MODIFIED YOLO WITH CUSTOM DATASET

DEEP LEARNING AND EMBEDDED SYSTEMS FINAL PROJECT

ALAN JACOB	NUID: 001056590
BASIL MIR	NUID: 001057701
KIRAN TULSULKAR	NUID : 001057882

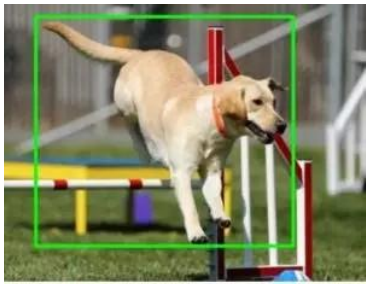
PROBLEM STATEMENT

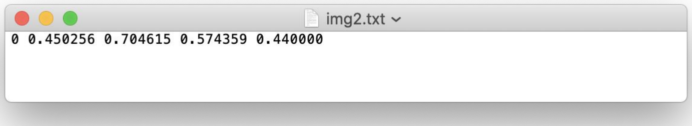
- Object detection models are extremely powerful, however, one of the biggest blockers keeping new applications from being built is adapting state-of-the-art, open source, and free resources to custom problems.
- In a security surveillance system or in any custom application we don’t need yolo to detect all the 80 classes but classes such as person, animals, bag, and other accessories
- We propose to reduce the number of classes that YOLOv3 algorithm detects
- We also plan to reduce the size of our model and reduce the number of trainable parameters in the model by modifying the number of channels at each level in the actual YOLOv3 algorithm
- By extracting few classes and their respective images from coco and open images dataset, we are going to train the model and detect only those classes

Dataset Description

Open Image : Custom Classes such as Person and Dog

- 15,851,536 boxes on 600 categories
- 2,785,498 instance segmentations on 350 categories
- 3,284,282 relationship annotations on 1,466 relationships
- 507,444 localized narratives
- 59,919,574 image-level labels on 19,957 categories
- Extension - 478,000 crowdsourced images with 6,000+ categories





```
0 0.450256 0.704615 0.574359 0.440000
```

Extracting Custom Dataset from COCO or Open Image Dataset

- Downloading the Images and Annotations using Open Images Tools or incase of Coco dataset we have used a pycocotools to extract the required images and annotations.
- Generating the annotations of Open Images Dataset and Coco dataset in YOLO format
- Generating coco.names which has name of the classes i.e in our case Person and Dog
- Generating coco.data which has path of train.txt, valid.txt, backup, coco.names and specifying number of classes in the dataset
- Generating train.txt and test.txt having path of each image

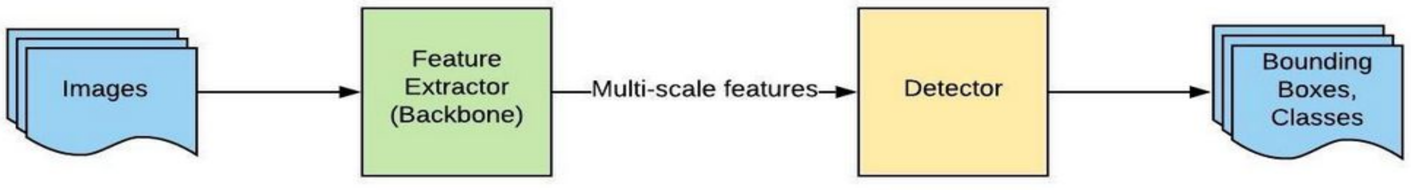
Bounding Box Format in Open Images :

