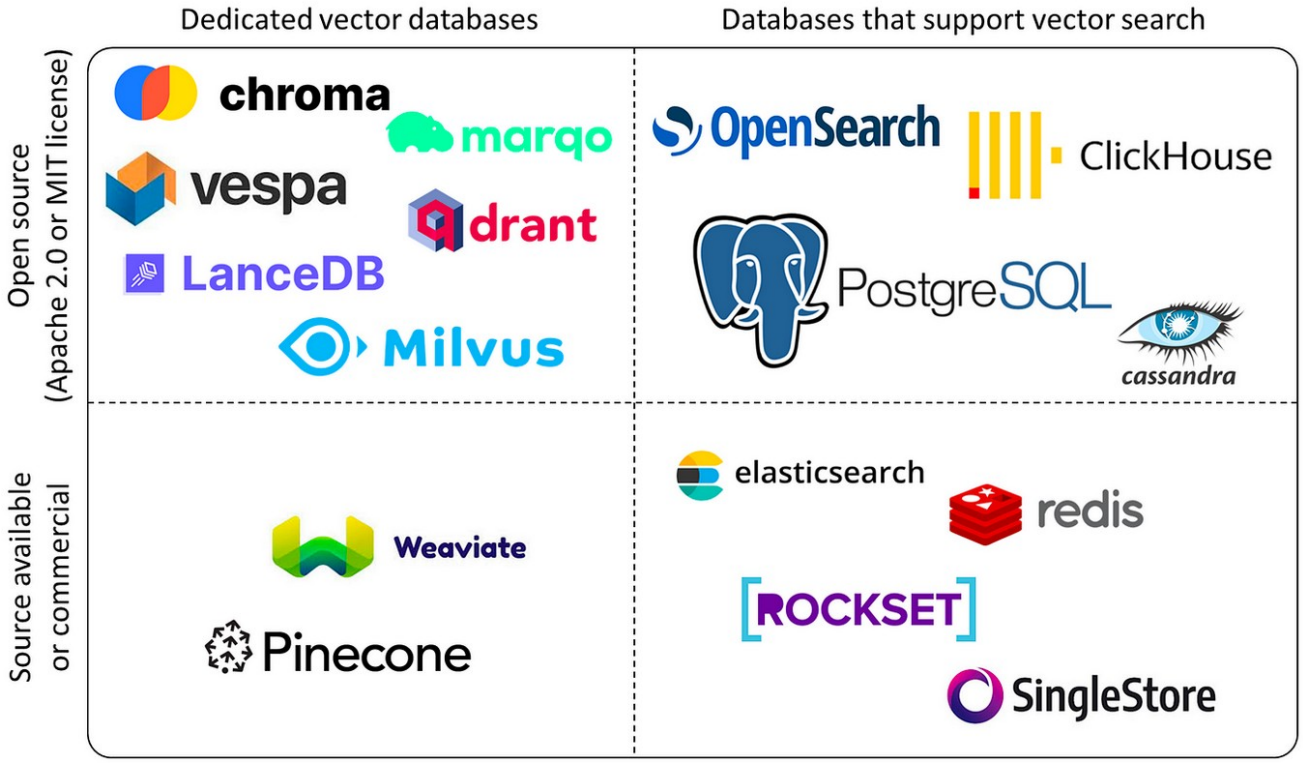
Vector Databases

Large ecosystem of vector databases

Advantages :

