CSM-392: Undergraduate Project – I Presentation

Topic: <u>Object Detection and Localization using YOLOv3</u>

Submitted by:
Aditya Kulkarni
Roll no:
20124054

Planning

- After formulating the work plan back in January 2023, I set up a reasonable schedule to work on the project regularly to complement my exploration of the field of computer vision.
- The link to the plan can be found out here: UG: Project Layout.
- This plan consisted of studying and researching along with coding and testing.
- The work plan had been duly submitted to **Prof SK Pandey**, and this project has been done under his mentorship and guidance.

Abstract

- The goal of this project was to build an object detection system that uses YOLOv3 to detect and localize objects within a bounding box.
- The major area of study was Convolutional Neural Networks (CNNs).
- The study involved the readings of various types of CNNs, different architectures comprising CNNs, their working and the intuition behind it.

- The study involved the readings of various types of CNNs, different architectures comprising CNNs, their working and the intuition behind it.
- Finally, this knowledge was applied in the form of code to build a YOLOv3 object detection model that selectively detects objects out of 80 possible classes and localizes them in a video.

Convolutional Neural Networks

- **Convolutional neural networks** are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs.
- In the case of MLPs (multilayer perceptrons), the amount of weights become unmanageable for large images.

- CNNs are typically grids of parameters that traverse the image in a set manner and essentially capture the information available in the cell it traverses.
- It starts traversing from the upper left corner of the image and moves along each 3x3 portion of the image, performing matrix scalar product
- It houses three layers which are Convolutional Layer, Pooling Layer, and Fully Connected (FC) Layer.

Object Detection and YOLO

- Object detection involves not only detecting what object is in an image, but also where it is located.
- This is done with the help of bounding boxes that encompass the object in the output image.
- The training for this is slightly different from that of an object classifier.

- While an object classifier outputs one of the various possible classes of the object after running a CNN model over the image, an object detector does the same in addition to finding the coordinates of the center of the bounding box and its height and width.
- Thus, the output in this case is larger than the one for an object classifier since the location of the object detected needs to be stored too.

- I read the research papers of some of the object detectors such as **U-Net** and **EfficientDet**.
- **YOLOv3** is one such object detector that passes over the image once, and fine-tunes preset bounding boxes (also known as anchor boxes) according to the locations of the objects it detects as the model passes over the image.
- This way, it is able to detect the object in one pass and is hence rightfully named, “You Only Look Once”.

- I developed a localized and personalized version of YOLOv3 myself by writing the code of the model myself and then applying the pre-trained weights instead of randomized ones.

- Finally, I ran the model over an image and a video and received satisfactory results.

