PRODUCT DEVELOPMENT LAB

TITLE-

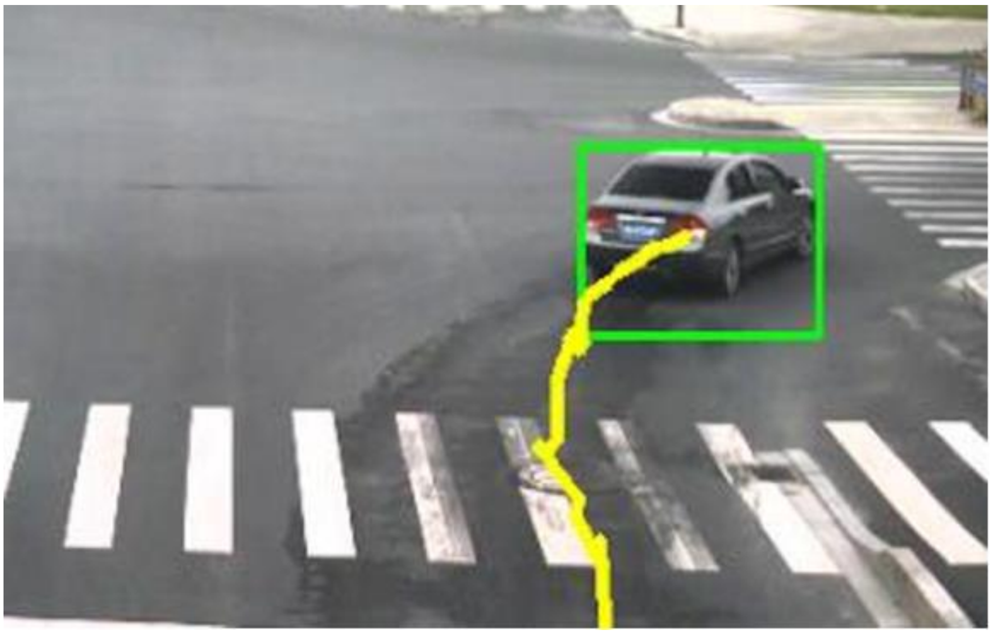
VEHICLE IDENTIFICATION AND ALERTING SYSTEM

TEAM MEMBERS-

1. ANWESH KUMAR SAMAL – 119EC0268
2. MALAYAJ RATH – 119EC0597
3. SRITAM PANDA – 119EC0195

OBJECTIVES-

- To detect traffic violation like taking wrong routes, crossing the zebra crossing at traffic
- To detect accidents in highways where there is not much access
- To detect any particular vehicle entering an area which can be implemented in automatic gate opening