- Flexibility
- Speed
- Scalable

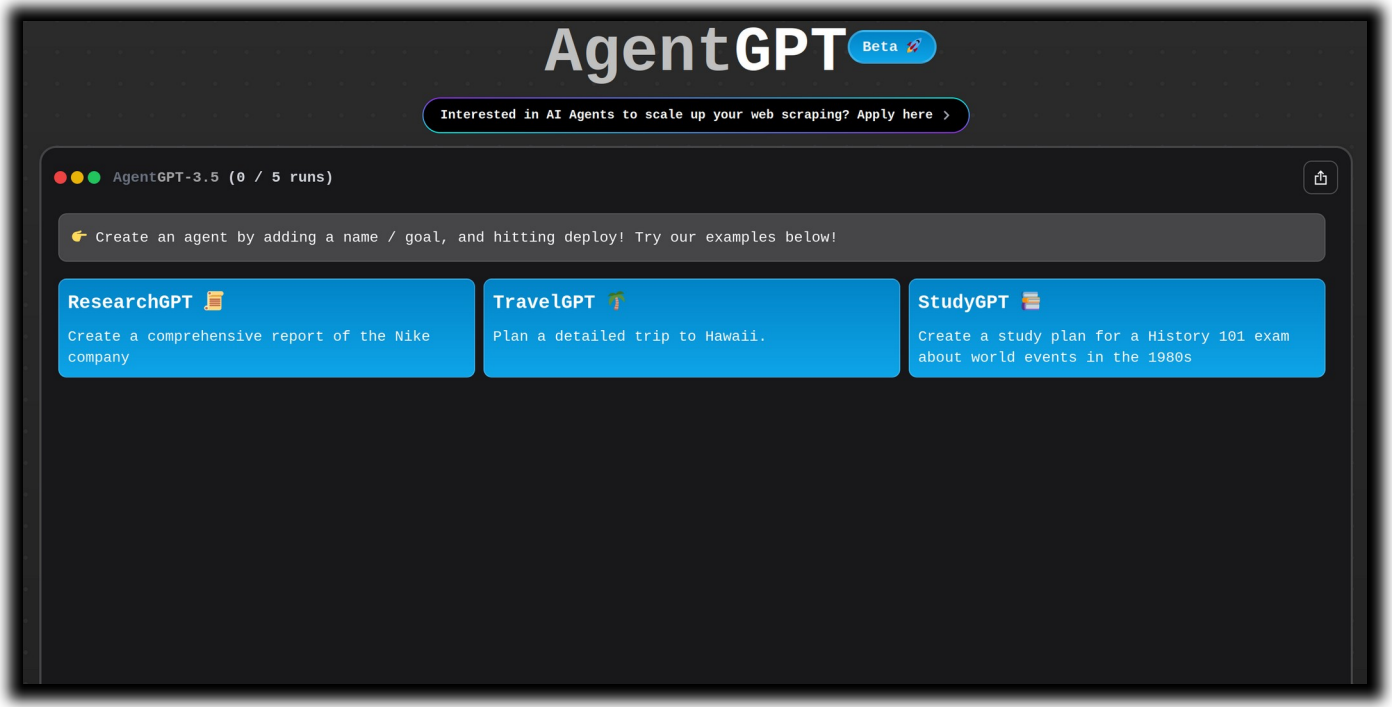
Old fashion DataBase such as SQL or NoSQL can be used



Practice !

