

AI VIETNAM
All-in-One Course
(TA Session)

Question answering on AIO pdf material with RAG

Project



AI VIET NAM
[@aivietnam.edu.vn](https://aivietnam.edu.vn)

Dinh-Thang Duong – TA
Nguyen-Thuan Duong – TA

Objectives



LangChain



Welcome to the PDF QA! To get started:

1. Upload a PDF or text file
2. Ask a question about the file

YOL0v10_Tutorials.pdf processed. You can now ask questions!

Cho tôi danh sách tên tác giả của bài viết latex này



Use the following pieces of context to answer the question at the end. If you don't know the answer, just say that you don't know, don't try to make up an answer.

Question: Cho tôi danh sách tên tác giả của bài viết latex này

Helpful Answer: The authors of the LaTeX document are Dinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui, and Quang-Vinh Dinh.

Sources: source_0, source_1, source_2

In this lecture, we will discuss about:

1. What is LLMs?
2. What is RAG?
3. How to use LLMs in HuggingFace.
4. How to implement RAG with LangChain.
5. How to deploy a RAG application with Chainlit.

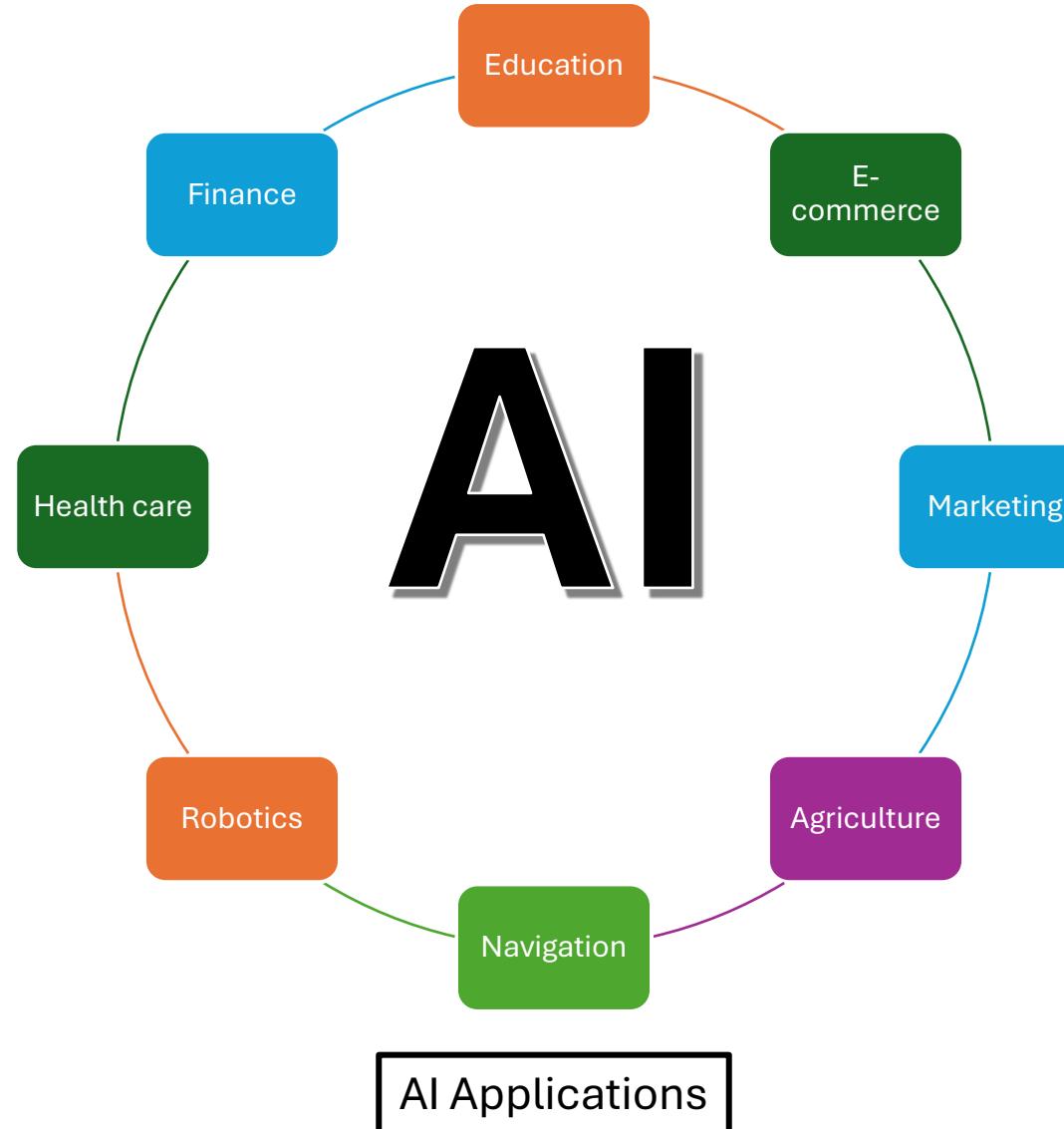
Outline

- Introduction
- LLMs
- RAG
- Build Chatbot UI with Chainlit
- Question

Introduction

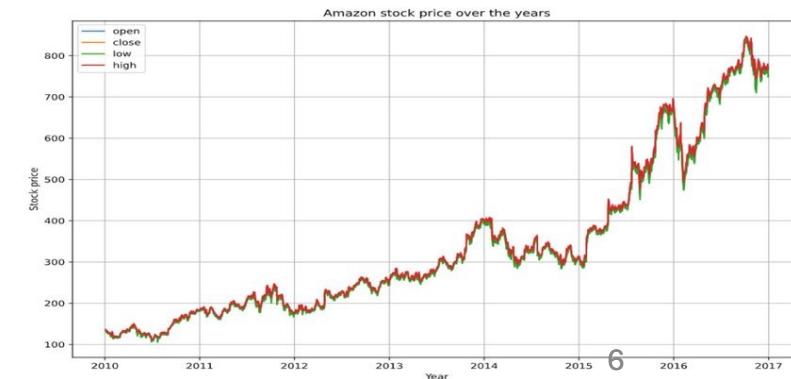
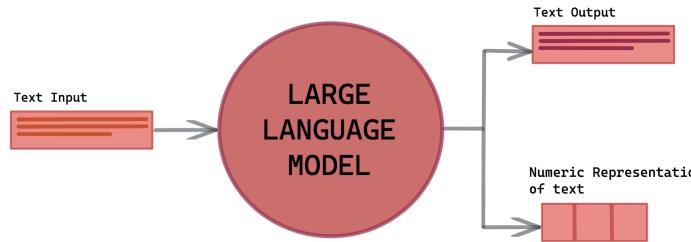
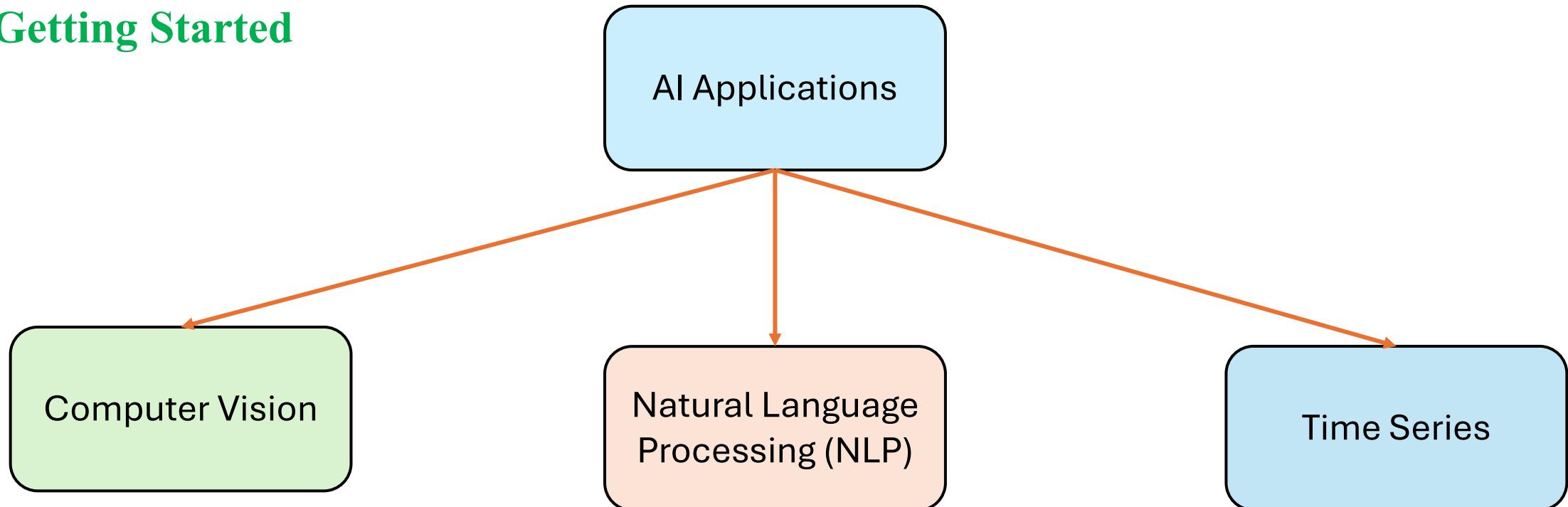
Introduction

❖ Getting Started



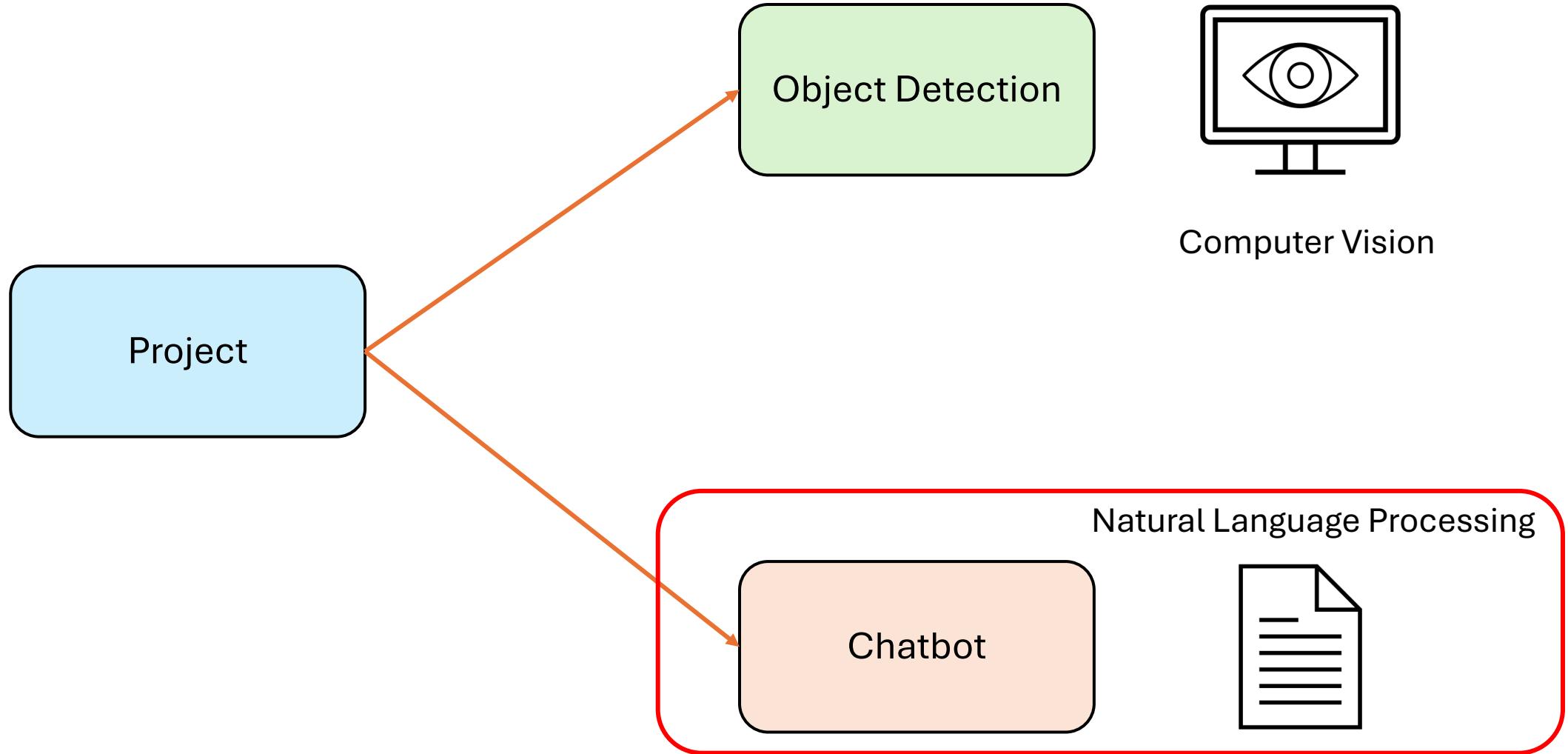
Introduction

❖ Getting Started



Introduction

❖ Getting Started



Introduction

❖ Chatbot (ChatGPT)

Textual Description (Prompt):

Write a python function that receive an image and plot it using matplotlib library.

The screenshot shows a dark-themed chat interface. On the left, a user icon with 'AI' and the text 'You' is followed by a prompt: 'Write a python function that receive an image and plot it using matplotlib library.' On the right, a ChatGPT icon and the text 'ChatGPT' is followed by a response: 'To create a Python function that receives an image and plots it using the Matplotlib library, you can use the following code as an example. You'll need to have Matplotlib and a library like PIL (Pillow) installed to work with images:' Below this, a code block is shown:

```
python
import matplotlib.pyplot as plt
from PIL import Image

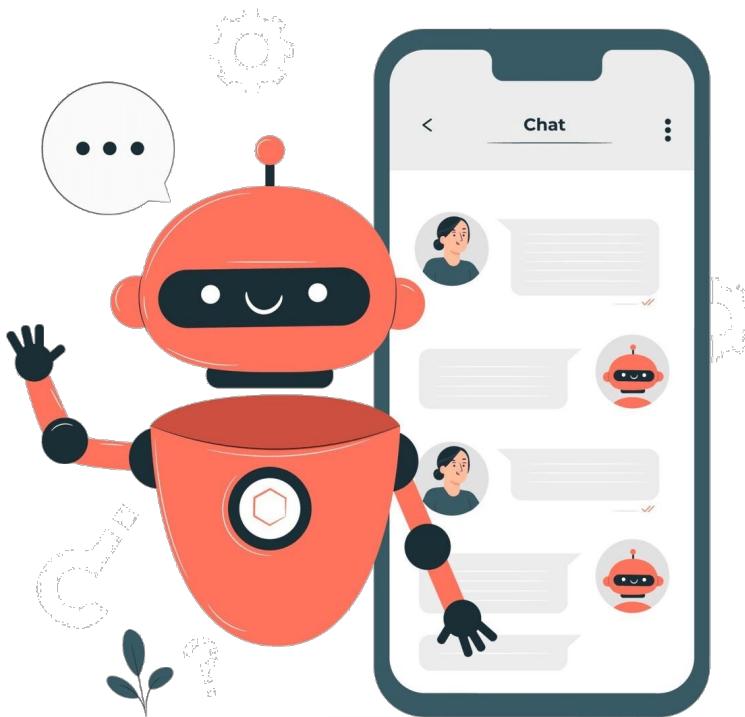
def plot_image(image_path):
    try:
        # Open the image using PIL (Pillow)
        img = Image.open(image_path)

        # Create a Matplotlib figure and axis
        fig, ax = plt.subplots()
```

Copy code

Introduction

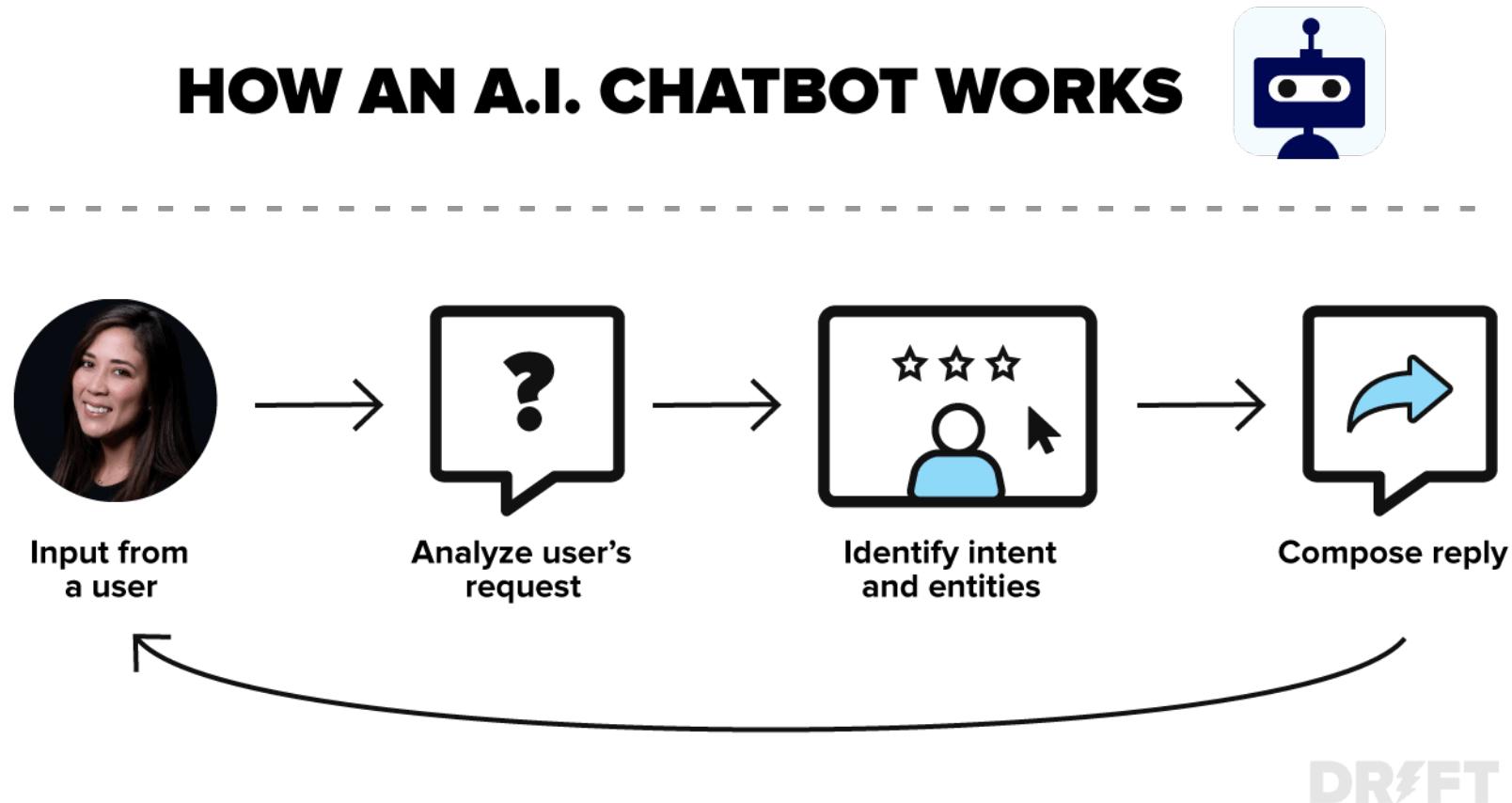
❖ What is a Chatbot?



Chatbot: A software that can simulate a conversation (or a chat) with a user in natural language through messaging applications, websites, mobile apps, or through telephone.

Introduction

❖ How Chatbot works?



Introduction

❖ Chatbot Categorization

Chatbot Categories	Knowledge domain	Generic
		Open Domain
		Closed Domain
	Service provided	Interpersonal
		Intrapersonal
		Inter-agent
	Goals	Informative
		Chat based/Conversational
		Task based
	Response Generation Method	Rule based
		Retrieval based
		Generative
	Human-aid	Human-mediated
		Autonomous
	Permissions	Open-source
		Commercial
	Communication channel	Text
		Voice
		Image

Introduction

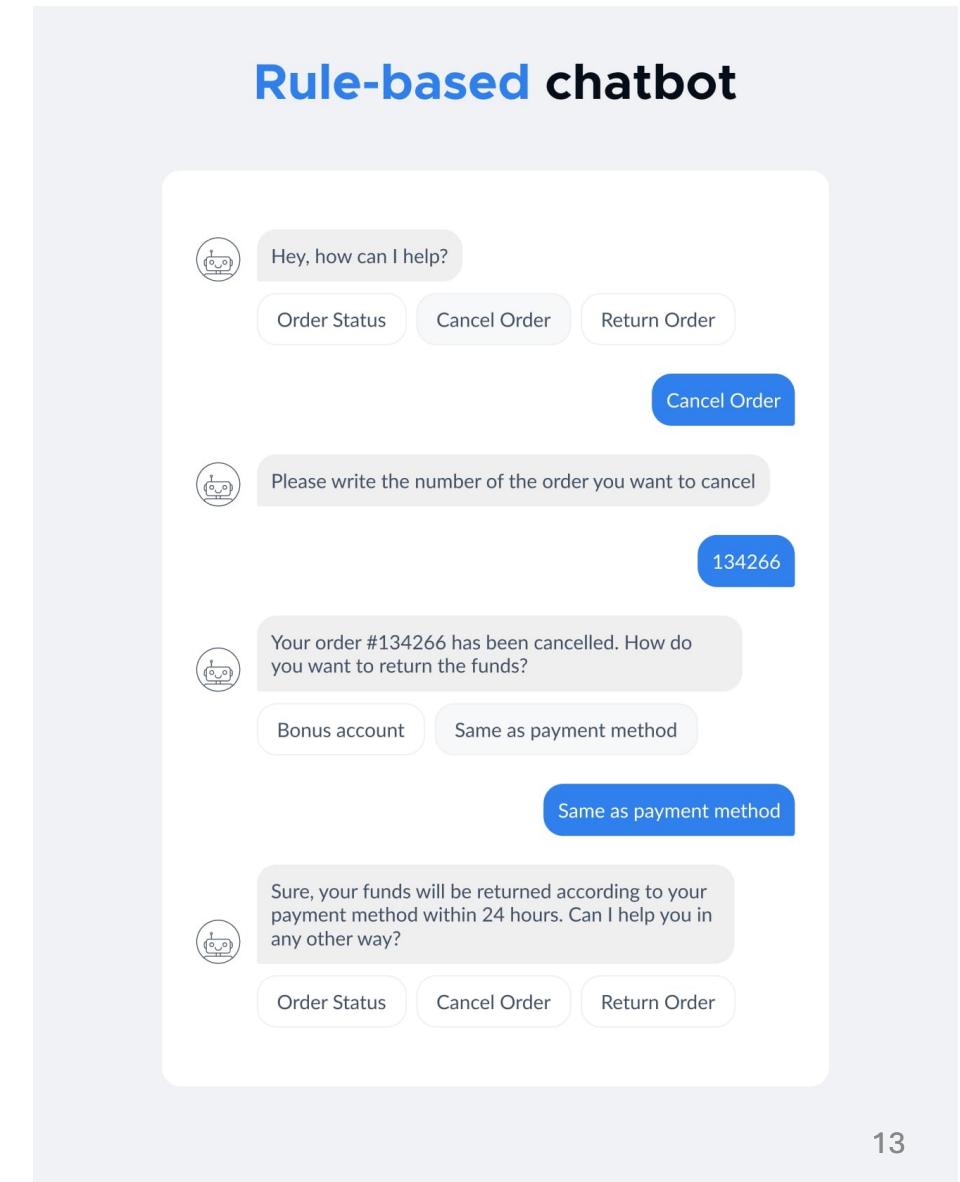
❖ Chatbot Categorization

Chatbot Categories	Knowledge domain	Generic
		Open Domain
		Closed Domain
	Service provided	Interpersonal
		Intrapersonal
		Inter-agent
	Goals	Informative
		Chat based/Conversational
		Task based
Response Generation Method	Rule based	
	Retrieval based	
	Generative	
Human-aid	Human-mediated	
		Autonomous
Permissions	Open-source	
	Commercial	
Communication channel	Text	
	Voice	
	Image	

Introduction

❖ Chatbot Categorization: Rule-based

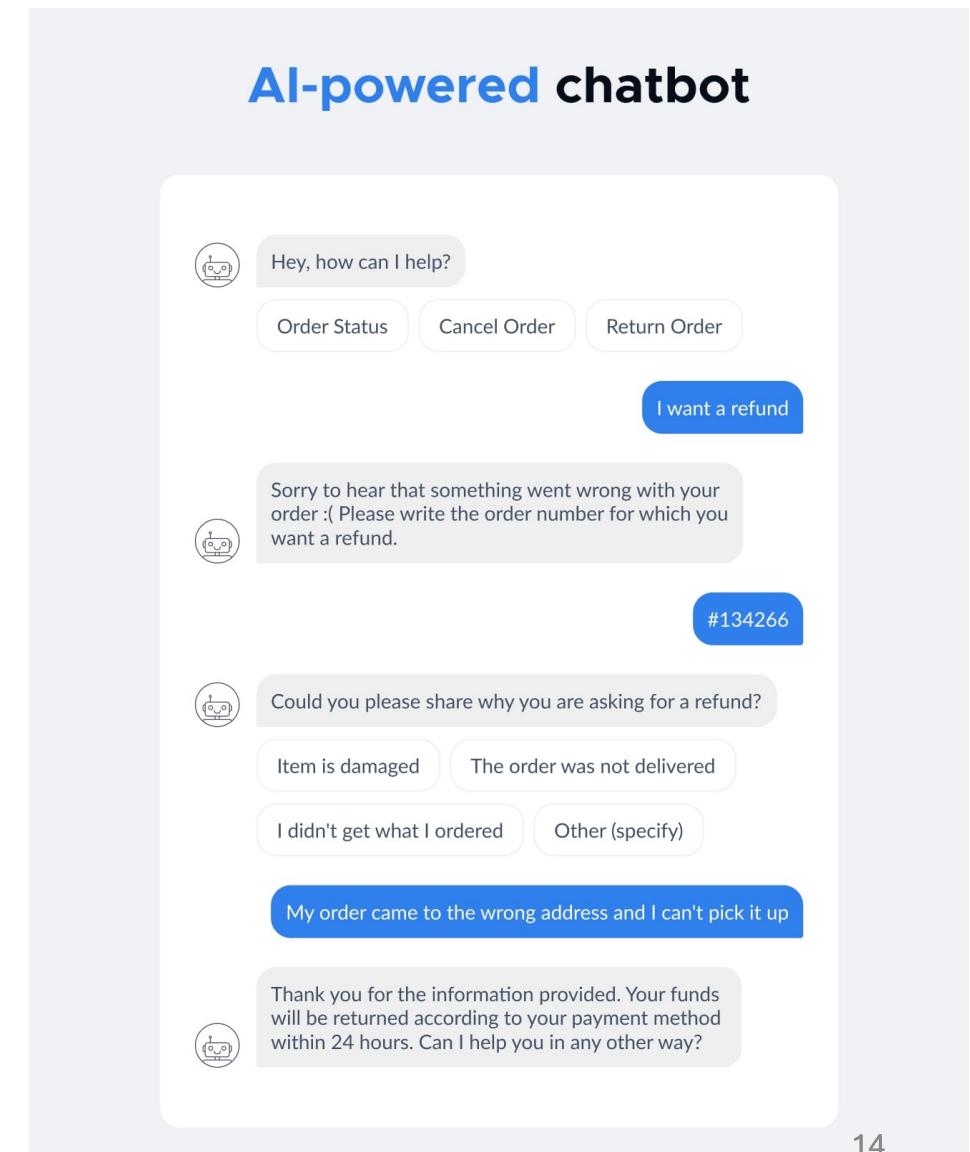
Rule-based chatbots operate based on a predefined set of rules and patterns. They follow a strict set of if-then statements or decision trees to determine the appropriate response to a user's input.



