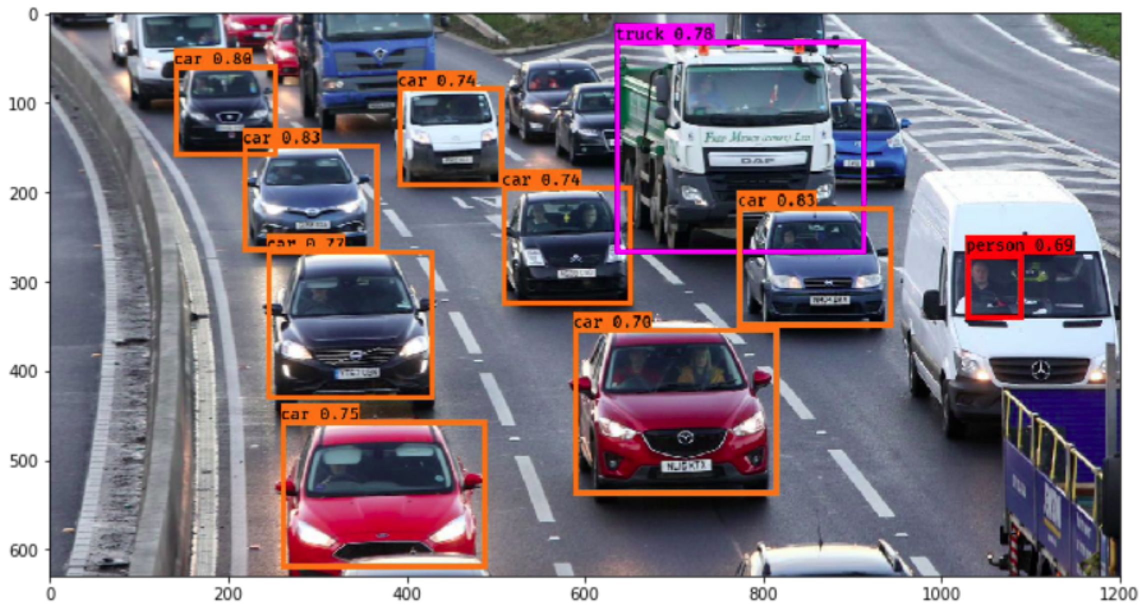
TOCI-2 (Deep learning)

Name: Wajiha Hanif Arain
Seat no: B18158064
Submitted to: Ma’am Farzeen Ashfaque

“Vehicle Counting, Classification & Detection using RCNN & Python”

Intelligent vehicle detection and counting are becoming increasingly important. However, due to the different sizes of vehicles, their detection remains a challenge that directly affects the accuracy of vehicle counts. The experimental results verify that using the p**roposed segmentation method can provide higher detection accuracy, especially for the detection of small vehicle objects.**
The object size of the vehicle changes greatly at this viewing angle, and the detection accuracy of a small object far away from the road is low. In the face of complex camera scenes, it is essential to effectively solve the above problems and further apply them. We apply the vehicle detection results to multi-object tracking and vehicle counting.

At present, vision-based vehicle object detection is divided into traditional machine vision methods and complex deep learning methods. Traditional machine vision methods use the motion of a vehicle to separate it from a fixed background image.



We’ll use the RCNN model with OpenCV-python. Open-CV is a real-time computer vision library of Python. We can use RCNN directly with OpenCV.

What is RCNN?

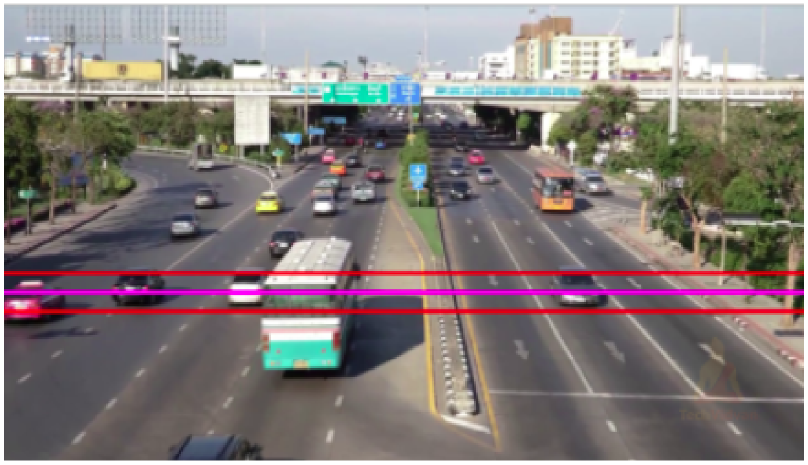
To bypass the problem of selecting a huge number of regions, a method where we use selective search to **extract just 2000 regions** from the image and he called them region proposals. Therefore, now, instead of trying to classify a huge number of regions, you can just work with 2000 regions. These 2000 region proposals are generated using the selective search algorithm which is written below.

In this project, we'll detect and classify cars, HMV (Heavy Motor Vehicle), and LMV (Light Motor Vehicle) on the road, and count the number of vehicles traveling on a road. And the data will be stored to analyze different vehicles that travel on the road.

A code sample where we calculate the distance of the vehicle and bounding samples through that counting confidence score

```
# Get center point of new object

for rect in objects_rect:

    x, y, w, h, index = rect

    cx = (x + x + w) // 2

    cy = (y + y + h) // 2



    # Find out if that object was detected already

    same_object_detected = False

    for id, pt in self.center_points.items():

        dist = math.hypot(cx - pt[0], cy - pt[1])



        if dist < 25:

            self.center_points[id] = (cx, cy)

            # print(self.center_points)
            objects_bbs_ids.append([x, y, w, h, id, index])

            same_object_detected = True

            break
```

- RCNN is trained on the XML dataset, so we read the file that contains all the class names and store the names in a list.
- The XML dataset contains 80 different classes.
- We need to detect only cars, motorbikes, buses, and trucks for this project.
- Read the video file through the video capture object and set the object as a cap.
- cap.read() reads each frame from the capture object.
- Using cv2.reshape() we reduced our frame by 50 percent.
- Then using the cv2.line() function we draw the crossing lines in the frame.
- And finally, we used cv2.imshow() function to show the output image.



- Track and count all vehicles on road

```
# Update the tracker for each object

  boxes_ids = tracker.update(detection)

  for box_id in boxes_ids:

    count_vehicle(box_id)
```

The problem we're facing in the end:

The problem we face was that our deep learning program required heavy processing and google collab did not support this short two days before. So this is all it could do.