```
%cd /content/drive/MyDrive/ImageBind finetune/ImageBind-LoRA
```

```
/content/drive/MyDrive/ImageBind finetune/ImageBind-LoRA
```

```
!pip install -r requirements.txt --quiet
```

```
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 890.1/890.1 MB 1.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.3/24.3 MB 64.8 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.2/4.2 MB 97.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 510.0/510.0 kB 50.9 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 54.4/54.4 kB 8.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 43.2/43.2 kB 6.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 50.2/50.2 kB 7.6 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.6/13.6 MB 98.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 42.2/42.2 kB 6.3 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.1/7.1 MB 118.5 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 11.6/11.6 MB 118.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 85.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 849.3/849.3 kB 64.9 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 557.1/557.1 MB 2.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 317.1/317.1 MB 2.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.0/21.0 MB 82.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 66.4/66.4 kB 9.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 80.8/80.8 kB 12.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 55.5/55.5 kB 8.4 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.2/1.2 MB 73.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 71.9/71.9 kB 11.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 868.8/868.8 kB 64.4 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.4/62.4 kB 8.8 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 129.9/129.9 kB 17.3 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 812.3/812.3 kB 65.1 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 34.3/34.3 MB 52.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 230.0/230.0 kB 31.3 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 268.9/268.9 kB 33.4 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 75.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.1/5.1 MB 106.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.5/1.5 MB 18.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.1/3.1 MB 95.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 64.3/64.3 kB 10.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.4/58.4 kB 8.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 139.2/139.2 kB 19.4 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.3/58.3 kB 7.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 812.3/812.3 kB 65.7 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 812.3/812.3 kB 66.0 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 812.2/812.2 kB 63.7 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 802.3/802.3 kB 55.3 MB/s eta 0:00:0
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.4/12.4 MB 106.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 82.7/82.7 kB 12.4 MB/s eta 0:00:00
  Building wheel for pytorchvideo (setup.py) ... done
  Building wheel for fvcore (setup.py) ... done
  Building wheel for iopath (setup.py) ... done
  Building wheel for mayavi (pyproject.toml) ... done
ERROR: pip's dependency resolver does not currently take into account all the pac
torchtext 0.18.0 requires torch>=2.3.0, but you have torch 1.13.0 which is incomp
```

```
import os
current_directory = os.getcwd()
```

Resources

You are not subscribed. Learn more
Available: 49.34 compute units
Usage rate: approximately 4.82 per hour
You have 1 active session.

Manage sessions

Want more memory and disk space? Upgrade to Colab Pro

Not connected to runtime.

