Project Report: Object Detection using YOLOv3 in Images and Videos

Mahdi Razzaghpour, Ghayoor Shah
{razzaghpour.mahdi, gshah8}@knights.ucf.edu
University of Central Florida, Orlando, FL

***Abstract*—Object detection is an integral component of computer vision as it can allow a computer to determine the existence and class of various objects in its vicinity. One of the biggest applications of object detection is in autonomous vehicles since they require situational awareness to ensure that the safety levels are met. In addition, autonomous vehicles require object detection to be performed in real-time. When it comes to real-time object detection for autonomous vehicles, the most widely used detection system is known as You Only Look Once (YOLO). In this paper, we perform object detection using pre-trained Yolo version 3 (YOLOv3) on images and then on videos. Next, we take it to the next step by fine-tuning the network and re-training it using the widely-used COCO dataset.**

***Index Terms*—Real-Time Object Detection, YOLO, COCO**

I. Introduction

Object detection is one of the main components of the field of Computer Vision since it enables a model to not only detect the existence of an object in an image/video, but is also provides localisation information where a model can identify a particular object and where it is situated in the input data. One application where the concept of object detection can be utilized is in autonomous vehicles where vehicles need to know the objects in their surrounding so that they can avoid collision with them and prevent any hazardous scenario. Autonomous vehicles not only require object detection based on an input, rather they require real-time object detection. You Look Only Once (Yolo) is one such detection system that can assist autonomous vehicles in making real-time object detection.

In this paper, we explore the implementation of Yolo version 3 (Yolov3) [1] in real-time object detection. We study the standard network architecture of Yolov3 and first use a pre-trained model to evaluate the performance of Yolov3 on a sample image. Next, we take it to the next step and evaluate the same performance using a sample input video of a road scenario with vehicles, pedestrians, traffic lights, etc. Once that is done, we move to the next phase of the project where we tune the parameters of Yolov3 and re-train the network. For initial weights, we use Darknet backbone weights. We finally show the performance of the re-trained network on the same sample image and traffic video.

The rest of the paper is organized as follows. Section II provides a brief overview of the COCO dataset. Section III provides an overview of Yolov3 and its network implementation. Section IV provides our implementation details where we show the results of pre-trained and re-trained Yolov3 on test images and videos. We also provide average precision (AP) and average recall (AR) values of all the scenarios that we run. We conclude the paper in Section V and in section VI, we present the contributions details of the authors regarding this project.

II. Details of COCO Dataset

The Common Objects in Context (COCO) dataset is one of the most popular open source object recognition databases used to train deep learning programs. This database includes hundreds of thousands of images with millions of already labeled objects for training. Sponsored by Microsoft, COCO segments images into categories and object, while also providing machine-readable context captions and tags.

COCO dataset contains 91 common object categories with 82 of them having more than 5,000 labeled instances. In total the dataset has 2,500,000 labeled instances in 328,000 images. In contrast to the popular ImageNet dataset, COCO has fewer categories but more instances per category. This can aid in learning detailed object models capable of precise 2D localization. The dataset is also significantly larger in number of instances per category than the PASCAL VOC and SUN datasets. [2]

III. YOLO v3

For it's time YOLO 9000 was the fastest, and also one of the most accurate algorithms. However, a couple of years down the line and it's no longer the most accurate with algorithms like RetinaNet, and SSD outperforming it in terms of accuracy. It still, however, was one of the fastest.But that speed has been traded off for boosts in accuracy in YOLO v3. This has to do with the increase in complexity of underlying architecture called Darknet.

YOLO v2 used a custom deep architecture darknet-19, an originally 19-layer network supplemented with 11 more layers for object detection. With a 30-layer architecture, YOLO v2 often struggled with small object detections. This was attributed to loss of fine-grained features as the layers downsampled the input. To remedy this, YOLO v2 used an identity mapping, concatenating feature maps from from a previous layer to capture low level features.

However, YOLO v2's architecture was still lacking some of the most important elements that are now staple in most of state-of-the art algorithms. No residual blocks, no skip connections and no upsampling. YOLO v3 incorporates all of these.

First, YOLO v3 uses a variant of Darknet, which originally has 53 layer network trained on Imagenet. For the task of detection, 53 more layers are stacked onto it, giving us a 106 layer fully convolutional underlying architecture for YOLO v3. This is the reason behind the slowness of YOLO v3 compared to YOLO v2. Here is how the architecture of YOLO now looks like in Fig.1 [1]