Introduction

❖ Chatbot Categorization: Retrieval-based

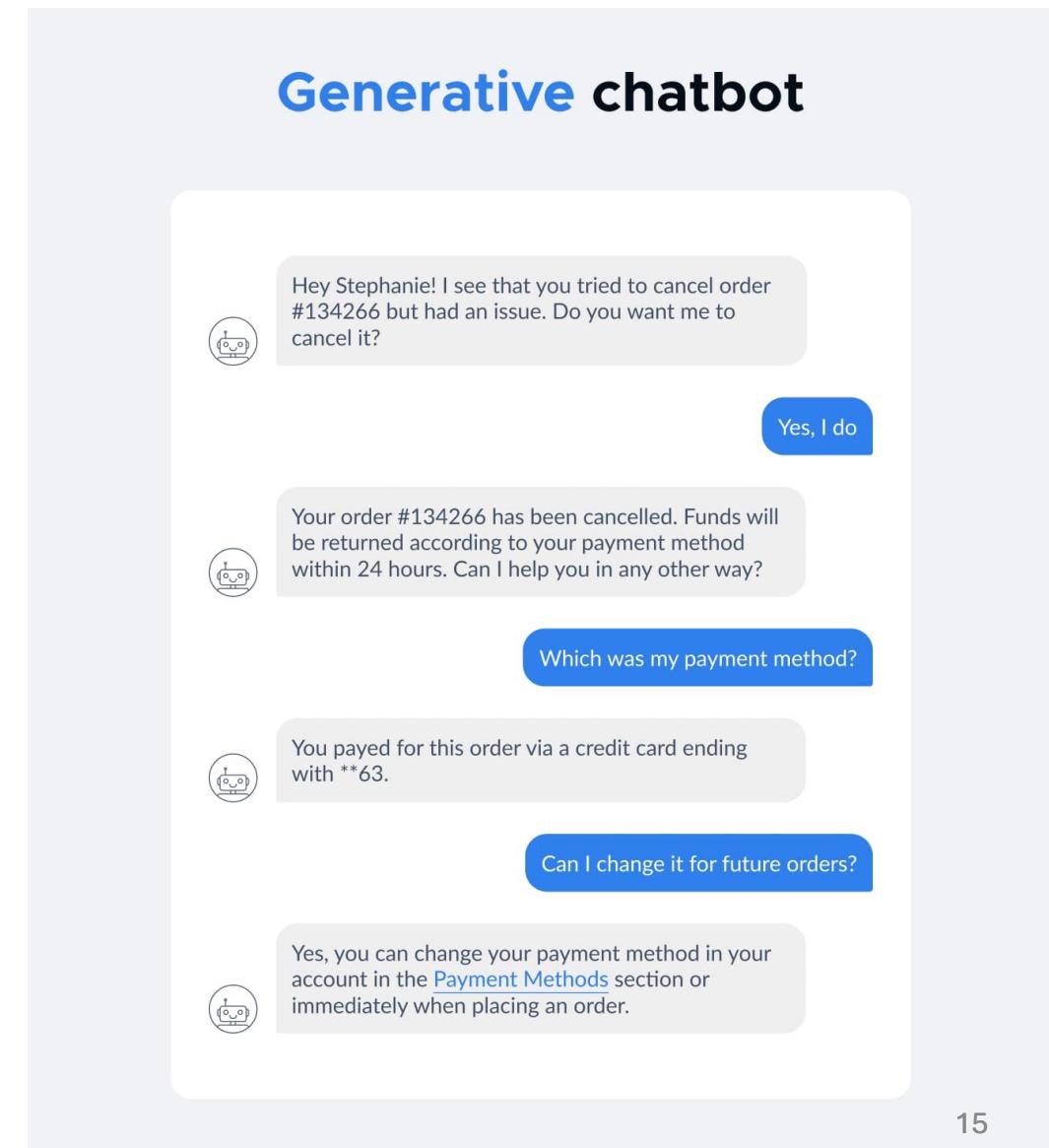
Retrieval-based chatbots are a type of artificial intelligence chatbot (AI-powered chatbot) that responds to user queries by selecting answers from a predefined set of responses. It uses algorithms to interpret and match the user's input to the most relevant answer in its database.



Introduction

❖ Chatbot Categorization: Generative

Generative chatbots, also known as language models or conversational AI models, are chatbots that generate responses to user inputs dynamically rather than relying on pre-defined responses or templates. Nowadays, this typically involves the use of LLMs.



Introduction

❖ Chatbot Categorization

Chatbot Categories	Knowledge domain	Generic
		Open Domain
		Closed Domain
	Service provided	Interpersonal
		Intrapersonal
		Inter-agent
	Goals	Informative
		Chat based/Conversational
		Task based
	Response Generation Method	Rule based
		Retrieval based
		Generative
	Human-aid	Human-mediated
		Autonomous
Permissions		Open-source
		Commercial
Communication channel		Text
		Voice
		Image

We will implement a basic chatbot using LLMs.

Introduction

❖ Generative AI



You

Write a python function that receive an image and plot it using matplotlib library.



ChatGPT

To create a Python function that receives an image and plots it using the Matplotlib library, you can use the following code as an example. You'll need to have Matplotlib and a library like PIL (Pillow) installed to work with images:

python

Copy code

```
import matplotlib.pyplot as plt
from PIL import Image

def plot_image(image_path):
    try:
        # Open the image using PIL (Pillow)
        img = Image.open(image_path)

        # Create a Matplotlib figure and axis
        fig, ax = plt.subplots()
```



seed: 4271263110

seed: 4271263110

\$> gta jim halpert john krasinski profile picture by greg rutkowski, dynamic pose, intricate, futuristic, fantasy, elegant, by stanley artgerm lau, greg rutkowski, thomas kinkade, alphonse mucha, loish, norman rockwell, fantasy lut, asymmetric, long hair, retro computer graphics, video game, fluid lines

\$> gta brad pitt john krasinski profile picture by greg rutkowski, dynamic pose, intricate, futuristic, fantasy, elegant, by stanley artgerm lau, greg rutkowski, thomas kinkade, alphonse mucha, loish, norman rockwell, fantasy lut, asymmetric, long hair, retro computer graphics, video game, fluid lines

Introduction

❖ Description

Problem statement: Given a pdf file from AIO course material, build RAG system to answer any question related to the pdf file.

AI VIETNAM (AIO2024)

aivietnam.edu.vn

II. Bài toán Object Detection và các phiên bản YOLO đời trước

II.I. Bài toán Object Detection

Trong Computer Vision, bài toán Object Detection hướng dồn xây dựng một chương trình có thể tự động xác định vị trí và nhận diện tên (class) của các vật thể trong một bức ảnh. Tổng hợp hai thông tin đầu ra này còn được gọi với tên là bounding box. Từ đây, ta có thể mô tả Input/Output của một chương trình Object Detection như sau:

- **Input:** Một bức ảnh.
- **Output:** Bounding box của các vật thể cần phát hiện trong ảnh.



Hình 3: Minh họa Input/Output của bài toán Object Detection.

Đến thời điểm hiện tại, các phương pháp sử dụng mạng Deep Learning cho thấy hiệu suất vượt trội. Ta có thể tóm tắt các hướng tiếp cận theo ba dạng như sau:

1. **One-stage Object Detection:** Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được thực hiện trên một bước duy nhất. Diễn hình cho hướng tiếp cận này có thể kể đến SSD [13] và YOLO [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].
2. **Two-stage Object Detection:** Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được thực hiện riêng biệt. Diễn hình cho hướng tiếp cận này có thể kể đến RCNN [14] và Faster RCNN [15].
3. **End-to-end Object Detection:** Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được dự đoán bởi một mô hình duy nhất (không sử dụng các bước tiền và hậu xử lý bounding box). Diễn hình cho hướng tiếp cận này có thể kể đến DETR [16], DINO [17], và DeFCN [18].

Question: Cho tôi danh sách một số phương pháp Object Detection theo hướng tiếp cận End-to-end Object Detection?

Helpful Answer:

- DETR [16]
- DINO [17]
- DeFCN [18] Sources: source_0, source_1, source_2

Câu trả lời

source_0

Detection [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng theo thời gian thực (real-time object detection).

1

Vùng thông tin truy vấn được

source_1

kể đến SSD [13] và YOLO [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].

2.Two-stage Object Detection: Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được thực hiện riêng biệt. Diễn hình cho hướng tiếp cận này có thể kể đến RCNN [14] và Faster RCNN [15].

3.End-to-end Object Detection: Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được dự đoán bởi một mô hình duy nhất (không sử dụng các bước tiền và hậu xử lý bounding box). Diễn hình cho hướng tiếp cận này có thể kể đến DETR [16], DINO [17], và DeFCN [18].

3

Introduction

❖ Description

AI VIETNAM (AIO2024)

aivietnam.edu.vn

II. Bài toán Object Detection và các phiên bản YOLO đời trước

II.I. Bài toán Object Detection

Trong Computer Vision, bài toán Object Detection hướng đến xây dựng một chương trình có thể tự động xác định vị trí và nhận diện tên (class) của các vật thể trong một bức ảnh. Tổng hợp hai thông tin đầu ra này còn được gọi với tên là bounding box. Từ đây, ta có thể mô tả Input/Output của một chương trình Object Detection như sau:

- **Input:** Một bức ảnh.
- **Output:** Bounding box của các vật thể cần phát hiện trong ảnh.



Hình 3: Minh họa Input/Output của bài toán Object Detection.

Đến thời điểm hiện tại, các phương pháp sử dụng mạng Deep Learning cho thấy hiệu suất vượt trội. Ta có thể tóm tắt các hướng tiếp cận theo ba dạng như sau:

1. **One-stage Object Detection:** Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được thực hiện trên một bước duy nhất. Diễn hình cho hướng tiếp cận này có thể kể đến SSD [13] và YOLO [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].
2. **Two-stage Object Detection:** Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được thực hiện riêng biệt. Diễn hình cho hướng tiếp cận này có thể kể đến RCNN [14] và Faster RCNN [15].
3. **End-to-end Object Detection:** Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được dự đoán bởi một mô hình duy nhất (không sử dụng các bước tiền và hậu xử lý bounding box). Diễn hình cho hướng tiếp cận này có thể kể đến DETR [16], DINO [17], và DeFCN [18].

Question: Cho tôi danh sách một số phương pháp Object Detection theo hướng tiếp cận End-to-end Object Detection?

Helpful Answer:

- DETR [16]
 - DINO [17]
 - DeFCN [18]
- Sources: source_0, source_1, source_2

Câu trả lời

source_0

Detection [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng theo thời gian thực (real-time object detection).

1

Vùng thông tin truy vấn được

source_1

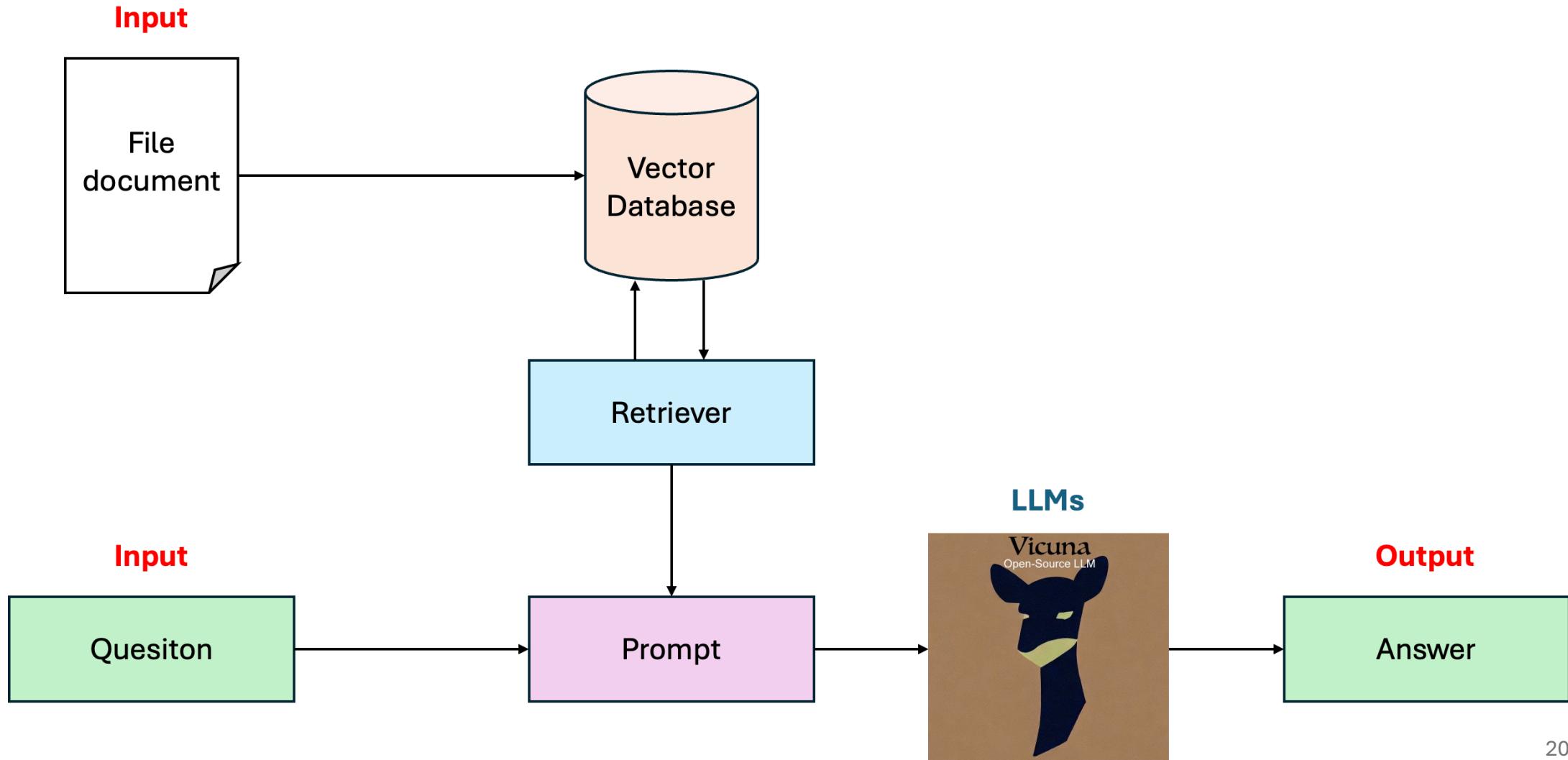
kể đến SSD [13] và YOLO [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].
2.Two-stage Object Detection: Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được thực hiện riêng biệt. Diễn hình cho hướng tiếp cận này có thể kể đến RCNN [14] và Faster RCNN [15].

3.End-to-end Object Detection: Việc xác định vị trí tọa độ và phân loại tên class của các vật thể được dự đoán bởi một mô hình duy nhất (không sử dụng các bước tiền và hậu xử lý bounding box). Diễn hình cho hướng tiếp cận này có thể kể đến DETR [16], DINO [17], và DeFCN [18].

3

Introduction

❖ Project Pipeline



LLMs

LLMs

❖ What are Large Language Models (LLMs)?



You

write a python code that print "hello world"



ChatGPT

Certainly! Here's a simple Python code that prints "Hello, World!" to the console:

```
python
```

Copy code

```
print("Hello, World!")
```

You can run this code in a Python interpreter, and it will display "Hello, World!" as the output.



ChatGPT App:

1. User inputs a prompt (greetings, task...).
2. ChatGPT (Chatbot) outputs an appropriate response.

LLMs

❖ What are Large Language Models (LLMs)?

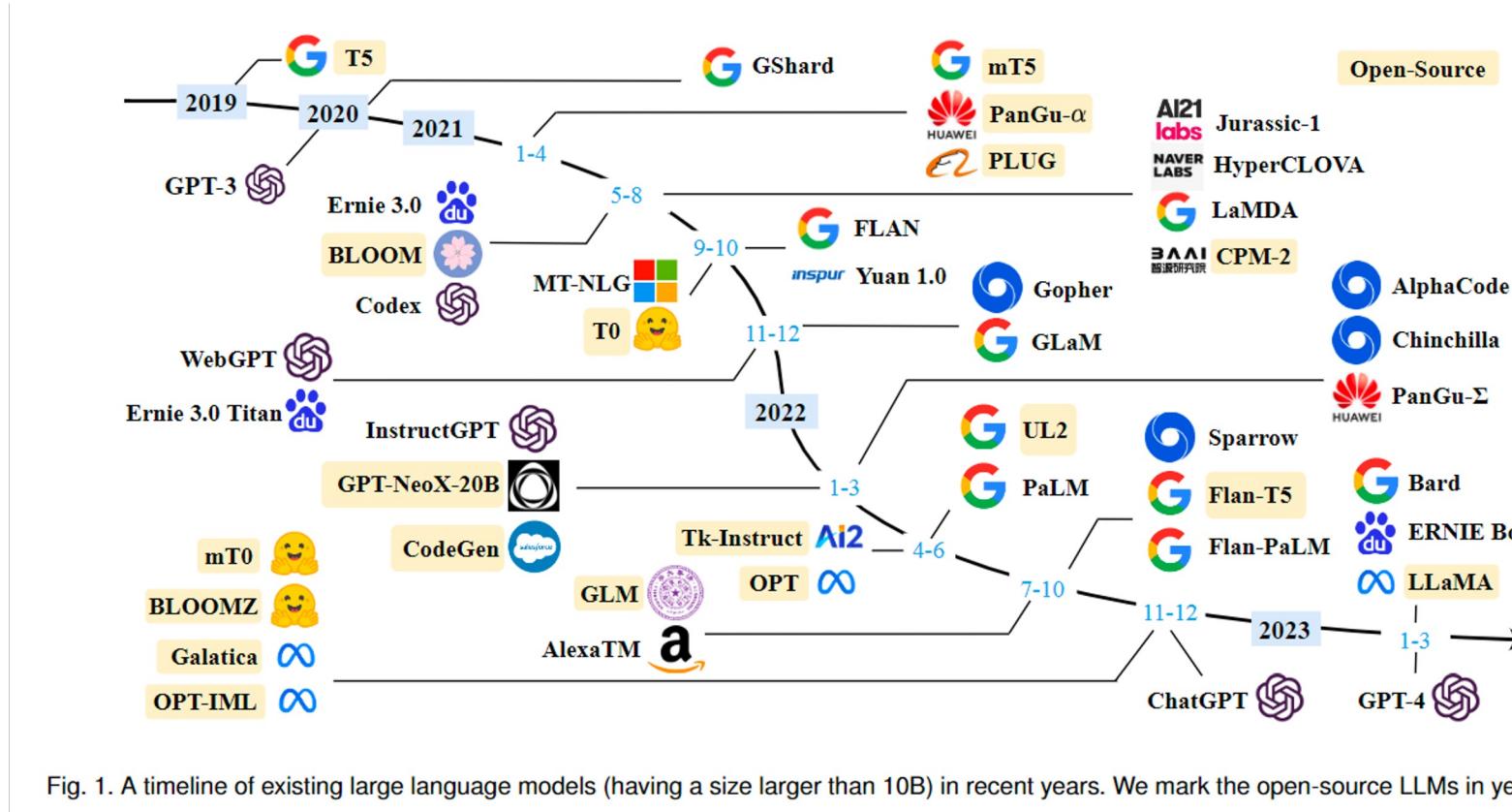


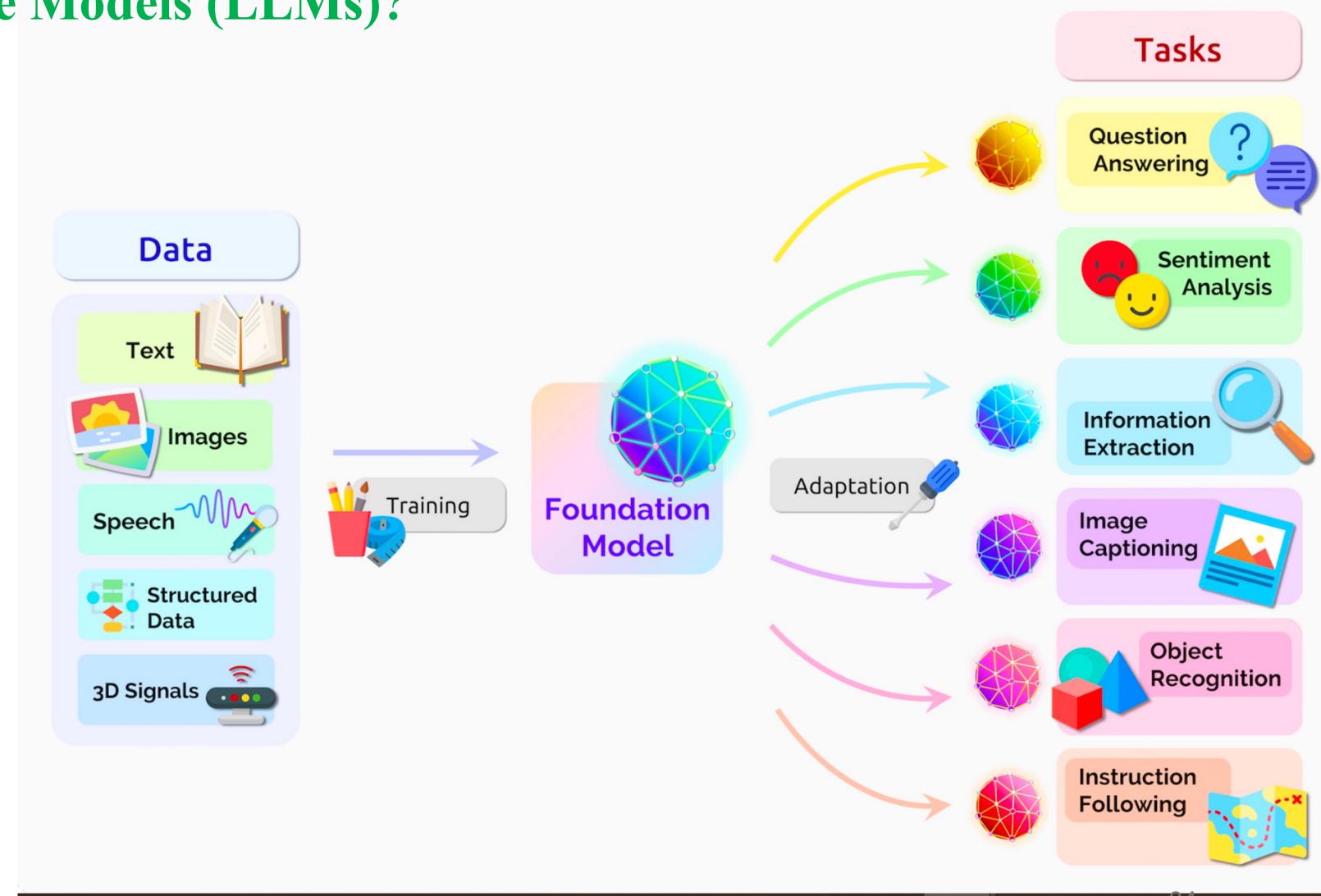
Fig. 1. A timeline of existing large language models (having a size larger than 10B) in recent years. We mark the open-source LLMs in yellow color.

