# Computer Vision (COL780)

## Assignment-3 Report

K N Ajay Shastry(2021CSY7547)

Suraj Patni (2021CSY7550)

## 1. Histogram of Oriented Gradients:

### Part(a): Pre-trained HoG detector.

Here we have implemented the Pedestrian Detection using a pre-trained model that is present in OpenCV.Following are the steps used in our method.

- For every input image, the dimensions are checked. If the image's width is greater than 400 px, then we resize the image to 400*300.
- Input images are sent to the pre-trained model, where they detect pedestrians in a multiscale fashion. In our model, we have used a stride value of (4,4) and a scale of 1.05.
- For each image, the model returns multiple bounding boxes over which we perform non-maximal suppression to get those bounding boxes that fit the pedestrian in the best possible way.

Below are the results obtained by this model.

| Metric | Value |
|---|---|
| Average Precision. | 0.0489 (4.89%) |
| Average Recall @ 1 detection per image. | 0.06 (6%) |
| Average Recall @ 10 detections per image | 0.120 (12%) |

**Part(b): Training an SVM for Pedestrian detection.**

Here we performed pedestrian detection by training an SVM. Following are the steps used in our method.

- We generated positive and negative image samples with the help of the provided dataset. The count of the negative samples is much greater than that of the positive samples. In our case, we had around 283 positive samples and 880 negative samples.

- We resized each image sample to the size of 64x128, Hence maintaining an aspect ratio of 1:2.

- We extracted the HOG features of each of these images, attached labels to them, and sent this as an input to our SVM model of its training.

- Once our model is trained, we perform detection at multiple scales and multiple window sizes on test images, and we get good results when the window size is (64,128) and stride of (9,9).

- For each image, the model returns multiple bounding boxes over which we perform non-maximal suppression to get those bounding boxes that fit the pedestrian in the best possible way.

Below are the results obtained by this model.

| **Metric** | **Value** |
|---|---|
| Average Precision. | 0.116 (11.59%) |
| Average Recall @ 1 detection per image. | 0.1057 (10.57%) |
| Average Recall @ 10 detections per image | 0.203 (20.35%) |

## 2. Faster-RCNN

Here we have implemented the Pedestrian Detection using a pre-trained PyTorch model. Following are the steps used in our method.

- We have loaded a pre-trained PyTorch model ( frcnn-resnet / frcnn-mobilenet) from the torch library.
- Prepared the Dataset Class, which is used in our DataLoader.
- The DataLoader is made with batch_size=5 and num_workers= 5.
- We iterated over DataLoader to find different batches.
- For all images in that batch, we've evaluated our model's output.
- Stored the output in COCO format and also drawn the bounding box in the original image.

Below are the results obtained by this model.

| **Metric** | **Value** |
|---|---|
| Average Precision. | 0.710 (71.04%) |
| Average Recall @ 1 detection per image. | 0.293 (29.35%) |
| Average Recall @ 10 detections per image | 0.757 (75.71%) |

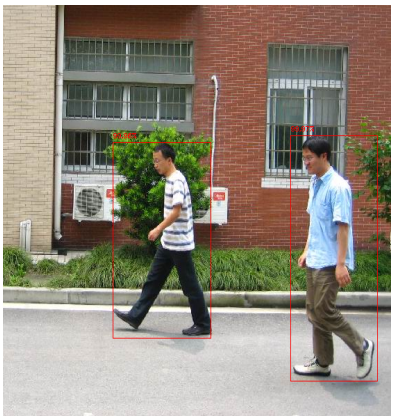
# Comparison:

