Zero-shot image classification

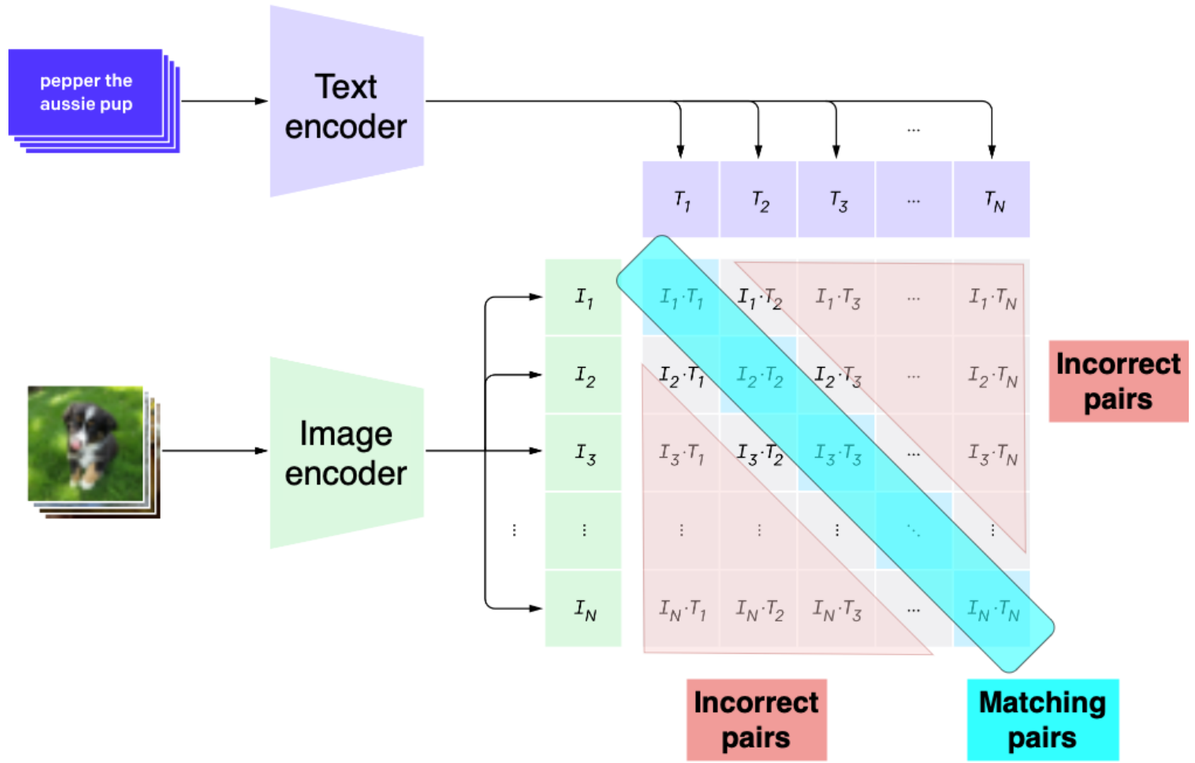
MULTI-MODAL MODELS WITH HUGGING FACE

James Chapman
Curriculum Manager, DataCamp

datacamp

CLIP

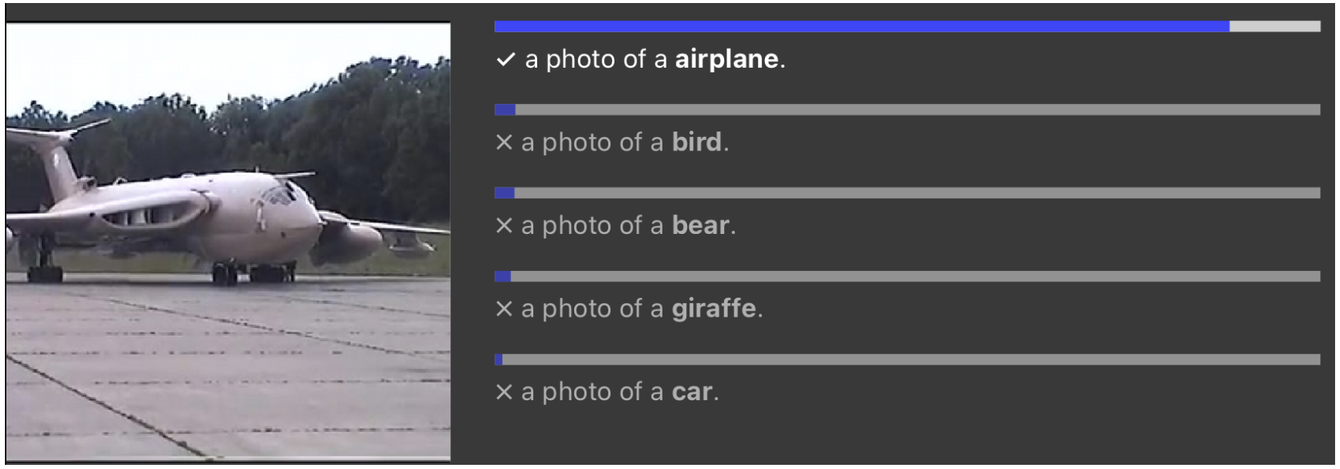
- Contrastive Language-Image Pre-training
- Score similarity between images and text
- Trained on 400M image-text pairs
- Two encoders:
  - Vision encoder
  - Text encoder
- Close image-text matches produce similar arrays



[1] https://openai.com/index/clip/

Zero-shot learning

- Perform tasks that the model wasn't trained for