Csv file : **XMin XMax YMin YMax**

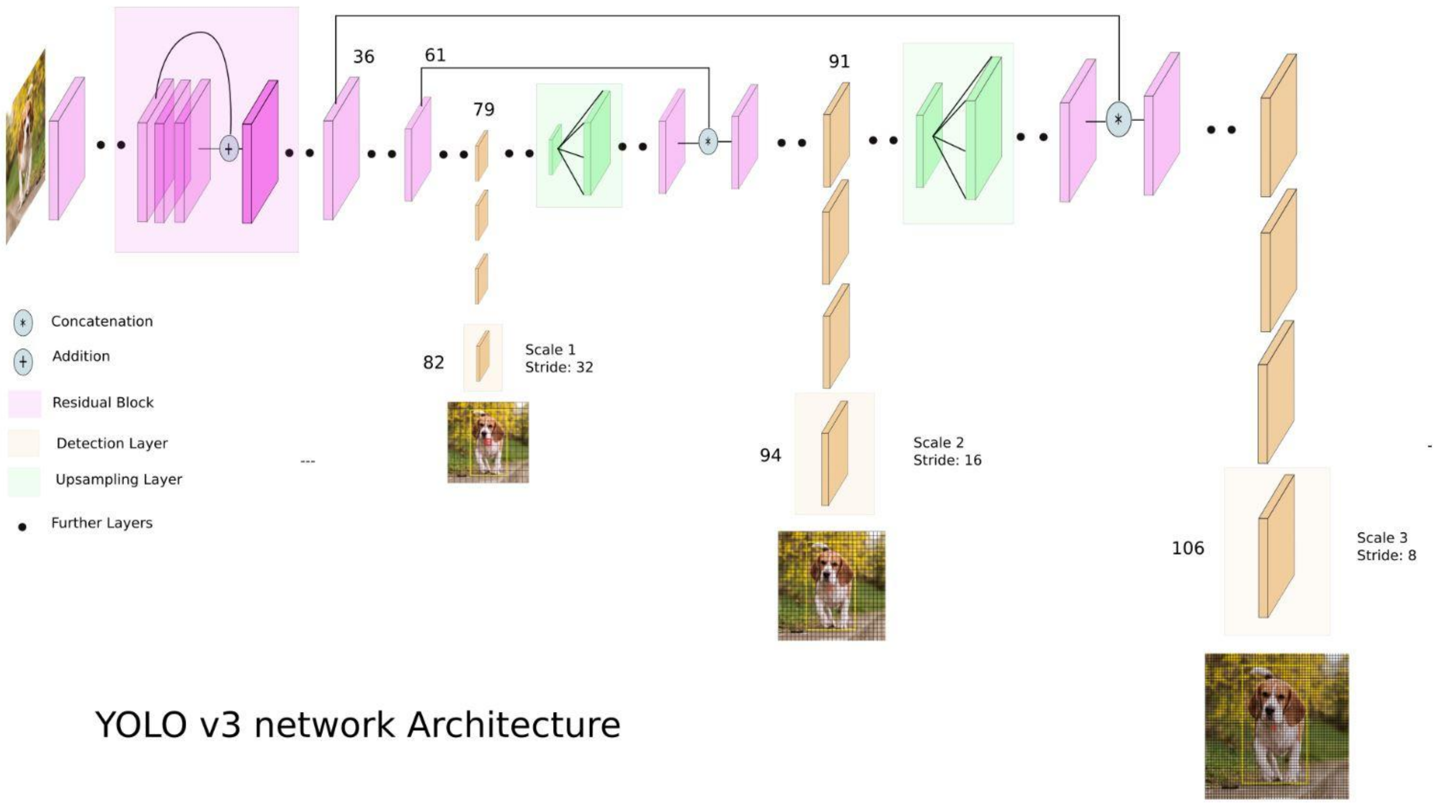
Yolo **: centre in x centre in y width height**