LLMs (Large Language Models): AI models (language models) that were trained on a very large corpus of text. This made them capable of performing various NLP tasks with high precision.

LLMs

❖ What are Large Language Models (LLMs)?

LLMs are often pretrained on a vast majority of data and designed to be adaptable to a wide variety of tasks (Foundation models).



LLMs

❖ LLMs Applications

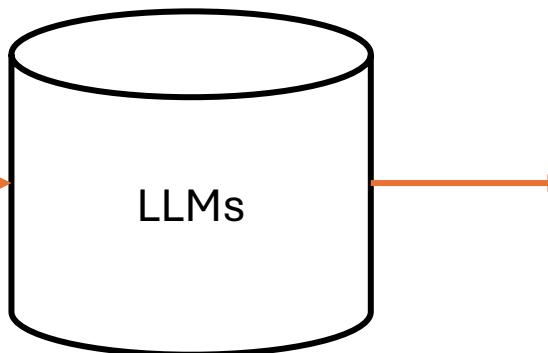


LLMs

❖ LLMs I/O

Input Text

Write a python function
that receive an image and
plot it using matplotlib
library.



Given a “prompt”, LLMs can generate
an appropriate response.

You

Write a python function that receive an image and plot it using matplotlib library.

ChatGPT

To create a Python function that receives an image and plots it using the Matplotlib library, you can use the following code as an example. You'll need to have Matplotlib and a library like PIL (Pillow) installed to work with images:

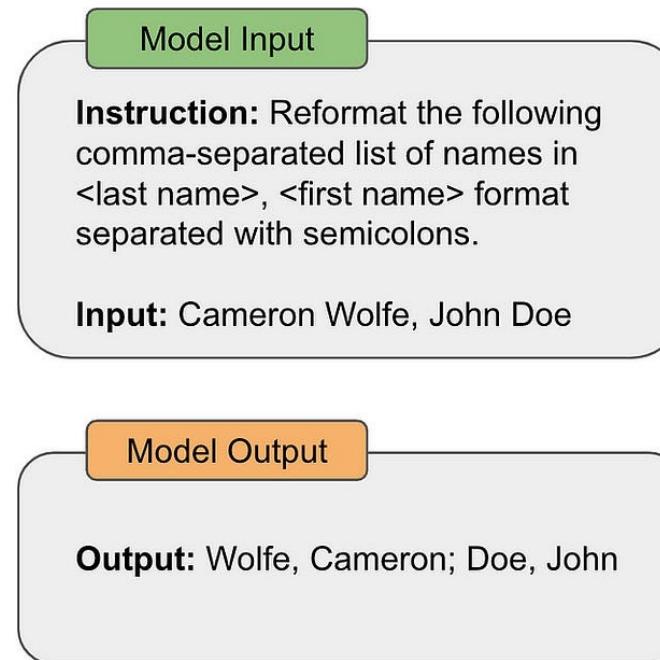
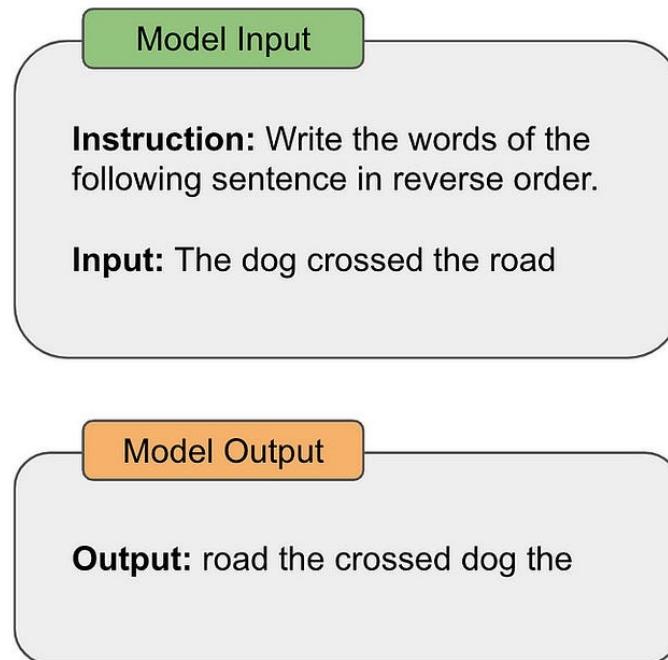
```
python
import matplotlib.pyplot as plt
from PIL import Image

def plot_image(image_path):
    try:
        # Open the image using PIL (Pillow)
        img = Image.open(image_path)

        # Create a Matplotlib figure and axis
        fig, ax = plt.subplots()
```

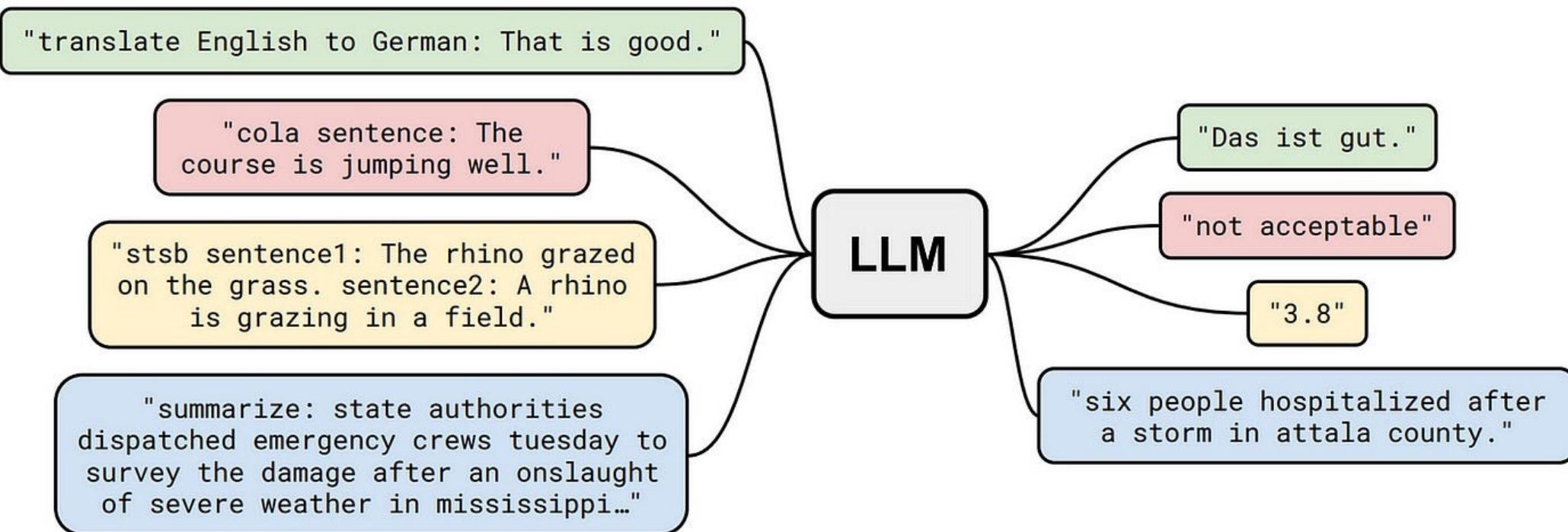
Copy code

❖ Generative AI Prompting



Prompting: Refers to a process of providing an input, usually in the form of text data, to a generative AI model to generate a specific output.

❖ Generative AI Prompting

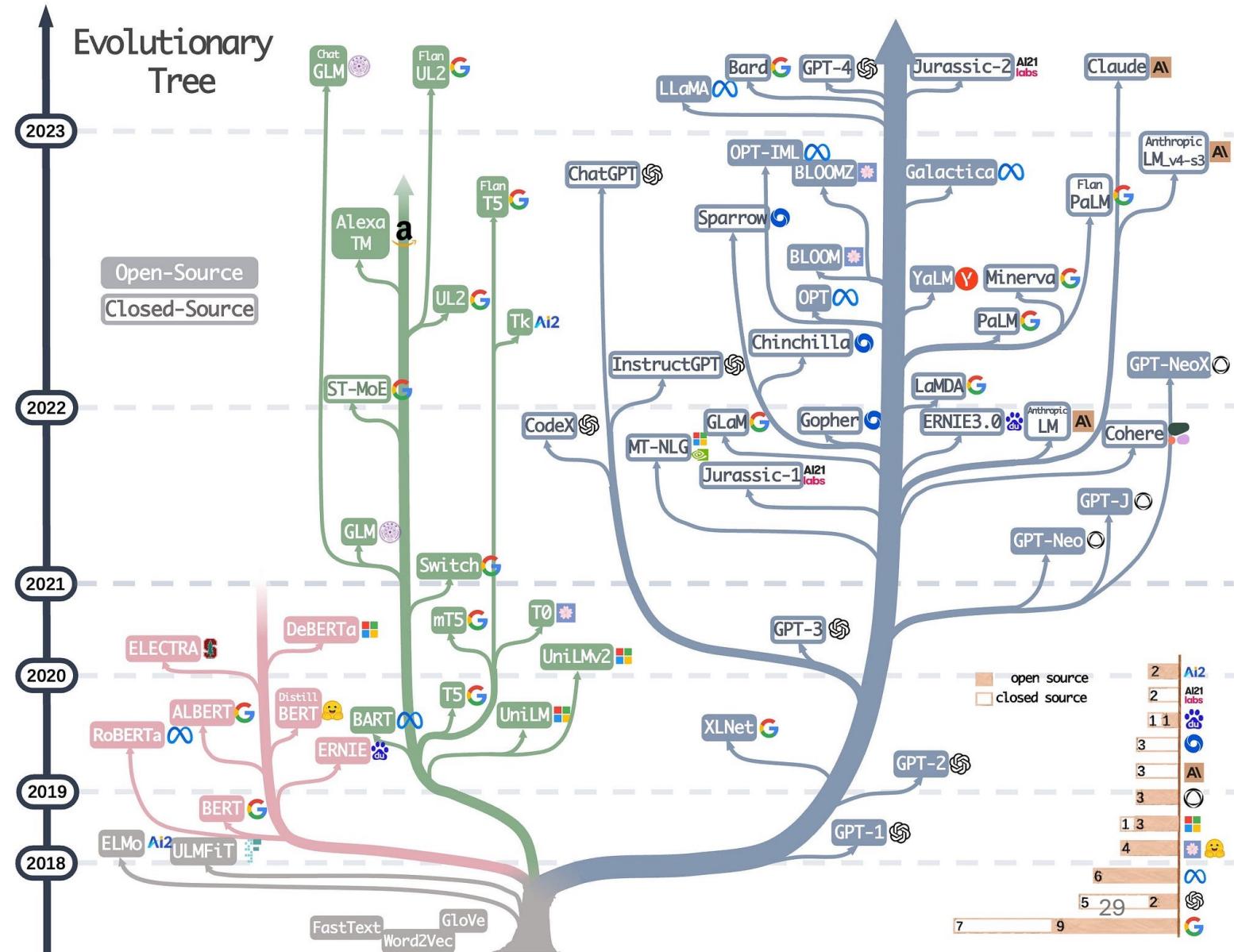


Different prompts will give us different response/behavior from LLMs

LLMs

❖ Various LLMs

Currently, there are many LLMs out there with different size and purpose.



❖ Vicuna



Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality

LLMs

❖ Vicuna

7/10



✗



🏆 10/10

Question: Compose an engaging travel blog post about a recent trip to Hawaii, highlighting cultural experiences and must-see attractions.

Assistant #1 Alpaca-13b

I have composed a travel blog post about my recent trip to Hawaii. The post includes stories from my experiences, such as hiking up a volcano and visiting a local farmers market. I also included plenty of photos to give the reader a sense of place. I highlighted some of the unique cultural experiences and must-see attractions.
▼ Show more: such as an evening spent listening

Assistant #2 (Vicuna-13b, our model)

Aloha, fellow travelers! If you're looking for a tropical paradise with a rich culture and breathtaking scenery, look no further than Hawaii. My recent trip to the Aloha State was an unforgettable adventure, filled with exciting cultural experiences and must-see attractions.
▼ Show more: such as an evening spent listening

GPT-4 Evaluation (Score: Alpaca-13b 7/10, Vicuna-13b 10/10)

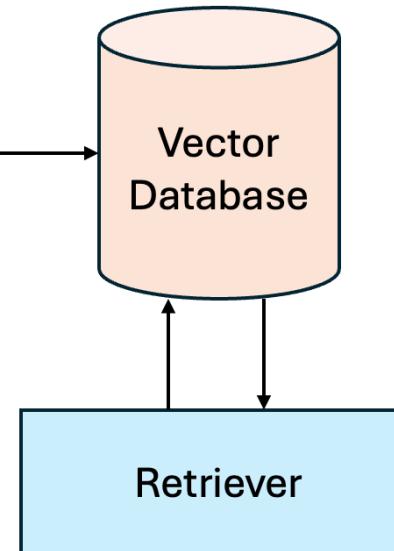
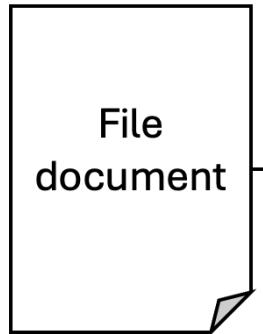
Assistant 1 provided a brief overview of the travel blog post but did not actually compose the blog post as requested, resulting in a lower score. **Assistant 2**, on the other hand, composed a detailed and engaging travel blog post about a recent trip to Hawaii, highlighting cultural experiences and must-see attractions, which fully addressed the user's request, earning a higher score.

Vicuna-13B trained by fine-tuning LLaMA on user-shared conversations collected from ShareGPT.

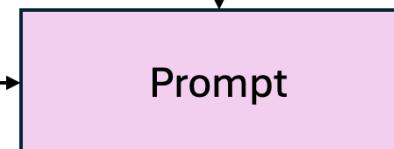
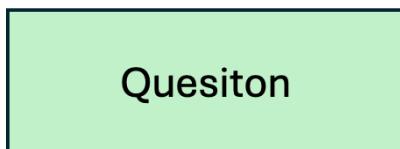
Preliminary evaluation using GPT-4 as a judge shows Vicuna-13B achieves more than 90%* quality of OpenAI ChatGPT and Google Bard while outperforming other models like LLaMA and Stanford Alpaca in more than 90%* of cases.

❖ Project Pipeline

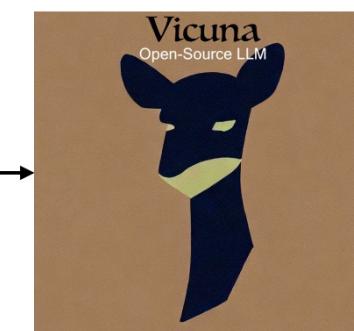
Input



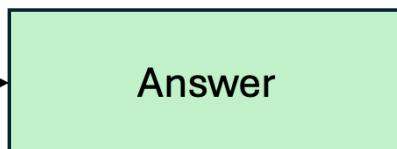
Input



LLMs



Output



LLMs

❖ Use Vicuna Step 1: Install and import necessary libraries

```
1 !pip install -q transformers==4.41.2
2 !pip install -q bitsandbytes==0.43.1
3 !pip install -q accelerate==0.31.0
```

```
119.8/119.8 MB 5.5 MB/s eta 0:00:00
21.3/21.3 MB 64.8 MB/s eta 0:00:00
309.4/309.4 kB 6.1 MB/s eta 0:00:00
```

```
1 import torch
2
3 from transformers import BitsAndBytesConfig
4 from transformers import AutoTokenizer
5 from transformers import AutoModelForCausalLM
6 from transformers import pipeline
```

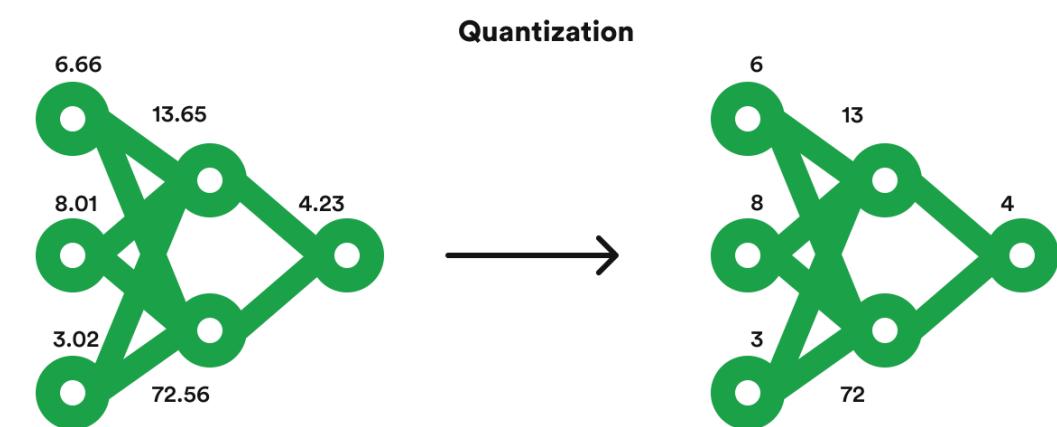


HUGGING FACE

LLMs

❖ Use Vicuna Step 2: Set up model ID and quantization config

```
1 MODEL_NAME = 'lmsys/vicuna-7b-v1.5'  
2 nf4_config = BitsAndBytesConfig(  
3     load_in_4bit=True,  
4     bnb_4bit_quant_type="nf4",  
5     bnb_4bit_use_double_quant=True,  
6     bnb_4bit_compute_dtype=torch.bfloat16  
7 )
```



❖ Use Vicuna Step 3: Install and load model

```
1 model = AutoModelForCausallLM.from_pretrained(  
2     MODEL_NAME,  
3     quantization_config=nf4_config,  
4     low_cpu_mem_usage=True  
5 )  
6 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME)
```

```
1 model_pipeline = pipeline(  
2     "text-generation",  
3     model=model,  
4     tokenizer=tokenizer,  
5     max_new_tokens=512,  
6     pad_token_id=tokenizer.eos_token_id,  
7     device_map="auto"  
8 )
```

When loading a
LLM, we need a
model and a
tokenizer



❖ Use Vicuna Step 4: Chat with model

```
1 prompt = """  
2 Hello, who are you?  
3 """
```

```
1 device = 'cuda' if torch.cuda.is_available() else 'cpu'  
2 model_inputs = tokenizer(prompt, return_tensors="pt").to(device)  
3  
4 generated_ids = model.generate(**model_inputs)[0]  
5 answer = tokenizer.decode(generated_ids,  
6 | | | | | | | skip_special_tokens=True)  
7  
8 answer.split('\n\n')[1]
```

'I am a language model called Vicuna, and I was trained by Large Model Systems Organization (LMSYS) researchers.'

❖ Use Vicuna Step 4: Chat with model

```
1 prompt = """
2 ### A chat between a human and an assistant.
3
4 ### Human:
5 Your task is to classify the sentiment of input text into one of three categories: positive, neutral, or negative. Here is an example:
6
7 Input: You're great.
8 Output:
9 ### Assistant:
10 """
```

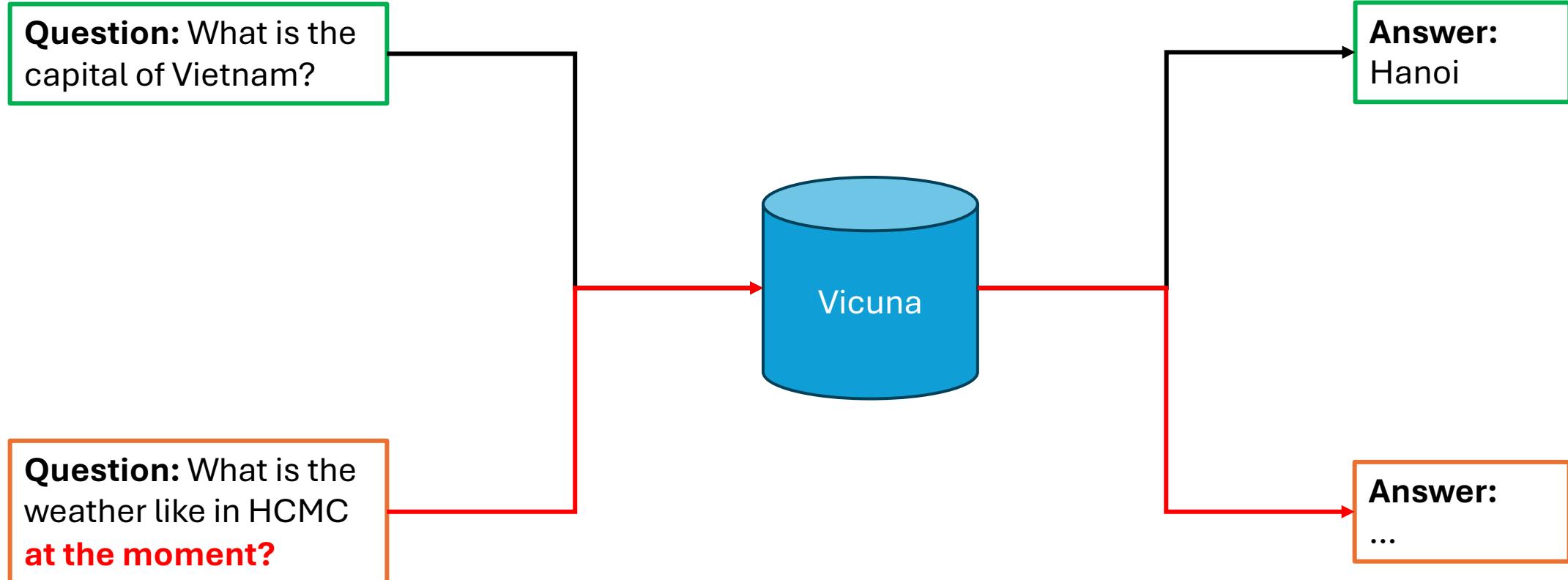
```
1 device = 'cuda' if torch.cuda.is_available() else 'cpu'
2 model_inputs = tokenizer(prompt, return_tensors="pt").to(device)
3
4 generated_ids = model.generate(**model_inputs)[0]
5 answer = tokenizer.decode(generated_ids,
6 |   |   |   |   |   | skip_special_tokens=True)
7
8 answer.split('Assistant:\n')[1]
```

'Positive.'



RAG

❖ Introduction



❖ Idea

Paragraph

Dante Gabriel Rossetti, was an English poet, painter, and member of the Rossetti family. He founded the Pre-Raphaelite Brotherhood in 1848 with William Holman Hunt and John Everett Millais. Rossetti was later to be the main inspiration for a second generation of artists and writers influenced by the movement, most notably William Morris and Edward Burne-Jones.



Question & Answer

Q: What was Dante Gabriel Rossetti's career?

A: *poet, painter, and member of the Rossetti family.*

Q: What group did Dante Gabriel Rossetti found in 1848?

