**The Codes\_For\_Training folder contains the codes that are being used to prepare datas for training.**

**xml\_to\_csv\_own\_code.py:**

We label images for training using a program named labelimage and it generates xml files that contain the object that we labeled. However, to generate TFrecord files we need to gather that information in csv files and put it in arrays. Hence, I wrote a code that reads xml files and takes that information and writes it to a csv document as an array.

**imresize.py:**

When we start to train a dataset, tensorflow converts the images according to pipeline configuration. Most of the pretrained models are different in terms of input image size. Hence, I write a code that converts the size of images that I found to which size one wants.

**find\_same\_images.py:**

I wrote this code to find the same images in the dataset, tensorflow may overfit or underfit if there are the same images according to some articles. I was manually doing it since my dataset was very low.

**txt\_to\_csv.py:**

I wrote this code to convert the txt files that contain annotations to the csv file I needed.

**head\_coordinates.py:**

This code helps to take the coordinates of mouse click on image. I used it for the head coordinates of people in crowd scenes.

**The Detection\_model\_and\_code folder contains the trained model and the python codes for detection.**

**Model\_person:**

This folder contains the trained model with a random dataset.

**model\_person\_top\_19k:**

This folder contains the trained model with only the top viewed image.

**Camera\_person\_detection.py:**

This script is used to detect people in video or internal cameras.

**person\_detection\_image.py:**

This script is used to detect people in an image.

**counting\_people\_in\_frame.py:**

It detects the number of people in one frame and returns the maximum number of people that counted

**count\_people\_in\_image.py:**

It counts the number of people in an image based on the number of detection which has a score bigger than the threshold.

**counting\_with\_center\_tracking.py:**

It stores the center of frame for two consecutive frames and checks whether it is less than or bigger than a threshold by looking at the euclidean distance between two centers. If it is less, the counter is not increased, otherwise the counter is increased by 1.

**counting\_people\_with\_open\_cv\_tracker.py:**

coordinate of person that is being tracked is stored in python dictionary with key as number of person function check\_new\_person is checking whether the new detected person is already being identified or not by checking euclidean distance between tracked coordinates and detected coordinates if its same person it adds to used\_ids the id of person if not it increases the counter by 1 and add to used\_ids. In function extract\_unseen\_person it creates a new dict and stores the only people seen. It takes id from used\_ids and find the center from the dict then store it in new dict and copy that new dict to one being used for classification

**in\_out\_withxcoord\_counting\_people\_with\_open\_cv\_tracker.py:**

It is used when the entrance door and exit door are different. Algorithm is the same, the only difference is it checks the x coordinate of detection to see whether the object goes out or in.

**in\_out\_counting\_people\_with\_open\_cv\_tracker.py:**

It is used when people get in and out through the same door. Algorithm is the same, the only difference is it checks the y coordinate of detection to see whether the object goes out or in by looking whether first detection is below a threshold value or above it.

**counting\_with\_tracker.py:**

The code is the same as counting\_people\_with\_open\_cv\_tracker.py The difference is I added a drawing bounding boxes function to draw boxes with my own code.

**Common Problems**

Error: UnicodeDecodeError: 'utf-8' codec can't decode byte 0xff in position 0: invalid start byte

Solution: Check whether you create a new directory for training. if not make a directory and copy only the pipeline to that directory and start the training from that directory.

Error: Resource exhausted: failed to allocate memory

Solution: Decrease the Batch\_size from pipeline.config

Error: Windows fatal access error:

Solution: Decrease the Batch\_size from pipeline.config

Error:

tensorflow.python.framework.errors\_impl.InvalidArgumentError: 2 root error(s) found.

(0) Invalid argument: assertion failed: [maximum box coordinate value is larger than 1.100000: ] [1.16451609]

[[{{node while/body/\_1/ToAbsoluteCoordinates/Assert/AssertGuard/else/\_2047/ToAbsoluteCoordinates/Assert/AssertGuard/Assert}}]]

[[while/body/\_1/BalancedPositiveNegativeSampler/Cast\_8/\_444]]

(1) Invalid argument: assertion failed: [maximum box coordinate value is larger than 1.100000: ] [1.16451609]

[[{{node while/body/\_1/ToAbsoluteCoordinates/Assert/AssertGuard/else/\_2047/ToAbsoluteCoordinates/Assert/AssertGuard/Assert}}]]

0 successful operations.

0 derived errors ignored. [Op:\_\_inference\_\_dist\_train\_step\_46931]

Function call stack:

\_dist\_train\_step -> \_dist\_train\_step

Solution:

The problem is caused by incorrect dimensions in record files. Coordinates of annotations should be checked and see whether there exist any coordinate that exceeds width or height.

Error: error: (-5:Bad argument) in function 'resize'

> - Can't parse 'dsize'. Sequence item with index 0 has a wrong type

> - Can't parse 'dsize'. Sequence item with index 0 has a wrong type

Solution:

change cv2.resize(image, pt1 , pt2 )

to cv2.resize(image, int(pt1),int(pt2)