YOLO V3 ALGORITHM

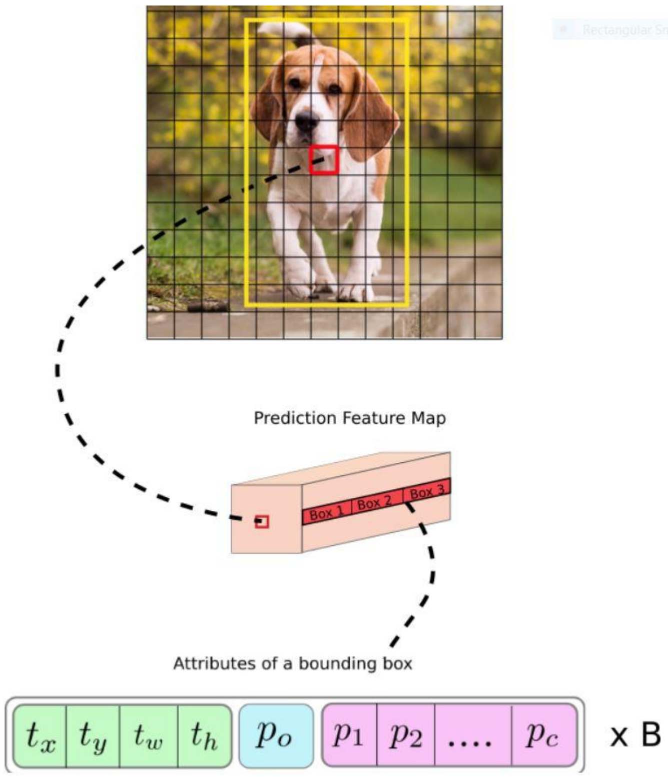
- It is one of the faster object detection algorithms out there and hence a very good choice when you need real-time detection, without loss of too much accuracy.
- In YOLOv3, a much deeper network Darknet-53 is used for feature extraction, i.e. 53 convolutional layers, each followed by batch normalization layer and Leaky ReLU activation.
- The most salient feature of v3 is that it makes detections at three different scales



Img source:https://towardsdatascience.com/dive-really-deep-into-yolo-v3-a-beginners-guide-9e3d2666280e



YOLO v3 network Architecture


Prediction Feature Map
Box 1
Box 2
Box 3
Attributes of a bounding box
t_x t_y t_w t_h p_o p_1 p_2 .... p_c
x B

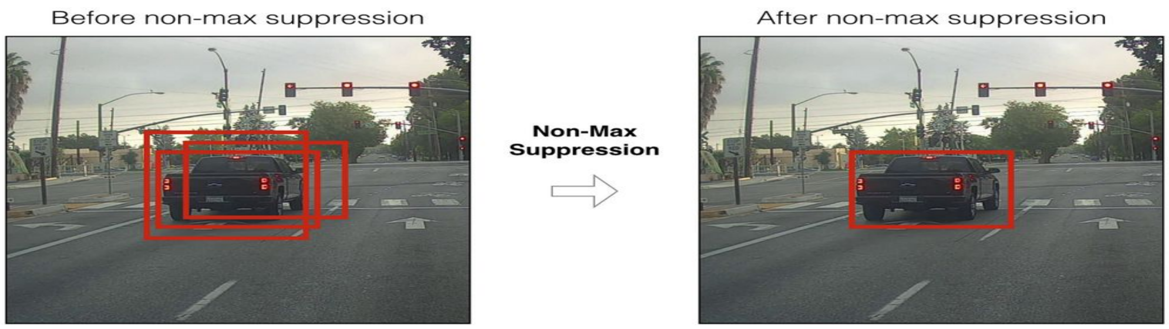
YOLO v3 has three anchors at each detection stage giving 3 boxes per cell. Each anchor box specifies:

- Location offset : *tx, ty, tw, th*. This has 4 values.
- The objectness p_0 to indicate if this box contains an object. This has 1 value.
- Class probabilities
- Shape of detection kernel where B=3 and C=2(Custom dataset)

  1 x 1 x (B x (5 + C))

YOLO V3 ALGORITHM WORKING (Contd.)

- The final component in this detection system is a post-processor. In order to eliminate duplicate results, an algorithm called non maximum suppression (NMS) is used.
- To find out the detection box with the best confidence , we eliminate all other boxes which have IOU over a certain threshold with this box. Next, you choose another box with the best confidence in the remaining boxes and do the same thing over and over until nothing is left.



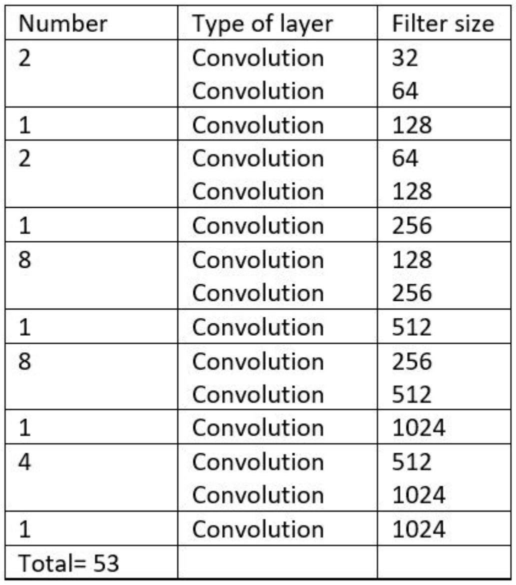
Img source:https://towardsdatascience.com/dive-really-deep-into-yolo-v3-a-beginners-guide-9e3d2666280e

