Computer Vision
hugomcp@di.ubi.pt, 2024/25

# Ficha Prática 8

## *Large Language Models*

Some platforms (like Hugging Face Inference API) provide access to LLaVA models without needing local setup.

**Option 1: Using LLaVA via an Online API**

Step 1: Get API Access

1. Sign up on Hugging Face.
2. Go to LLaVA models.
3. If there's an API endpoint, obtain the token.

Step 2: Use Python to Call the API

```
import requests
API_URL = "https://api-inference.huggingface.co/models/liuhaotian/llava-v1.5-7b"
HEADERS = {"Authorization": "Bearer YOUR_HF_API_KEY"}
def query_llava(text):
    payload = {"inputs": text}
    response = requests.post(API_URL, headers=HEADERS, json=payload)
    return response.json()
print(query_llava("Describe the content of an image."))
```

**Option 2: Using LLaVA Locally**

Step 1: Install Dependencies

Ensure you have PyTorch installed:

```
pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu121
```

Install other dependencies:

```
pip install transformers accelerate bitsandbytes
pip install git+https://github.com/huggingface/transformers.git
```

Step 2: Download and Load LlaVA

```
from transformers import AutoProcessor, AutoModelForCausalLM
import torch
model_name = "liuhaotian/llava-v1.5-7b"
device = "cuda" if torch.cuda.is_available() else "cpu"
processor = AutoProcessor.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name,
torch_dtype=torch.float16).to(device)
def llava_inference(prompt):
   inputs = processor(text=prompt, return_tensors="pt").to(device)
   output = model.generate(**inputs, max_new_tokens=100)
   return processor.batch_decode(output, skip_special_tokens=True)[0]
print(llava_inference("Hello, LLaVA!"))
```

1) **Unimodal Text Prompting.** Create a Pyhton script that interacts with the LLM, using text queries.
    a. Start with a vague prompt (e.g., “Explain blockchain”).
    b. Analyze its shortcomings (e.g., lack of depth, missing aspects).
    c. Apply refinement techniques:
        i. Specify required details (e.g., “Explain blockchain focusing on consensus mechanisms”).
        ii. Use delimiters (e.g., “Provide a structured response: Introduction | Key Components | Challenges”).
    d. Add role-based instructions (e.g., “Act as a cybersecurity expert”).
    e. Compare responses before and after refinement.
    f. Repeat until an optimal prompt is achieved.

2) **Image Generation.** Create a Python script that asks for a specific kind of image. Then, based in the feedback provided by the user, the model should regenerate the image, to obtain the specific desired features

3) **Multimodal Prompting.** Create a Python script that fuses image plus text data as input, in order to extract meaningful information from the image.
    a. Ask about specific properties of the image being given as input.

b. Ask the LLM to edit the given image, in a specoified direction.