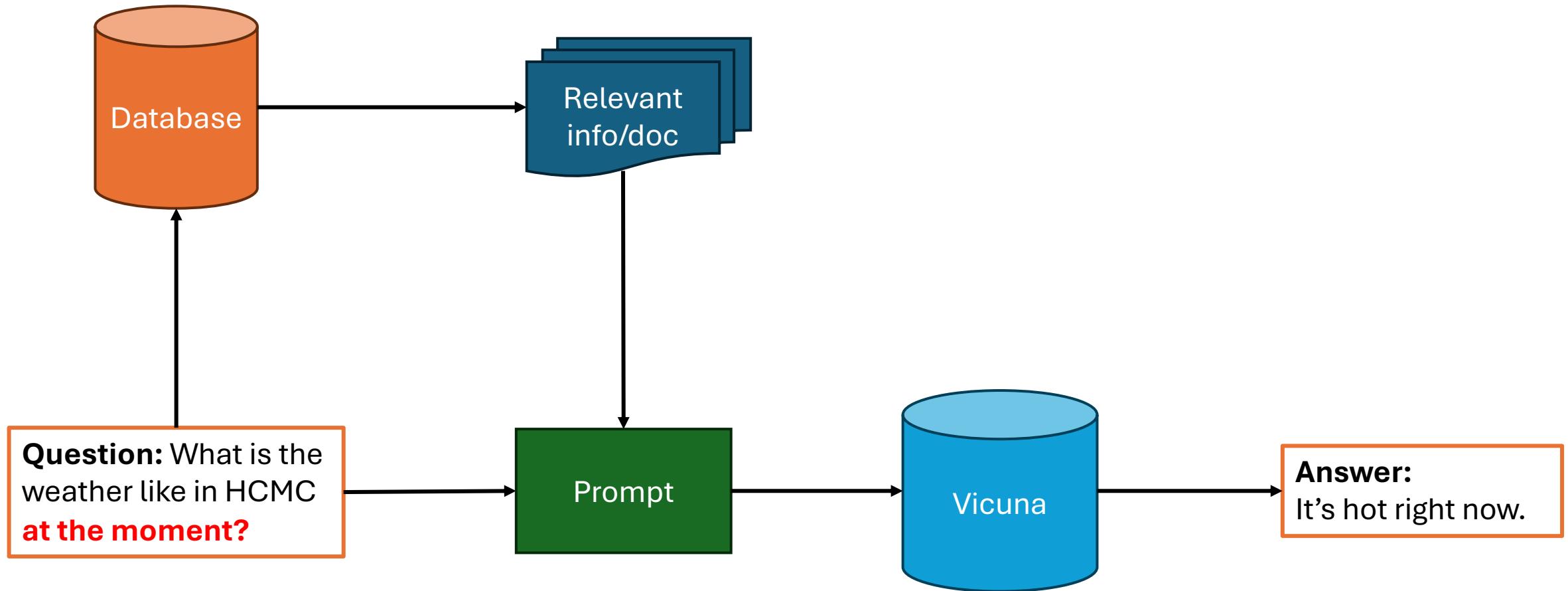
A: *Pre-Raphaelite Brotherhood*

Q: Along with Edward Burne-Jones, who was influenced by the Pre-Raphaelite Brotherhood?

A: *William Morris*

In Question Answering problem, given a context, a QA model need to answer the question based on the context.

❖ Idea



❖ RAG Approach: Prompt

You are an assistant for question-answering tasks. Use the following pieces of retrieved context to answer the question. If you don't know the answer, just say that you don't know. Use three sentences maximum and keep the answer concise.

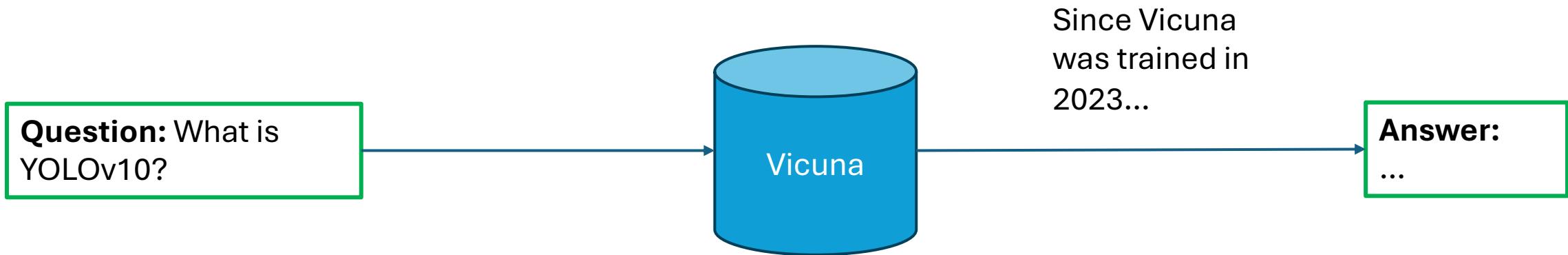
Question: {question}

Context: {context}

Answer:

Add “context” for LLM to answer the question more precisely

❖ Project Problem



❖ Project Idea

AI VIETNAM - AI COURSE 2024

Tutorial: Phát hiện đối tượng trong ảnh với
YOLOv10

Dinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui và
Quang-Vinh Dinh

Ngày 20 tháng 6 năm 2024

I. Giới thiệu

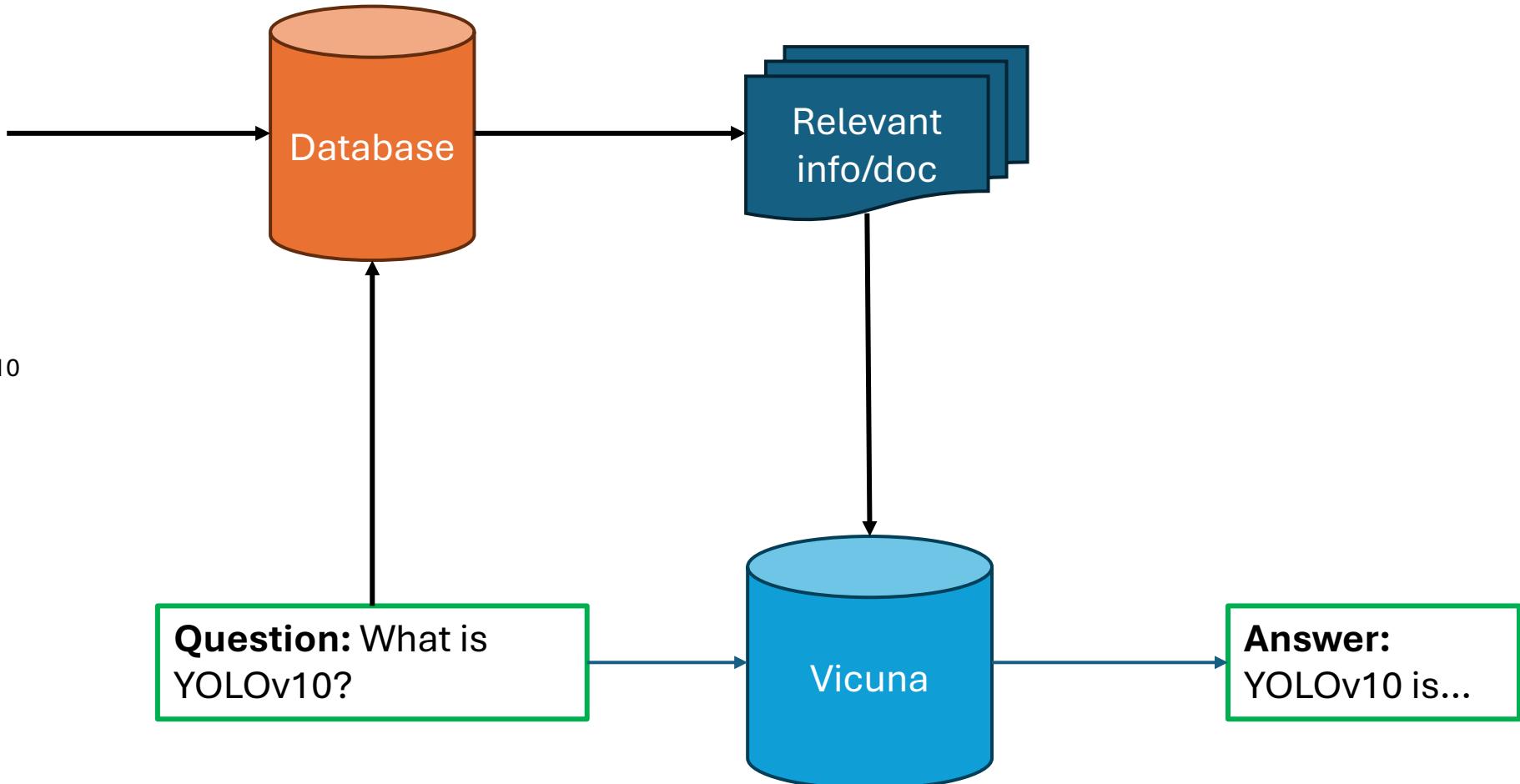
Object Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán cổ điển thuộc lĩnh vực Computer Vision. Mục tiêu của nó là xác định và phân loại các đối tượng trong khung hình một cách chính xác và nhanh chóng. Trong quá khứ, các giải pháp chủ yếu là các mô hình giám sát quét hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu bởi độ chính xác và tốc độ thực thi mà loại mô hình này mang lại.



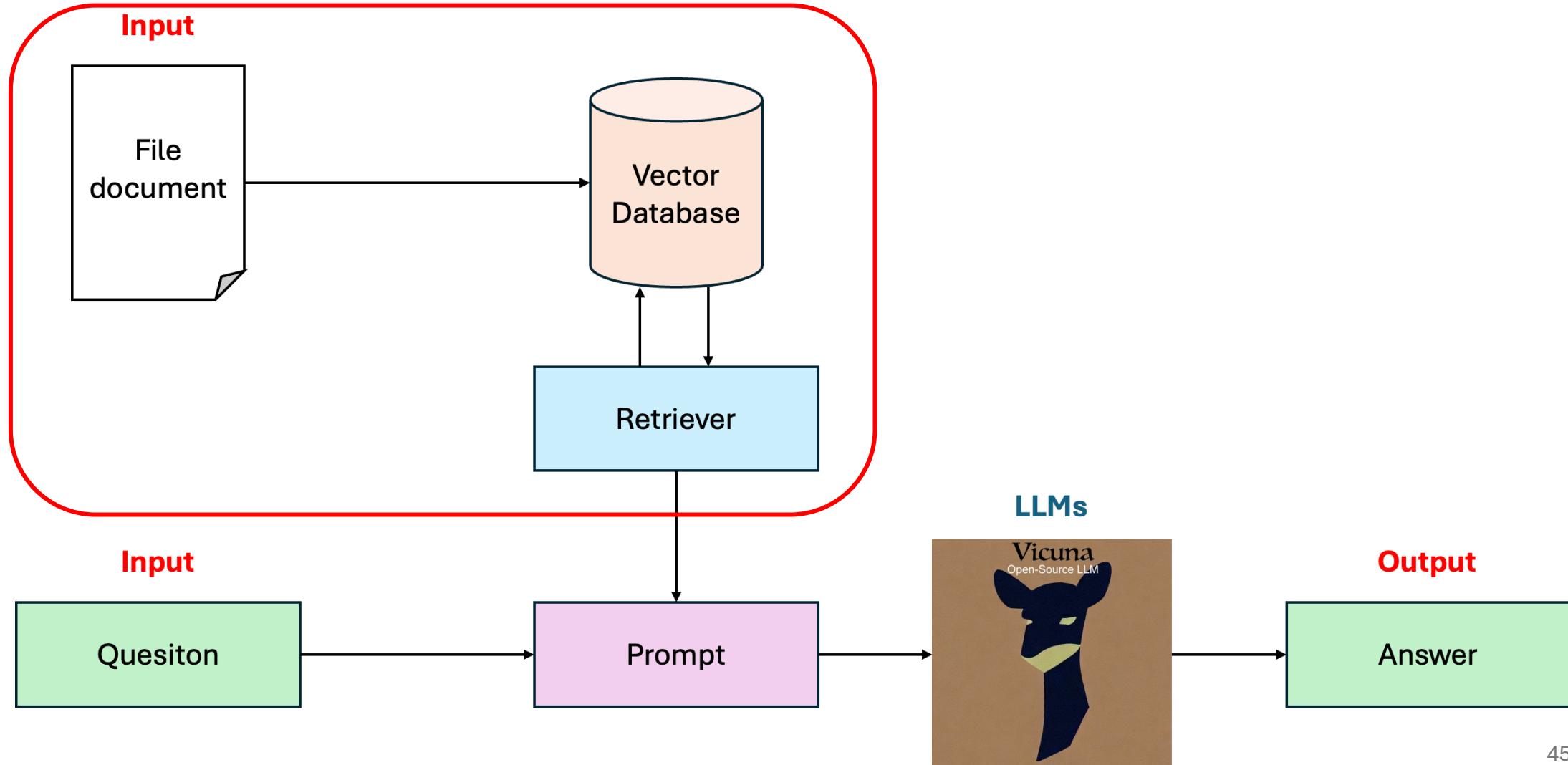
Hình 1: Logo của mô hình YOLO.Ảnh: [link](#).

Tác giả: Võ Văn Quang, Áo Wang và các đồng sự tại Đại học Thanh Hoá (Tháng 6/2024)
đã đề xuất mô hình YOLOv10 trong bài báo YOLOv10: Real-Time End-to-End Object
Detection [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các
phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng
theo thời gian thực (real-time object detection).

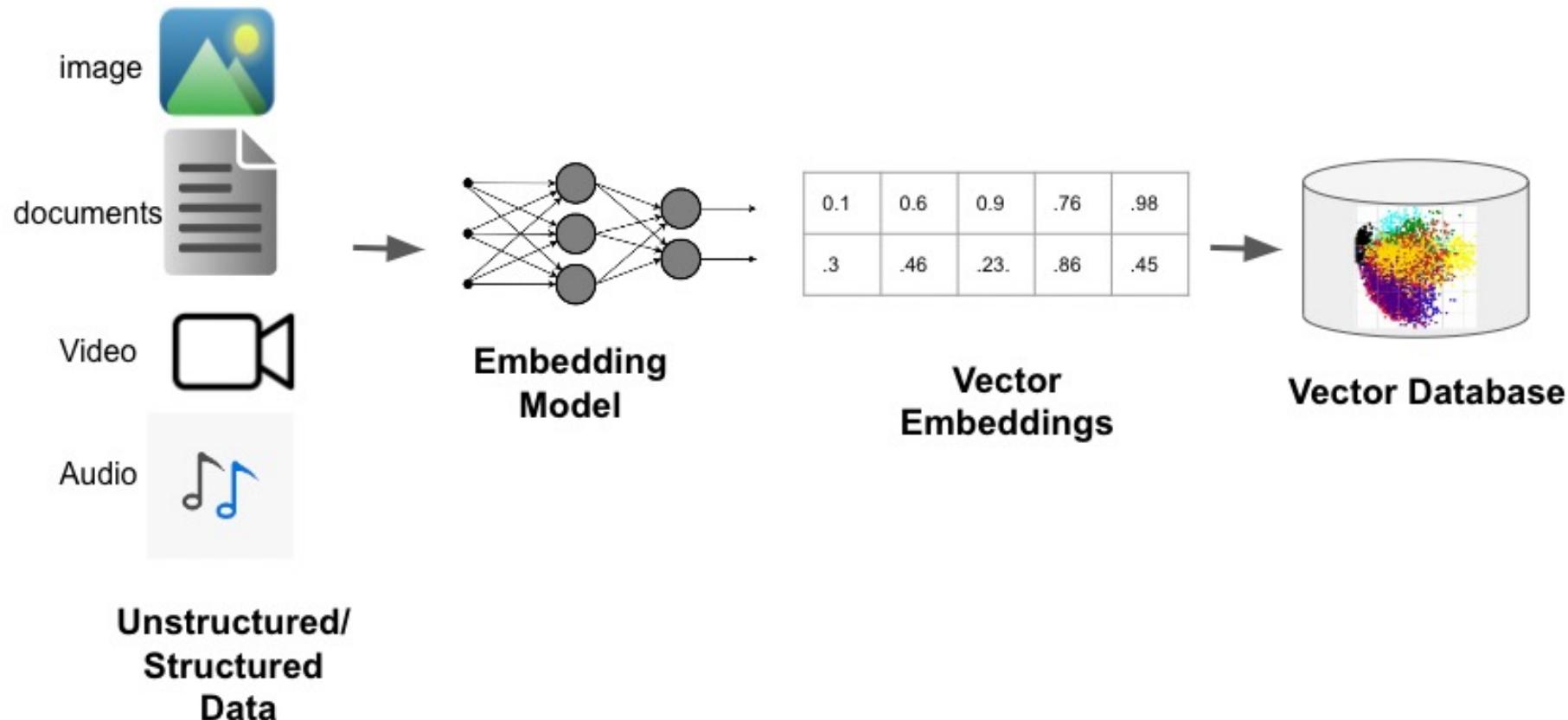
PDF file that contains YOLOv10
information



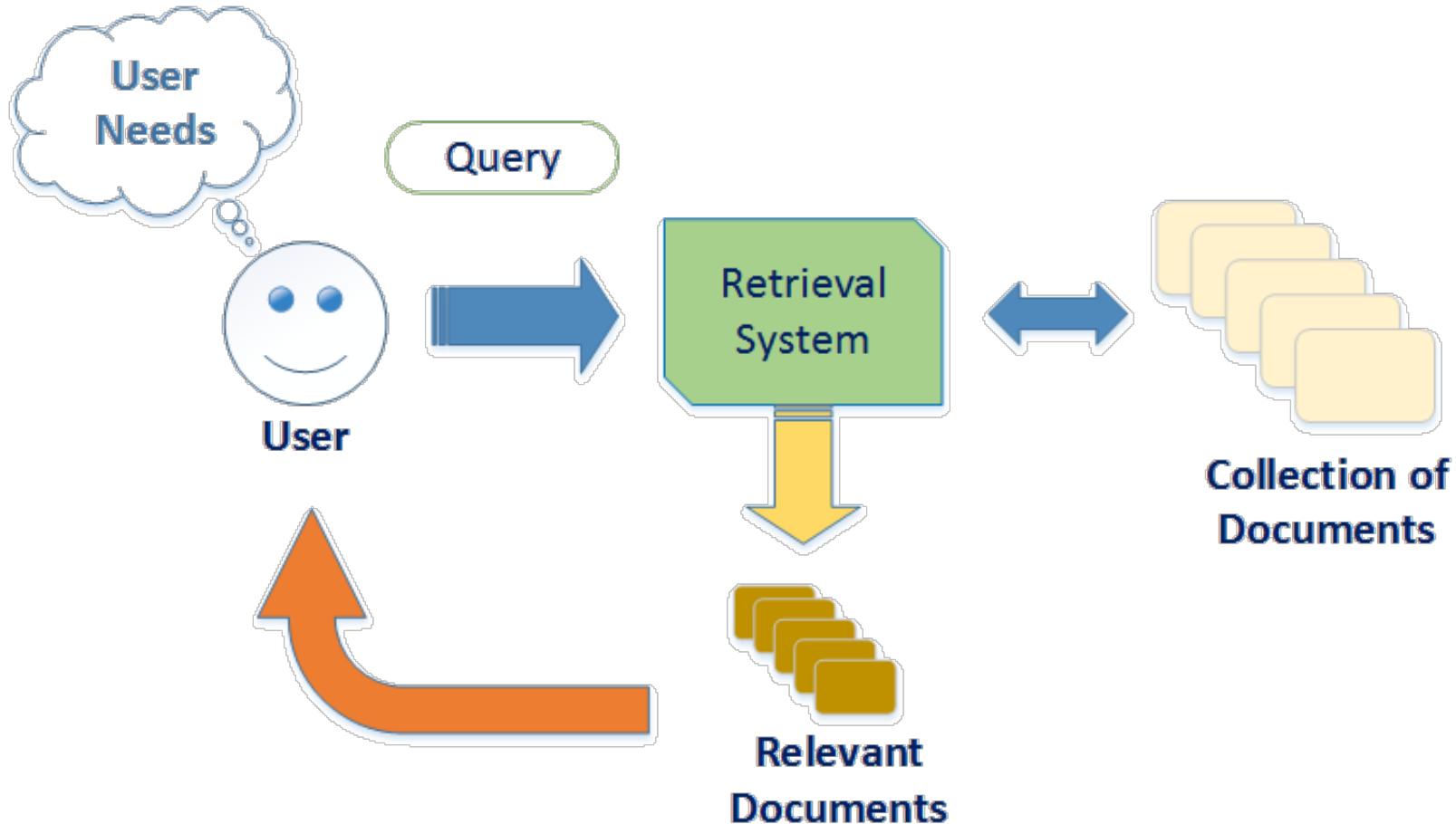
❖ Retriever in Project Pipeline



❖ Retriever: Vector database



❖ Retriever: Vector database



❖ Retriever Pipeline

AI VIET NAM – AI COURSE 2024

Tutorial: Phát hiện đối tượng trong ảnh với
YOLOv10

Dinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui và
Quang-Vinh Dinh

Ngày 20 tháng 6 năm 2024

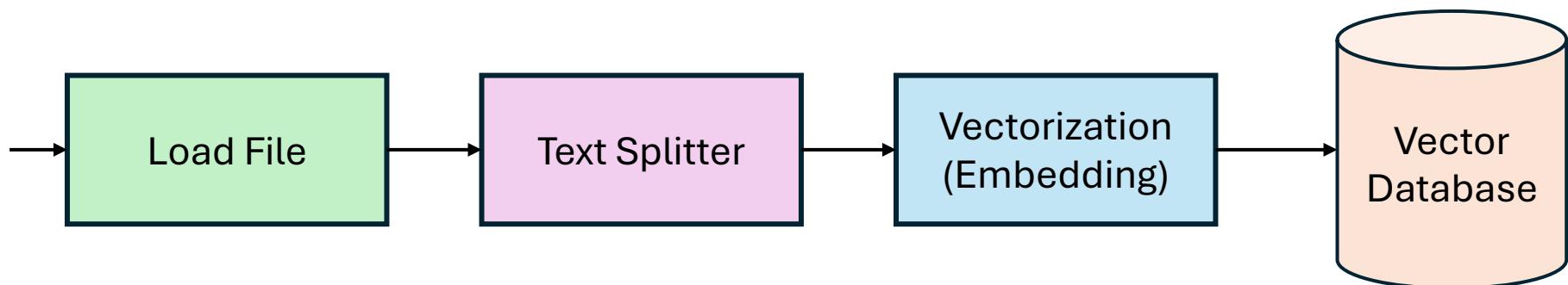
I. Giới thiệu

Object Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán cổ điển thuộc lĩnh vực Computer Vision. Mục tiêu của bài toán này là tự động xác định vị trí của các đối tượng trong một tấm ảnh. Tính tối thiểu điểm hiện tại, đã có rất nhiều phương pháp được phát triển nhằm giải quyết hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu do chính xác và tốc độ thực thi mà loại mô hình này mang lại.



Hình 1: Logo của mô hình YOLO. Ảnh: [link](#).

Thời gian vừa qua, Ao Wang và các cộng sự tại Đại học Thanh Hoá (Tsinghua University) đã đề xuất mô hình YOLOv10 trong bài báo **YOLOv10: Real-Time End-to-End Object Detection** [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng theo thời gian thực (real-time object detection).