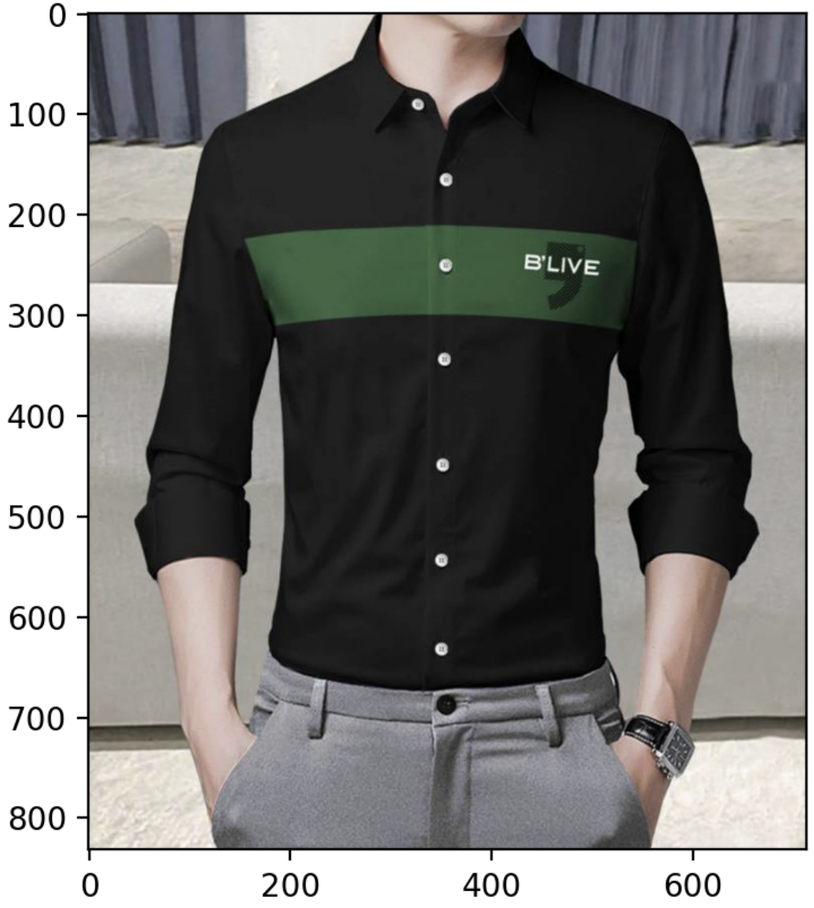
[1] https://openai.com/index/clip/

Use case: product categorization

```
from datasets import load_dataset
import matplotlib.pyplot as plt

dset = "rajuptvs/ecommerce_products_clip"
dataset = load_dataset(dset)
print(dataset["train"][0]["Description"])
plt.imshow(dataset["train"][0]["image"])
plt.show()
```

```
Blive High quality premium Full sleeves printed
Shirt direct from the manufacturers.Gives you
a clean and classy look while also
making you feel comfortable.Trusted
brand online and no compromise on quality.
```