YOLO v3 Idea

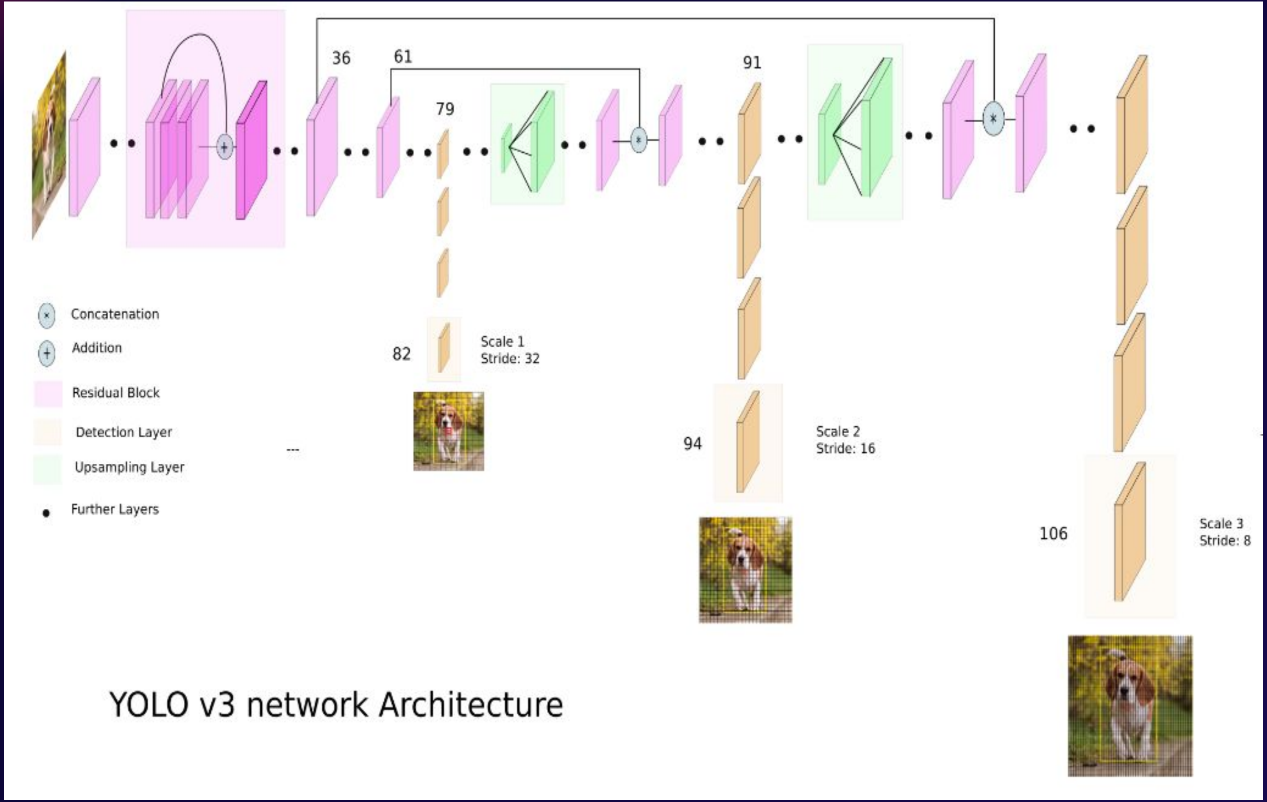
- Consider the image as a S × S grid. If the center of a target falls into a grid, the grid is responsible for detecting the target.
- Each grid will output a bounding box, confidence, and class probability map.
- The bounding box contains four values: x, y, w, h, (x, y) represents the center of the box. (W, h) defines the width and height of the box.

- Confidence indicates the probability of containing objects in this prediction box, which is the IoU value between the prediction box and the actual box.
- The class probability indicates the class probability of the object, and the YOLOv3 uses a two-class method.

YOLO v3 Architecture

- YOLO (You Only Look Once) applies a single forward pass neural network to the whole image and predicts the bounding boxes and their class probabilities as well, which makes it quite fast.
- YOLOv3 has 53 convolutional layers called Darknet 53, the layers are in the form of convolutional and residual structures.
- The last three layers Avgpool, Connected, and softmax layer, are used for classification training on the Imagenet dataset. When using the Darknet-53 layer to extract features from the picture, these three layers are not used.

The YOLO v3 Architecture



Transfer Learning

A technique to reuse the weights in one or more layers from a pre-trained network in a new model by