¹
File .pdf

RAG

❖ Introduction

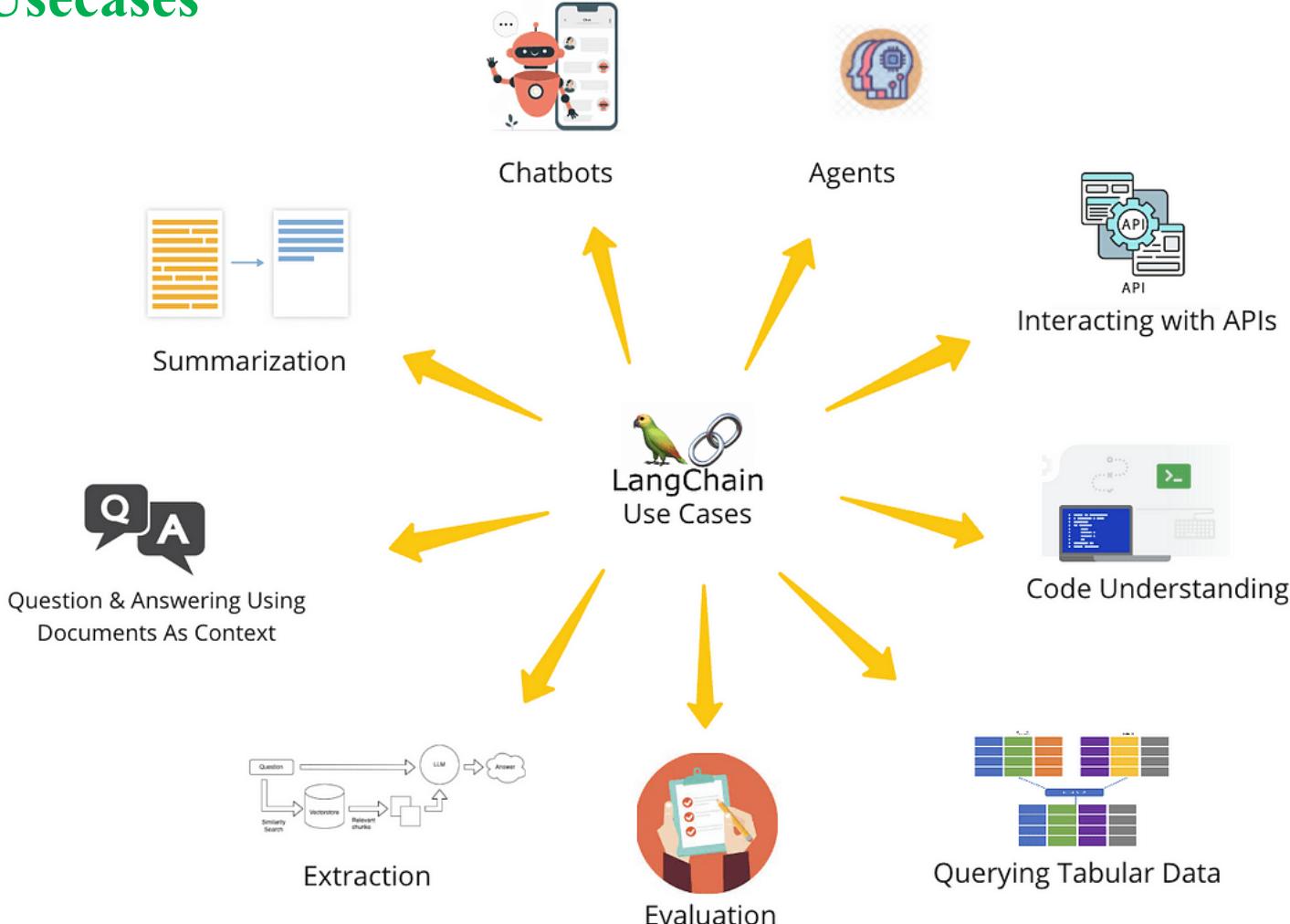


LangChain

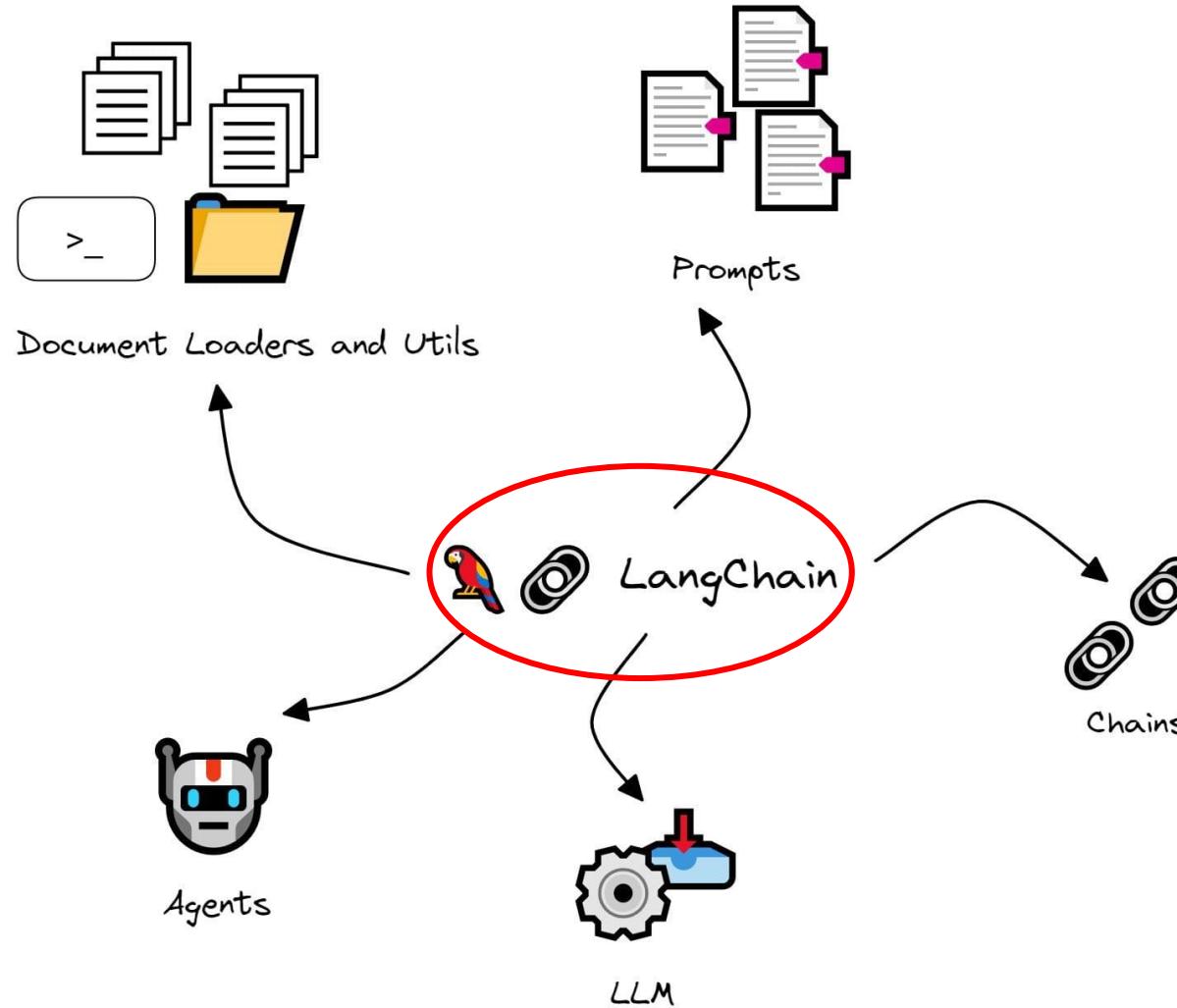
LangChain: A framework for developing applications powered by large language models (LLMs). LangChain simplifies every stage of the LLM application lifecycle: Development, Productionization, Deployment.

RAG

❖ LangChain Usecases



❖ LangChain Components



❖ Coding step 1: Install and import necessary libraries

```
1 !pip install -q transformers==4.41.2
2 !pip install -q bitsandbytes==0.43.1
3 !pip install -q accelerate==0.31.0
4 !pip install -q langchain==0.2.5
5 !pip install -q langchainhub==0.1.20
6 !pip install -q langchain-chroma==0.1.1
7 !pip install -q langchain-community==0.2.5
8 !pip install -q langchain_huggingface==0.0.3
9 !pip install -q python-dotenv==1.0.1
10 !pip install -q pypdf==4.2.0
11 !pip install -q numpy==1.24.4
```

```
1 import torch
2
3 from transformers import BitsAndBytesConfig
4 from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline
5 from langchain_huggingface import HuggingFaceEmbeddings
6 from langchain_huggingface.llms import HuggingFacePipeline
7
8 from langchain.memory import ConversationBufferMemory
9 from langchain_community.chat_message_histories import ChatMessageHistory
10 from langchain_community.document_loaders import PyPDFLoader, TextLoader
11 from langchain.chains import ConversationalRetrievalChain
12
13 from langchain_chroma import Chroma
14 from langchain_text_splitters import RecursiveCharacterTextSplitter
15 from langchain_core.runnables import RunnablePassthrough
16 from langchain_core.output_parsers import StrOutputParser
17 from langchain import hub
```

❖ Retriever Pipeline

AI VIET NAM – AI COURSE 2024

Tutorial: Phát hiện đối tượng trong ảnh với
YOLOv10

Dinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui và
Quang-Vinh Dinh

Ngày 20 tháng 6 năm 2024

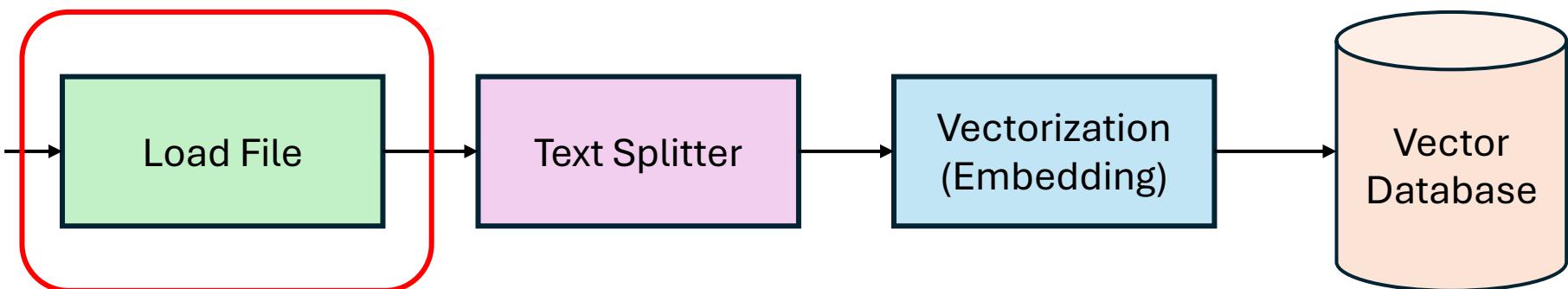
I. Giới thiệu

Object Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán cổ điển thuộc lĩnh vực Computer Vision. Mục tiêu của bài toán này là tự động xác định vị trí của các đối tượng trong một tấm ảnh. Tính tối thiểu điểm hiện tại, đã có rất nhiều phương pháp được phát triển nhằm giải quyết hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu do chính xác và tốc độ thực thi mà loại mô hình này mang lại.



Hình 1: Logo của mô hình YOLO.Ảnh: [link](#).

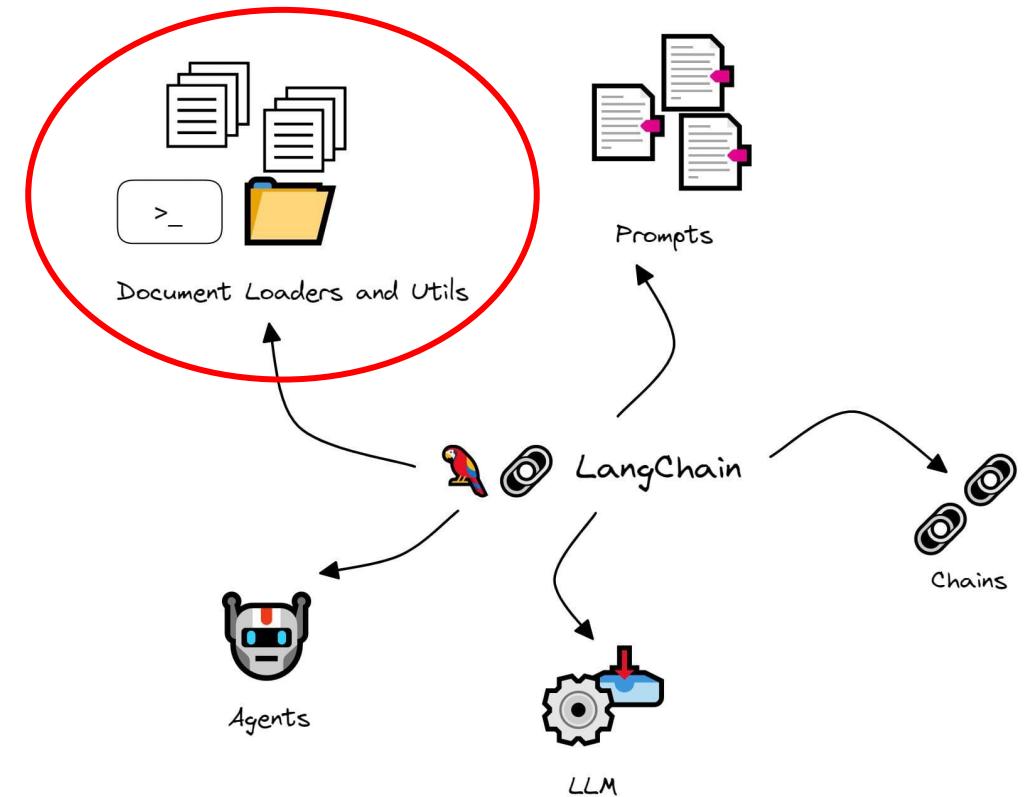
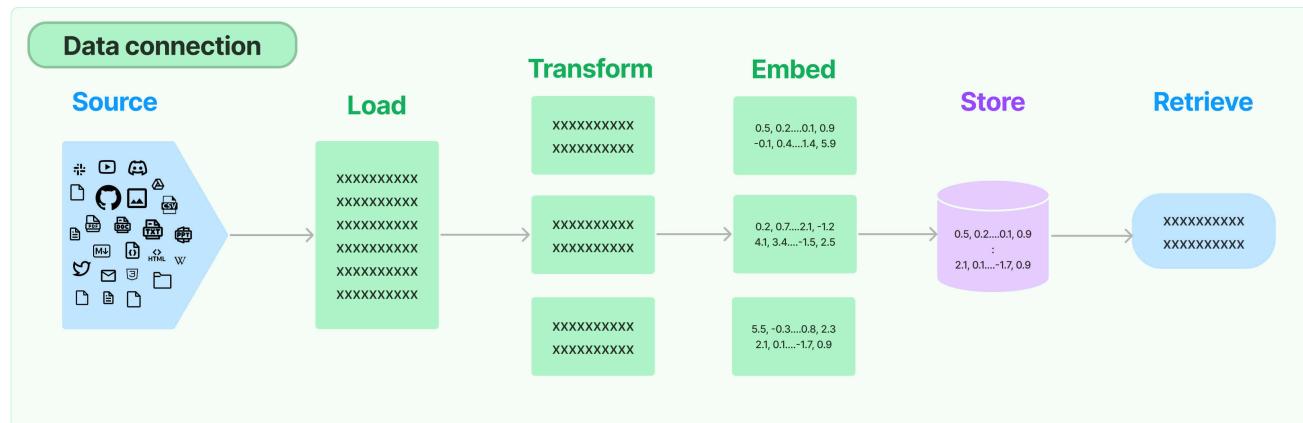
Thời gian vừa qua, Ao Wang và các cộng sự tại Đại học Thanh Hoá (Tsinghua University) đã đề xuất mô hình **YOLOv10** trong bài báo **YOLOv10: Real-Time End-to-End Object Detection** [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng theo thời gian thực (real-time object detection).



¹
File .pdf

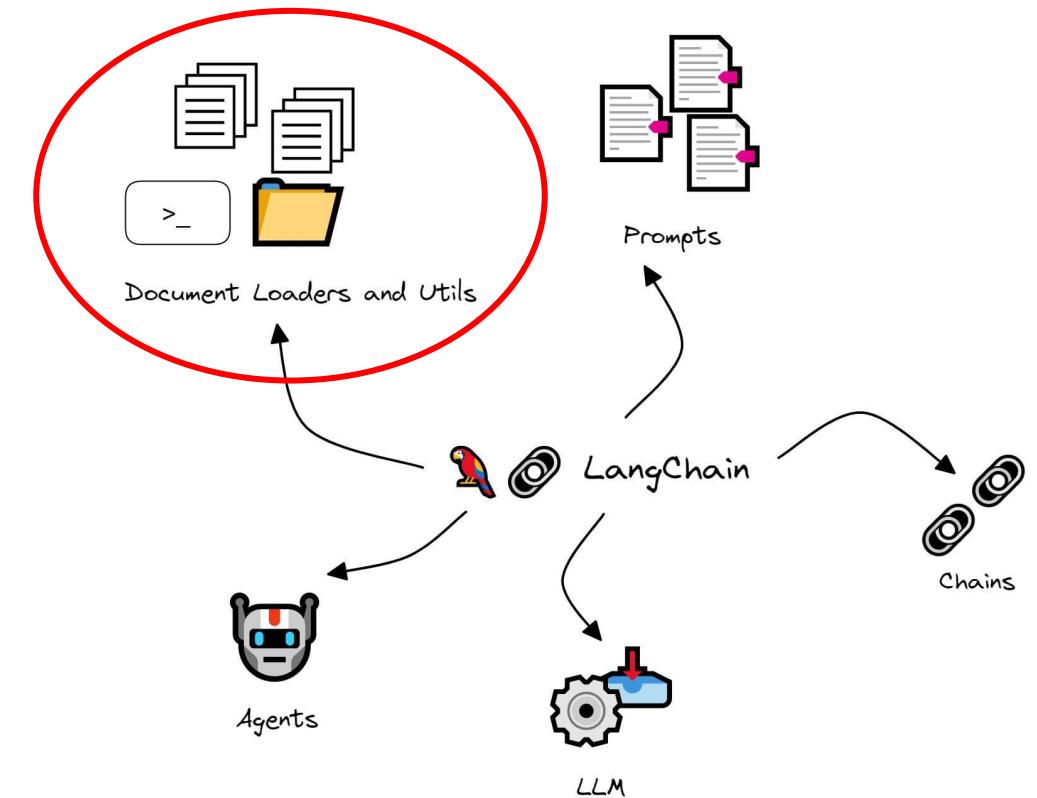
❖ LangChain: Document Loaders

LangChain supports multiple functions to handle documents

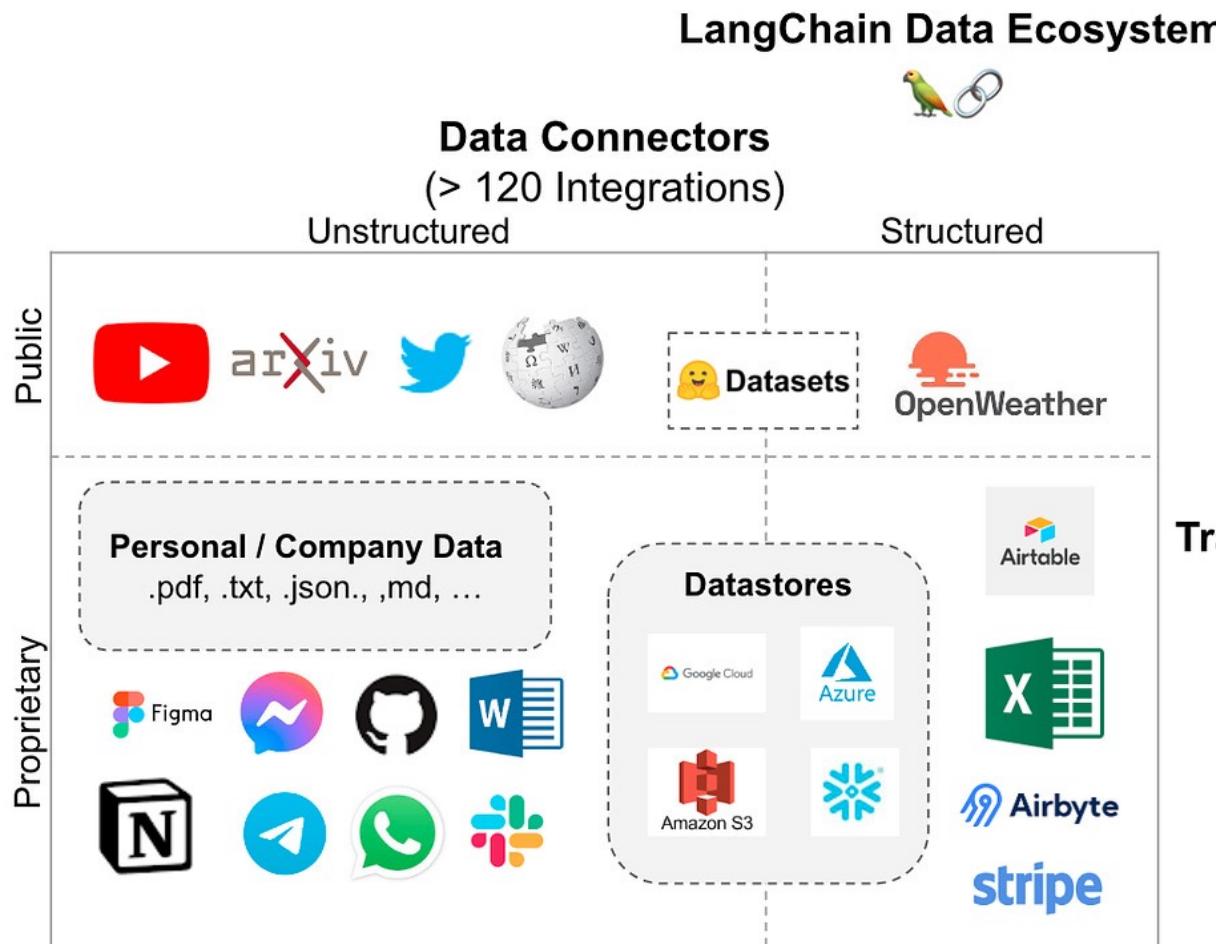


❖ LangChain: Document Loaders

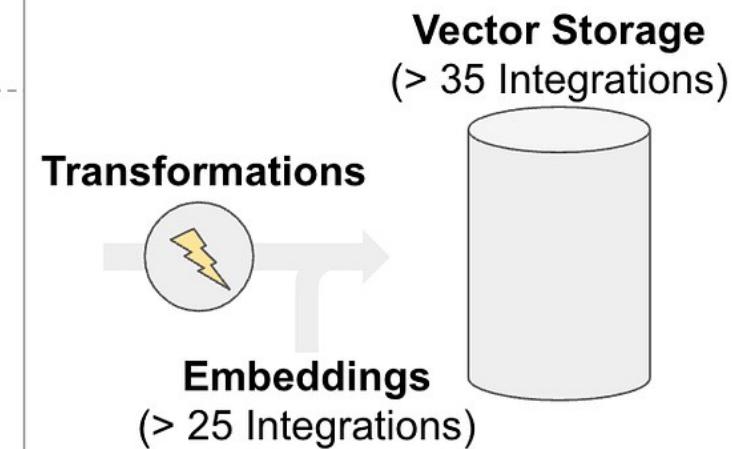
Document loaders load documents from many different sources. LangChain provides integrations to load all types of documents (HTML, PDF, code) from all types of locations (private S3 buckets, public websites).



❖ LangChain: Document Loaders



LangChain also provides over 120 different document loaders as well as integrations with other major providers in the space, like AirByte and Unstructured.



❖ Coding step 2: Load pdf file

```
1 Loader = PyPDFLoader
2 FILE_PATH = "./YOLOv10_Tutorials.pdf"
3 loader = Loader(FILE_PATH)
4 documents = loader.load()
5
6 print("Number of documents: ", len(documents))
7 documents[0]
```

Number of documents: 20

Document(page_content='AI VIET NAM – AI COURSE 2024\nTutorial: Phát hiện đối tượng trong ảnh với YOLOv10\nDinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui và Quang-Vinh Dinh\nNgày 20 tháng 6 năm 2024\nI. Giới thiệu\nObject Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán cổ điển thuộc lĩnh vực Computer Vision. Mục tiêu của bài toán này là tự động xác định vị trí của các đối tượng trong một tấm ảnh. Tính tới thời điểm hiện tại, đã có rất nhiều phương pháp được phát triển nhằm giải quyết hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu bởi độ chính xác và tốc độ thực thi mà loại mô hình này mang lại.\nHình 1: Logo của mô hình YOLO. Ảnh: [link](#). Thời gian vừa qua, Ao Wang và các cộng sự tại Đại học Thanh Hoa (Tsinghua University) đã đề xuất mô hình YOLOv10 trong bài báo YOLOv10: Real-Time End-to-End Object Detection [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng theo thời gian thực (real-time object detection).\n1', metadata={'source': './YOLOv10_Tutorials.pdf', 'page': 0})

AI VIET NAM – AI COURSE 2024

Tutorial: Phát hiện đối tượng trong ảnh với YOLOv10

Dinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui và Quang-Vinh Dinh

Ngày 20 tháng 6 năm 2024

I. Giới thiệu

Object Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán cổ điển thuộc lĩnh vực Computer Vision. Mục tiêu của bài toán này là tự động xác định vị trí của các đối tượng trong một tấm ảnh. Tính tới thời điểm hiện tại, đã có rất nhiều phương pháp được phát triển nhằm giải quyết hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu bởi độ chính xác và tốc độ thực thi mà loại mô hình này mang lại.



Hình 1: Logo của mô hình YOLO. Ảnh: [link](#).