Zero-shot learning with CLIP

```
model = CLIPModel.from_pretrained("openai/clip-vit-base-patch32")
processor = CLIPProcessor.from_pretrained("openai/clip-vit-base-patch32")

categories = ["shirt", "trousers", "shoes", "dress", "hat", "bag", "watch"]

inputs = processor(text=categories, images=dataset["train"][0]["image"],
                   return_tensors="pt", padding=True)
outputs = model(**inputs)

probs = outputs.logits_per_image.softmax(dim=1)
categories[probs.argmax().item()]
```

```
shirt
```

The CLIP score

- Similarity of encoded image and encoded description
- Range from `100` (perfect agreement) to `0` (no agreement)

```
from torchmetrics.functional.multimodal import clip_score

image = dataset["train"][0]["image"]
description = dataset["train"][0]["Description"]

from torchvision.transforms import ToTensor
image = ToTensor()(image)*255

score = clip_score(image, description, "openai/clip-vit-base-patch32")
print(f"CLIP score: {score}")
```

```
CLIP score: 28.495952606201172
```

Let's practice!

MULTI-MODAL MODELS WITH HUGGING FACE

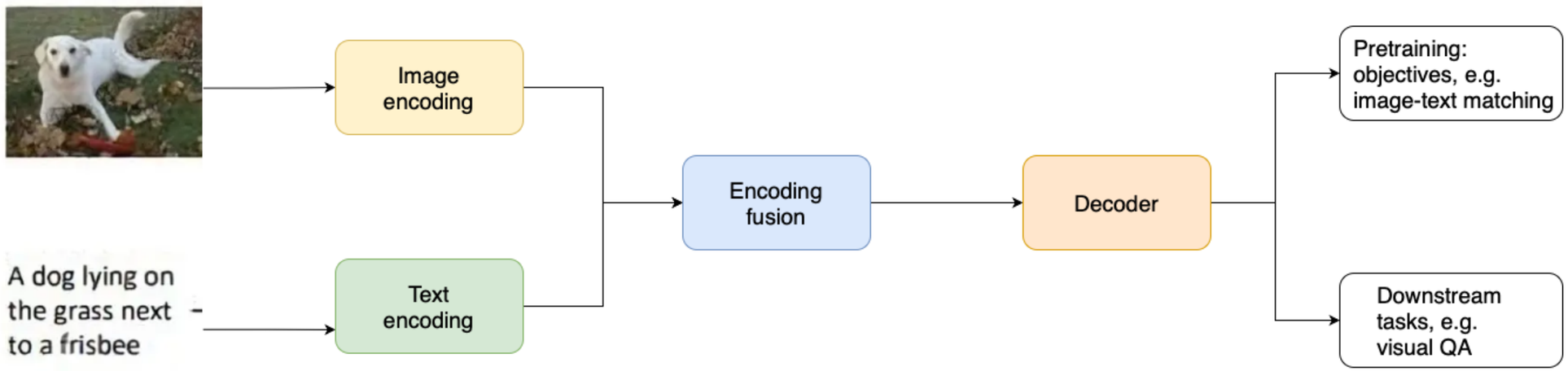
Multi-modal sentiment analysis

MULTI-MODAL MODELS WITH HUGGING FACE

James Chapman
Curriculum Manager, DataCamp

datacamp

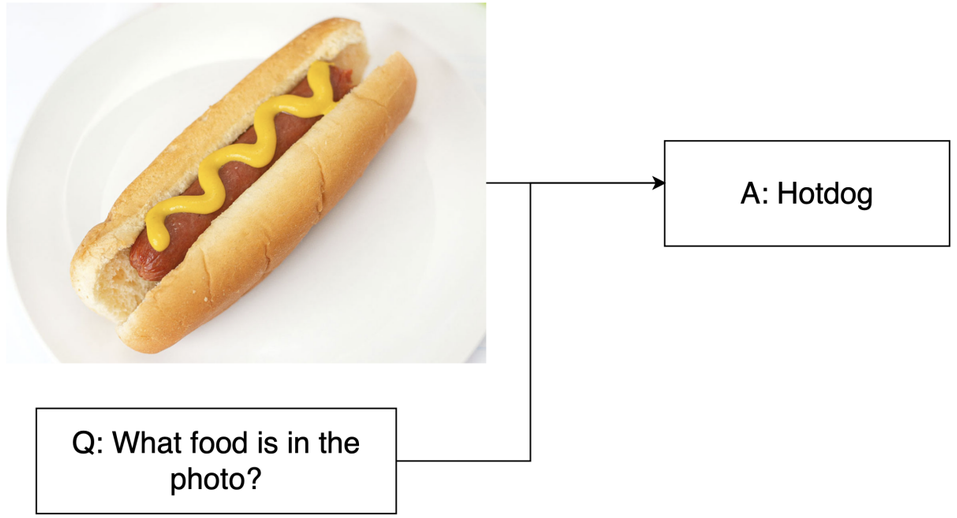
Vision Language Models (VLMs)