MODIFICATIONS TO MODEL

- We modified the YOLOv3 algorithm using two approaches:
    - One we completely removed three convolution layers from the original model.
        - Improvements:
            - Trainable parameters: Reduced from 61 million in the original to 46 million
            - The size of the model also dropped to 1044 MB from 1118 MB
    - Second approach that we took involved keeping all the convolution layers as it is but reducing the number of filters in each layer.
        - Improvements:
            - The size of our model dropped to 769 MB
            - Trainable parameters reduced to 39 million.

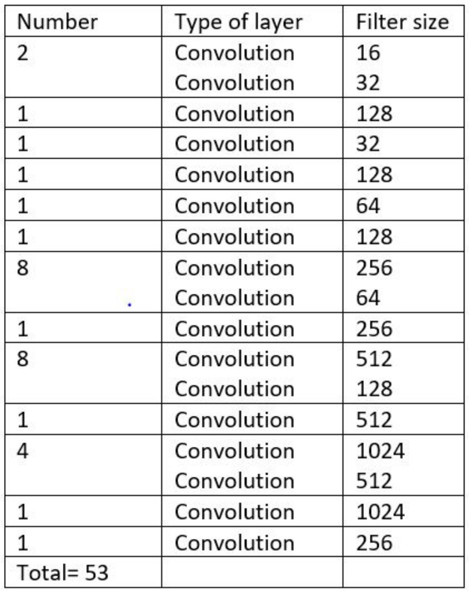
Original YOLO V3 VS Model with Reduced Number of Filters

Original Model



Number	Type of layer	Filter size
2	Convolution Convolution	32 64
1	Convolution	128
2	Convolution Convolution	64 128
1	Convolution	256
8	Convolution Convolution	128 256
1	Convolution	512
8	Convolution Convolution	256 512
1	Convolution	1024
4	Convolution Convolution	512 1024
1	Convolution	1024
Total= 53		

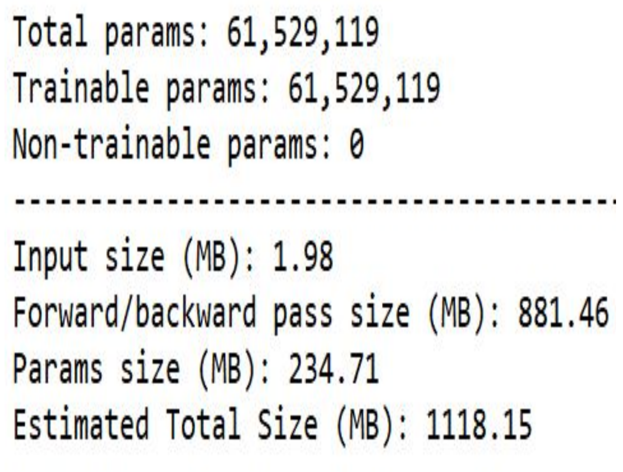
Modified Model



Number	Type of layer	Filter size
2	Convolution Convolution	16 32
1	Convolution	128
1	Convolution	32
1	Convolution	128
1	Convolution	64
1	Convolution	128
8	Convolution Convolution	256 64
1	Convolution	256
8	Convolution Convolution	512 128
1	Convolution	512
4	Convolution Convolution	1024 512
1	Convolution	1024
1	Convolution	256
Total= 53		

