LangChain and HuggingFace: make your life easier!

V. Bencini
MLCF
13/12/2024

Huggingface 🤗 +Langchain 🦜🔗 LangChain

Huggingface: is a platform that collects tools to build ML applications

- Models can be accessed and used directly using a dedicated API
- Models for almost every task (https://huggingface.co/tasks)

LangChain: is a framework designed to simplify the creation of applications using large language models.

- API provides impressive amount of tools to preprocess data (data retrieving, embeddings, tokenizer ..) for LLMs and to use the models

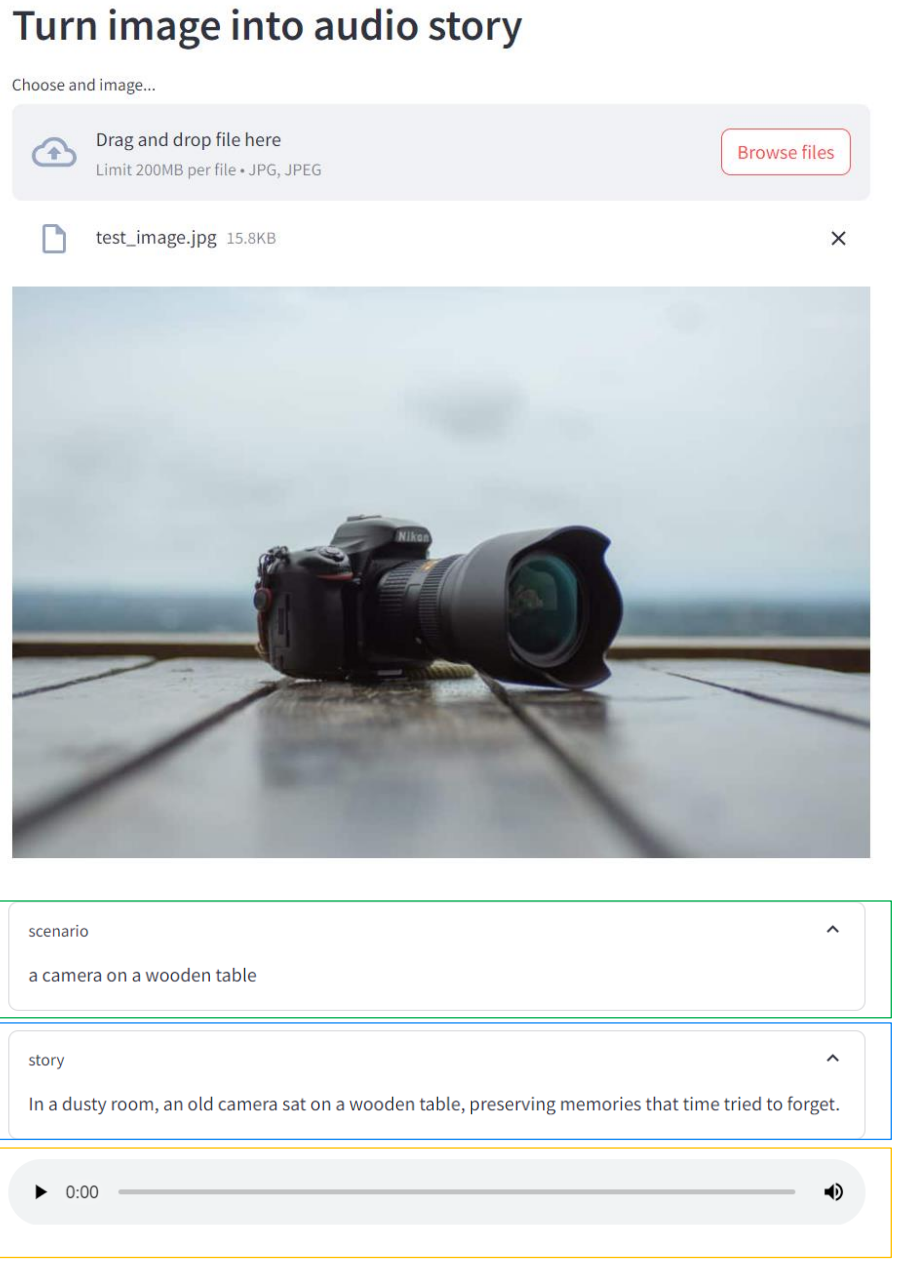
Huggingface +Langchain

Huggingface: is a platform that collects tools to build ML applications

- Models can be accessed and used directly using a dedicated API
- Models for almost every task (https://huggingface.co/tasks)

LangChain: is a framework designed to simplify the creation of applications using large language models.

- API provides impressive amount of tools to preprocess data (data retrieving, embeddings, tokenizer ..) for LLMs and to use the models



Example: Build an app that:

- Captions an image
- Creates a story based on the caption
- Generate a synthetic audio file telling the story

project source:
https://www.youtube.com/watch?v=_j7JEDWuqLE

Libraries

```
from dotenv import find_dotenv, load_dotenv
from transformers import pipeline
from langchain.prompts import PromptTemplate
from langchain.chains import LLMChain
from langchain.llms.huggingface_hub import HuggingFaceHub
import requests
import os
from langchain.llms import OpenAI
import streamlit as st
load_dotenv(find_dotenv())

OPENAI_API_TOKEN = os.getenv("OPENAI_API_TOKEN")
os.environ["OPENAI_API_KEY"] = OPENAI_API_TOKEN
HUGGINGFACEHUB_API_TOKEN = os.getenv("HUGGINGFACEHUB_API_TOKEN")
```

Story generation

```
def generate_story(scenario):
    template = '''
    You are a story teller;
    You can generate a short story based on a simple narrative, the story should
be no more than 20 words;

    CONTEXT: {scenario}
    STORY:
    '''

    prompt = PromptTemplate(template=template, input_variables=['scenario'])

    story_line = LLMChain(llm=OpenAI(temperature=1, model_name='gpt-3.5-turbo'),
prompt=prompt, verbose=True)

    story = story_line.run(scenario=scenario)
    print(story)
    return story
```

Image to text

```
def img2text(url):
    image_to_text = pipeline("image-to-text", model="Salesforce/blip-image-
captioning-base")

    text = image_to_text(url)[0]['generated_text']

    return text


scenario = img2text('test_image.jpg')
```

Text to speech

```
def text2speach(message):

    API_URL = "https://api-inference.huggingface.co/models/espnet/kan-
bayashi_ljspeech_vits"
    headers = {"Authorization": f"Bearer {HUGGINGFACEHUB_API_TOKEN}"}
    def query(payload):
        response = requests.post(API_URL, headers=headers, json=payload)
        return response

    response = query({
        "inputs": message,
    })
    # You can access the audio with IPython.display for example
    with open('audio.flac', 'wb') as file:
        file.write(response.content)
```

Useful resources

Projects: Image to speech: https://gitlab.cern.ch/vbencini/image_to_speach
Object detection: https://gitlab.cern.ch/vbencini/object_detection

Huggingface:

Website:
https://huggingface.co/

Education:
https://huggingface.co/learn
https://www.youtube.com/watch?v=_j7JEDWuqLE

LangChain:

Website:
https://python.langchain.com/docs/get_started/introduction

Education:
https://www.deeplearning.ai/short-courses/langchain-for-llm-application-development/
https://www.deeplearning.ai/short-courses/langchain-chat-with-your-data/