- Image and text separately encoded
- Feature fusion
- Create shared representations
  - Multiple tasks from one model

Visual reasoning tasks

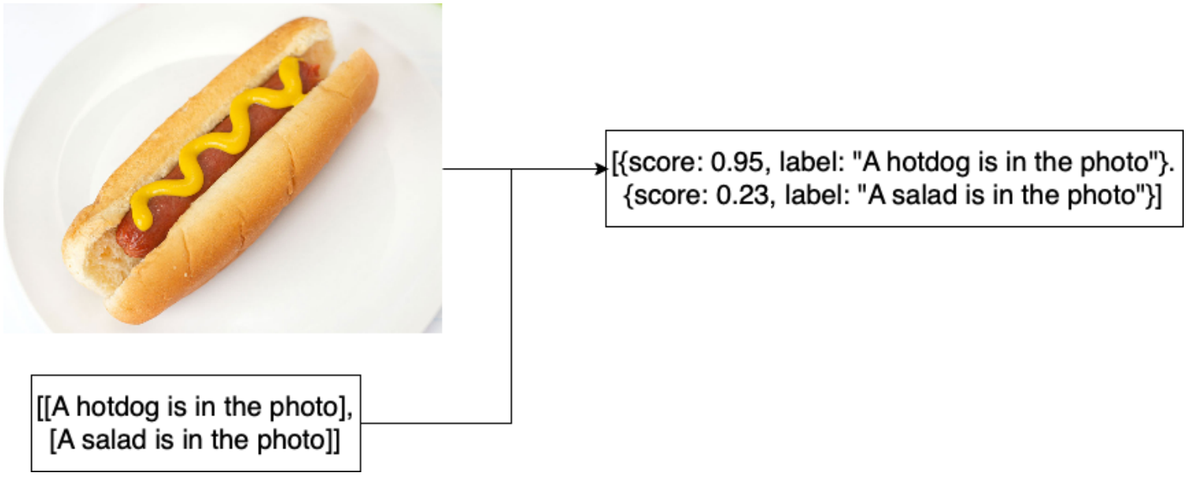
- **VQA**: Image + question → Direct answers about image content



[1] https://arxiv.org/pdf/1505.00468

Visual reasoning tasks

- **VQA:** Image + question → Direct answers about image content
- **Matching:** Image + statement → True/False validations



Visual reasoning tasks

- **VQA:** Image + question → Direct answers about image content
- **Matchings:** Image + statement → True/False validations
- **Entailment:** Image + contrasting texts → Semantic relationship checks

Use case: share price impact

```
from datasets import load_dataset
dset = "RealTimeData/bbc_news_alltime"
dataset = load_dataset(dset, '2017-01',
                       split="train")

image = dataset[87]["top_image"]
content = dataset[87]["content"]
print(content)
```

```
'Ford\'s decision to cancel a $1.6bn
investment in Mexico and
invest an extra $700m in Michigan ...
```

Qwen 2 VLMs

```
from transformers import Qwen2VLForConditionalGeneration
from qwen_vl_utils import process_vision_info

vl_model = Qwen2VLForConditionalGeneration.from_pretrained(
    "Qwen/Qwen2-VL-2B-Instruct", device_map="auto", torch_dtype="auto"
)
```

The preprocessor

```
from transformers import Qwen2VLProcessor


min_pixels = 224 * 224
max_pixels = 448 * 448
vl_model_processor = Qwen2VLProcessor.from_pretrained(
    "Qwen/Qwen2-VL-2B-Instruct", min_pixels=min_pixels, max_pixels=max_pixels
)
```

