REPORT FOR NFT ASSIGNMENT 2

Pre-trained model used = YOLOv3

Dataset MSCOCO
- 80 object categories

Fully convolutional neural network.
24 convolution layers and 2 fully connected layers
Convolution layers for image detection that is make the input of less size with more information.
Convolution is filled in application that is apply some filters to the image to detect some specified features.
After convolution there is a leaky RE-Lu function to make the output non-linear as convolution on linear output is also linear so the composition of convolutions would be like a single convolution.
Not very profitable
Then after that we apply the max-pool layer with different strides to detect any spacial orientation
Last 3 layers are only convolution of layers that is to detect multiple features.

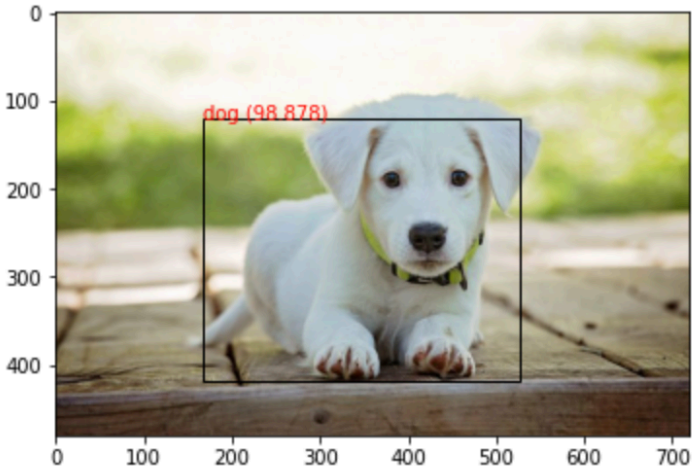
YOLO trains on the full images while others like RCNN uses sliding window or region based technique so this helps YOLO to see the image in a generalised manner and then predict the output. YOLO divides the image into multiple grid and if an object centre falls on the grid it detects it.

Strong spatial limitations since each group can only have one class. Limits the number of objects detected nearby struggles with small objects that appear in groups
Another limitation is there calculations that is treats error same in small boxes versus large boxes so a small error in the large box is irrelevant but a small error in the small box can affect. Main reason for error is incorrect localisation.

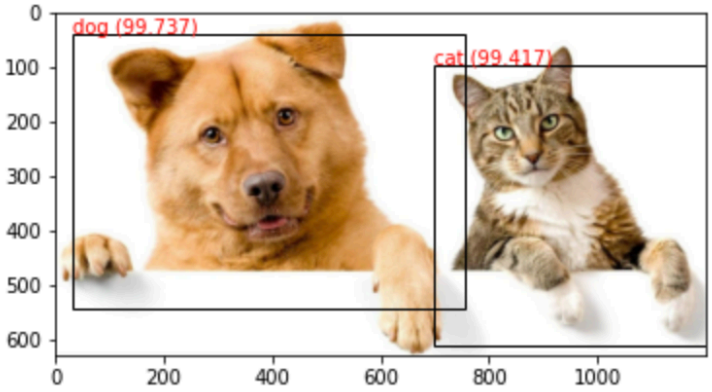
2 activation functions used
1. Convolutional layers = leaky ReLu
2. Fully connected layer (output layer) = Soft-max function

Efficiency was good >95% but due to Strong spacial limitations it gave wrong predictions on some inputs
3rd figure win the next page where multiple animals are on top of each other is such a case

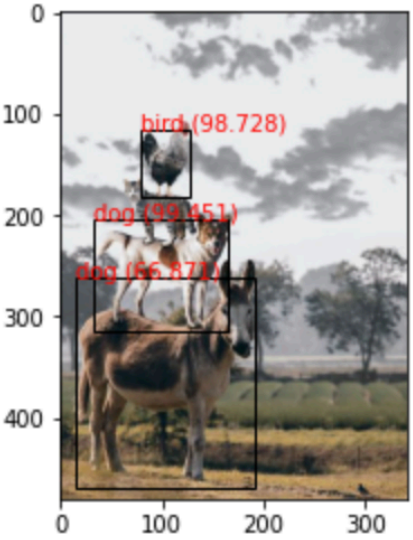
```
[(1, 13, 13, 255), (1, 26, 26, 255), (1, 52, 52, 255)]
dog 98.87832403182983
```




```
[(1, 13, 13, 255), (1, 26, 26, 255), (1, 52, 52, 255)]
dog 99.73706007003784
cat 99.41670298576355
```




```
[(1, 13, 13, 255), (1, 26, 26, 255), (1, 52, 52, 255)]
dog 99.45053458213806
dog 66.8711245059967
bird 98.72798323631287
```



My - Model

Activation function used is sigmoid function

2 hidden layer
Epoch = 2

Hidden1 = 50
Hidden2 = 80
Alpha = 0.5

```
cr = 0
for i in range(10):
  cnn_img.forward(dogs[i])
  lst = list(cnn_img.fout[0])
  i = lst.index(max(lst))
  if(i==0):
      cr+=1
  cnn_img.which_class()
for i in range(10):
  cnn_img.forward(cats[i])
  i = lst.index(max(lst))
  if(i==1):
      cr+=1
  cnn_img.which_class()
for i in range(10):
  cnn_img.forward(horse[i])
  i = lst.index(max(lst))
  if(i==2):
      cr+=1
  cnn_img.which_class()
print(cr)

print('efficiency = ' , cr*10/3)
```

```
3
3
2
1
2
1
1
1
1
1
2
2
1
1
1
1
1
1
1
3
1
1
1
1
2
3
1
2
1
1
6
efficiency =  20.0
```

Hidden layer - 2
Alpha = 0.5
Hidden1 = 40
Hidden2 =50
Epoch = 2

```
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
10
efficiency =  33.333333333333336
```

Hidden layer -2
Alpha = 0.1
Hidden1 = 60
Hidden2 = 80
Epoch = 2

```
2
2
3
1
1
1
3
2
1
2
2
1
2
2
2
2
2
2
1
3
3
1
1
1
2
1
2
2
3
2
14
efficiency =  46.666666666666664
```