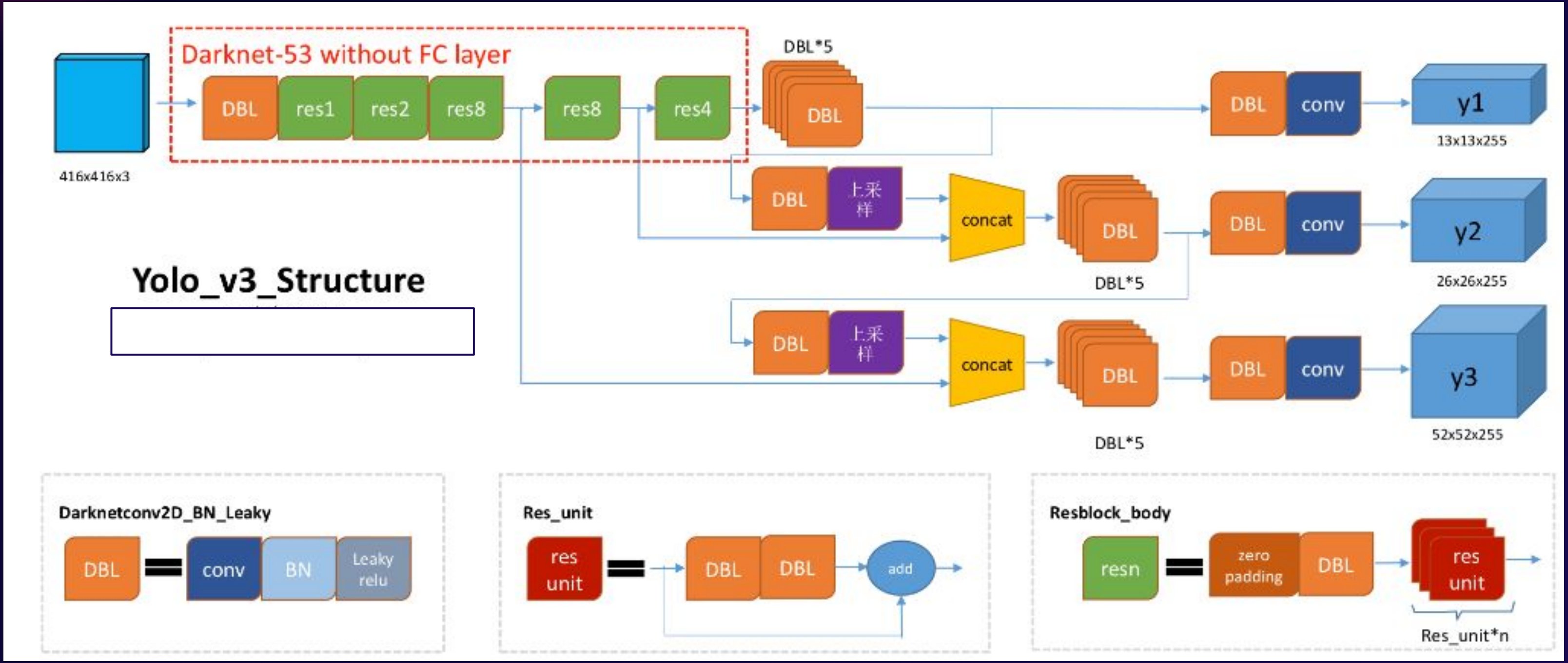
1. Keeping the weights
2. Fine-tuning the weights
3. Adapting the weights entirely when when training a new model

Structure of YOLO v3

- YOLO makes detection in 3 different scales to accommodate various objects sizes by using strides of 32, 16, and 8.
- This means, if we feed an input image of size 416x416, YOLOv3 will make detection on the scale of 13x13, 26x26, and 52x52.
- YOLOv3, downsamples the input image into 13 x 13 and predicts the 82nd layer for the first scale. The 1st detection scale yields a 3-D tensor of size 13x13x255.

- After that, YOLOv3 takes the feature map from layer 79 and applies one convolutional layer before upsampling it by a factor of 2 to have a size of 26 x 26.
- This upsampled feature map is then concatenated with the feature map from layer 61. The concatenated feature map is subjected to a few more convolutional layers until the 2nd detection scale is performed at layer 94.
- The second prediction scale produces a 3-D tensor of size 26 x 26 x 255.

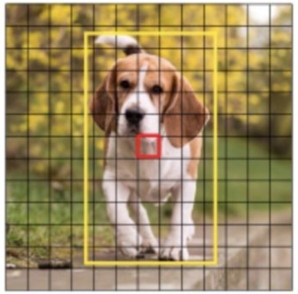
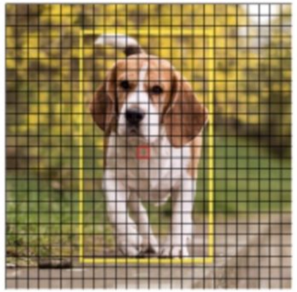
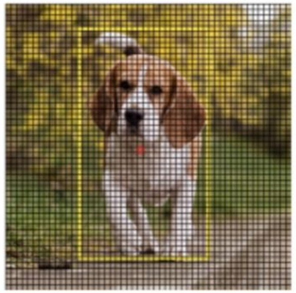
The overall structure can be understood in more detail with the help of the following diagram