Multi-modal prompts

```
text_query = f"Is the sentiment of the following content good or bad for the Ford share price:
{article_text}. Provide reasoning."

chat_template = [
    {
        "role": "user",
        "content": [
            {"type": "image", "image": article_image},
            {"type": "text", "text": text_query}
        ]
    }
]
```

VLM classification

```
text = vl_model_processor.apply_chat_template(chat_template, tokenize=False,
                                              add_generation_prompt=True)
image_inputs, _ = process_vision_info(chat_template)

inputs = vl_model_processor(text=[text], images=image_inputs, padding=True,
                            return_tensors="pt")

generated_ids = vl_model.generate(**inputs, max_new_tokens=500)
generated_ids_trimmed = [out_ids[len(in_ids) :] for in_ids, out_ids\
                         in zip(inputs.input_ids, generated_ids)]
```

VLM classification

```
output_text = vl_model_processor.batch_decode(
    generated_ids_trimmed, skip_special_tokens=True, clean_up_tokenization_spaces=False
)
print(output_text[0])
```

```
The sentiment of the provided text is negative. The author is expressing concern and
skepticism ...
```

Let's practice!

MULTI-MODAL MODELS WITH HUGGING FACE

Zero-shot video classification

MULTI-MODAL MODELS WITH HUGGING FACE

James Chapman
Curriculum Manager, DataCamp

datacamp

Zero-shot video classification

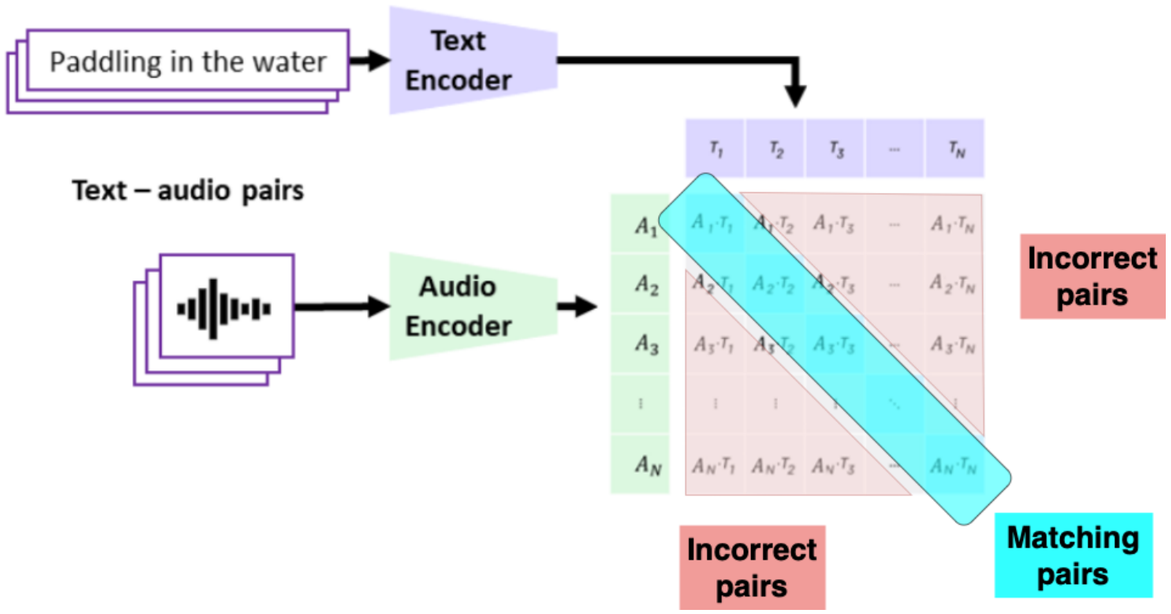
Business case:

- A competitor's ads are leading to more sales
- Is our competitor more positive in ads?