WORK FLOW-

1. **VEHICLE DETECTION (OBJECT DETECTION)**

2. **ALERTING SYSTEM**

3. **DEVELOPING OF A WEBPAGE/APP TO SEND THE ALERT NOTIFICATION**

4. **IMPLEMENTATION ON RASPBERRY PI WITH A CAMERA ON IT TO GIVE THE INPUT FEED.**

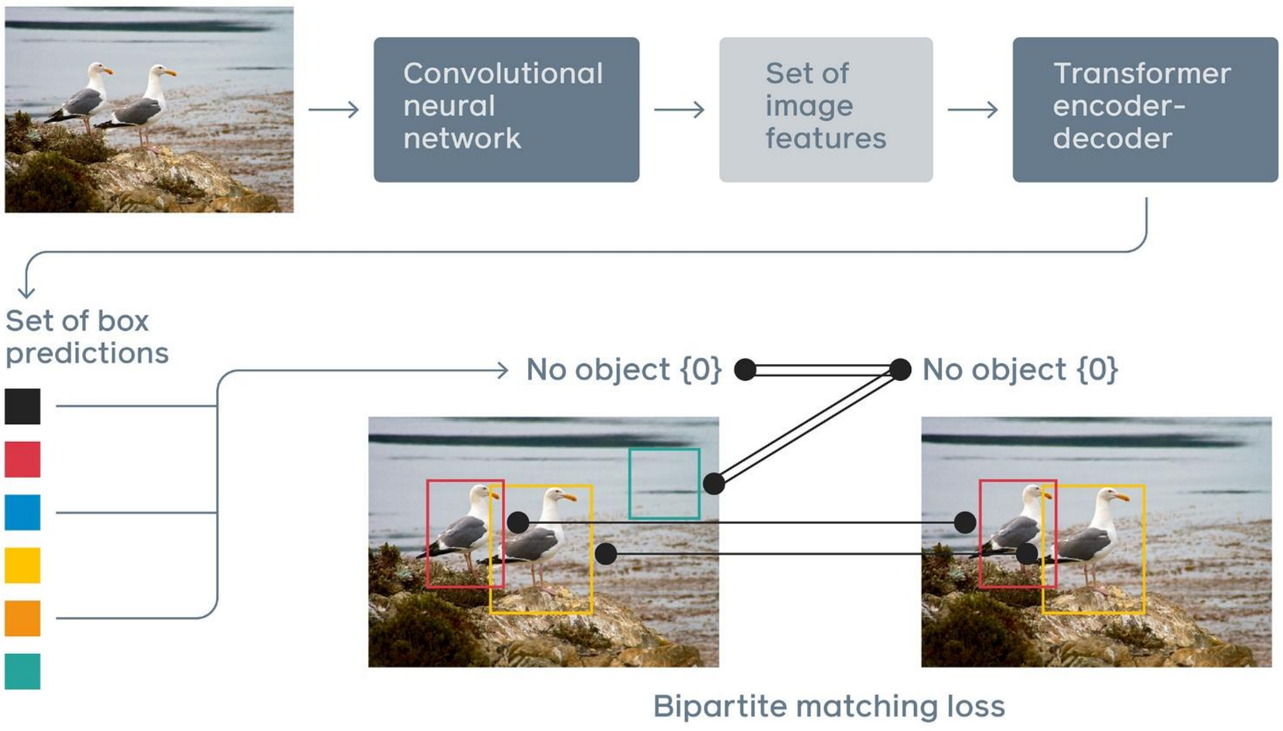
ALGORITHM: -

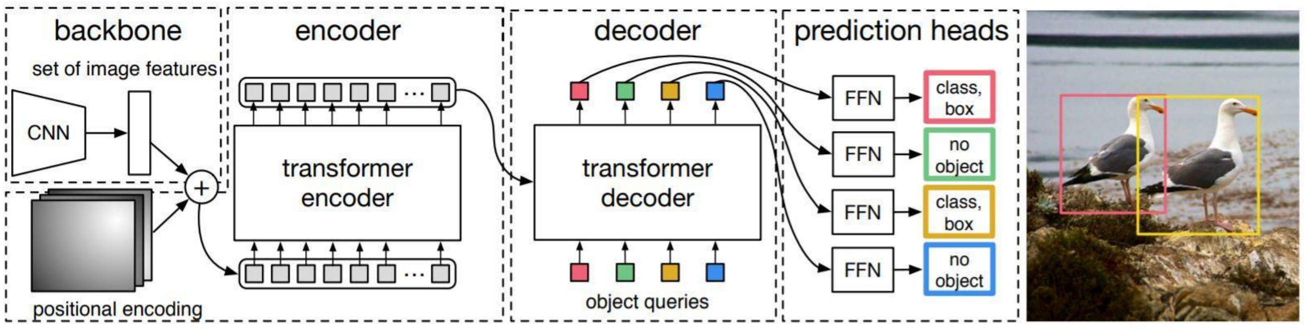
- **Transformer based object detection (DETR)**

This algorithm was implemented by Facebook for human detection but our main motive is to use this algorithm on real time video feed so that we can use it on vehicle detection.

Transformers have been widely applied on problems with sequential data, in particular in natural language processing (NLP) tasks such as language modelling and machine translation. But, perhaps surprisingly, computer vision has not yet been swept up by the Transformer revolution.

To help bridge this gap, we are releasing Detection Transformers (DETR), an important new approach to object detection and panoptic segmentation.





1. DETR (Detection Transfer) is a simpler model for object detection avoiding classical approaches like non maximum suppression.

2. In this technique we first use a CNN to extract the image features, then we have transformer encoder - decoder to recognize and return detected objects in the image.

