VEHICLE DETECTION – Project 5

The Goal of the project is to identify the vehicle in a video.

I have not used the classroom materials to solve the problem using hog, spatial and color. Instead I used YOLO model to identify the cars in the image.

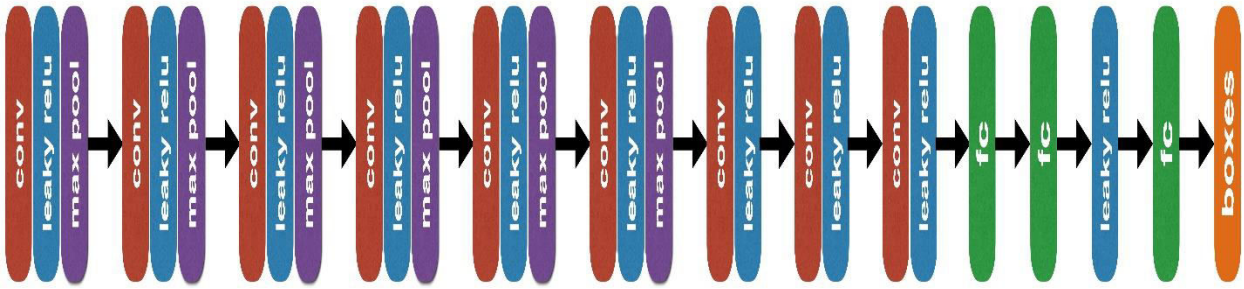
1. A brief intro to YOLO (You Look Only Once)

The YOLO is a pre-trained model with architecture as below.

```
Layer (type)                     Output Shape          Param #     Connect
ed to
====================================================================================
==========================
convolution2d_1 (Convolution2D)  (None, 16, 448, 448)  448         convolu
tion2d_input_1[0][0]
____________________________________________________________________________
_____________________
leakyrelu_1 (LeakyReLU)          (None, 16, 448, 448)  0           convolu
tion2d_1[0][0]
____________________________________________________________________________
_____________________
maxpooling2d_1 (MaxPooling2D)    (None, 16, 224, 224)  0           leakyre
lu_1[0][0]
____________________________________________________________________________
_____________________
convolution2d_2 (Convolution2D)  (None, 32, 224, 224)  4640        maxpool
ing2d_1[0][0]
____________________________________________________________________________
_____________________
leakyrelu_2 (LeakyReLU)          (None, 32, 224, 224)  0           convolu
tion2d_2[0][0]
____________________________________________________________________________
_____________________
maxpooling2d_2 (MaxPooling2D)    (None, 32, 112, 112)  0           leakyre
lu_2[0][0]
```

```
_________________________________________________________________
convolution2d_3 (Convolution2D)  (None, 64, 112, 112)  18496       maxpool
ing2d_2[0][0]
_________________________________________________________________
leakyrelu_3 (LeakyReLU)          (None, 64, 112, 112)  0           convolu
tion2d_3[0][0]
_________________________________________________________________
maxpooling2d_3 (MaxPooling2D)    (None, 64, 56, 56)    0           leakyre
lu_3[0][0]
_________________________________________________________________
convolution2d_4 (Convolution2D)  (None, 128, 56, 56)   73856       maxpool
ing2d_3[0][0]
_________________________________________________________________
leakyrelu_4 (LeakyReLU)          (None, 128, 56, 56)   0           convolu
tion2d_4[0][0]
_________________________________________________________________
maxpooling2d_4 (MaxPooling2D)    (None, 128, 28, 28)   0           leakyre
lu_4[0][0]
_________________________________________________________________
convolution2d_5 (Convolution2D)  (None, 256, 28, 28)   295168      maxpool
ing2d_4[0][0]
_________________________________________________________________
leakyrelu_5 (LeakyReLU)          (None, 256, 28, 28)   0           convolu
tion2d_5[0][0]
_________________________________________________________________
maxpooling2d_5 (MaxPooling2D)    (None, 256, 14, 14)   0           leakyre
lu_5[0][0]
_________________________________________________________________
convolution2d_6 (Convolution2D)  (None, 512, 14, 14)   1180160     maxpool
ing2d_5[0][0]
_________________________________________________________________
leakyrelu_6 (LeakyReLU)          (None, 512, 14, 14)   0           convolu
tion2d_6[0][0]
_________________________________________________________________
maxpooling2d_6 (MaxPooling2D)    (None, 512, 7, 7)     0           leakyre
lu_6[0][0]
_________________________________________________________________
convolution2d_7 (Convolution2D)  (None, 1024, 7, 7)    4719616     maxpool
ing2d_6[0][0]
_________________________________________________________________
```

```
leakyrelu_7 (LeakyReLU)          (None, 1024, 7, 7)    0           convolu
tion2d_7[0][0]
_____________________________
convolution2d_8 (Convolution2D)  (None, 1024, 7, 7)    9438208     leakyre
lu_7[0][0]
_____________________________
leakyrelu_8 (LeakyReLU)          (None, 1024, 7, 7)    0           convolu
tion2d_8[0][0]
_____________________________
convolution2d_9 (Convolution2D)  (None, 1024, 7, 7)    9438208     leakyre
lu_8[0][0]
_____________________________
leakyrelu_9 (LeakyReLU)          (None, 1024, 7, 7)    0           convolu
tion2d_9[0][0]
_____________________________
flatten_1 (Flatten)              (None, 50176)         0           leakyre
lu_9[0][0]
_____________________________
dense_1 (Dense)                  (None, 256)           12845312    flatten
_1[0][0]
_____________________________
dense_2 (Dense)                  (None, 4096)          1052672     dense_1
[0][0]
_____________________________
leakyrelu_10 (LeakyReLU)         (None, 4096)          0           dense_2
[0][0]
_____________________________
dense_3 (Dense)                  (None, 1470)          6022590     leakyre
lu_10[0][0]
==========================================================================
==========================
Total params: 45,089,374
Trainable params: 45,089,374
Non-trainable params: 0
```



What is does his it has trained the model such that it sparses 7*7 grid of an image of input size 448,448,3 and get the label which contains 1470 values. The count is obtained by the formula **S x S x (B x 5 + C)**, Where s is 7 and B is 2 and C is 20

2. **Explaining the Labels:**

The 1470 vector output is divided into three parts, giving the probability, confidence and box coordinates.