Thời gian vừa qua, Ao Wang và các cộng sự tại Đại học Thanh Hoa (Tsinghua University) đã đề xuất mô hình YOLOv10 trong bài báo YOLOv10: Real-Time End-to-End Object Detection [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng theo thời gian thực (real-time object detection).

❖ Retriever Pipeline

AI VIET NAM – AI COURSE 2024
Tutorial: Phát hiện đối tượng trong ảnh với
YOLOv10
Dinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui và
Quang-Vinh Dinh
Ngày 20 tháng 6 năm 2024

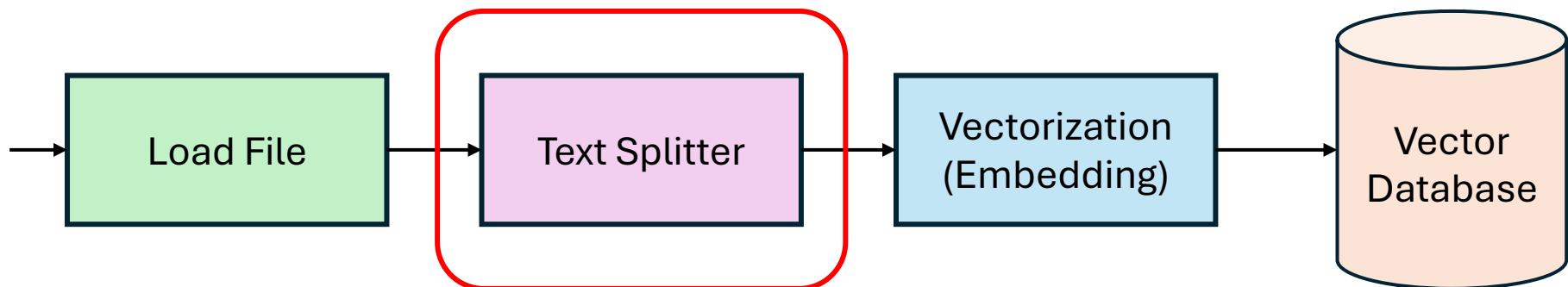
I. Giới thiệu

Object Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán cổ điển thuộc lĩnh vực Computer Vision. Mục tiêu của bài toán này là tự động xác định vị trí của các đối tượng trong một tấm ảnh. Tính tối thiểu điểm hiện tại, đã có rất nhiều phương pháp được phát triển nhằm giải quyết hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu do chính xác và tốc độ thực thi mà loại mô hình này mang lại.



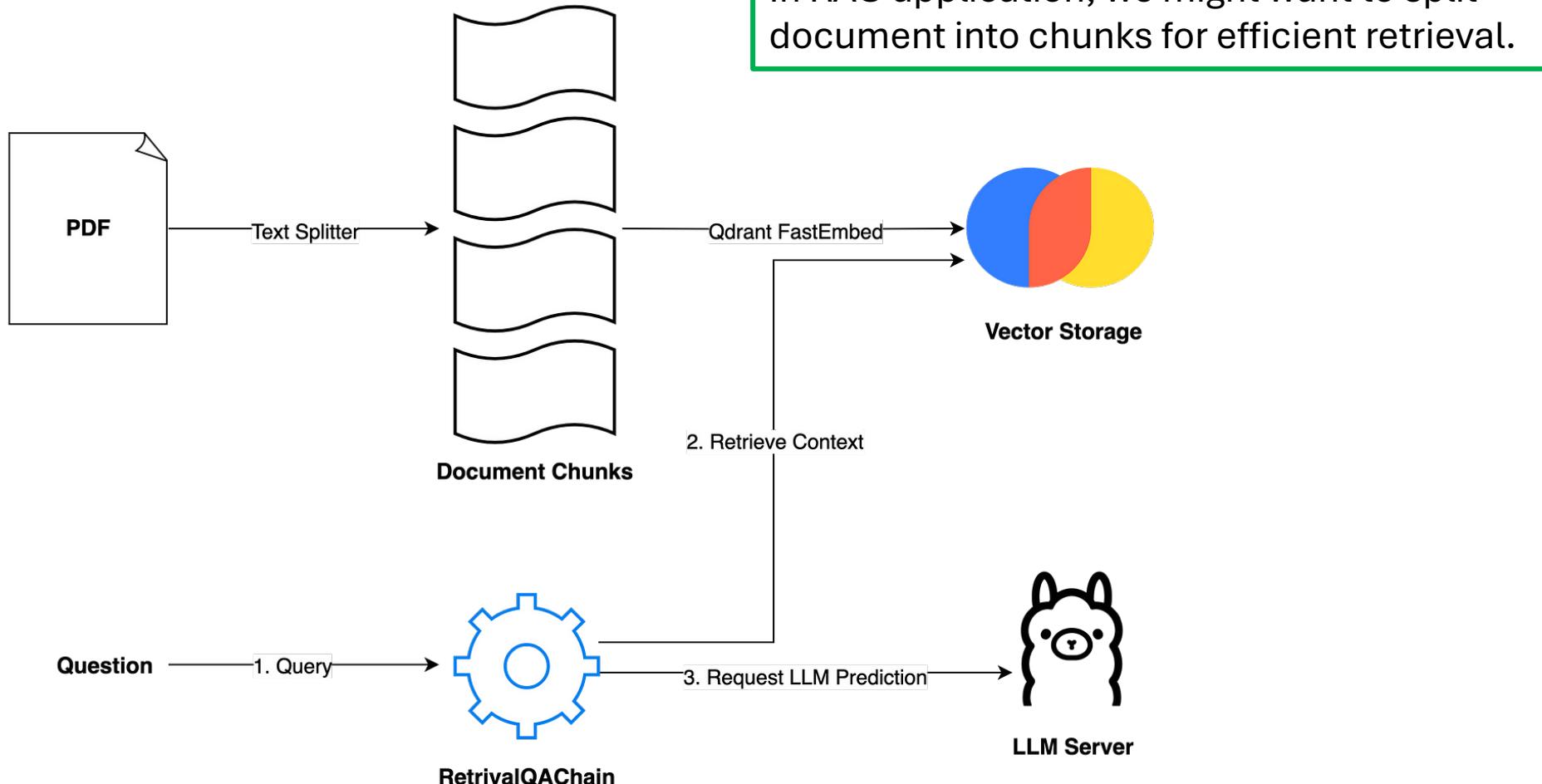
Hình 1: Logo của mô hình YOLO. Ảnh: [link](#).

Thời gian vừa qua, Ao Wang và các cộng sự tại Đại học Thanh Hoá (Tsinghua University) đã đề xuất mô hình YOLOv10 trong bài báo **YOLOv10: Real-Time End-to-End Object Detection** [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng theo thời gian thực (real-time object detection).



¹
File .pdf

❖ LangChain: Document Splitter

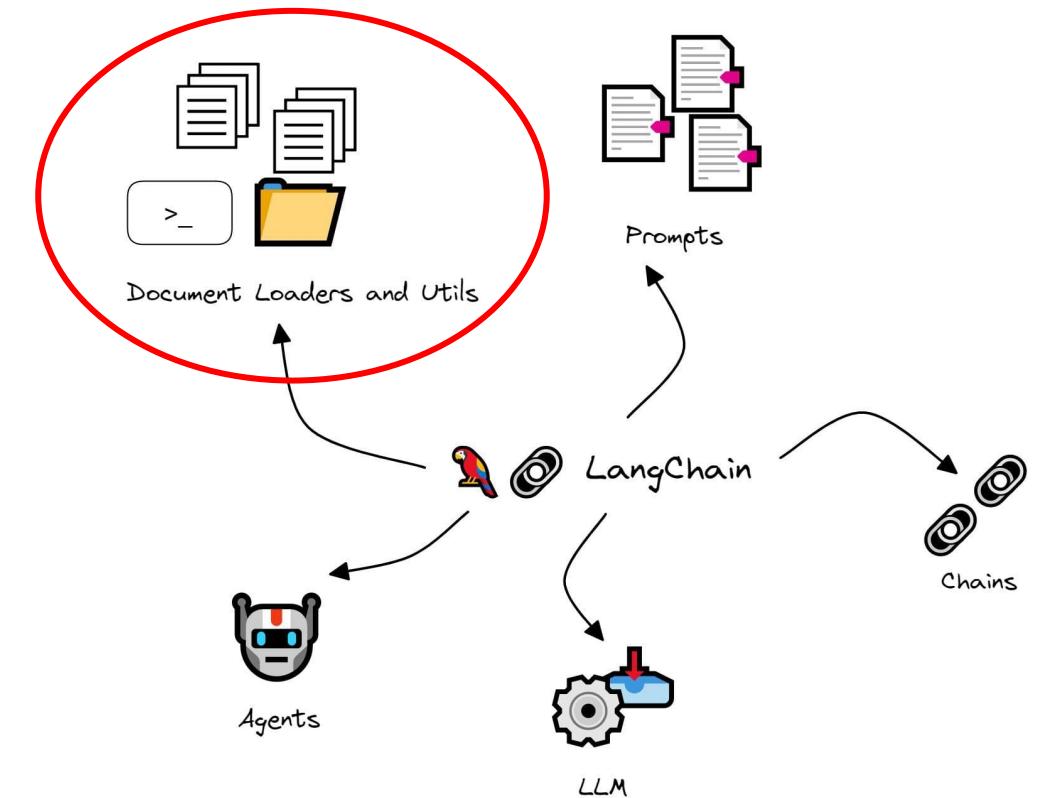


❖ LangChain: Document Splitter

A key part of retrieval is fetching only the relevant parts of documents.

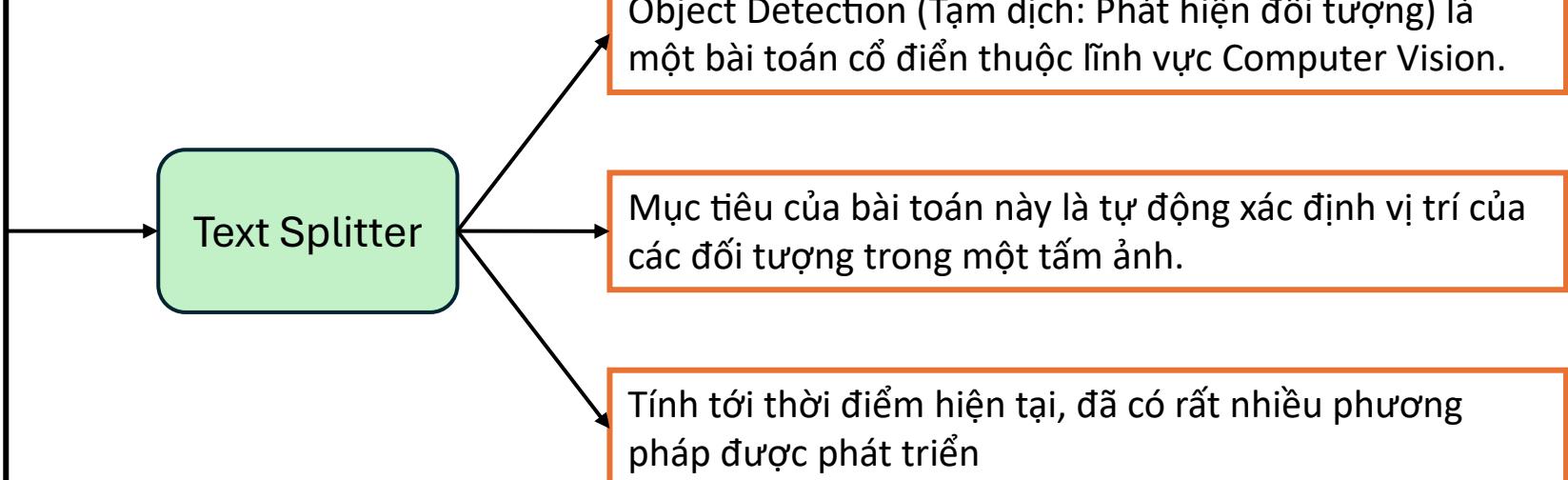
One of the primary ones here is splitting (or chunking) a large document into smaller chunks.

LangChain provides several transformation algorithms for doing this, as well as logic optimized for specific document types (code, markdown, etc).



❖ LangChain: Document Splitter

"Object Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán cổ điển thuộc lĩnh vực Computer Vision. Mục tiêu của bài toán này là tự động xác định vị trí của các đối tượng trong một tấm ảnh. Tính tới thời điểm hiện tại, đã có rất nhiều phương pháp được phát triển nhằm giải quyết hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu bởi độ chính xác và tốc độ thực thi mà loại mô hình này mang lại."



• • •

❖ Coding step 3: Run text splitter

```
1 docs = text_splitter.split_documents(documents)
2
3 print("Number of mini-documents: ", len(docs))
4 docs[0]
```

Number of mini-documents: 33
Document(page_content='AI VIET NAM – AI COURSE 2024\nTutorial: Phát hiện đối tượng trong ảnh với YOLOv10\nDinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui và Quang-Vinh Dinh\nNgày 20 tháng 6 năm 2024\nI. Giới thiệu\nObject Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán có điều kiện thuộc lĩnh vực Computer Vision. Mục tiêu của bài toán này là tự động xác định vị trí của các đối tượng trong một tấm ảnh. Tính tới thời điểm hiện tại, đã có rất nhiều phương pháp được phát triển nhằm giải quyết hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu bởi độ chính xác và tốc độ thực thi mà loại mô hình này mang lại.\nHình 1: Logo của mô hình YOLO. Ảnh: link.\nThời gian vừa qua, Ao Wang và các cộng sự tại Đại học Thanh Hoa (Tsinghua University)\nđã đề xuất mô hình YOLOv10 trong bài báo YOLOv10: Real-Time End-to-End Object Detection [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các', metadata={'source': './YOLOv10_Tutorials.pdf', 'page': 0})

❖ Retriever Pipeline

AI VIET NAM – AI COURSE 2024
Tutorial: Phát hiện đối tượng trong ảnh với
YOLOv10
Dinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui và
Quang-Vinh Dinh
Ngày 20 tháng 6 năm 2024

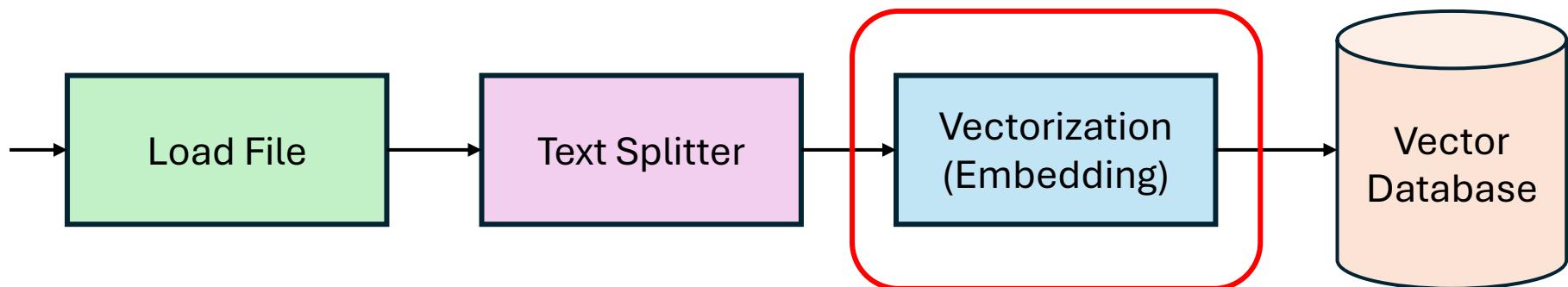
I. Giới thiệu

Object Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán cổ điển thuộc lĩnh vực Computer Vision. Mục tiêu của bài toán này là tự động xác định vị trí của các đối tượng trong một tấm ảnh. Tính tối thiểu điểm hiện tại, đã có rất nhiều phương pháp được phát triển nhằm giải quyết hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu do chính xác và tốc độ thực thi mà loại mô hình này mang lại.



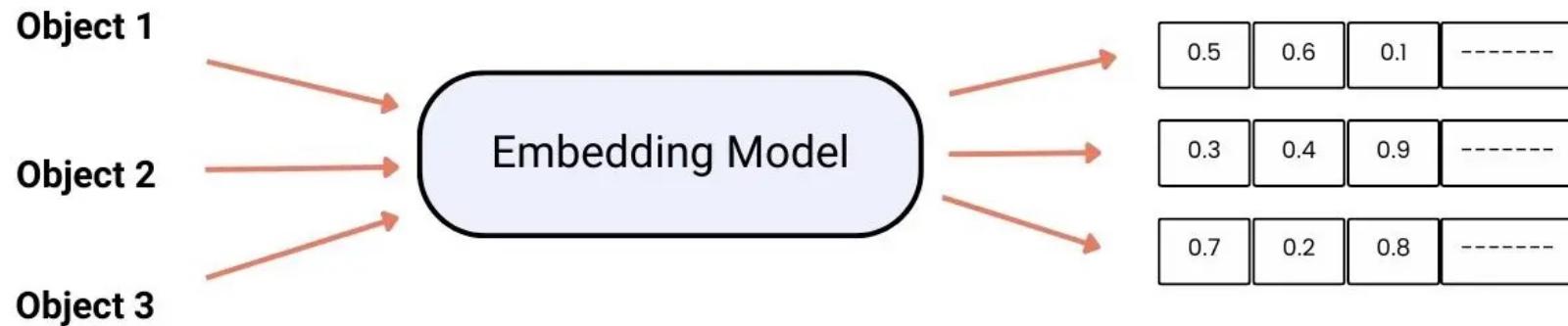
Hình 1: Logo của mô hình YOLO. Ảnh: [link](#).

Thời gian vừa qua, Ao Wang và các cộng sự tại Đại học Thanh Hoá (Tsinghua University) đã đề xuất mô hình YOLOv10 trong bài báo **YOLOv10: Real-Time End-to-End Object Detection** [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng theo thời gian thực (real-time object detection).

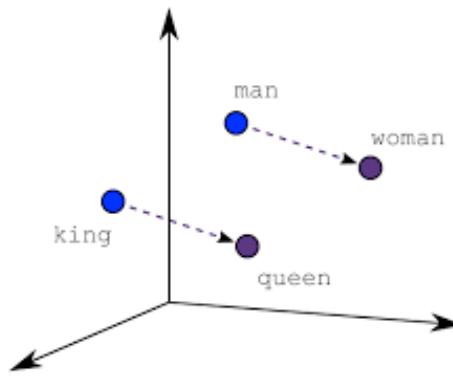


¹
File .pdf

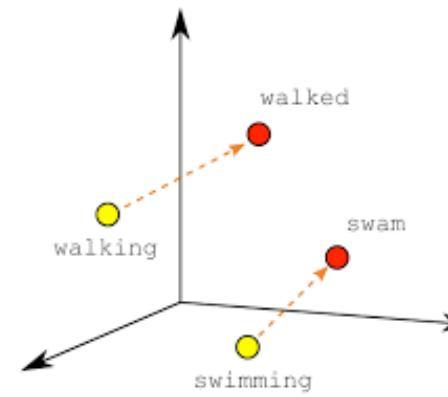
❖ LangChain: Text Embedding



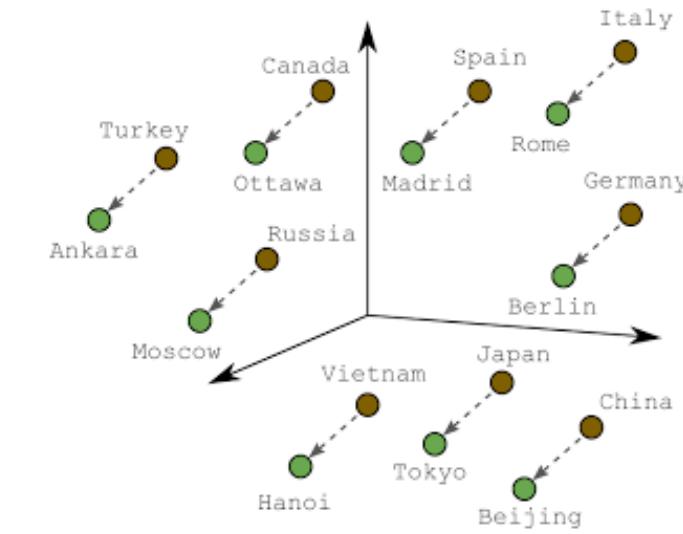
❖ Text Embedding



Male-Female



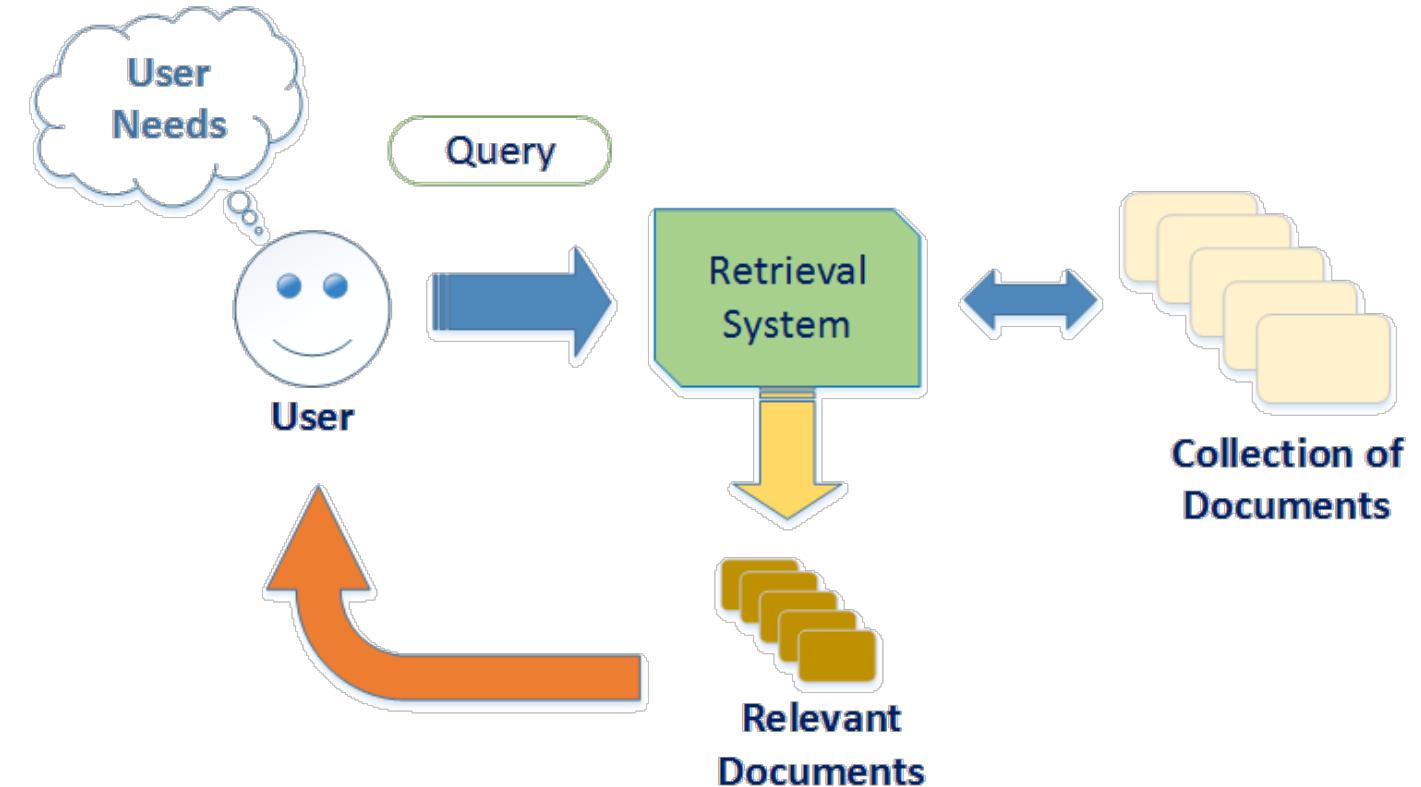
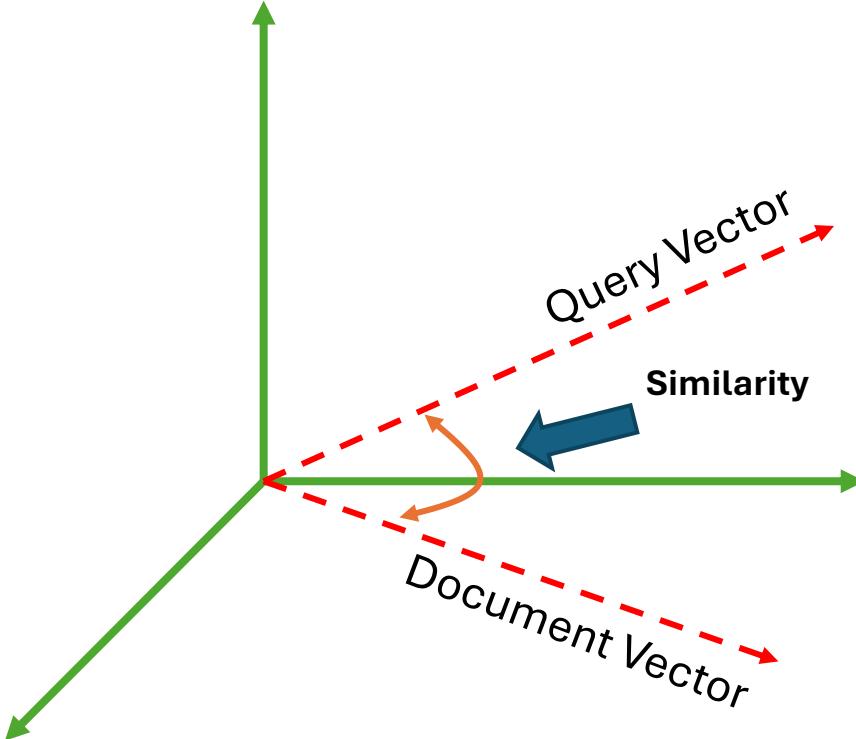
Verb Tense