3. DETR uses a loss function (Bipartite loss matching - Hungerian Biparatite matching algorithm) between predicted and ground truth objects.

4. Transformer encoder takes the vector of features and process them using multi head self-attention module and a feed - forward network.

5. Transformer decoder takes object queries and decode them into independent box coordinates, which is levelled by a feed forward network. Using self- and encoder-decoder attention over these embeddings, the model globally reasons about all objects together using pair-wise relations between them, while being able to use the whole image as context.

6. The final prediction is computed by a 3-layer perceptron with ReLU activation function and hidden dimension d, and a linear projection layer. The FFN predicts the normalized center coordinates, height and width of the box w.r.t. the input image, and the linear layer predicts the class label using a softmax function.

STEPS to perform the algorithm-

1. Instantiate the DETR model and push it to GPU.
2. Define the classes that DETR can detect.
3. Load your image and normalize it. Be careful here, it needs to be an RGB image in a shape (800, 600).
4. Push your image to GPU with cuda support.
5. Forward your image through the model and get the predictions:

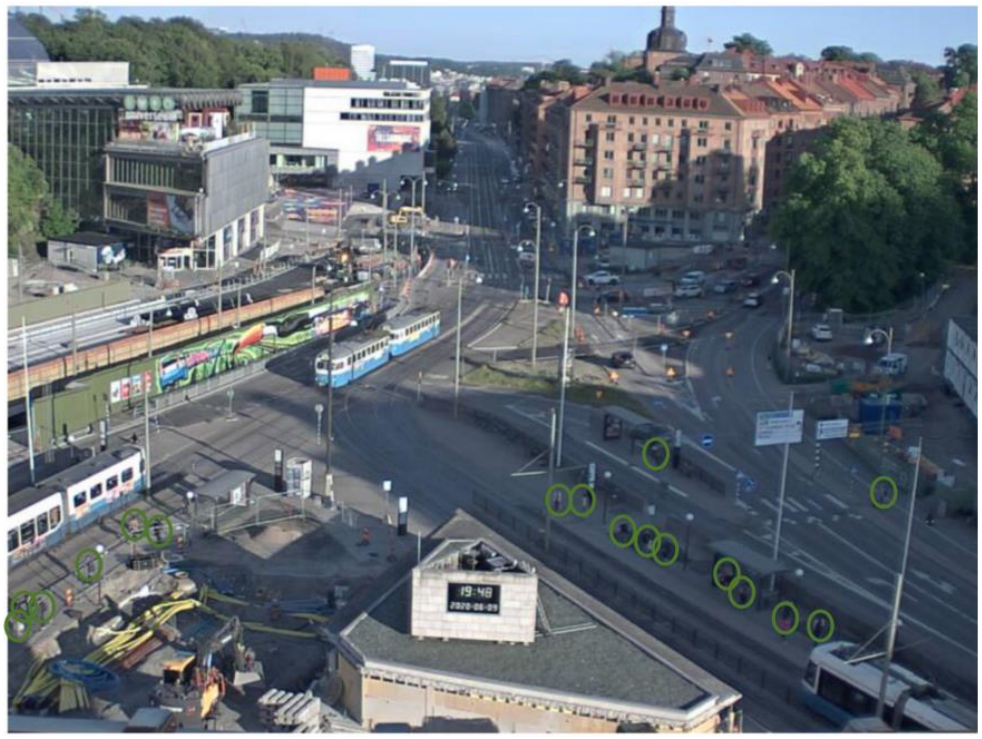
   I) predicted boxes with the position in the image of the detected objects

   II) ii) predicted probabilities for each of the 100 images detected. For the latter, we have for each image detected, a distribution of values corresponding to each class.