Change runtime type

```
import torch
import logging
import data
from models import imagebind_model
from models.imagebind_model import ModalityType, load_module
from models import lora as LoRA

logging.basicConfig(level=logging.INFO, force=True)

lora = False
linear_probing = False
device = "cuda:0" if torch.cuda.is_available() else "cpu"
load_head_post_proc_finetuned = True

assert not (linear_probing and lora), \
    "Linear probing is a subset of LoRA training procedure for ImageBind. " \
    "Cannot set both linear_probing=True and lora=True. "

if lora and not load_head_post_proc_finetuned:
    lora_factor = 12 / 0.07
else:
    lora_factor = 1
```

```
/usr/local/lib/python3.10/dist-packages/torchvision/transforms/_functional_video.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/torchvision/transforms/_transforms_video.
  warnings.warn(
```

```
image_paths = ["test_data/00088_2.jpg", "test_data/00095.jpg", "test_data/00158_4.jpg",
               "test_data/00169_3.jpg", "test_data/00220.jpg", "test_data/00222_3.jpg"]
audio_paths = ["test_data/00088_2.wav", "test_data/00095.wav", "test_data/00158_4.wav",
               "test_data/00169_3.wav", "test_data/00220.wav", "test_data/00222_3.wav"]
```

```
for i in range(len(image_paths)):
    image_paths[i] = current_directory + "/" + image_paths[i]
for i in range(len(audio_paths)):
    audio_paths[i] = current_directory + "/" + audio_paths[i]
```

```
model = imagebind_model.imagebind_huge(pretrained=True)
```

```
# if lora:
#     model.modality_trunks.update(
#         LoRA.apply_lora_modality_trunks(model.modality_trunks, rank=4,
#                                         modality_names=[ModalityType.TEXT, Modality

#     LoRA.load_lora_modality_trunks(model.modality_trunks,
#                                    checkpoint_dir=".checkpoints/lora/", postfix="_l

#     if load_head_post_proc_finetuned:
#         load_module(model.modality_postprocessors, module_name="postprocessors",
#                     checkpoint_dir=".checkpoints/lora/", postfix="_last")
#         load_module(model.modality_heads, module_name="heads",
#                     checkpoint_dir=".checkpoints/lora/", postfix="_last")
# elif linear_probing:
#     load_module(model.modality_heads, module_name="heads",
#                 checkpoint_dir="./.checkpoints/lora/", postfix="_last")
```

```
model.eval()
model.to(device)
```

```
            (mlp): Mlp(
              (fc1): Linear(in_features=768, out_features=3072, bias=True)
              (act): GELU(approximate='none')
              (fc2): Linear(in_features=3072, out_features=768, bias=True)
              (drop): Dropout(p=0.0, inplace=False)
            )
            (norm_2): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
          )
          (6): BlockWithMasking(
            (attn): MultiheadAttention(
              (out_proj): NonDynamicallyQuantizableLinear(in_features=768,
out_features=768, bias=True)
            )
            (drop_path): DropPath(drop_prob=0.055)
            (norm_1): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
            (mlp): Mlp(
              (fc1): Linear(in_features=768, out_features=3072, bias=True)
              (act): GELU(approximate='none')
              (fc2): Linear(in_features=3072, out_features=768, bias=True)
              (drop): Dropout(p=0.0, inplace=False)
            )
            (norm_2): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
          )
          (7): BlockWithMasking(
            (attn): MultiheadAttention(
              (out_proj): NonDynamicallyQuantizableLinear(in_features=768,
out_features=768, bias=True)
            )
            (drop_path): DropPath(drop_prob=0.064)
            (norm_1): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
            (mlp): Mlp(
              (fc1): Linear(in_features=768, out_features=3072, bias=True)
              (act): GELU(approximate='none')
              (fc2): Linear(in_features=3072, out_features=768, bias=True)
              (drop): Dropout(p=0.0, inplace=False)
            )
            (norm_2): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
          )
          (8): BlockWithMasking(
            (attn): MultiheadAttention(
              (out_proj): NonDynamicallyQuantizableLinear(in_features=768,
out_features=768, bias=True)
            )
            (drop_path): DropPath(drop_prob=0.073)
            (norm_1): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
            (mlp): Mlp(
```

```
# Load data
#audio_query_paths = [current_directory + "/test_data/00222_3.wav"]  # Example: Choos

inputs_audio = {
    ModalityType.AUDIO: data.load_and_transform_audio_data(audio_paths, device),
}

inputs_image = {
    ModalityType.VISION: data.load_and_transform_vision_data(image_paths, device),
}

# Generate embeddings for audio query
with torch.no_grad():
    embeddings_audio = model(inputs_audio)
with torch.no_grad():
    embeddings_image = model(inputs_image)

vision_embeddings = embeddings_image[ModalityType.VISION]
audio_query_embedding = embeddings_audio[ModalityType.AUDIO]

count_crt=0
for audio_index, audio_query_embedding in enumerate(embeddings_audio['audio']):

  similarity_scores = torch.softmax(vision_embeddings @ audio_query_embedding.T * (lo

  most_relevant_image_index = torch.argmax(similarity_scores)

  most_relevant_image_path = image_paths[most_relevant_image_index]

  if most_relevant_image_index==audio_index:
    count_crt+=1

  print("Audio query:",audio_paths[audio_index].split('/')[-1] )
  print("Most relevant image for audio query:", most_relevant_image_path.split('/')[-
```

```
Audio query: 00088_2.wav
Most relevant image for audio query: 00222_3.jpg
Audio query: 00095.wav
Most relevant image for audio query: 00095.jpg
Audio query: 00158_4.wav
Most relevant image for audio query: 00169_3.jpg
Audio query: 00169_3.wav
Most relevant image for audio query: 00222_3.jpg
Audio query: 00220.wav
Most relevant image for audio query: 00220.jpg
Audio query: 00222_3.wav
Most relevant image for audio query: 00220.jpg
```

```
count_crt
```

```
2
```

```
for index, audio_query_emb in enumerate(embeddings_audio['audio']):
  print(index, audio_query_emb)
```

```
0 tensor([ 0.0478, -0.2131,  0.2315,  ...,  0.5279, -0.7604, -0.0132],
       device='cuda:0')
1 tensor([ 0.2381, -0.0772, -0.2055,  ..., -0.4301,  0.3229, -0.4043],
       device='cuda:0')
2 tensor([ 0.0859, -0.0992,  0.6365,  ..., -0.1347, -0.8802,  0.0288],
       device='cuda:0')
3 tensor([-0.3175, -0.6430, -0.4416,  ...,  0.0071,  0.6742,  0.1453],
       device='cuda:0')
4 tensor([ 0.5646, -0.1296, -0.1822,  ..., -0.1235,  0.5655,  0.2960],
       device='cuda:0')
5 tensor([-0.2691, -0.3681, -0.1953,  ..., -0.4585,  0.3020,  0.2901],
       device='cuda:0')
```

```
vision_embeddings.shape
```

```
torch.Size([6, 1024])
```

```
vision_embeddings
```

```
tensor([[ 0.0016, -0.0102,  0.0057,  ..., -0.0128,  0.0134,  0.0019],
        [ 0.0013, -0.0126,  0.0278,  ...,  0.0009,  0.0452, -0.0210],
        [ 0.0209,  0.0147,  0.0075,  ..., -0.0229,  0.0221, -0.0286],
        [-0.0012,  0.0201, -0.0152,  ...,  0.0071, -0.0067, -0.0100],
        [-0.0206, -0.0143,  0.0086,  ...,  0.0110,  0.0229, -0.0065],
        [-0.0201,  0.0095,  0.0293,  ...,  0.0029, -0.0007, -0.0067]],
       device='cuda:0')
```