YOLO v3 Detection

Multiscale Detection

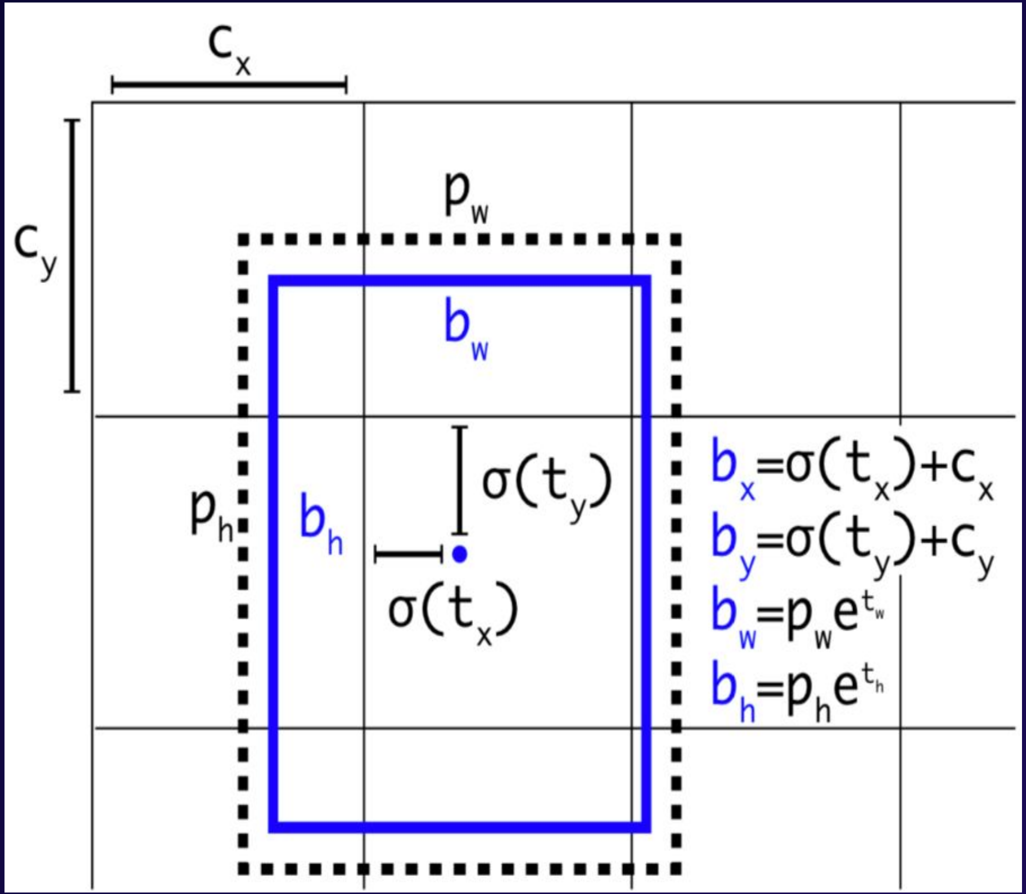
YOLO performs coarse, medium, and fine meshing of the input image to predict large, medium, and small objects, respectively. In this way, it is scaled by 32, 16, and 8 times in length and width, respectively:

