DMY1401TT_Assignment2_Option2

Done by Marcus Tan Wei, A0149980N

My classifier Web app hosted at link

Task Description

The task for this option is to deploy an object detection tool or service using existing pre-trained models.

The pretrained Object Detector and detection function can be found in the following colab notebook. Alternatively, you can use your own object detection model or code. Original model is also attached at Object_detection.ipynb.

Application

In this task, I have created a Web Service using Flask that allows user to input an image and be shown the image with boxes drawn to classify the objects.

Server

1. Takes as input an image and displays or returns an image with bounding boxes drawn.
2. Communication between client and server (Rest API) must be in JSON format.

Client

1. Web client that takes as input an image path and uploads/sends that image to the server.
2. Receives JSON response from server with the image with the plots itself.

Steps to Run Locally

1. Clone this repository

```
git clone https://github.com/MarcusTw/DMY1401TT_Assignment2_Option2.git
```

2. (Optional) If you don't have Python, you can follow the guide to install Python 3.9.8 here. Do not use the latest Python 3.10 version as Tensorflow is not supported.

3. Install the required dependencies

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

or

```
pip3 install -r requirements.txt
```

4. Lastly, you can run the application using Flask

```
export FLASK_APP=app
export FLASK_ENV=development
flask run
```

5. (Optional) If you want to expose host and trust users on your network, you can make the server publicly available by adding --host=0.0.0.0 to the Flask run command.

```
flask run --host=0.0.0.0
```

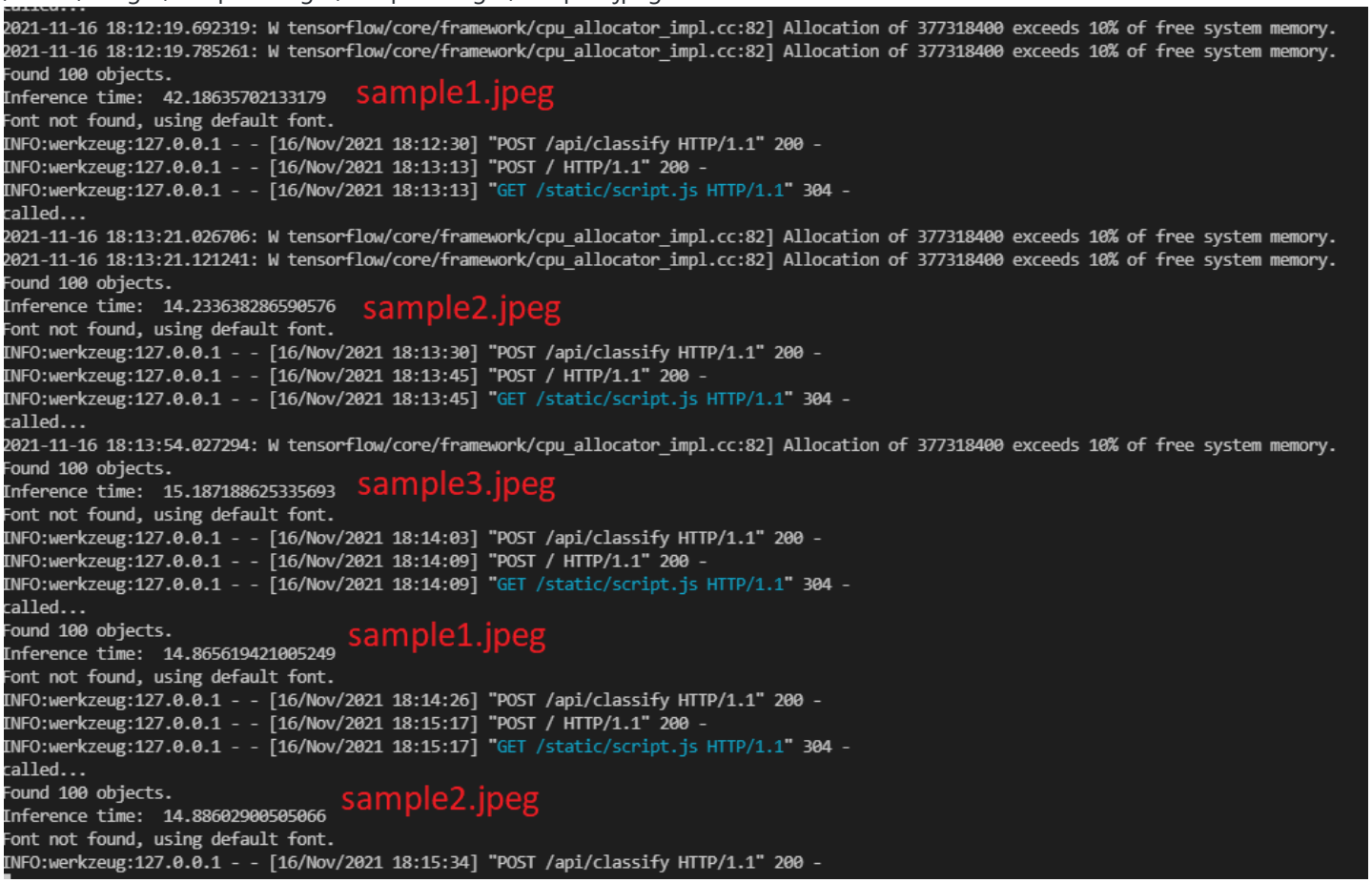
Then you will be able to access the local server at http://localhost:5000.

6. Go to the hosted local server http://127.0.0.1:5000.

Submission

1. Client and server code for Service: Github Repo
2. Client Website host: Link
3. Demonstration: Link
4. Inference time (in order):

- /assets/images/sample-images/sample-images/sample1.jpeg : 42.18635702133179s
- /assets/images/sample-images/sample-images/sample2.jpeg : 14.233638286590576s
- /assets/images/sample-images/sample-images/sample3.jpeg : 15.187188625335693s
- /assets/images/sample-images/sample-images/sample1.jpeg : 14.865619421005249s
- /assets/images/sample-images/sample-images/sample2.jpeg : 14.88602900505066s



- Explanation: The first image would take a longer period of inference time perhaps because of creating and writing to metadata file.

5. Simple usage guide:

- Go to Link
- Click on upload an image
- Select an image that could be classified (recommend using those in `/assets/images/sample-images`)
- Click on Submit and wait for detector the classify! After it is done, you should be able to see the picture with drawn boxes to classify them.

Note: API Calls to classification API endpoint is done through passing of JSON.