Each of these three parts is also further divided into 49 small regions, corresponding to the predictions at each cell.

In post processing steps, we take this 1470 vector output from the network to generate the boxes that with a probability higher than a certain threshold.

All this are explained in the comments on the code.

3. **Loading the weights**

We Load the pre-trained weights of the model downloaded from internet. The name of the file is yolo-tiny. weights

4. **Predicting the model with new image**

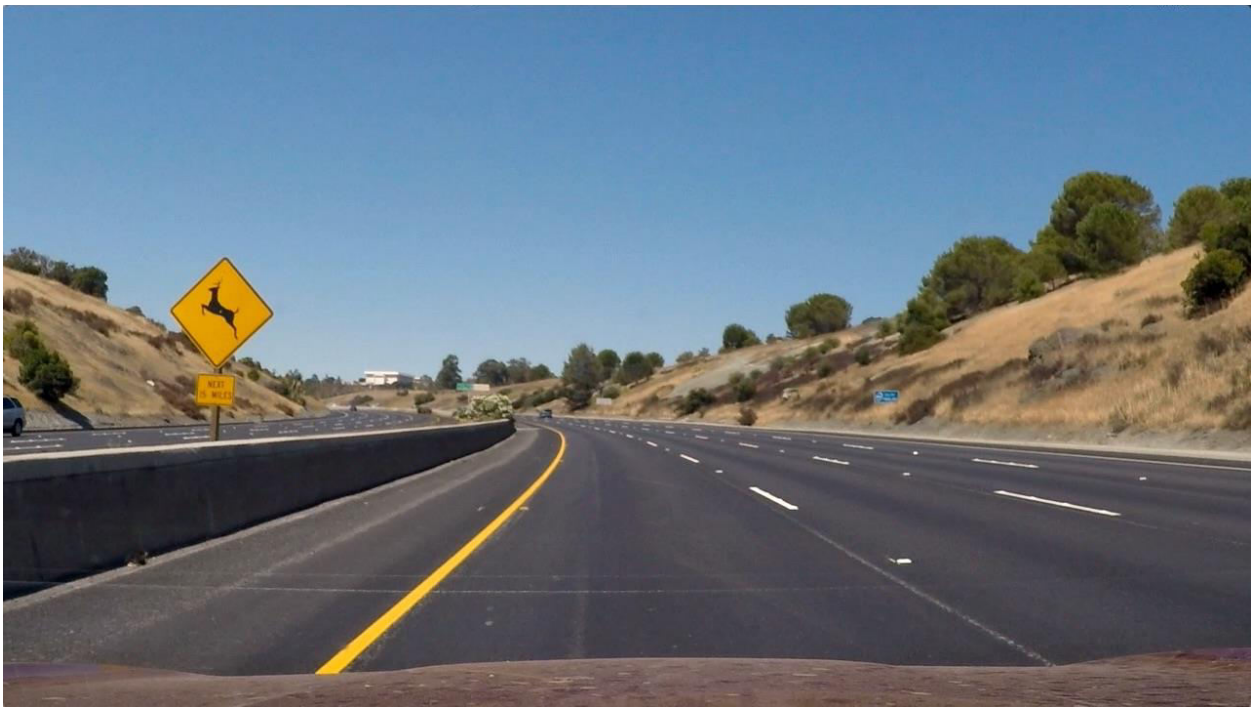
The model is fed a new test image to predict the labels,

Which has bounding box, probability and confidence score.

5. **Drawing the bounding boxes with a score higher than threshold**

The we use those labels to get the bounding boxes scores and draw the boxes in the image.

The output of this for test images are:



UPDATES FOR RE-SUBMISSION:

1. **Some discussion is given around how you improved the reliability of the classifier i.e., fewer false positives and more reliable car detections (this could be things like choice of feature vector, thresholding the decision function, hard negative mining etc.)**
   The writeup did not explain how the reliability of the vehicles detected was improved.
   How does the implementation improve the reliability of the already trained model?
   Did the model use any form of thresholding to make sure the vehicles detected are actually vehicles?

The model that I have used is a tiny yolo model. So it is tradeoff between reliability and speed. So the reliability of this model is less compared to other yolo models with are trained on a larger dataset.

I have used thresholds to make sure vehicles are correctly detected.

I took the p_classnum as 6(car in yolo tiny) and made sure there the product of the prob and confident score is greater than 0.2

2. The writeup did not explain how the reliable predictions from previous frames of the video were used to improve the reliability of the vehicles detected in the current frame. Was this implemented in the model?

This is not implemented in the model as the yolo detects objects in every frame accurately.

3. Discussion includes some consideration of problems/issues faced, what could be improved about their algorithm/pipeline, and what hypothetical cases would cause their pipeline to fail.

A higher yolo model could predict the vehicles better.

The model may fail in cases where the lower threshold may pick non-car data.

To improve this, we can use higher yolo models which can give more accuracy.