Crawl Image Directory

Introduction

In other tutorials, we demonstrated how to start a video stream and run inference, a common scenario. But sometimes you might want to run inference on images instead of videos. This tutorial will demonstrate how to do that using BrainFrame.

The use case in this tutorial is pretty simple. We want to iterate over all images in a directory to find the ones with cats in them. You can find the complete script and sample images on our GitHub repository.

Setup The Environment

In a previous tutorial, we installed BrainFrame server, client, and Python API libraries. The API functions we are going to use in this tutorial are:

- `api.get_plugins(...)`
- `api.process_image(...)`

In this tutorial, we will use one of our publicly available capsules: `detector_people_and_vehicles_fast`. You can download it from our downloads page.

Before we start, you should already have the BrainFrame server and client running, and the capsule downloaded.

Check the Existing Capsules

As usual, let's import the dependencies first:

```
from pathlib import Path

import cv2

from brainframe.api import BrainFrameAPI
```

And connect to the server:

```
api = BrainFrameAPI("http://localhost")
```

Before we start processing images, we want to check the existing capsules to verify that `detector_people_and_vehicles_fast` is loaded:

```
# Get the names of existing capsules
loaded_capsules = api.get_plugins()
loaded_capsules_names = [capsule.name for capsule in loaded_capsules]

# Print out the capsules names
print(f"Loaded Capsules: {loaded_capsules_names}")
```

Make sure `detector_people_and_vehicles_fast` is present.

```
Loaded Capsules: ['detector_people_and_vehicles_fast']
```

You can also check the loaded capsules using the client.

Iterate through the Image Directory

With the capsule loaded, we can iterate over all the images in the directory, and get the inference results for each image. Then we will filter for detections with `class_name == "cat"`.

```
 # Root directory containing the images.
IMAGE_ARCHIVE = Path("../images")

# Iterate through all images in the directory
for image_path in IMAGE_ARCHIVE.iterdir():
    # Use only PNGs and JPGs
    if image_path.suffix not in [".png", ".jpg"]:
        continue

    # Get the image array
    image_array = cv2.imread(str(image_path))

    # Perform inference on the image and get the results
    detections = api.process_image(
        # Image array
        img_bgr=image_array,
        # The names of capsules to enable while processing the image
        plugin_names=["detector_people_and_vehicles_fast"],
        # The capsule options you want to set. You can check the available
        # capsule options with the client. Or in the code snippet above that
        # printed capsule names, also print the capsule metadata.
        option_vals={
            "detector_people_and_vehicles_fast": {
                # This capsule is able to detect people, vehicles, and animals.
                # In this example we want to filter out detections that are not
                # animals.
                "filter_mode": "only_animals",
                "threshold": 0.9,
            }
        }
    )

    print()
    print(f"Processed image {image_path.name} and got {detections}")

    # Filter the cat detections using the class name
    cat_detections = [detection for detection in detections
                      if detection.class_name == "cat"]

    if len(cat_detections) > 0:
        print(f"This image contains {len(cat_detections)} cat(s)")
```

Now the script will tell you if there are cats in those images:

```
 Processed image one-person.jpg and got []

Processed image no_people.jpg and got []

Processed image one-person-png.png and got []

Processed image one_cat.jpg and got [Detection(class_name='cat', coords=[[800, 0], [1566, 0], [1566, 850], [800, 850]],
children=[], attributes={}, with_identity=None, extra_data={'detection_confidence': 0.9875224233}, track_id=None)]
This image contains 1 cat(s)

Processed image two_people_and_dtag.png and got []

Processed image two_people.jpg and got []
```