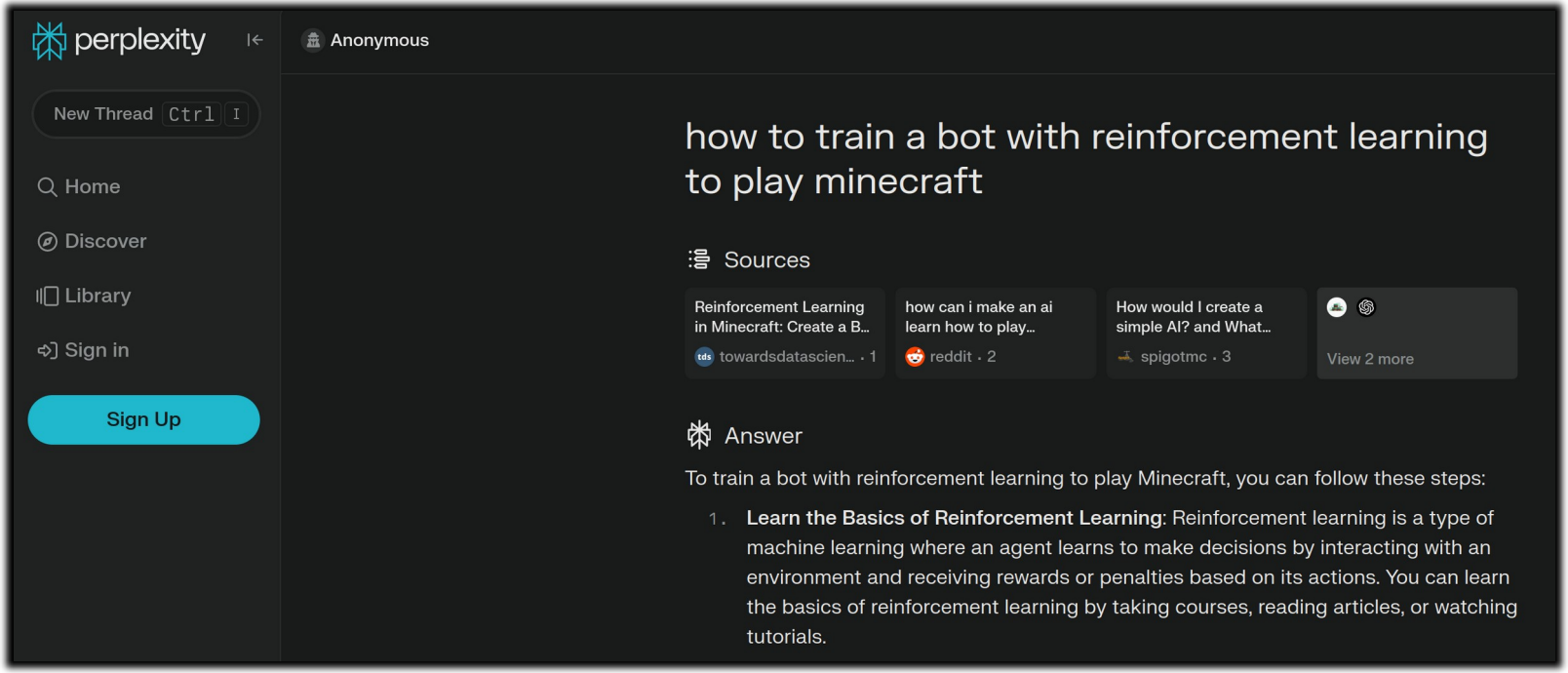
Intelligent Agents

Best way to understand What is an agent !



Alexandre Gazagnes

Perplexity : Google + LLM



AutoGPT

AutoGPT: build & use AI agents

AutoGPT 52506 members | Follow @Auto_GPT | License MIT

AutoGPT is the vision of the power of AI accessible to everyone, to use and to build on. Our mission is to provide the tools, so that you can focus on what matters:

- 🏗️ **Building** - Lay the foundation for something amazing.
- 🧪 **Testing** - Fine-tune your agent to perfection.
- 🤝 **Delegating** - Let AI work for you, and have your ideas come to life.

Be part of the revolution! **AutoGPT** is here to stay, at the forefront of AI innovation.