[1] https://repository.duke.edu/dc/adviews/dmbb37204

CLAP

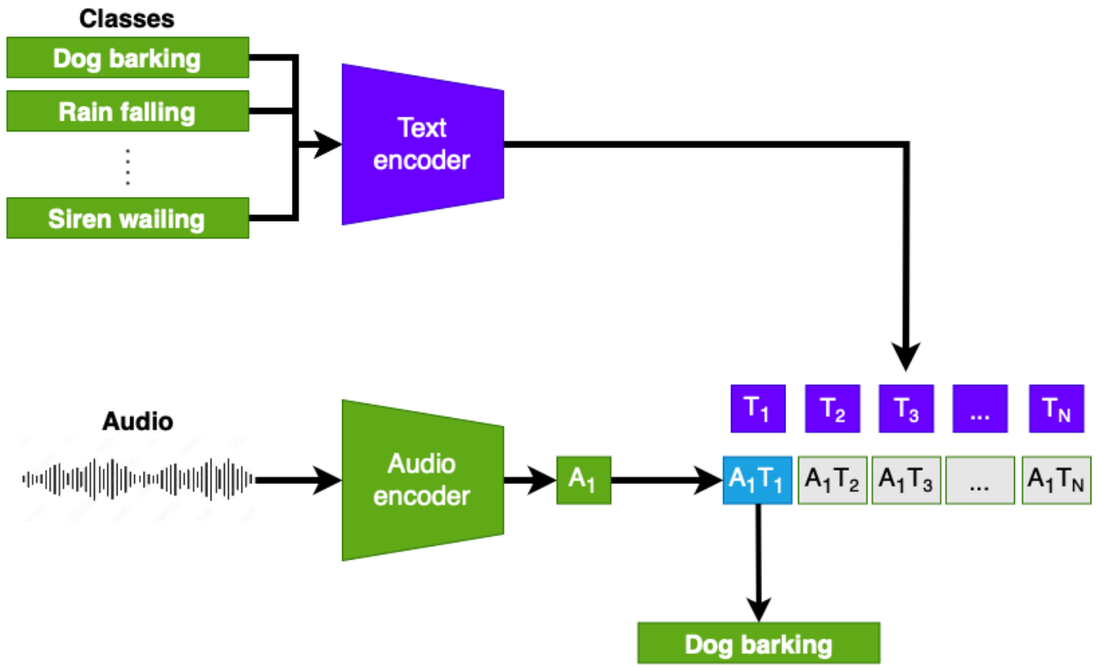
- Contrastive Language Audio Pretraining
- Analogous to method used to train CLIP
  - Separate text encoder and audio encoder
- **633k** matched pairs of audio + descriptions
- Contrastive pretraining aligns audio + text embeddings



