Assignment 3

COL780, October 2021

The aim of this assignment is to perform Pedestrian Detection in images. A single image may contain multiple pedestrians. Your task is to detect each person and return a bounding box for each detection.

Dataset: The dataset for this assignment can be downloaded from here. The dataset comprises of 170 images with 345 labeled pedestrians. The labels comprise segmentation masks as well as bounding box annotations, but for the purpose of this assignment we only need the bounding boxes. This dataset is to be used for the purpose of training and validation. Your code will be evaluated on held-back test set.

Objective: You are required to detect pedestrians using two different methods:

A. ***Histogram of Oriented Gradients:*** For this part, you will be using a HoG person detector.
   a. First, use a pretrained HoG detector (trained on the INRIA dataset as per the original paper [1]). You can use OpenCV's `HOGDescriptor()` with `getDefaultPeopleDetector()` [link].
   b. Next, extract HoG features and train your own classifier on top of it, using the dataset provided.
      i. Prepare training data – positive and negative samples.
      ii. Extract HoG features of the training samples using scikit-image HoG feature extractor.
      iii. Train a linear SVM classifier on the extracted features. You can use the scikit-learn implementation.
      iv. Experiment with different window sizes.

B. ***Faster-RCNN:*** The objective in this part is to familiarize with the data loading and inference pipeline of a deep learning-based object detection model. You are not required to train your own model, instead use a pre-trained model for inference.
   - You should use a pre-trained Faster-RCNN detector [2].
   - You are free to use either Keras or PyTorch deep learning libraries.
   - You can use any pre-trained open-source Faster-RCNN model, with any backbone. Note that the model should be trained on a generic object detection dataset, not for the specific task of pedestrian detection.
   - Write your own data loader to work with the provided dataset.
   - For inference, note that the pre-trained model will be trained to detect multiple object categories, but we are only concerned with the 'person' category. For example, if you use a model trained on the COCO dataset, you only need the predictions for class label 1, which corresponds to the 'person' category.
   - Useful resources: pytorch faster rcnn tutorial, keras faster-rcnn tutorial (This is actually a Mask-RCNN implementation, but you can use only the output ROIs and ignore the masks).

Finally, compare the three models.

a. Compare the Average Precision, Average Recall, Miss Rate.
b. ~~Plot Detection Error Tradeoff curves (Miss Rate vs False Positive Per Image (FPPI)) for FPPI in the range [10^{-2}, 10^{0}].~~
c. Show 5 images from your validation set on which predictions of the three models differ. What is the reason behind the difference?

Evaluation: Detection output must be saved in the COCO Results format. Each bounding box must be stored separately in its own dict of the following format:

```
[{
    "image_id"          : int,
    "category_id"       : int,
    "bbox"              : [x,y,width,height],
    "score"             : float,
}]
```

Results for the whole dataset must be aggregated in a single array and saved as a JSON file. You can find sample results JSON files here.

The following metrics will be used for evaluation:

1. Average Precision (AP): AP evaluated and averaged over 10 IoU thresholds of .50:.05:.95
2. Average Recall (AR): AR is the maximum recall given a fixed number of detections, averaged over IoUs. AR will be evaluated at 1 and 10 detections per image.
3. ~~Miss Rate [3]: Calculated by averaging the miss rate at False Positive Per Image (FPPI) values over the range of [10^{-2}, 10^{0}].~~

The evaluation scripts will be released shortly.

Submission Instructions: You are required to submit:

- Your source code, along with a readme, and a detailed report mentioning the methodology, design choices, results and analysis.
- Upload your model weights file in dropbox/onedrive/google-drive and share the link in the report.
- Zip the code and report in a single file, rename the zip as <Your-entry-number>.zip and submit on moodle.

Evaluation Rubrik:

- Pre-trained HoG detector – 2 points
- Training of HoG detector – 3 points
- Inference using Faster-RCNN – 3 points

- Comparison and interpretation of results – 1 point
- Report + demo – 1 point

References:

[1] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. Ieee, 2005.

[2] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems* 28 (2015): 91-99.

[3] Dollar, Piotr, et al. "Pedestrian detection: An evaluation of the state of the art." *IEEE transactions on pattern analysis and machine intelligence* 34.4 (2011): 743-761.