Intermediate Workshop

Intermediate Activities

- Write a program using Pandas/Matplotlib that takes in a CSV of student data, then produces a report about various student grade metrics
- Create a web application that allows the user to create a list of accounts with details they can edit
- Try the interactive PyTorch yolov3 demo
- **Challenge:** Build a web application in Flask that allows you to upload an image, then displays the image to the user with every dog circled.
- Anything else cool!

Useful tools for you pitch

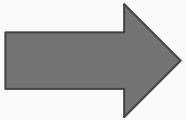
OpenCV + PyTorch

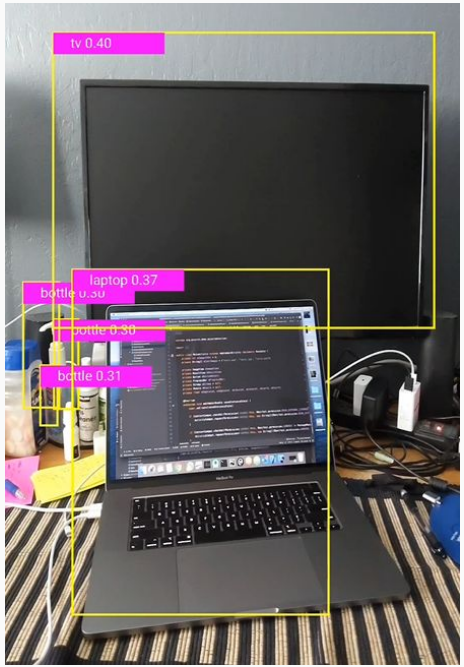
- OpenCV - library for the processing/manipulation of images
  - Can hook it up to your webcam for live video feeds!
- PyTorch is a library for machine learning
  - Contains many models prepared for you!
  - Often as simple as a function call to an OpenCV image

A nice demo here: https://github.com/Moritz-Bergemann/YOLOv5Demo

OpenCV + PyTorch

```python
# Get the libraries we need to run the program
import numpy as np  # numpy
import cv2 as cv    # OpenCV
import torch        # PyTorch

# Get input from device 0 (the webcam)
CAMERA_DEVICE_NUM = 0

# Download YOLOv5 machine learning model from PyTorch Hub
model = torch.hub.load('ultralytics/yolov5', 'yolov5s')

# Get the webcam video capture stream
capture = cv.VideoCapture(CAMERA_DEVICE_NUM)

# For every frame in the image...
while capture.isOpened():
    ret, frame = capture.read()

    # End the program if we didn't get a new frame
    if not ret:
        print("Can't receive frame (stream end?). Exiting ...")
        break

    # Use the model to run a prediction on the frame
    results = model(frame)

    # Get the prediction results and format them
    result_img = np.array(results.render())
    result_img = np.squeeze(result_img)

    # Show the results on screen
    cv.imshow('YOLOv5 computer vision demo', result_img)

    # Allow the program to be quit using the 'q' key
    if cv.waitKey(1) & 0xFF == ord('q'):
        break
```



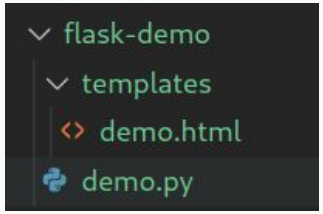


Flask or Django

- Very quick python tools to make web applications
- Flask is simpler (more suitable for demos), Django is more versatile
- Build a quick web application in minutes!*
- Some basic web terminology
  - Your browser/computer makes requests to the server (Flask/Django)
  - Requests are made to a url on the server (for example “myserver.com/index”
  - Requests are transferred using HTTP
  - “GET” requests ask for information from the server
  - “POST” requests give information to the server

*usually hours

The simplest Flask Demo



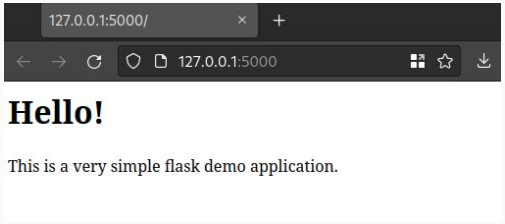
flask-demo > demo.py > ...

```python
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def index():
    return render_template('demo.html')
```

flask-demo > templates > demo.html > html

```html
<html>
    <h1>Hello!</h1>

    <p>This is a very simple flask demo application.</p>
</html>
```

```
$ python -m flask run
```



Data Visualisation - Matplotlib + Pandas

- Pandas is for reading in/managing tabular data in python (called DataFrames)
  - Works on column/heading syntax
  - Way nicer than just using arrays
- Matplotlib allows for hugely versatile graph creation
  - A bit annoying
  - Compatible with Pandas!
  - Seaborn is an alternative

Intermediate Activities

- Write a program using Pandas/Matplotlib that takes in a CSV of student data, then produces a report about various student grade metrics
- Create a web application that allows the user to create a list of accounts with details they can edit
- Try the interactive PyTorch yolov3 demo
- **Challenge:** Build a web application in Flask that allows you to upload an image, then displays the image to the user with every dog circled.
- Anything else cool!