[1] https://arxiv.org/html/2211.06687v4

Approach: multi-modal ZSL

Zero-shot learning with **CLAP**



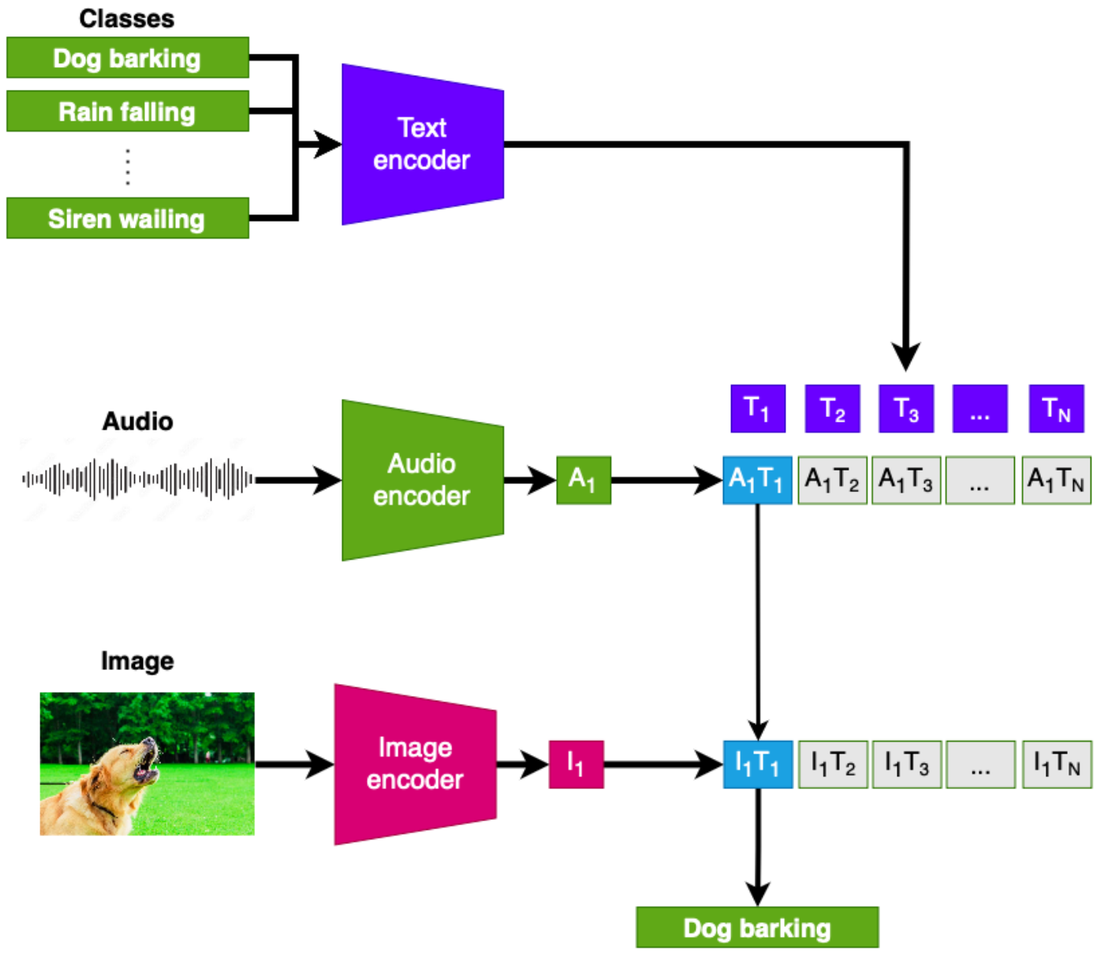
[1] https://arxiv.org/html/2310.02298v3

Approach: multi-modal ZSL

Zero-shot learning with **CLAP**

+ zero-shot learning with **CLIP**

- Common text classes
- Combined probabilities from CLIP and CLAP



Video and audio

Separating `.mp4` audio and video streams with `moviepy`:

```
from moviepy.editor import VideoFileClip
from moviepy.video.io.ffmpeg_tools import ffmpeg_extract_subclip

ffmpeg_extract_subclip("advert.mp4", 0, 5, "advert_5s.mp4")

video = VideoFileClip("advert_5s.mp4")
audio = video.audio
audio.write_audiofile("advert_5s.mp3")
```

Preparing the audio and video

```
from decord import VideoReader
from PIL import Image
video_reader = VideoReader(video_path)
video = video_reader.get_batch(range(20)).asnumpy()
video = video[:, :, :, ::-1]
video = [Image.fromarray(frame) for frame in video]

from datasets import Dataset, Audio
audio_dataset = Dataset.from_dict({"audio": [audio_path]}).cast_column("audio", Audio())
audio_sample = audio_dataset[0]["audio"]["array"]
```

Video predictions

```
emotions = ["joy", "fear", "anger", "sadness", "disgust", "surprise", "neutral"]

image_class = pipeline(model="openai/clip-vit-large-patch14",
                       task="zero-shot-image-classification")

predictions = image_classifier(video, candidate_labels=emotions)
scores = [
    {l['label']: l['score'] for l in prediction}
    for prediction in predictions
]

avg_image_scores = {emotion: sum([s[emotion] for s in scores])/len(scores) for emotion in emotions}
print(f"Average scores: {avg_image_scores}")
```

```
Average scores: {'joy': 0.100633262679912 16, 'fear': 0.0868348691612482, ...}
```

Audio predictions and combination

```
audio_class = pipeline(model="laion/clap-htsat-unfused",
                       task="zero-shot-audio-classification")

audio_scores = audio_class(audio_sample, candidate_labels=emotions)

audio_scores = {l['label']: l['score'] for l in audio_scores}
multimodal_scores = {emotion: (avg_image_scores[emotion] + audio_scores[emotion])/2
                     for emotion in emotions}
print(f"Multimodal scores: {multimodal_scores}")
```

```
Multimodal scores: {'joy': 0.3109628591220826, 'fear': 0.09013736313208938,
'anger': 0.0114543550764210 53, 'sadness': 0.060181018337607 38, 'disgust': 0.07207315033301712,
'surprise': 0.252118631079793, 'neutral': 0.2030726027674973}
```

Let's practice!

MULTI-MODAL MODELS WITH HUGGING FACE