Project: Vehicle-Detection
Name: Dongdong Wu

Overview:

Vehicle detection in video-stream or in image is popular issue of Computer vision. The main idea is, by using pre-trained model, detect vehicle object. As a classification problem, the image are divided into small patches, each of which will be run through a classifier to determine whether there are objects in the patch. Then the bounding boxes will be assigned to locate around patches that are classified with high probability of present of an object. In the regression approach, the whole image will be run through a convolutional neural network to directly generate one or more bounding boxes for objects in the images.

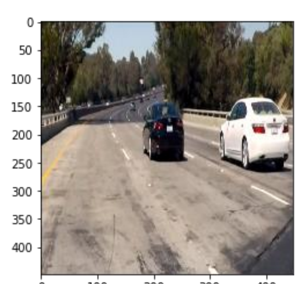
During this project, we will use tiny-YOLO which is based off of the Darknet reference network and is much faster but less accurate than the normal YOLO model. For real time vehicle detection, speed(FPS) are critical import comparing to detection on image. By comparing model's feature:

Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
Old YOLO	VOC 2007+2012	2007	63.4	40.19 Bn	45		link
SSD300	VOC 2007+2012	2007	74.3	-	46		link
SSD500	VOC 2007+2012	2007	76.8	-	19		link
YOLOv2	VOC 2007+2012	2007	76.8	34.90 Bn	67	cfg	weights
YOLOv2 544x544	VOC 2007+2012	2007	78.6	59.68 Bn	40	cfg	weights
Tiny YOLO	VOC 2007+2012	2007	57.1	6.97 Bn	207	cfg	weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	-	-	7.07 Bn	200	cfg	weights

That is why I choose this model to do detection.

Image Pre-process:

Since Yolo model's input format limit, I crop and resize the image to 448*448 format. Then do normalization to {-1, 1} range. The example image :



Model Architecture:

During this project, i use keras as API to implement yolo-model. According to pre-trained yolo model, this model consist of 9 convolution layers and 3 full connected layers. Each convolution layer consists of convolution, relu and max-pooling operations. The first 9 convolution layers can be understood as the feature extractor, whereas the last three full connected layers can be understood as the "regression head" that predicts the bounding boxes. The model summary reports:

The detail information can be found:

https://pjreddie.com/darknet/yolov1/

```
Layer (type)                     Output Shape          Param #     Connected to
==========================================================================================
===========
convolution2d_1 (Convolution2D)  (None, 16, 448, 448)  448
convolution2d_input_1[0][0]
__________________________________________________________________________________________
___________
leakyrelu_1 (LeakyReLU)          (None, 16, 448, 448)  0           convolution2d_1[0][0]
__________________________________________________________________________________________
___________
maxpooling2d_1 (MaxPooling2D)    (None, 16, 224, 224)  0           leakyrelu_1[0][0]
__________________________________________________________________________________________
___________
convolution2d_2 (Convolution2D)  (None, 32, 224, 224)  4640        maxpooling2d_1[0][0]
__________________________________________________________________________________________
___________
leakyrelu_2 (LeakyReLU)          (None, 32, 224, 224)  0           convolution2d_2[0][0]
__________________________________________________________________________________________
___________
maxpooling2d_2 (MaxPooling2D)    (None, 32, 112, 112)  0           leakyrelu_2[0][0]
__________________________________________________________________________________________
___________
convolution2d_3 (Convolution2D)  (None, 64, 112, 112)  18496       maxpooling2d_2[0][0]
__________________________________________________________________________________________
___________
leakyrelu_3 (LeakyReLU)          (None, 64, 112, 112)  0           convolution2d_3[0][0]
```

```
____________________________________________________________________________________________________
maxpooling2d_3 (MaxPooling2D)    (None, 64, 56, 56)    0           leakyrelu_3[0][0]
____________________________________________________________________________________________________
convolution2d_4 (Convolution2D)  (None, 128, 56, 56)   73856       maxpooling2d_3[0][0]
____________________________________________________________________________________________________
leakyrelu_4 (LeakyReLU)          (None, 128, 56, 56)   0           convolution2d_4[0][0]
____________________________________________________________________________________________________
maxpooling2d_4 (MaxPooling2D)    (None, 128, 28, 28)   0           leakyrelu_4[0][0]
____________________________________________________________________________________________________
convolution2d_5 (Convolution2D)  (None, 256, 28, 28)   295168      maxpooling2d_4[0][0]
____________________________________________________________________________________________________
leakyrelu_5 (LeakyReLU)          (None, 256, 28, 28)   0           convolution2d_5[0][0]
____________________________________________________________________________________________________
maxpooling2d_5 (MaxPooling2D)    (None, 256, 14, 14)   0           leakyrelu_5[0][0]
____________________________________________________________________________________________________
convolution2d_6 (Convolution2D)  (None, 512, 14, 14)   1180160     maxpooling2d_5[0][0]
____________________________________________________________________________________________________
leakyrelu_6 (LeakyReLU)          (None, 512, 14, 14)   0           convolution2d_6[0][0]
____________________________________________________________________________________________________
maxpooling2d_6 (MaxPooling2D)    (None, 512, 7, 7)     0           leakyrelu_6[0][0]
____________________________________________________________________________________________________
convolution2d_7 (Convolution2D)  (None, 1024, 7, 7)    4719616     maxpooling2d_6[0][0]
____________________________________________________________________________________________________
leakyrelu_7 (LeakyReLU)          (None, 1024, 7, 7)    0           convolution2d_7[0][0]
____________________________________________________________________________________________________
```

```
convolution2d_8 (Convolution2D)  (None, 1024, 7, 7)    9438208     leakyrelu_7[0][0]

____________________________________________________________________________________________________

leakyrelu_8 (LeakyReLU)          (None, 1024, 7, 7)    0           convolution2d_8[0][0]

____________________________________________________________________________________________________

convolution2d_9 (Convolution2D)  (None, 1024, 7, 7)    9438208     leakyrelu_8[0][0]

____________________________________________________________________________________________________

leakyrelu_9 (LeakyReLU)          (None, 1024, 7, 7)    0           convolution2d_9[0][0]

____________________________________________________________________________________________________

flatten_1 (Flatten)              (None, 50176)         0           leakyrelu_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     leakyrelu_10[0][0]
====================================================================================================

Total params: 45,089,374

Trainable params: 45,089,374

Non-trainable params: 0
```