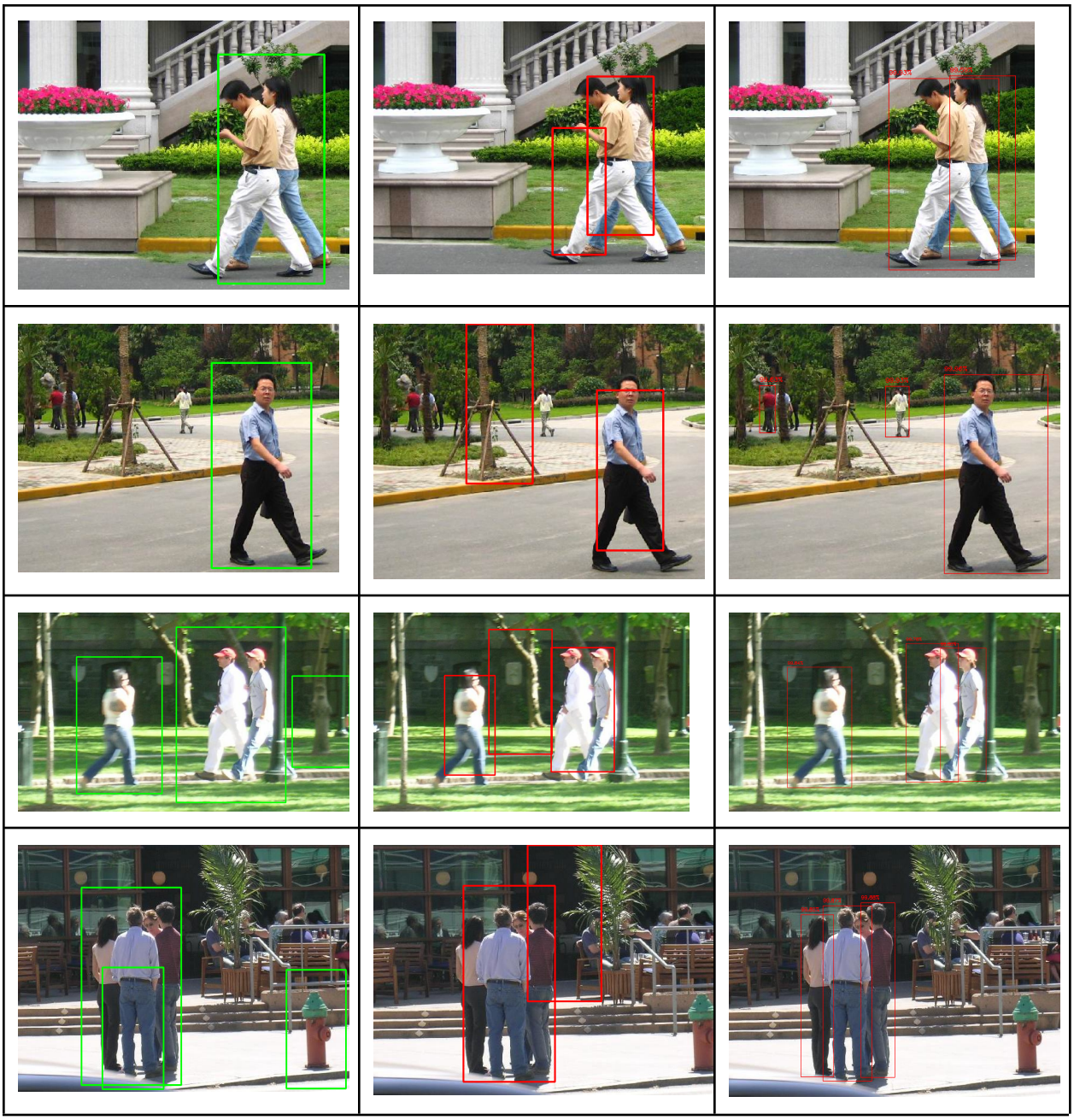
When we compare the models, we can infer that the Average Precision value is highest for the **Faster-RCNN**.

Below are a few image samples from all three models.

The images with

- Green bounding boxes are from the Pre-trained HOG model.
- Red bounding boxes are from the custom HOG-SVM model.
- Red bounding boxes with labels are from the Faster-RCNN model.

| **Pretrained HOG** | **Custom HOG-SVM** | **Faster-RCNN** |
| --- | --- | --- |
|  |  |  |



We can see that the bounding boxes placed by the Faster-RCNN are very accurate compared to the other two.

There were some miss classifications in a few images. The possible reason for them are.

- The object may have had the same structure as that of a human. Eg: structure of the tree.
- Due to non-maximal suppression, there are chances that the area of the bounding boxes are decreased.

Link for the drive folder

END OF REPORT