Large	Medium	Small
 13 x 13	 26 x 26	 52 x 52

Dimensions of the Bounding box

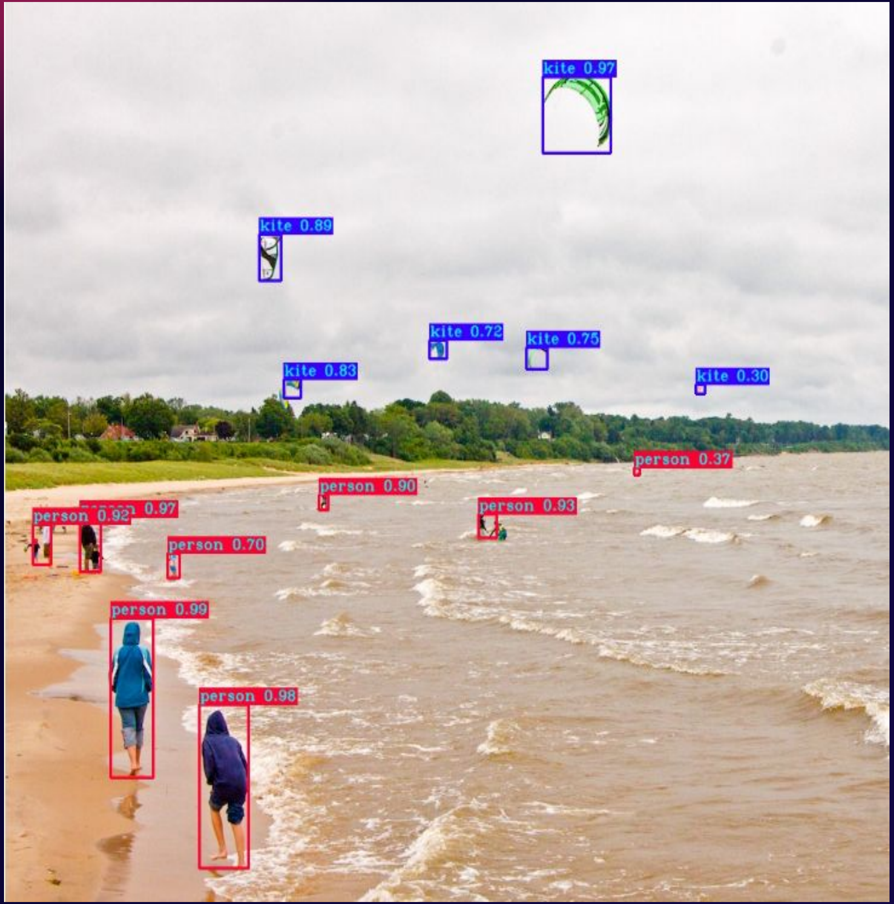
- The output of the three branches of the YOLOv3 network will be sent to the decode function to decode the channel information of the Feature Map.
- The dimensions of the bounding box are predicted by applying a log-space transformation to the output and then multiplying with an anchor: in the following picture: the black dotted box represents the a priori box (anchor), and the blue box represents the prediction box.

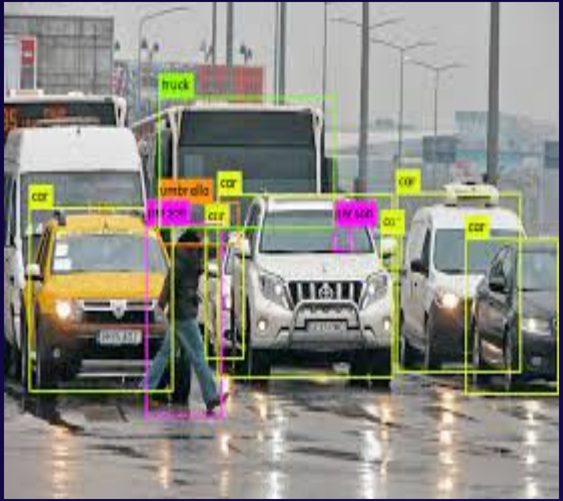
- **b** denotes the length and width of the prediction frame, respectively, and **P** represents the length and width of the a priori frame.
- **t** represents the offset of the object's center from the upper left corner of the grid, and **C** represents the coordinates of the upper left corner of the grid.



Results

- Hence applying the above knowledge, I constructed and compiled the YOLO v3 model in Tensorflow and keras.
- It also required transferring weights from original Darknet weights to constructed model.
- Object Detection was carried out and satisfactory results were obtained.
- Complete codebase can be found out at my Github Repository.


kite 0.97
kite 0.89
kite 0.72
kite 0.75
kite 0.83
kite 0.30
person 0.37
person 0.90
person 0.97
person 0.92
person 0.93
person 0.70
person 0.99
person 0.98


truck
umbrella
car
car
car
person
car
person
car
car
car

Future Scope

- The YOLO v3, can be further extended to detect tiny objects, which can be easily done with using YOLOv3-tiny.
- The model can be tweaked according to the needs, as there is a tradeoff between speed and accuracy, as YOLOv3-tiny is much faster yet less accurate.
- Further, this object detection model can be extended to real time object tracking. Where approaches such as Deep Regression Networks, ROLO (Recurrent - YOLO).
- Also, using DEEP SORT (Simple Real-time Tracker), helps us to track objects in real time and detect them.

Conclusion

- Exploring object detection was an excellent learning experience. I was able to realize the wide range of applications that this technology can have.
- This technology can be used in autonomous vehicles, face detectors, image filters, satellite mapping, robot vision, and so much more.
- Computer vision is still in a growing age and after my experience with it in the last few months, I am sure that it has the potential to completely change the way we live.

Thank You