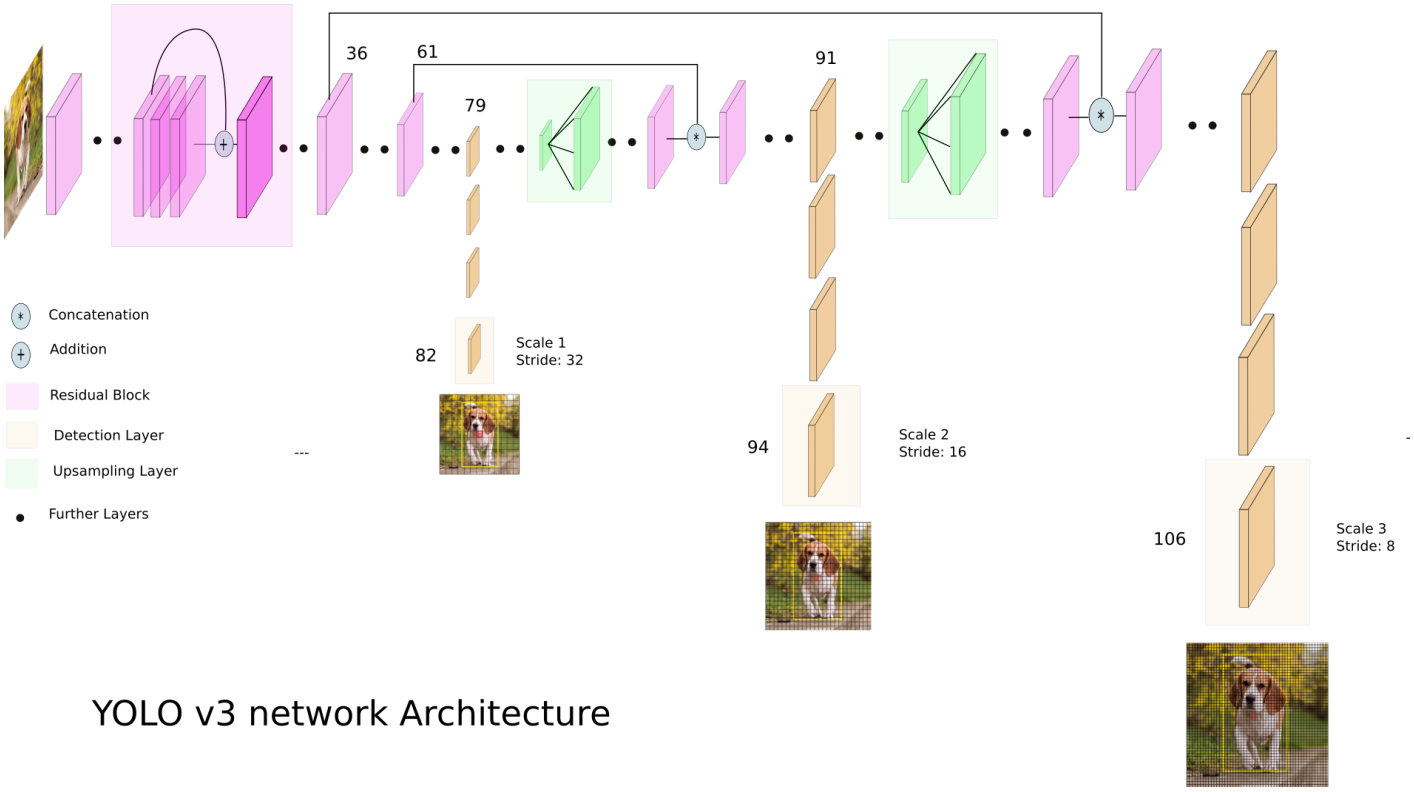


Fig. 1: YOLO v3 Network

IV. Implementation Details

In order to observe the performance of the original pre-trained Yolov3 system with YOLOv3 weights, we evaluate it on a sample image. As shown in Fig.2d, the pre-trained Yolo performs very nicely and detects most of the objects in the image. It should be noted that this model was originally trained for 500,000 epochs. Although the pre-trained model evaluation is shown to be effective, it is important to test the pre-trained model on videos as well. Thus, we implement a program that allows this model to be tested with videos in addition to images. We present the sample video for this test in the presentation. It can be seen that the performance of sample traffic videos is very good and shows the accuracy and effectiveness of the model.

After showing the performance of the pre-trained model, we want to re-train the model with slightly different parameters. The number of epochs for re-training are limited to 6000 due to time constraints. The reason behind re-training the network is to check the progress of training for every 2000 epochs and to visualize how the object detection improves as the number of epochs increase. Additionally, one more reason for re-training is so that we can test out how the tweaked hyper-parameters affect the object detection process as the number of epochs increase. The learning rate is increased from 0.001 to 0.003 because of fewer epochs. We also reduce the Non-Maximal Suppresion (NMS) threshold from 0.7 to 0.5 to facilitate easier detections.

The following figures Fig.2b, c, d, show the object detection performance of re-trained model after every 2000 epochs. These images can assist the user in visualizing how object detection gradually improves as the number of epochs increase. This can also assist in implying that as the model is trained for even more number of epochs as the original model (around 500000 epochs), it can detect more number of objects and can also have better bounding boxes around the detected objects.

