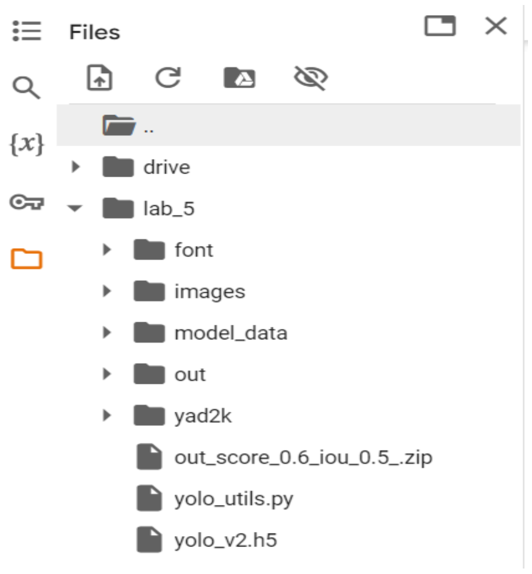
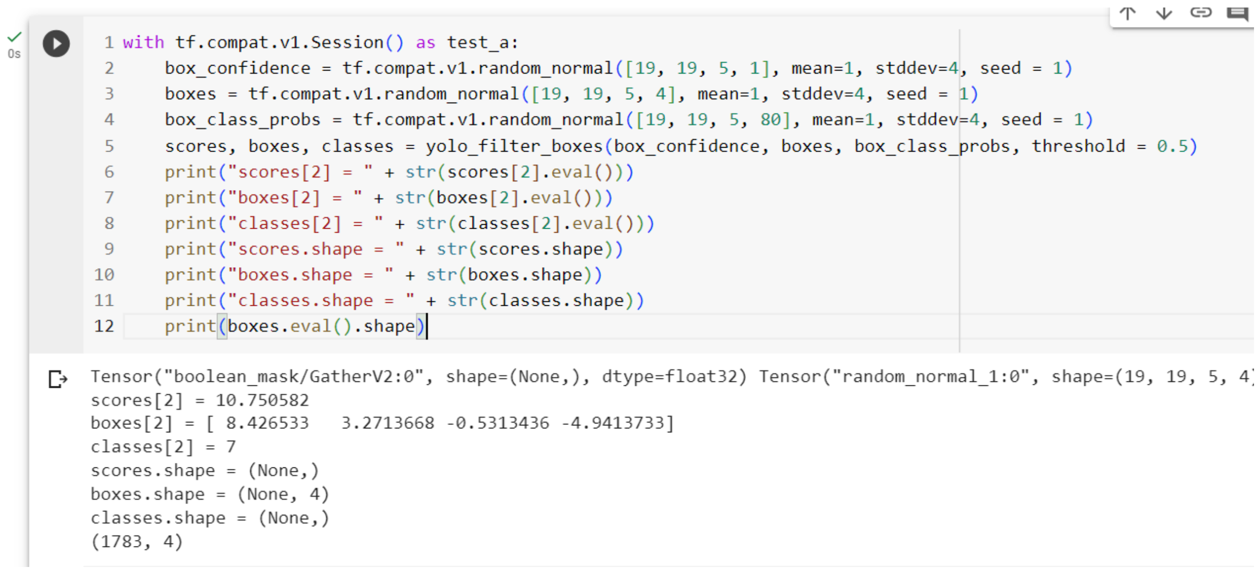
Deep Learning - Lab 4 – YOLO object detection

1. Download the yolo_v2.h5 weights file from the below google drive link. It contains the Yolo version 2 pre-trained model weights. Copy the downloaded file to the 'model_data' directory.
    - Link: https://drive.google.com/drive/folders/1ogvfqQsLADkVLWt64m-wABDfszHaE8EV?usp=sharing
2. Zip all folders and file in the lab5.zip except the lab5_labsheet.docx and Autonomous_driving_application_car_detection.ipynb and upload the zip file to a google drive folder.
3. Upload the jupyter notebook (i.e., Autonomous_driving_application_car_detection.ipynb) to Google colab root directory.