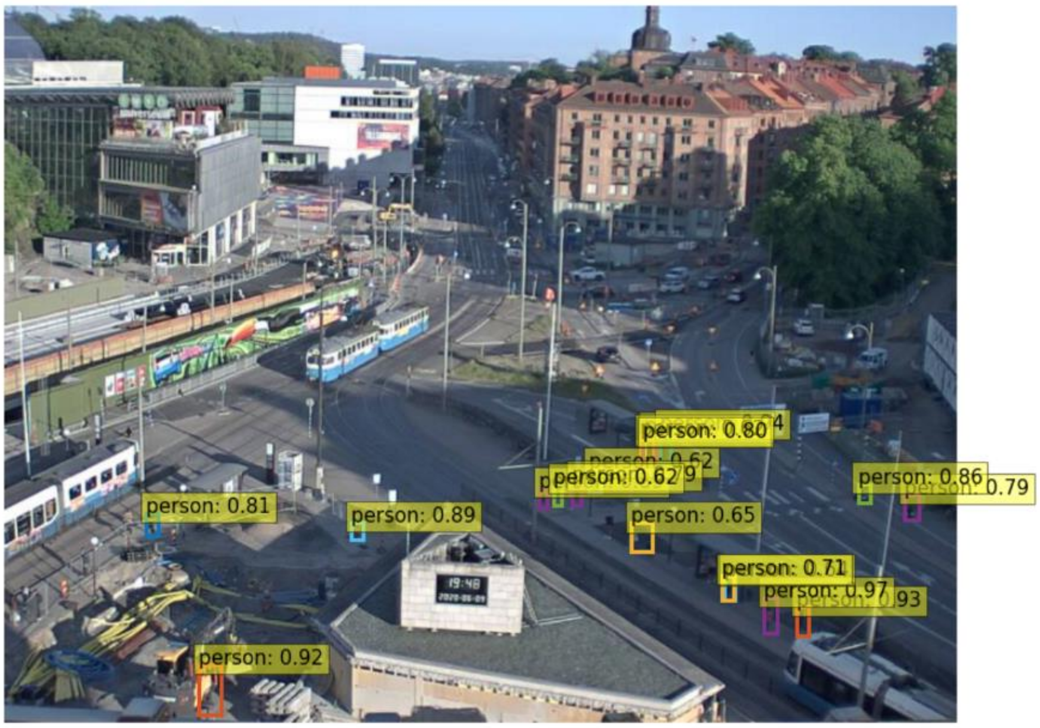
6. We then apply argmax over each row of the previous output to get the index of the class maximizing the probability of the object being drawn from one of the classes.
7. Now that we have our predicted class for each object detected in our image, we can draw on the image a box in order to show the object with its associated class. The coordinates of such box is found from one of the outputs of DETR in step 5.