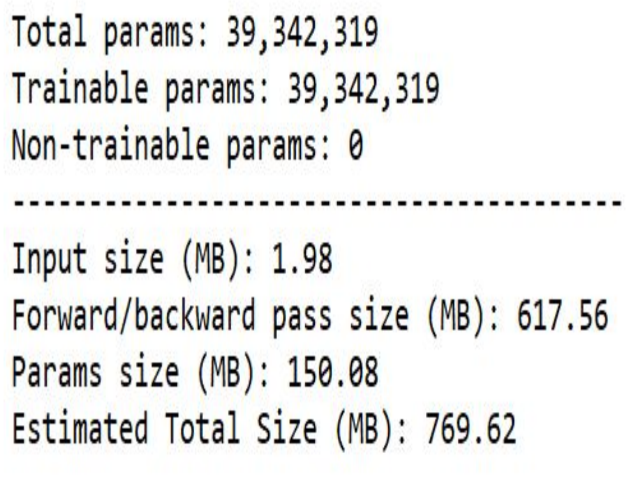
DIFFERENCE BETWEEN ORIGINAL YOLOV3 AND OUR CUSTOM YOLO V3

ORIGINAL YOLO v3



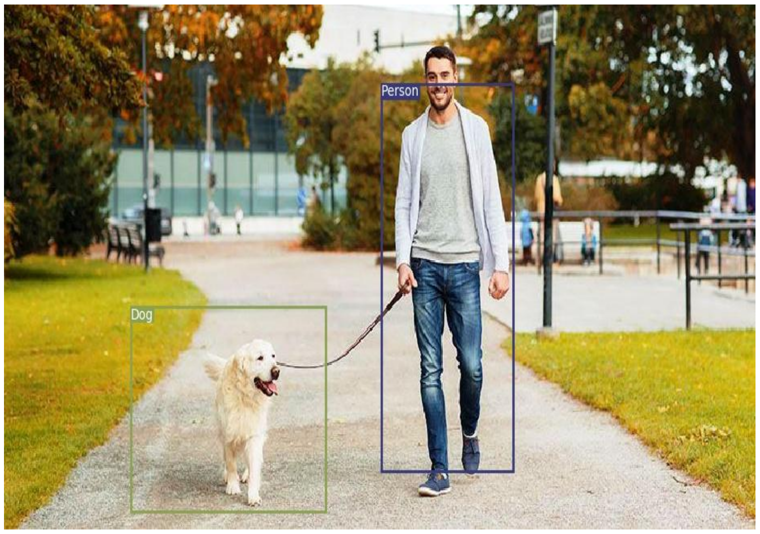
```
Total params: 61,529,119
Trainable params: 61,529,119
Non-trainable params: 0
----------------------------------------
Input size (MB): 1.98
Forward/backward pass size (MB): 881.46
Params size (MB): 234.71
Estimated Total Size (MB): 1118.15
```

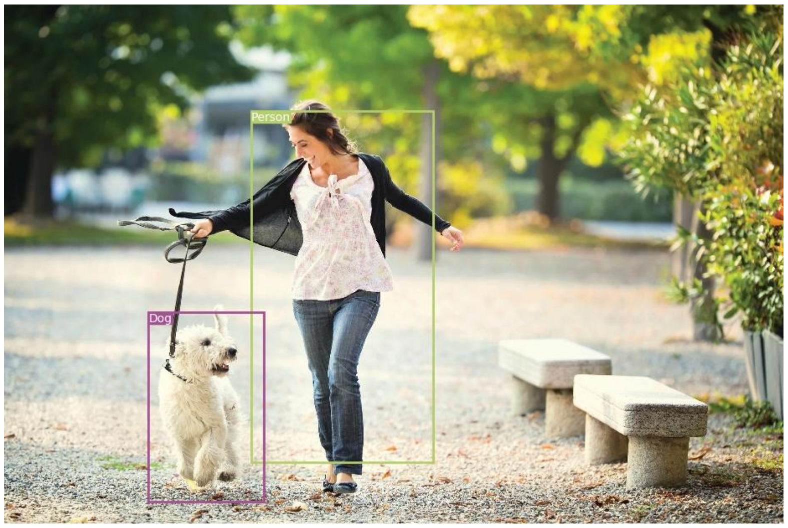
CUSTOM YOLO



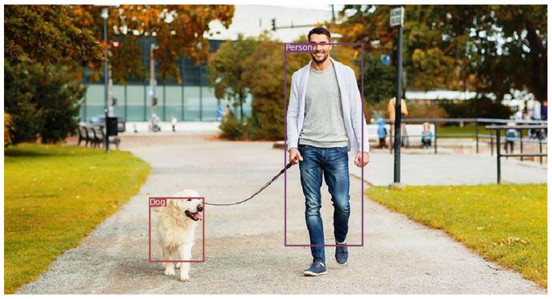
```
Total params: 39,342,319
Trainable params: 39,342,319
Non-trainable params: 0
----------------------------------------
Input size (MB): 1.98
Forward/backward pass size (MB): 617.56
Params size (MB): 150.08
Estimated Total Size (MB): 769.62
```