4. In the uploaded notebook, change the path in the second cell to the path of the zip file (one created and uploaded in step 2) and run the first two cells. Root directory should look similar to the below image after this.
5. Go through the Autonomous_driving_application_car_detection.ipynb notebook and try to understand the code.
6. Run the notebook. Make sure to change the dest_folder path in the final cell before running it. At the end you should see the,
    - Result of the car detection demo (i.e., test.jpg).
    - Results of the random images (i.e., giraffe_resized.jpg and DSC_1643_resized.jpg)
    - Results of all image in the autonomous driving dataset (i.e., 0100.jpg to 0120.jpg) saved to out directory.
    - A zip file containing all above images in the colab root as well as in your google drive.

7. In the below given cell, shape of the boxes.eval() is (1783,4). Why are there 1783 boxes? Explain the reason for it. What is the maximum number and minimum number you can get for that? Write these answers in a word file.
    - Change the values like mean and stddev in lines 2 and 4 as well as threshold value in line 5 and observe the different values you get for the boxes.eval().shape.



```
1 with tf.compat.v1.Session() as test_a:
2     box_confidence = tf.compat.v1.random_normal([19, 19, 5, 1], mean=1, stddev=4, seed = 1)
3     boxes = tf.compat.v1.random_normal([19, 19, 5, 4], mean=1, stddev=4, seed = 1)
4     box_class_probs = tf.compat.v1.random_normal([19, 19, 5, 80], mean=1, stddev=4, seed = 1)
5     scores, boxes, classes = yolo_filter_boxes(box_confidence, boxes, box_class_probs, threshold = 0.5)
6     print("scores[2] = " + str(scores[2].eval()))
7     print("boxes[2] = " + str(boxes[2].eval()))
8     print("classes[2] = " + str(classes[2].eval()))
9     print("scores.shape = " + str(scores.shape))
10    print("boxes.shape = " + str(boxes.shape))
11    print("classes.shape = " + str(classes.shape))
12    print(boxes.eval().shape)
```

```
Tensor("boolean_mask/GatherV2:0", shape=(None,), dtype=float32) Tensor("random_normal_1:0", shape=(19, 19, 5, 4
scores[2] = 10.750582
boxes[2] = [ 8.426533   3.2713668 -0.5313436 -4.9413733]
classes[2] = 7
scores.shape = (None,)
boxes.shape = (None, 4)
classes.shape = (None,)
(1783, 4)
```

8. yolo_anchors.txt contains 10 values. They can be considered as height and width of 5 anchor boxes. What is the advantage of using such anchor boxes? What was the method used to determine the sizes of these anchor boxes? Give the answers to these questions in the word file.
9. Upload a new traffic image to images and edit the code as needed to detect vehicles in that image.
10. Download the output images zip file from the google drive and observe the bounding boxes in the autonomous driving dataset (i.e., 21 images from 0100.jpg to 0120.jpg). Select 2 images from these 21 images and,
    - Write what you observe regarding correctly detected objects, incorrectly detected objects, undetected objects and incorrect bounding boxes in the word file.
    - Include these output 2 images as well as the original 2 images in the word file.
11. Adjusting parameters like max_boxes, score_threshold, and iou_threshold of the yolo_eval function can potentially address the limitations you noticed in step 10.
    - Change the max_boxes [integer value] to a different value but use the original values for other 2 variables. Rerun the required cells to get the output images for the autonomous driving dataset. Observe if this result in improvement compared to step 10 for the same two images. If there are any improvements, write them in the word file. Include the new 2 output images in the word file.
    - Change the score_threshold [value between 0-1] to a different value but use the original values for other 2 variables. Rerun the required cells to get the output images for the autonomous driving dataset. Observe if this result in improvement compared to step 10

for the same two images. If there are any improvements, write them in the word file. Include the new 2 output images in the word file.

- Change the iou_threshold [value between 0-1] to a different value but use the original values for other 2 variables. Rerun the required cells to get the output images for the autonomous driving dataset. Observe if this result in improvement compared to step 10 for the same two images. If there are any improvements, write them in the word file. Include the new 2 output images in the word file.

Submission.

Download the final modified Autonomous_driving_application_car_detection.ipynb notebook. Add this notebook, the new image used in step 9 and the word file to a new zip file. Upload this zip file to the courseweb submission link. The file name should be your registration number.