📖 Documentation | 🚀 Contributing | 🛠️ Build your own Agent - Quickstart

🥇 Current Best Agent: evo.ninja

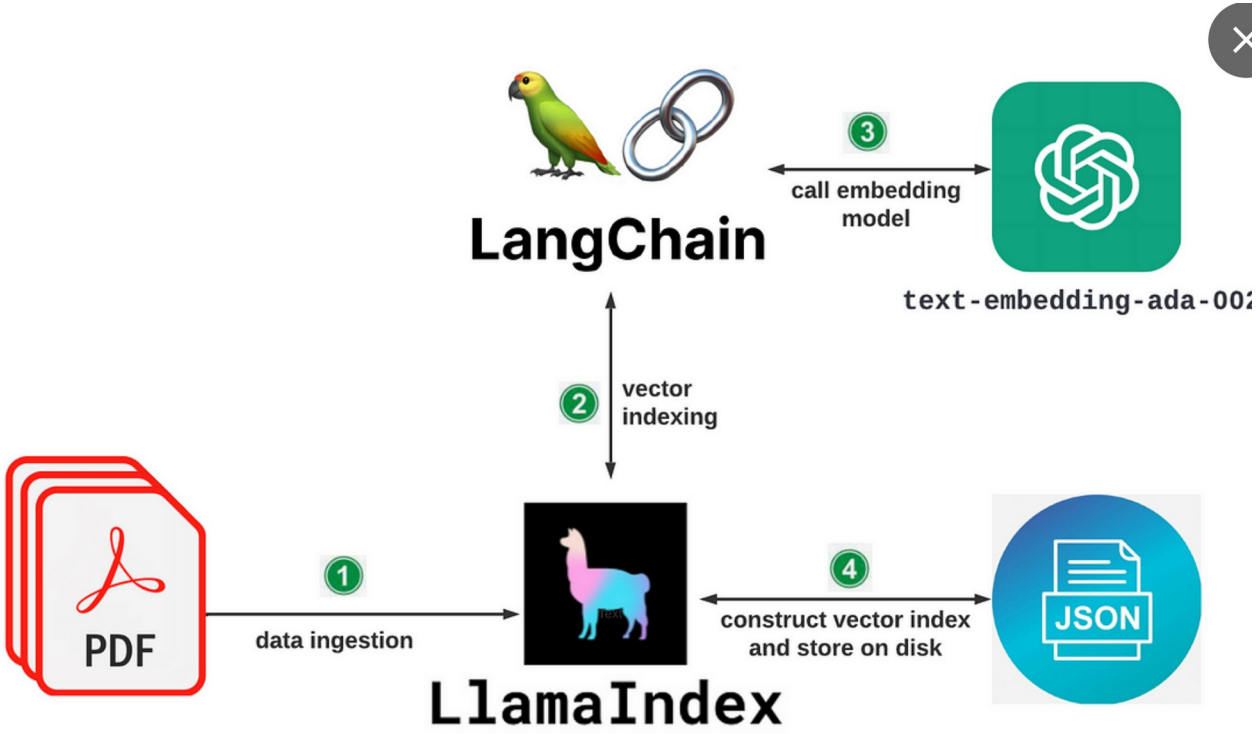
SuperAGI

Alexandre Gazagnes

Practice !

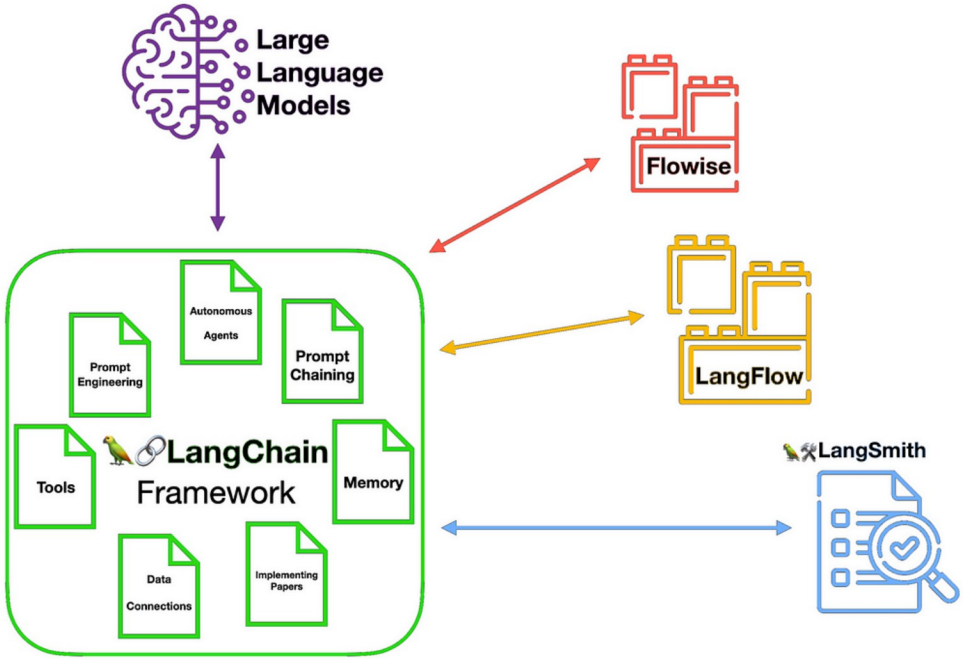
Annexes

Annexes


LangChain
3
call embedding model
text-embedding-ada-00
2
vector indexing
PDF
1
data ingestion
LlamaIndex
4
construct vector index and store on disk
JSON