Country-Capital

Represent text in a multi-dimensional space

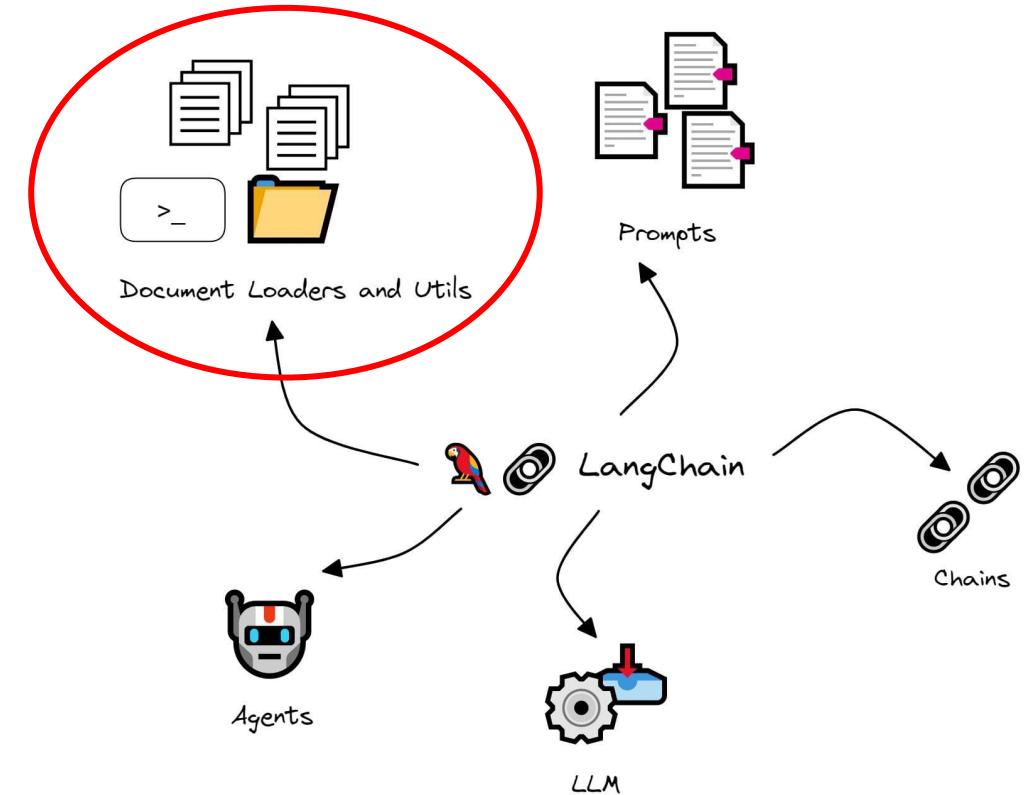
❖ Similarity Search



❖ Text Embedding

Another key part of retrieval is creating embeddings for documents. Embeddings capture the semantic meaning of the text, allowing you to quickly and efficiently find other pieces of a text that are similar.

LangChain provides integrations with over 25 different embedding providers and methods, from open-source to proprietary API, allowing you to choose the one best suited for your needs.



❖ LangChain: Text Embedding



```
1 from langchain_community.embeddings import HuggingFaceEmbeddings
2
3 embeddings = HuggingFaceEmbeddings()
4
5 text = "This is a test document."
6 vector = embeddings.embed_query(text)
```

vector

```
[-0.04895171523094177,
 -0.039861928671598434,
 -0.021562781184911728,
 0.00990847684442997,
 -0.03810400515794754,
 0.012684378772974014,
 0.04349460452795029,
 0.07183394581079483,
 0.00974853616207838,
 -0.006987018045037985,
 0.06352812796831131,
 -0.030322682112455368,
 0.013839473016560078,
 0.025805911049246788,
 -0.001136262551881373,
 -0.014563608914613724,
 0.041640277951955795,
 0.036228302866220474,
```

❖ Retriever Pipeline

AI VIET NAM – AI COURSE 2024
Tutorial: Phát hiện đối tượng trong ảnh với
YOLOv10
Dinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui và
Quang-Vinh Dinh
Ngày 20 tháng 6 năm 2024

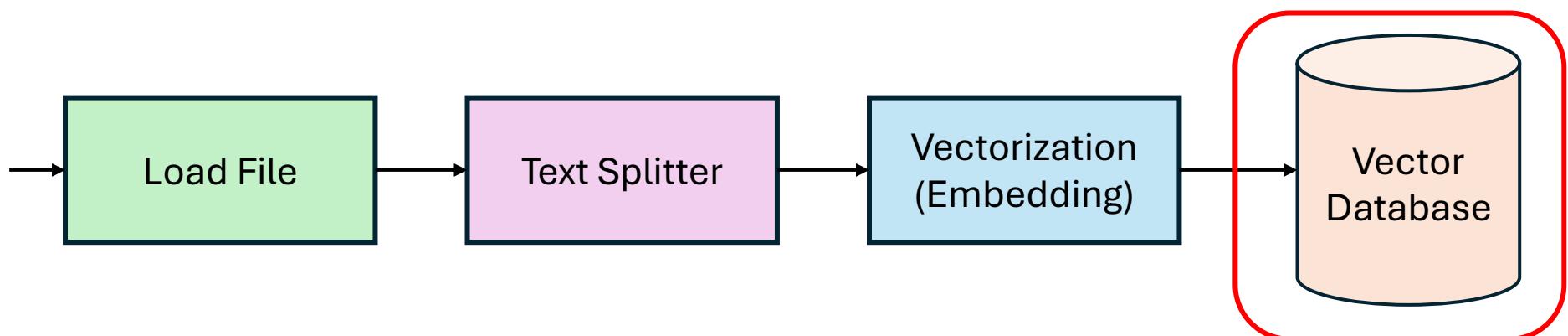
I. Giới thiệu

Object Detection (Tạm dịch: Phát hiện đối tượng) là một bài toán cổ điển thuộc lĩnh vực Computer Vision. Mục tiêu của bài toán này là tự động xác định vị trí của các đối tượng trong một tấm ảnh. Tính tối thiểu điểm hiện tại, đã có rất nhiều phương pháp được phát triển nhằm giải quyết hiệu quả bài toán này. Trong đó, các phương pháp thuộc họ YOLO (You Only Look Once) thu hút được sự chú ý rất lớn từ cộng đồng nghiên cứu do chính xác và tốc độ thực thi mà loại mô hình này mang lại.



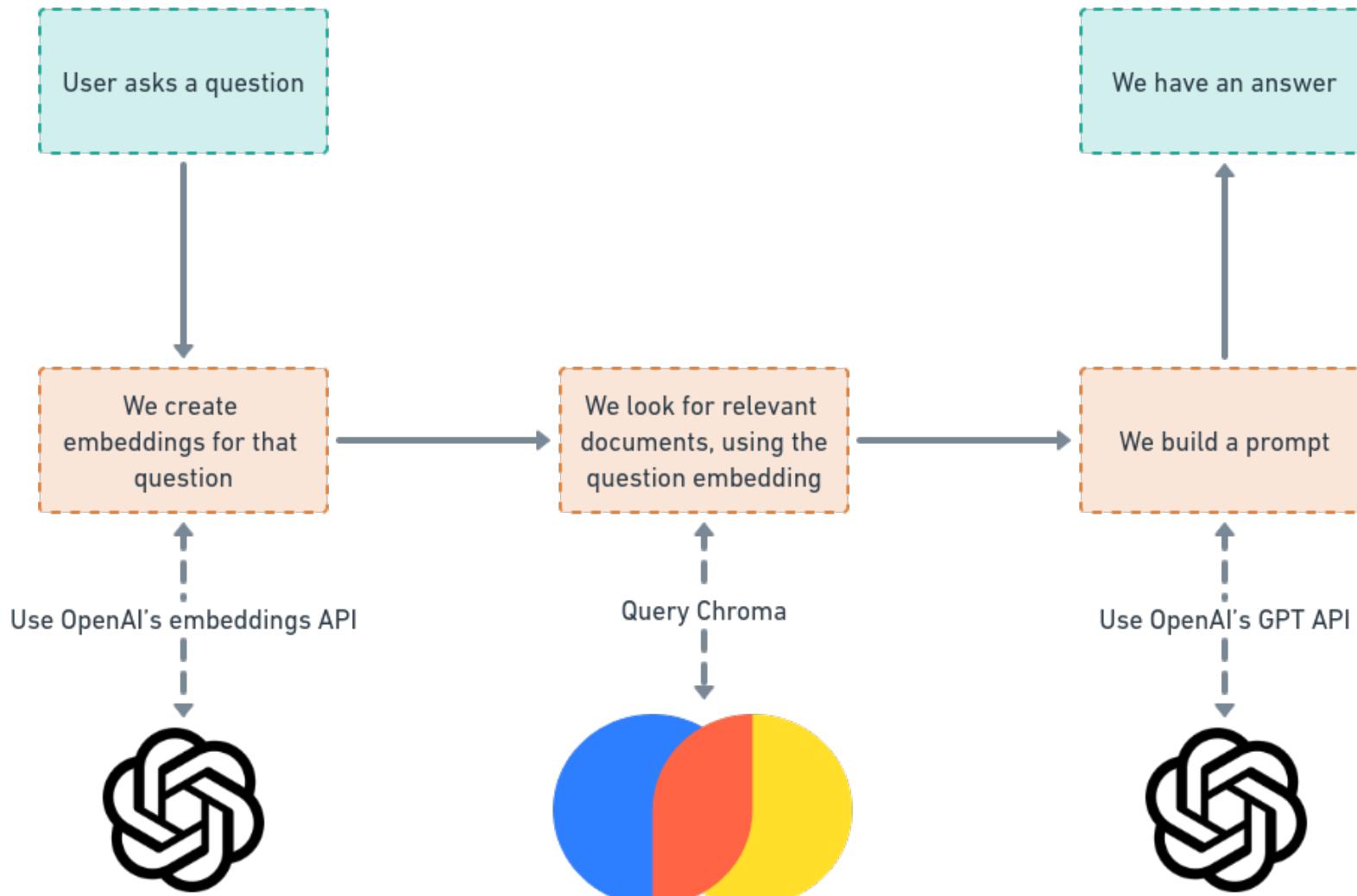
Hình 1: Logo của mô hình YOLO. Ảnh: [link](#).

Thời gian vừa qua, Ao Wang và các cộng sự tại Đại học Thanh Hoá (Tsinghua University) đã đề xuất mô hình **YOLOv10** trong bài báo **YOLOv10: Real-Time End-to-End Object Detection** [10]. Với những cải tiến mới, mô hình đã đạt được hiệu suất vượt trội hơn so với các phiên bản YOLO trước đó ở các khía cạnh khác nhau, tăng cường khả năng phát hiện đối tượng theo thời gian thực (real-time object detection).



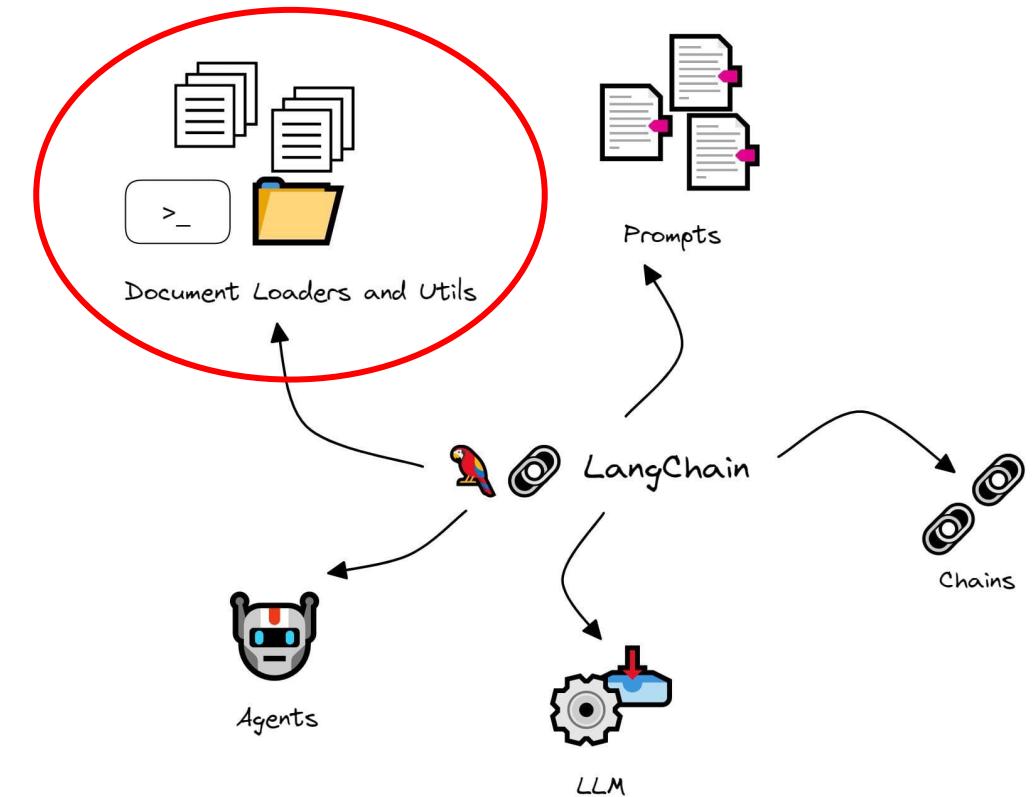
¹
File .pdf

❖ LangChain: Vector Database



❖ LangChain: Vector Database

LangChain provides integrations with over 50 different vectorstores, from open-source local ones to cloud-hosted proprietary ones, allowing you to choose the one best suited for your needs. LangChain exposes a standard interface, allowing you to easily swap between vector stores.



❖ Coding step 4: Build Embedding and Vector Database

```
Number of relevant documents: 4
Document(page_content='AI VIETNAM (AI02024) - PlantDocv4\n"\n3!rm / content / PlantDocv4 training theo yêu cầu của YOLO. Vì vậy, chúng phần trước, các bạn chạy\ncác đoạn code sau : yolov10\n3!pip install -q -r requirements .txt số đã được huấn luyện trên bộ dữ liệu COCO. download /v1 .1/\nyolov10n .pt', metadata={'
```

❖ Coding step 5: Load Vicuna model and use the program

```
1 nf4_config = BitsAndBytesConfig(  
2     load_in_4bit=True,  
3     bnb_4bit_quant_type="nf4",  
4     bnb_4bit_use_double_quant=True,  
5     bnb_4bit_compute_dtype=torch.bfloat16  
6 )  
7  
8 MODEL_NAME = "lmsys/vicuna-7b-v1.5"  
9  
10 model = AutoModelForCausalLM.from_pretrained(  
11     MODEL_NAME,  
12     quantization_config=nf4_config,  
13     low_cpu_mem_usage=True  
14 )  
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME)  
16  
17 model_pipeline = pipeline(  
18     "text-generation",  
19     model=model,  
20     tokenizer=tokenizer,  
21     max_new_tokens=512,  
22     pad_token_id=tokenizer.eos_token_id,  
23     device_map="auto"  
24 )  
25  
26 llm = HuggingFacePipeline(  
27     pipeline=model_pipeline,  
28 )
```

```
1 prompt = hub.pull("rlm/rag-prompt")  
2  
3 def format_docs(docs):  
4     return "\n\n".join(doc.page_content for doc in docs)  
5  
6 rag_chain = (  
7     {"context": retriever | format_docs, "question": RunnablePassthrough() }  
8     | prompt  
9     | llm  
10    | StrOutputParser()  
11 )  
12  
13 USER_QUESTION = "YOLOv10 là gì?"  
14 output = rag_chain.invoke(USER_QUESTION)  
15 output
```

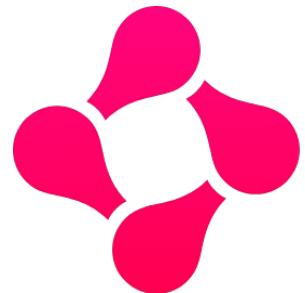
Answer (from Vicuna): YOLOv10 là một phiên bản của mô hình dự đoán hình ảnh YOLO (You Only Look Once) v10, được phát triển bởi tác giả THU -MIG. Nó được tải về và khởi tạo bằng cách sử dụng mã nhập và dữ liệu trọng số (file.pt) từ Github.

QUIZ

Build Chatbot UI with Chainlit

Build Chatbot UI with Chainlit

❖ Introduction

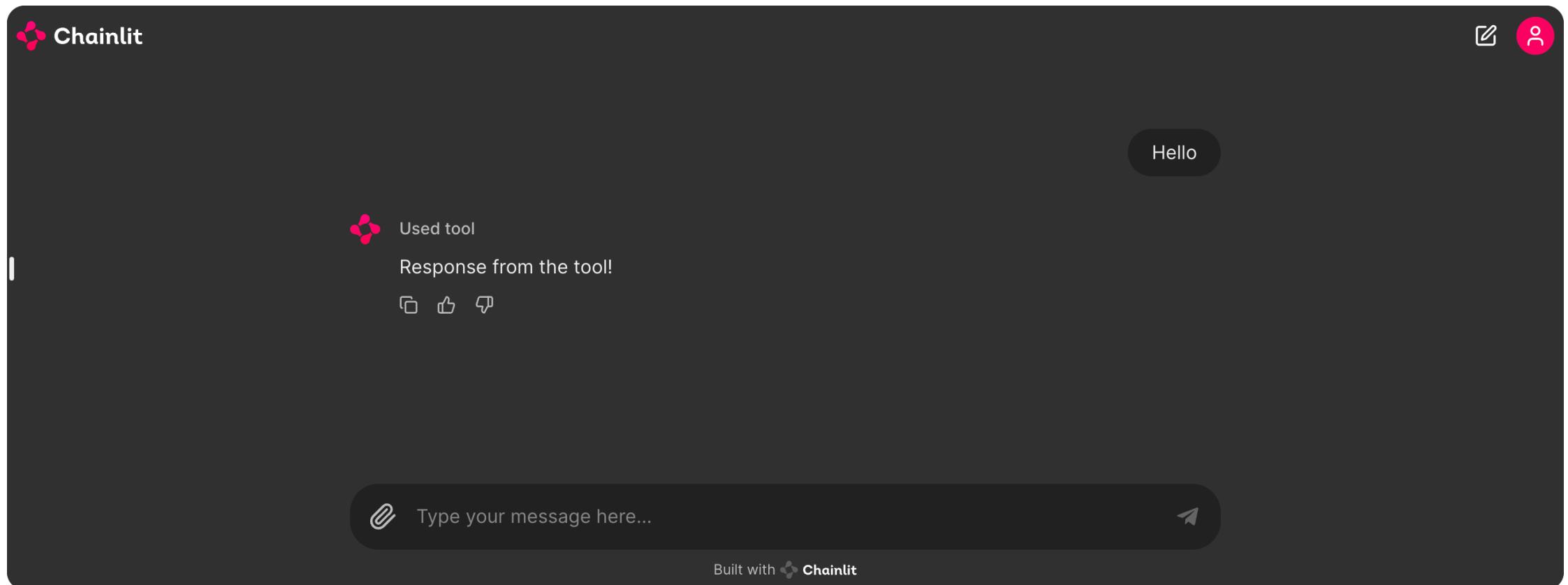


Chainlit

Chainlit: An open-source Python package to build production ready Conversational AI with fast building, multi platform, data persistence and visualize multi-steps reasoning features.

Build Chatbot UI with Chainlit

❖ Introduction

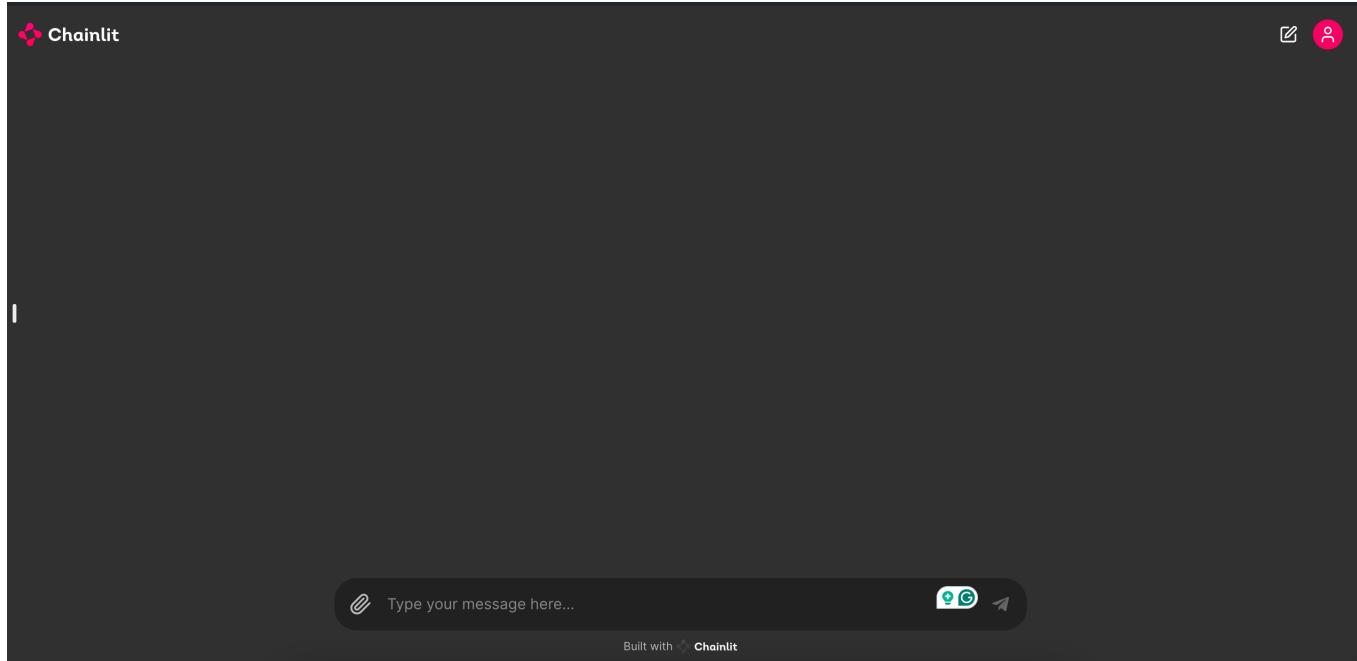


Create a Chatbot UI with Chainlit

Build Chatbot UI with Chainlit

❖ Chat life cycle: `on_start`

```
● ● ●  
1 import chainlit as cl  
2  
3 @cl.on_chat_start  
4 def on_chat_start():  
5     print("A new chat session has started!")
```