- Examples to show the advantage of using the DETR Algorithm: -

Manual detection-

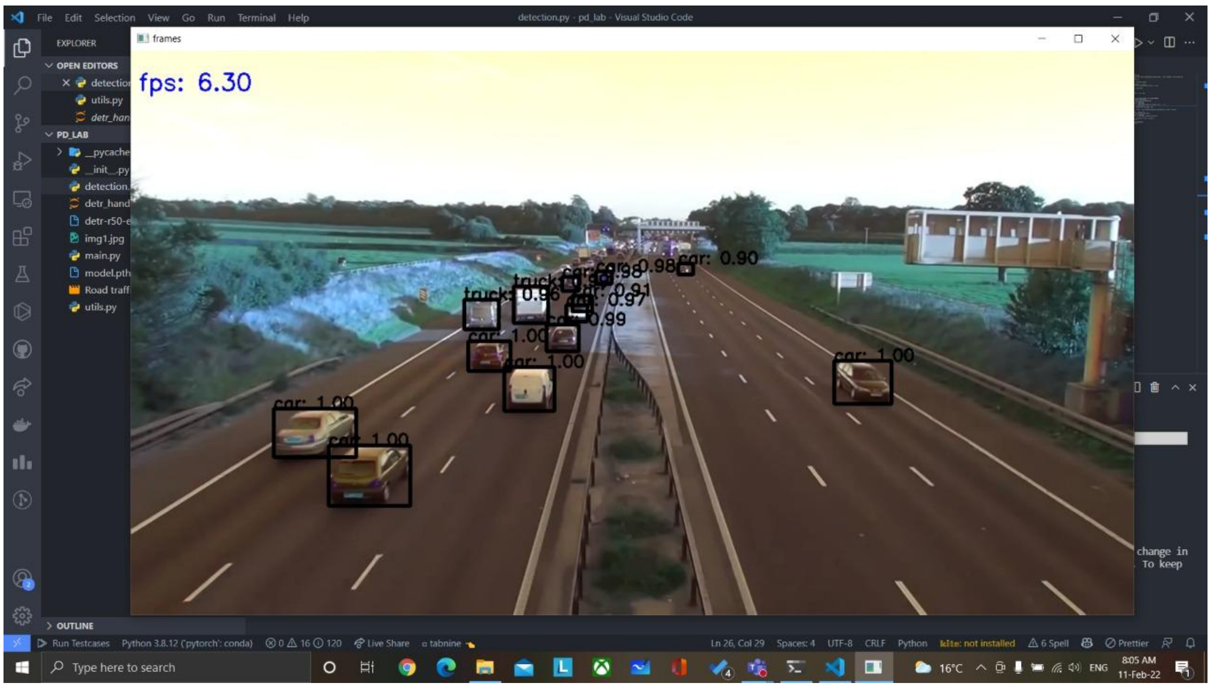


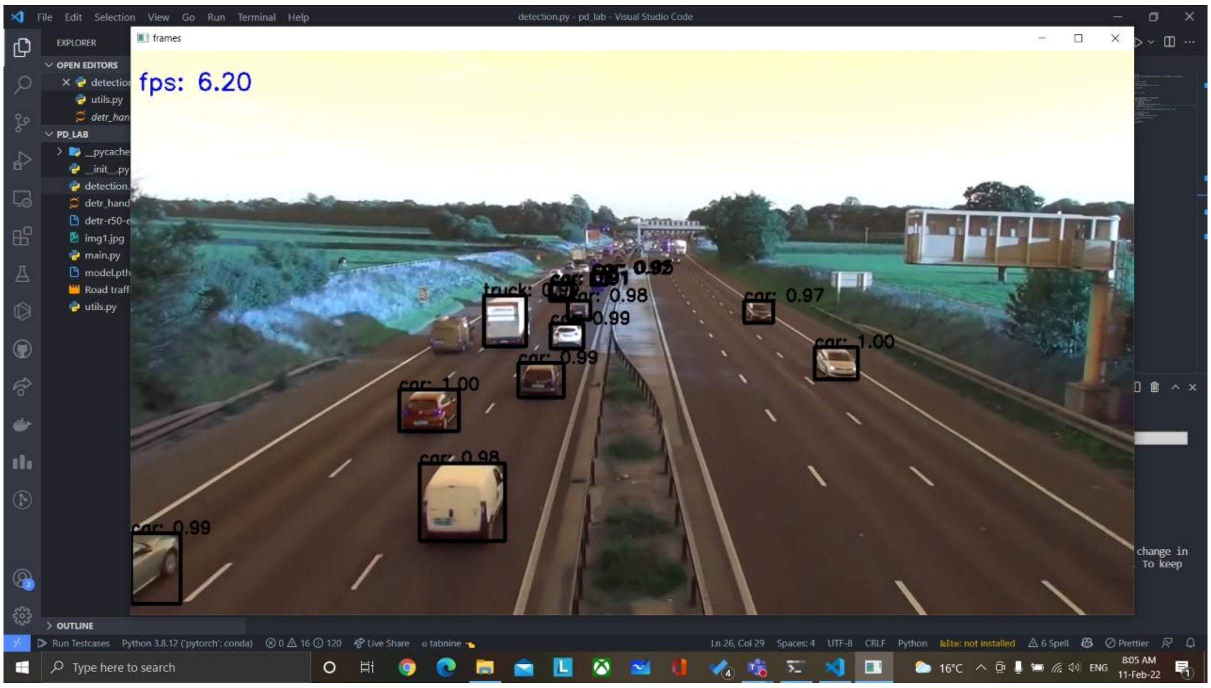
USING DETR-



OBSERVATIONS-

(Sample testing code):


detection.py - pd_lab - Visual Studio Code
frames
fps: 6.30
car: 1.00
car: 0.90
truck: 0.96
car: 0.99
car: 0.97
Ln 26, Col 29
Type here to search
8:05 AM
11-Feb-22


detection.py - pd_lab - Visual Studio Code
frames
fps: 6.20
car: 1.00
car: 0.97
car: 0.98
car: 0.99
Ln 26, Col 29
Type here to search
8:05 AM
11-Feb-22