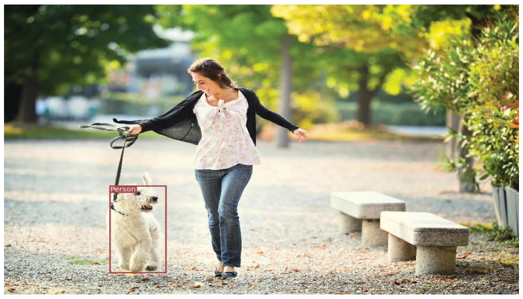
OUTPUT OF ORIGINAL YOLOv3 ON DIFFERENT IMAGES FROM CUSTOM DATASET

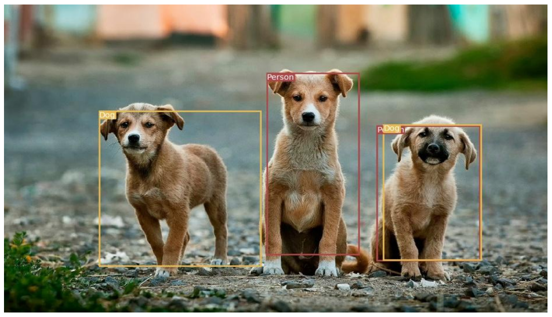


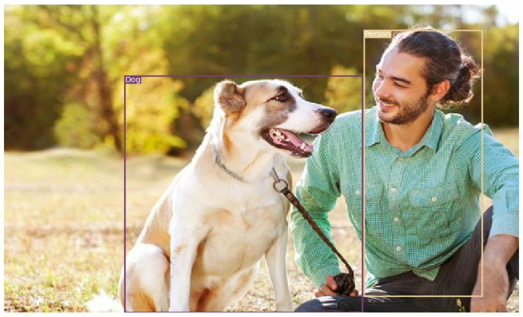


OUTPUT OF OUR CUSTOM MODEL(Reduced number of filters) ON DIFFERENT IMAGES

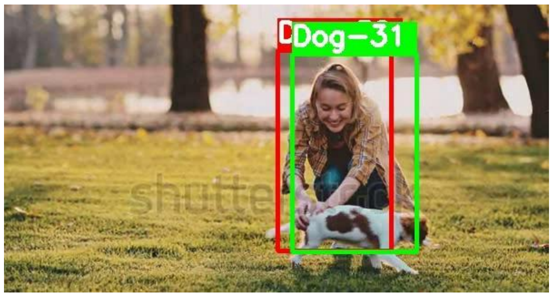


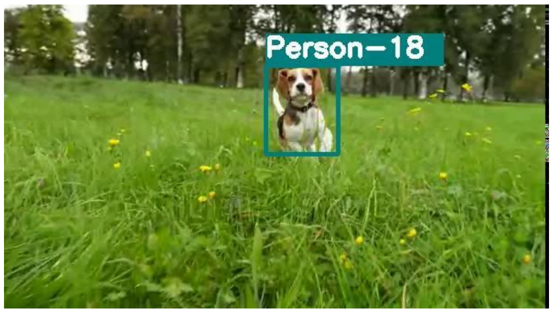


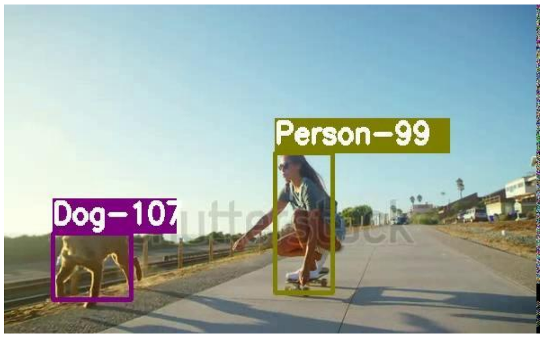




OUTPUT OF OUR CUSTOM MODEL ON DIFFERENT VIDEOS







THANK YOU