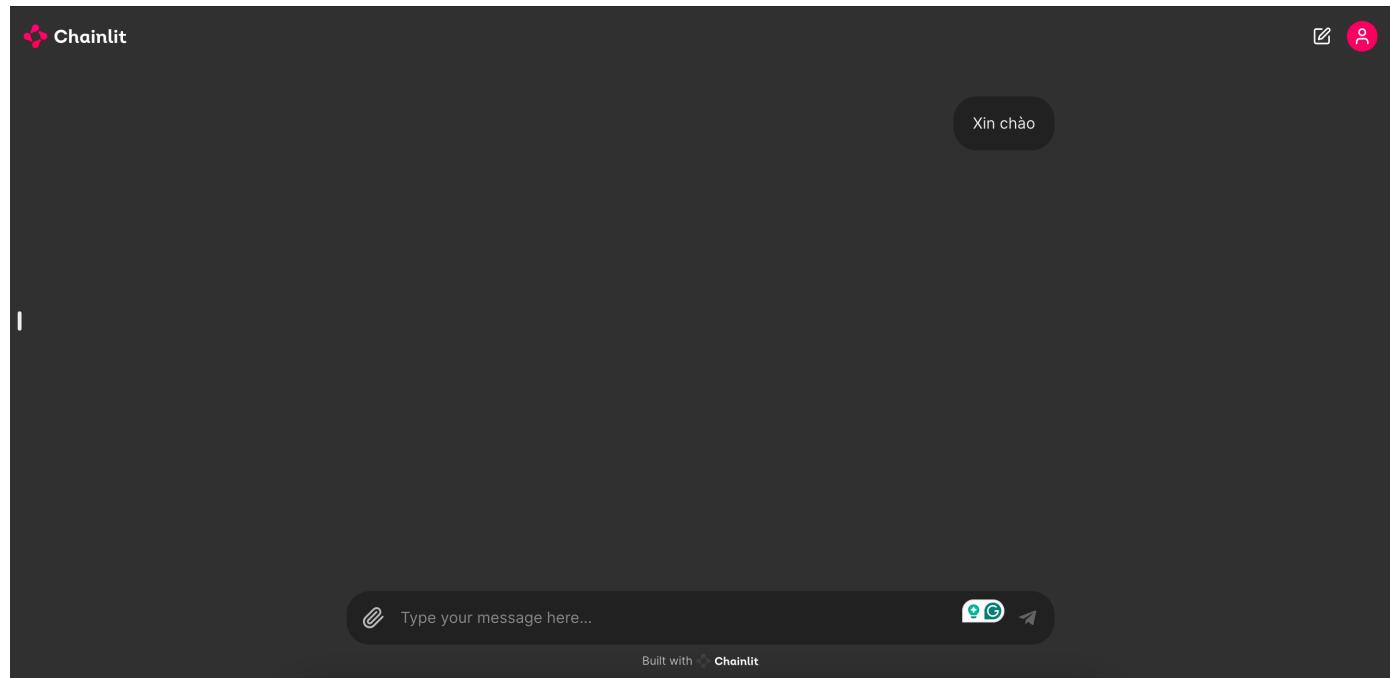


```
● ● ●  
1 2024-06-29 17:14:13 - Your app is available at http://localhost:8000  
2 A new chat session has started!
```

Build Chatbot UI with Chainlit

❖ Chat life cycle: on_message

```
● ● ●  
1 import chainlit as cl  
2  
3 @cl.on_message  
4 def on_message(msg: cl.Message):  
5     print("The user sent: ", msg.content)
```

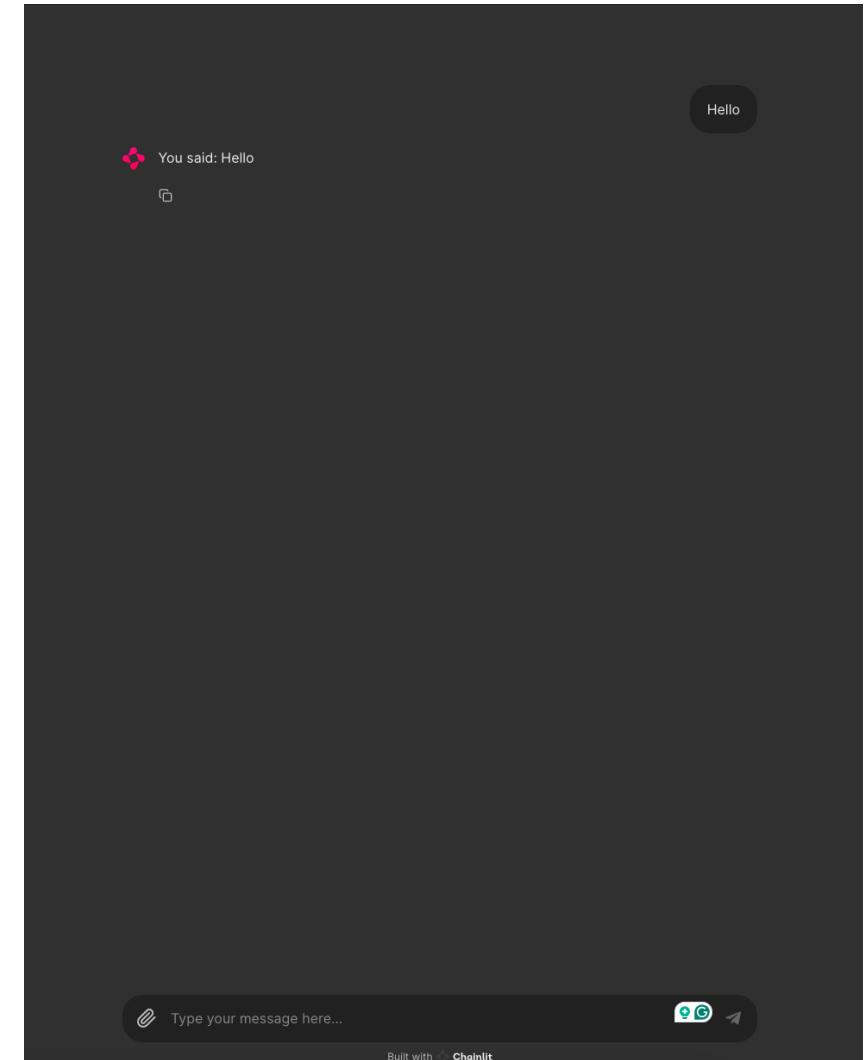


```
● ● ●  
1 2024-06-29 18:22:46 - Your app is available at http://localhost:8000  
2 2024-06-29 18:22:55 - Translated markdown file for en-US not found. Defaulting to chainlit.md.  
3 The user sent: Hi  
4
```

Build Chatbot UI with Chainlit

❖ Return message

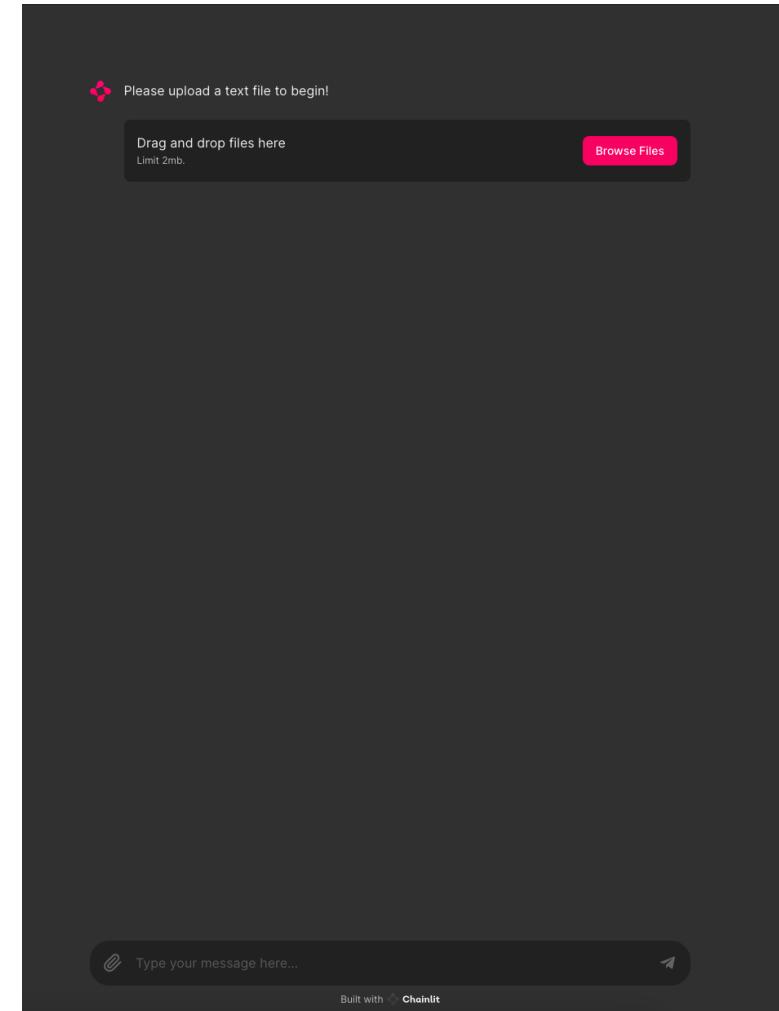
```
1 import chainlit as cl
2
3 @cl.on_chat_start
4 def on_chat_start():
5     pass
6
7 @cl.on_message
8 async def on_message(message: cl.Message):
9     user_msg = message.content
10
11     await cl.Message(content=f"You said: {user_msg}").send()
```



Build Chatbot UI with Chainlit

❖ File upload (PDF)

```
1 import chainlit as cl
2 from langchain_community.document_loaders import PyPDFLoader
3
4 @cl.on_chat_start
5 async def start():
6     files = None
7
8     while files == None:
9         files = await cl.AskFileMessage(
10             content="Please upload a text file to begin!",
11             accept=["application/pdf"]
12         ).send()
13
14     pdf_file = files[0]
15     documents = PyPDFLoader(pdf_file.path).load()
16
17     await cl.Message(
18         content=f"`{pdf_file.name}` uploaded"
19     ).send()
```

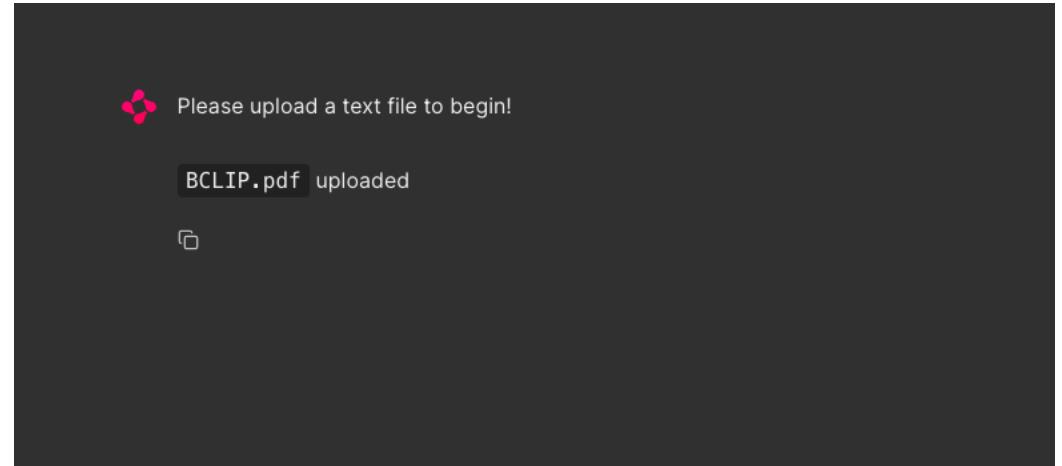


Build Chatbot UI with Chainlit

❖ File upload (PDF)



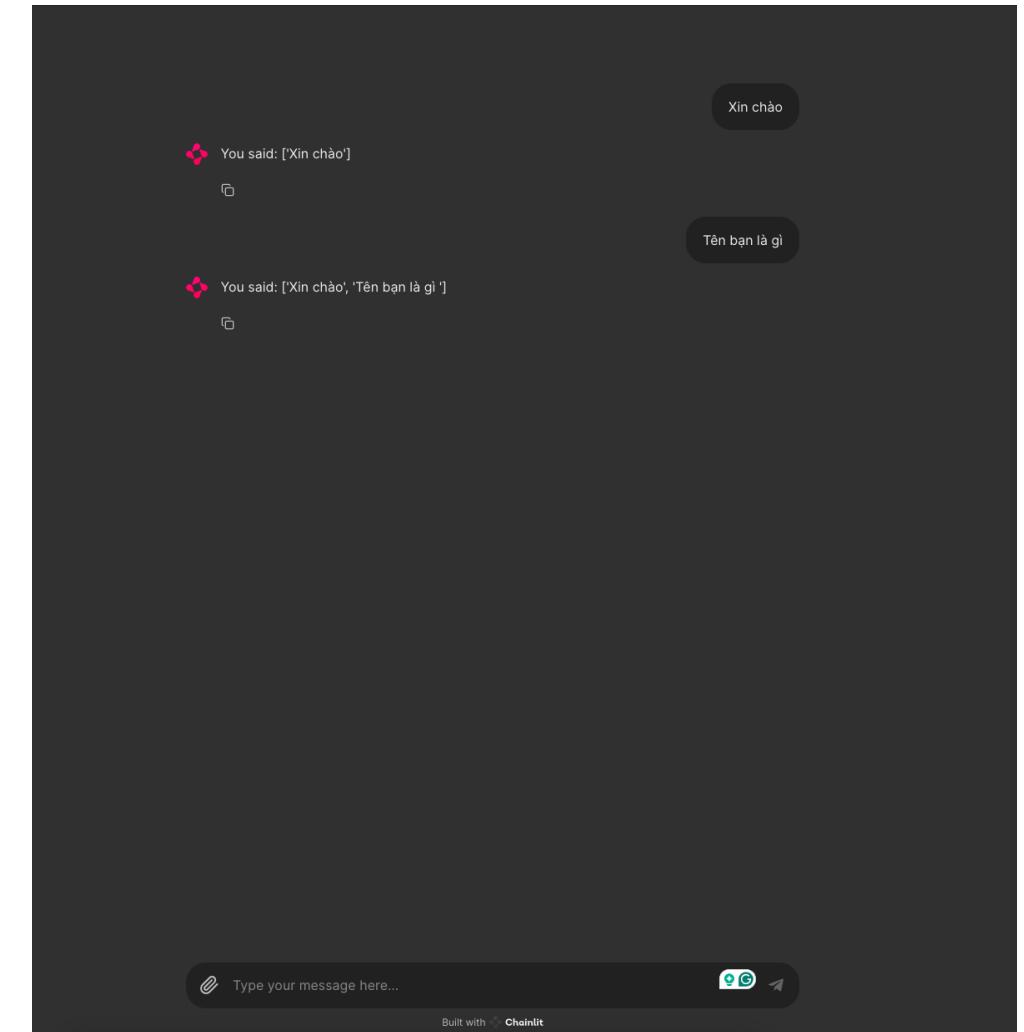
```
1 import chainlit as cl
2 from langchain_community.document_loaders import PyPDFLoader
3
4 @cl.on_chat_start
5 async def start():
6     files = None
7
8     while files == None:
9         files = await cl.AskFileMessage(
10             content="Please upload a text file to begin!",
11             accept=["application/pdf"]
12         ).send()
13
14     pdf_file = files[0]
15     documents = PyPDFLoader(pdf_file.path).load()
16
17     await cl.Message(
18         content=f"`{pdf_file.name}` uploaded"
19     ).send()
```



Build Chatbot UI with Chainlit

❖ User Session

```
● ● ●  
1 import chainlit as cl  
2  
3 @cl.on_chat_start  
4 def on_chat_start():  
5     cl.user_session.set("history", [])  
6  
7 @cl.on_message  
8 async def on_message(message: cl.Message):  
9     history = cl.user_session.get("history")  
10    history.append(message.content)  
11  
12    cl.user_session.set("history", history)  
13    await cl.Message(content=f"You said: {history}").send()
```



Build Chatbot UI with Chainlit

❖ Coding step 1: Install and import libraries

```
1 !pip install -q transformers==4.41.2
2 !pip install -q bitsandbytes==0.43.1
3 !pip install -q accelerate==0.31.0
4 !pip install -q langchain==0.2.5
5 !pip install -q langchainhub==0.1.20
6 !pip install -q langchain-chroma==0.1.1
7 !pip install -q langchain-community==0.2.5
8 !pip install -q langchain-openai==0.1.9
9 !pip install -q langchain_huggingface==0.0.3
10 !pip install -q chainlit==1.1.304
11 !pip install -q python-dotenv==1.0.1
12 !pip install -q pypdf==4.2.0
13 !npm install -g localtunnel
14 !pip install -q numpy==1.24.4
```

```
1 import chainlit as cl
2 import torch
3
4 from chainlit.types import AskFileResponse
5
6 from transformers import BitsAndBytesConfig
7 from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline
8 from langchain_huggingface import HuggingFaceEmbeddings
9 from langchain_huggingface.llms import HuggingFacePipeline
10
11 from langchain_community.chat_message_histories import ChatMessageHistory
12 from langchain_community.document_loaders import PyPDFLoader, TextLoader
13 from langchain.chains import ConversationalRetrievalChain
14 from langchain.memory import ConversationBufferMemory
15 from langchain_chroma import Chroma
16 from langchain_text_splitters import RecursiveCharacterTextSplitter
17 from langchain_core.runnables import RunnablePassthrough
18 from langchain_core.output_parsers import StrOutputParser
19 from langchain import hub
```

Build Chatbot UI with Chainlit

❖ Coding step 2: Build vector database

```
1 def process_file(file: AskFileResponse):
2     if file.type == "text/plain":
3         Loader = TextLoader
4     elif file.type == "application/pdf":
5         Loader = PyPDFLoader
6
7     loader = Loader(file.path)
8     documents = loader.load()
9     docs = text_splitter.split_documents(documents)
10    for i, doc in enumerate(docs):
11        doc.metadata["source"] = f"source_{i}"
12    return docs
```

```
1 def get_vector_db(file: AskFileResponse):
2     docs = process_file(file)
3     cl.user_session.set("docs", docs)
4     vector_db = Chroma.from_documents(documents=docs,
5                                         embedding=embedding)
6     return vector_db
```

Build functions for the retriever.

❖ Coding step 3: Create Vicuna model

```
1 def get_huggingface_llm(model_name: str = "lmsys/vicuna-7b-v1.5",
2 |                         max_new_token: int = 512):
3     nf4_config = BitsAndBytesConfig(
4         load_in_4bit=True,
5         bnb_4bit_quant_type="nf4",
6         bnb_4bit_use_double_quant=True,
7         bnb_4bit_compute_dtype=torch.bfloat16
8     )
9     model = AutoModelForCausalLM.from_pretrained(
10        model_name,
11        quantization_config=nf4_config,
12        low_cpu_mem_usage=True
13    )
14     tokenizer = AutoTokenizer.from_pretrained(model_name)
15
16     model_pipeline = pipeline(
17         "text-generation",
18         model=model,
19         tokenizer=tokenizer,
20         max_new_tokens=max_new_token,
21         pad_token_id=tokenizer.eos_token_id,
22         device_map="auto"
23     )
24
25     llm = HuggingFacePipeline(
26         pipeline=model_pipeline,
27     )
28     return llm
29
30 LLM = get_huggingface_llm()
```



Build Chatbot UI with Chainlit

❖ Coding step 4: Setup on_chat_start()

```
1 @cl.on_chat_start
2 async def on_chat_start():
3     files = None
4     while files is None:
5         files = await cl.AskFileMessage(
6             content=welcome_message,
7             accept=["text/plain", "application/pdf"],
8             max_size_mb=20,
9             timeout=180,
10            ).send()
11     file = files[0]
12
13     msg = cl.Message(content=f"Processing `{file.name}`...",
14                      | disable_feedback=True)
15     await msg.send()
16
17     vector_db = await cl.make_async(get_vector_db)(file)
18
19     message_history = ChatMessageHistory()
20     memory = ConversationBufferMemory(
21         memory_key="chat_history",
22         output_key="answer",
23         chat_memory=message_history,
24         return_messages=True,
25     )
26     retriever = vector_db.as_retriever(search_type="mmr",
27                                         | search_kwargs={'k': 3})
```

```
29     chain = ConversationalRetrievalChain.from_llm(
30         llm=LLM,
31         chain_type="stuff",
32         retriever=retriever,
33         memory=memory,
34         return_source_documents=True
35     )
36
37     msg.content = f"`{file.name}` processed. You can now ask questions!"
38     await msg.update()
39
40     cl.user_session.set("chain", chain)
```

Setup Retriever and LLM when starting the chat.

Build Chatbot UI with Chainlit

❖ Coding step 5: Setup on_message()

```
1 @cl.on_message
2 async def on_message(message: cl.Message):
3     chain = cl.user_session.get("chain")
4     cb = cl.AsyncLangchainCallbackHandler()
5     res = await chain.invoke(message.content, callbacks=[cb])
6     answer = res["answer"]
7     source_documents = res["source_documents"]
8     text_elements = []
9
10    if source_documents:
11        for source_idx, source_doc in enumerate(source_documents):
12            source_name = f"source_{source_idx}"
13            text_elements.append(
14                cl.Text(content=source_doc.page_content,
15                       name=source_name)
16            )
17    source_names = [text_el.name for text_el in text_elements]
18
19    if source_names:
20        answer += f"\nSources: {', '.join(source_names)}"
21    else:
22        answer += "\nNo sources found"
23
24    await cl.Message(content=answer, elements=text_elements).send()
```



Use the following pieces of context to answer the question at the end. If you don't know the answer, just say that you don't know, don't try to make up an answer.

AI VIETNAM (AIO2024) aivietnam.edu.vn

1! mkdir datasets

2! unzip -q "/content /PlantDocv4 .zip" -d "/content /datasets /PlantDocv4 /"

3!rm /content /PlantDocv4 .zip

Quan sát thư mục giải nén, có thể thấy bộ dữ liệu này đã được gán nhãn và đưa vào format cấu trúc dữ liệu training theo yêu cầu của YOLO. Vì vậy, chúng ta sẽ không cần thực hiện bước chuẩn bị dữ liệu ở bài này.

2.Cài đặt và import các thư viện cần thiết: Tương tự như phần trước, các bạn chạy các đoạn code sau để cài đặt các gói thư viện để sử dụng được YOLOv10:

1!git clone https://github.com/THU-MIG/yolov10 .git

2%cd yolov10

3!pip install -q -r requirements .txt

4!pip install -e .

3.Khởi tạo mô hình YOLOv10: Chúng ta sẽ khởi tạo mô hình YOLOv10 với phiên bản nano (n) từ trọng số đã được huấn luyện trên bộ dữ liệu COCO. Để tải trọng số yolov10n.pt, các bạn chạy đoạn code sau:

1! wget https://github.com/THU-MIG/yolov10 /releases /download /v1 .1/ yolov10n .pt

Build Chatbot UI with Chainlit

❖ Result

Welcome to the PDF QA! To get started:

1. Upload a PDF or text file
2. Ask a question about the file

YOL0v10_Tutorials.pdf processed. You can now ask questions!

Cho tôi danh sách tên tác giả của bài viết latex này

Use the following pieces of context to answer the question at the end. If you don't know the answer, just say that you don't know, don't try to make up an answer.

Question: Cho tôi danh sách tên tác giả của bài viết latex này

Helpful Answer: The authors of the LaTeX document are Dinh-Thang Duong, Nguyen-Thuan Duong, Minh-Duc Bui, and Quang-Vinh Dinh.

Sources: source_0, source_1, source_2

Build Chatbot UI with Chainlit

❖ Result

1. Upload a PDF or text file
2. Ask a question about the file

YOLOv10_Tutorials.pdf processed. You can now ask questions!

Link github của YOLOv10 là gì?

❖ Use the following pieces of context to answer the question at the end. If you don't know the answer, just say that you don't know, don't try to make up an answer.

AI VIETNAM (AIO2024) aivietnam.edu.vn
1! mkdir datasets
2! unzip -q "/ content / PlantDocv4 . zip" -d "/ content / datasets / PlantDocv4
/"
3!rm / content / PlantDocv4 .zip
Quan sát thư mục giải nén, có thể thấy bộ dữ liệu này đã được gán nhãn và đưa vào format cấu trúc dữ liệu training theo yêu cầu của YOLO. Vì vậy, chúng ta sẽ không cần thực hiện bước chuẩn bị dữ liệu ở bài này.
2.Cài đặt và import các thư viện cần thiết: Tương tự như phần trước, các bạn chạy các đoạn code sau để cài đặt các gói thư viện để sử dụng được YOLOv10:
1!git clone https://github.com/THU-MIG/yolov10 .git
2%cd yolov10

Question: Link github của YOLOv10 là gì?
Helpful Answer: The link to the YOLOv10 GitHub repository is <https://github.com/THU-MIG/yolov10>.
Sources: source_0, source_1, source_2

Summary

In this lecture, we have discussed:

- 1. What is LLMs?**
- 2. What is RAG?**
- 3. How to implement a RAG application using LangChain**
 - 1. Learn how to initialize a LLM using HuggingFace.**
 - 2. Learn the basic of LangChain.**
 - 3. How to combine LLM and vector database to create a RAG application.**
- 4. How to implement a Chatbot application using Chainlit.**
- 5. Apply Chainlit to perform our RAG application.**

Question

