```
!pip install accelerate
```

```
!pip install -Uqqq
!pip -qqq install bitsandbytes accelerate
```

```
ERROR: You must give at least one requirement to install (see "pip help install")
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 102.2/102.2 MB 8.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 290.1/290.1 kB 13.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 23.7/23.7 MB 25.8 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 823.6/823.6 kB 31.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 14.1/14.1 MB 49.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 731.7/731.7 MB 1.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 410.6/410.6 MB 2.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 121.6/121.6 MB 8.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.5/56.5 MB 14.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 124.2/124.2 MB 8.4 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 196.0/196.0 MB 6.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 166.0/166.0 MB 6.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 99.1/99.1 kB 14.8 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 75.5 MB/s eta 0:00:00
```

```
from transformers import AutoProcessor, LlavaForConditionalGeneration, AutoTokenizer, AutoImageProcessor
import torch
from PIL import Image
import requests

# MODEL_PATH = "/content/drive/MyDrive/llava-1.5-7b-hf"
MODEL_PATH = "llava-hf/llava-1.5-7b-hf"
model = LlavaForConditionalGeneration.from_pretrained(MODEL_PATH, torch_dtype=torch.float16,low_cpu_mem_usage
# model = LlavaForConditionalGeneration.from_pretrained(MODEL_PATH, low_cpu_mem_usage=True)
model.to("cuda:0")# this runs out of memory?

# processor = LlavaNextProcessor.from_pretrained(MODEL_PATH, cache_dir="./cache")
tokenizer = AutoTokenizer.from_pretrained(MODEL_PATH)
imgprocessor = AutoImageProcessor.from_pretrained(MODEL_PATH)
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:88: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/se
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
config.json: 100%  950/950 [00:00<00:00, 23.6kB/s]
model.safetensors.index.json: 100%  70.1k/70.1k [00:00<00:00, 1.60MB/s]
Downloading shards: 100%  3/3 [02:14<00:00, 41.59s/it]
model-00001-of-00003.safetensors: 100%  4.99G/4.99G [00:52<00:00, 139MB/s]
model-00002-of-00003.safetensors: 100%  4.96G/4.96G [00:52<00:00, 112MB/s]
model-00003-of-00003.safetensors: 100%  4.18G/4.18G [00:27<00:00, 177MB/s]
Loading checkpoint shards: 100%  3/3 [00:01<00:00, 1.90it/s]
generation_config.json: 100%  141/141 [00:00<00:00, 10.2kB/s]
tokenizer_config.json: 100%  1.33k/1.33k [00:00<00:00, 89.9kB/s]
tokenizer.model: 100%  500k/500k [00:00<00:00, 31.7MB/s]
tokenizer.json: 100%  1.84M/1.84M [00:00<00:00, 6.87MB/s]
added_tokens.json: 100%  41.0/41.0 [00:00<00:00, 1.98kB/s]
special_tokens_map.json: 100%  438/438 [00:00<00:00, 18.9kB/s]
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned
preprocessor_config.json: 100%  557/557 [00:00<00:00, 20.1kB/s]
```

```
def embed_t():
  # prepare image and text prompt, using the appropriate prompt template
  image = Image.open("/content/drive/MyDrive/image/clean.jpeg")
  prompt = "A chat between a curious human and an artificial intelligence assistant. The assistant gives helpfu

  # get text embedding
  input_ids = tokenizer(prompt, return_tensors="pt").input_ids.to("cuda:0")
  input_embeds = model.get_input_embeddings()(input_ids).cpu()
  return input_embeds

t_embed = embed_t()
print(t_embed)
print(t_embed.shape)
```

```
tensor([[[ 0.0045, -0.0038,  0.0017,  ..., -0.0088,  0.0025, -0.0025],
         [-0.0112, -0.0129, -0.0121,  ...,  0.0090,  0.0118, -0.0081],
         [ 0.0195, -0.0058,  0.0061,  ...,  0.0171, -0.0052, -0.0212],
         ...,
         [-0.0187, -0.0017,  0.0177,  ...,  0.0238,  0.0052,  0.0101],
         [ 0.0066, -0.0161,  0.0117,  ..., -0.0103,  0.0148,  0.0073],
         [ 0.0039,  0.0015,  0.0055,  ..., -0.0042,  0.0151,  0.0024]]],
       dtype=torch.float16, grad_fn=<ToCopyBackward0>)
torch.Size([1, 50, 4096])
```

Resources ✕ •••

You are not subscribed. Learn more
You currently have zero compute units available. Resources offered free of charge are not guaranteed. Purchase more units here.

Manage sessions

Python 3 Google Compute Engine backend (GPU)
Showing resources from 2:34 PM to 2:46 PM

System RAM
2.6 / 12.7 GB

GPU RAM
14.2 / 15.0 GB

Disk
42.4 / 78.2 GB

```
def embed_i(image):
  # get image embedding
  pixel_value = imgprocessor(image, return_tensors="pt").pixel_values.to("cuda:0")
  image_outputs = model.vision_tower(pixel_value, output_hidden_states=True)
  # print(image_outputs.hidden_states)
  # Tuple of torch.FloatTensor (one for the output of the embeddings, if the model
  # has an embedding layer, + one for the output of each layer) of
  # shape (batch_size, sequence_length, hidden_size).
  selected_image_feature = image_outputs.hidden_states[model.config.vision_feature_layer]
  selected_image_feature = selected_image_feature[:, 1:] # by default
  image_features = model.multi_modal_projector(selected_image_feature).cpu()
  return image_features

img = Image.open("/content/drive/MyDrive/image/clean.jpeg")
i_embed = embed_i(img)
print(i_embed)
print(i_embed.shape)
```

```
tensor([[[-0.2830,  0.1919, -0.3965,  ...,  0.0248, -0.0055, -0.0535],
         [ 0.8623,  0.6191, -0.4436,  ..., -0.1350, -0.2805, -0.1148],
         [ 1.1797, -0.1567, -0.7183,  ...,  0.0803,  0.7251,  0.5850],
         ...,
         [ 0.9492, -0.2242, -0.6724,  ...,  0.0057,  0.7280,  0.4758],
         [ 0.0447,  0.4231, -0.2920,  ..., -0.1610,  0.0742, -0.2947],
         [ 0.2825,  0.2184, -0.2264,  ..., -0.2017,  0.2123, -0.3271]]],
       dtype=torch.float16, grad_fn=<ToCopyBackward0>)
torch.Size([1, 576, 4096])
```

Change runtime type