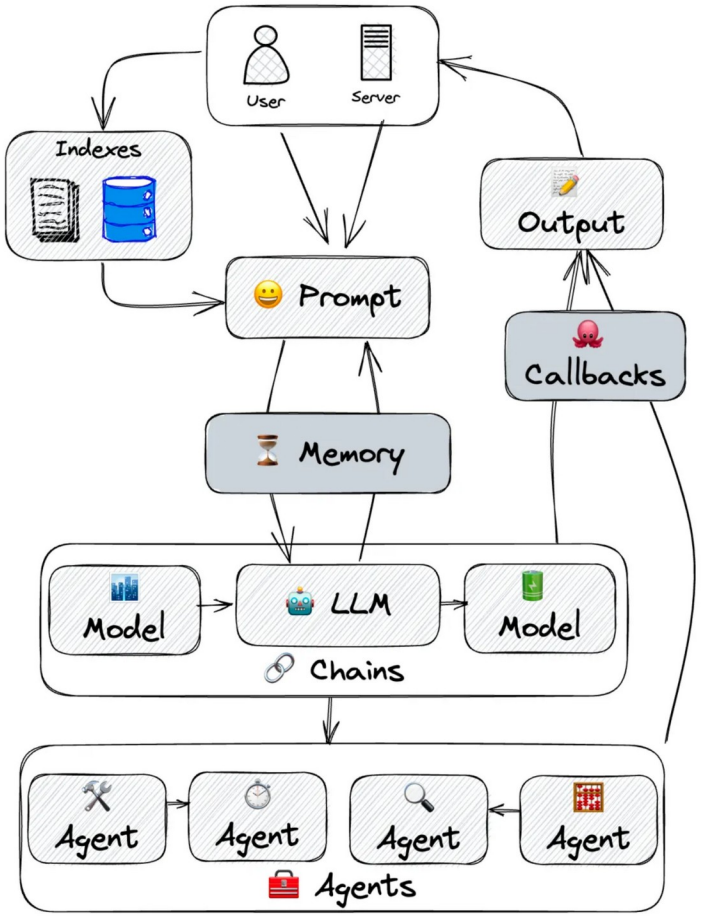
Annexes

LangChain Ecosystem

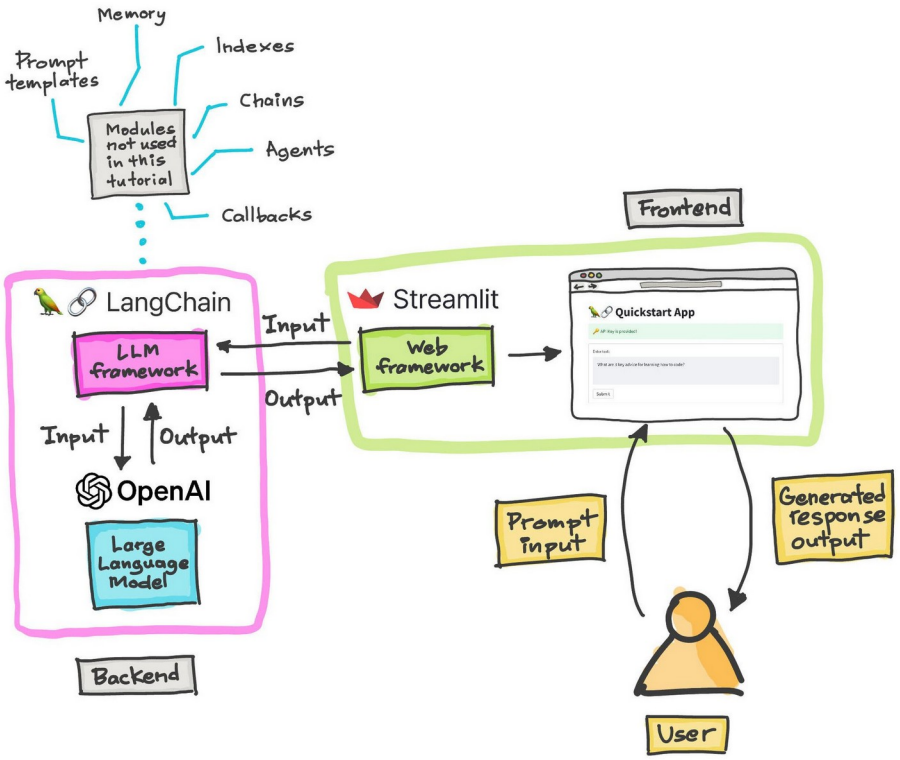


Alexandre Gazagnes

Annexes


User
Server
Indexes
Output
Prompt
Callbacks
Memory
Model
LLM
Model
Chains
Agent
Agent
Agent
Agent
Agents

Annexes


Memory
Prompt templates
Indexes
Chains
Modules not used in this tutorial
Agents
Callbacks
LangChain
LLM framework
Input
Output
Input
Output
OpenAI
Large Language Model
Backend
Streamlit
Web framework
Frontend
Quickstart App
Prompt Input
Generated response output
User