V. Conclusion

In this paper, we have attempted to explore the area of real-time object detection that can be useful for autonomous vehicles. We have utilized the widely used detection system known as Yolov3 along with COCO dataset. We not only show the performance of the pre-trained Yolov3 in images and videos, but we also re-train it and show the performance using the same test images and videos. Although the training on the pre-trained model of the original Yolov3 was done for 500000 epochs, we limit our number of epochs to 6000 due to time constraints. Although the limit in the number of epochs for the re-trained model is way lesser than that of the pre-trained model, we show that we can still detect a good number of objects in real-time if we efficiently choose the parameters.

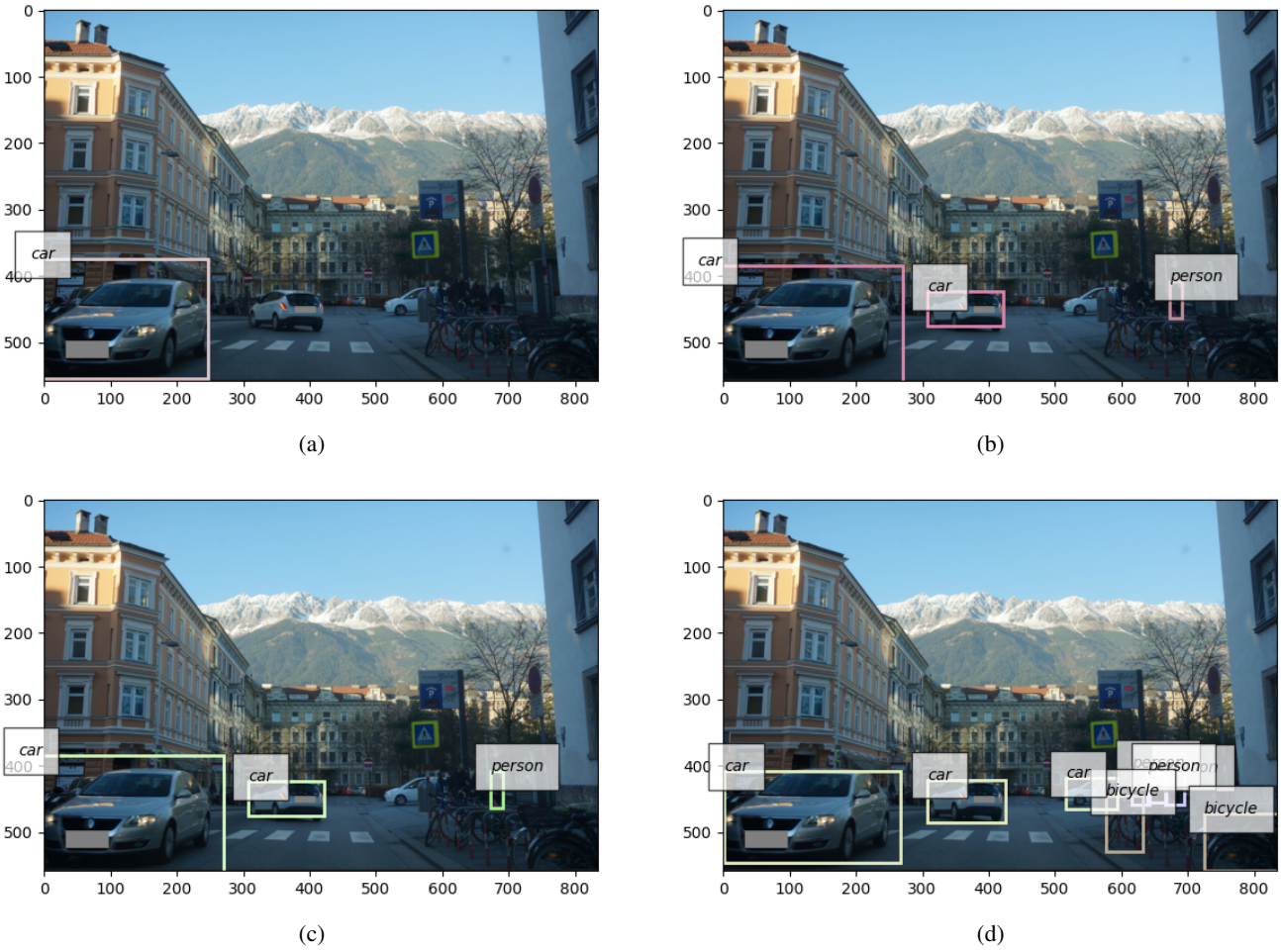
VI. Contributions

Both the authors have contributed equally in this project, including code and report. For the coding, both Ghayoor and Mahdi worked on getting the pre-trained Yolov3 run for images. Once that was done, Mahdi worked on getting the pre-trained network run for videos. For the next phase, Ghayoor contributed in re-training the network with different parameters and making it work for images. The report writing was also divided up equally between the two authors.

References

[1] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv*, 2018.

[2] Schiele B. Tuytelaars T. Fleet D., Pajdla T. Microsoft coco: Common objects in context.



(a) (b) (c) (d)

Fig. 2: (a) Object Detection using 2000 iteration (b) Object Detection using 3000 iteration (c) Object Detection using 6000 iteration (d) Object Detection using pretrained network