Load Pre-trained Yolo weights and Train one image

Load pretrained weights of yolo provided from

https://drive.google.com/file/d/0B1tW_VtY7onibmdQWE1zVERxcjQ/view

Data output for bounding box and probability of each class which recorded in “data.md” file.

The bounding box of vehicle in training image:

[-6.2443885803222656, 22.23392595563616, 2177.4734942449722, 1823.8973605994979]

[8.1763703482491632, 2.4748785836356029, 420.87584318220615, 633.60562289156951]

[-7.6249623979840964, -4.1853103637695312, 3655.9590410698584, 11427.779967978189]

[39.074292864118306, 5.6808046613420755, 42097.977821622044, 32.97495707947769]

[-3.2266101837158203, -8.3505581447056354, 3163.0630805657856, 3610.5449726496736]

[-13.818937029157366, 18.408437456403458, 213.93004241378458, 5060.5619251030148]

[1.5318431173052107, -5.3705831255231589, 371.11749337709989, 10926.247901296301]

[-0.36035473006112234, -3.1194964817592075, 137.45633478127365, 16792.321420513326]

[-18.485462733677455, 24.96602303641183, 31048.148896806175, 263.02700191329131]

[18.209995814732142, -6.85974611554827, 4270.4358333386481, 650.95543337824347]

[4.1464734758649557, -19.608760288783483, 403.16572382865706, 2888.5655044692103]

[10.283409118652344, 11.294999258858818, 197.08981332987605, 22.75501516451186]

[10.924356733049665, 4.104710715157645, 1433.11478402793, 4985.7506377550308]

[-2.2601637159075056, 6.2452278137207031, 18.584896667613293, 2015.9841025611968]

[14.397278921944755, 6.0225312369210382, 12340.756753989495, 2348.5042427848239]

[21.816515241350448, -6.2865398951939175, 5.1132214578594812, 28.415502496536874]

[-6.5367927551269531, 10.683708190917969, 18.260654216897819, 9512.4072314335499]

[3.9819752829415456, 0.15059631211417063, 37338.503499800572, 2321.6925270596985]

[-35.955019269670757, 5.9842344011579245, 2445.5093257385452, 14.417907359584433]

Draw bounding box in image

Result:

Pipeline for video stream

Generate “project_output.mp4”