**Custom Object Detection using YOLO V3**

-by Samarth Anand

B.Tech., CSE Dept. MIET, Meerut

**ABSTRACT:**

Human beings the most wonderful creation of god can easily identify and detect objects in the surroundings, without consideration of their circumstances, like it doesn’t matter what type of object it is, where the object is placed, what is the color/texture/shape of the object. Therefore, it is easy to view the objects present around us. But to detect and recognize similar objects without the interference of human beings i.e. using computers requires lots of computer processing to extract information from the shapes and objects in a picture. So, we are using YOLO (You Look Only Once) which scans the provided image only once and provides an output based on that single processing. As it traverses the image only once hence speed to detect an object in this algorithm is very fast besides its fast speed it also provides high accuracy than another object detection algorithm.

**INTRODUCTION:**

In computer vision, object detection refers to detecting and identifying of an object from the images or videos. Here we have used object detection in Grocery Shops to identify different grocery products and maintain their records on the main server so that it becomes very easy to identify the availability of stocks on different grocery shops and dealer supply the products to the merchant when dealer finds the requirement.

The main steps used in object detection are:-

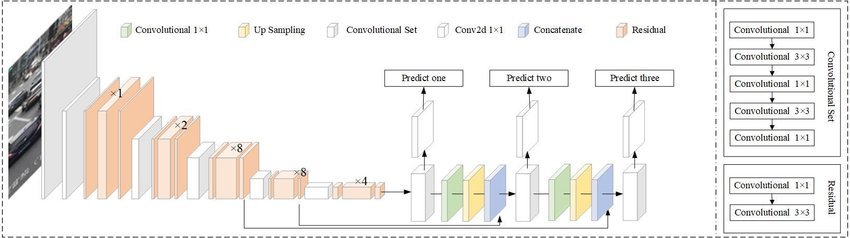
1. Feature Extraction
2. Feature Processing
3. Object Classification
4. Object Detection

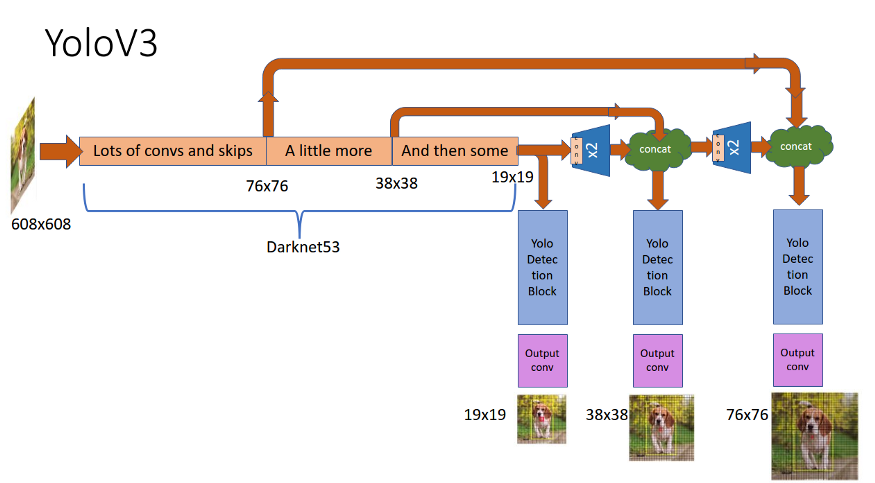
Main challenge behind this system is to detect custom object with high accuracy and fast speed so we are using YOLO algorithm which detects object with high accuracy and very fast speed.

**How YOLOv3 works?**

YOLOv3 is a Convolutional Neural Network (CNN) for doing object detection. CNN’s are classifier-based systems that can process provided images in the structured arrays of data and detects patterns presents between them. It has the advantage of being very much faster than other networks and still maintains the accuracy of detected object.

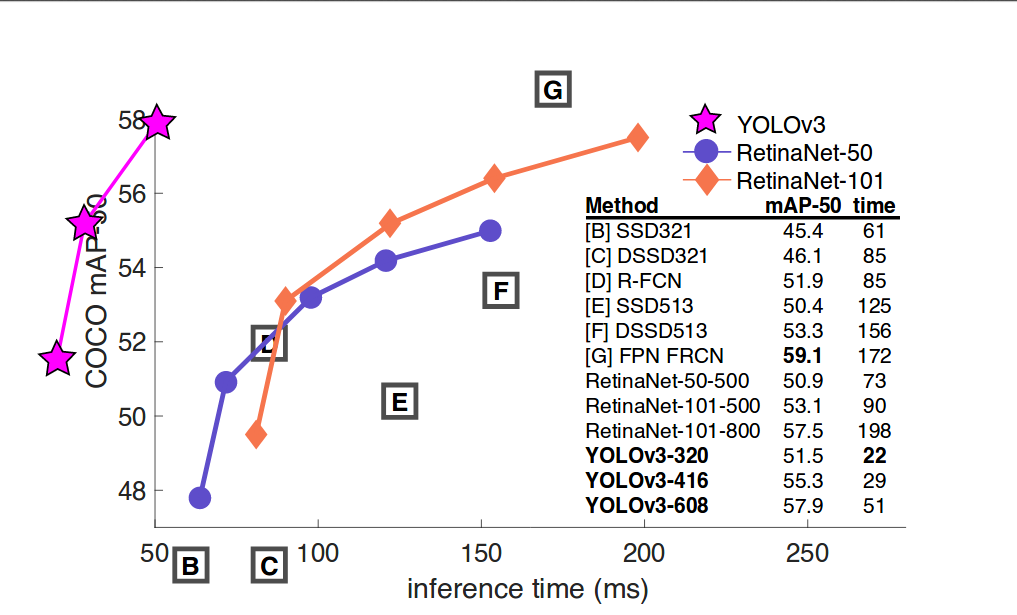
It allows the model to look at the whole sets of images at test time so its predictions are informed by the global context in the image. YOLO and other convolutional neural network (CNN) algorithms “score” regions are based on their similarities to predefined classes. High-scoring regions refers to the positive detections of whatever class they most closely identify with. For example, in a live feed of traffic, YOLO can be used to detect different kinds of vehicles depending on which regions video belongs score highly in comparison to predefined classes of vehicles.





Yolov3 Architecture

**Comparison of Yolov3 with other Algorithms:**

****

**Files Requirement for Yolov3 model?**

To detect custom object using yolov3 we require following files: -

1. Weight’s file
2. Configuration file (cfg)
3. Classes file
4. OpenCV

Weight’s file: - It contains the detail of objects we want to detect through yolov3 so that the detection occurs.

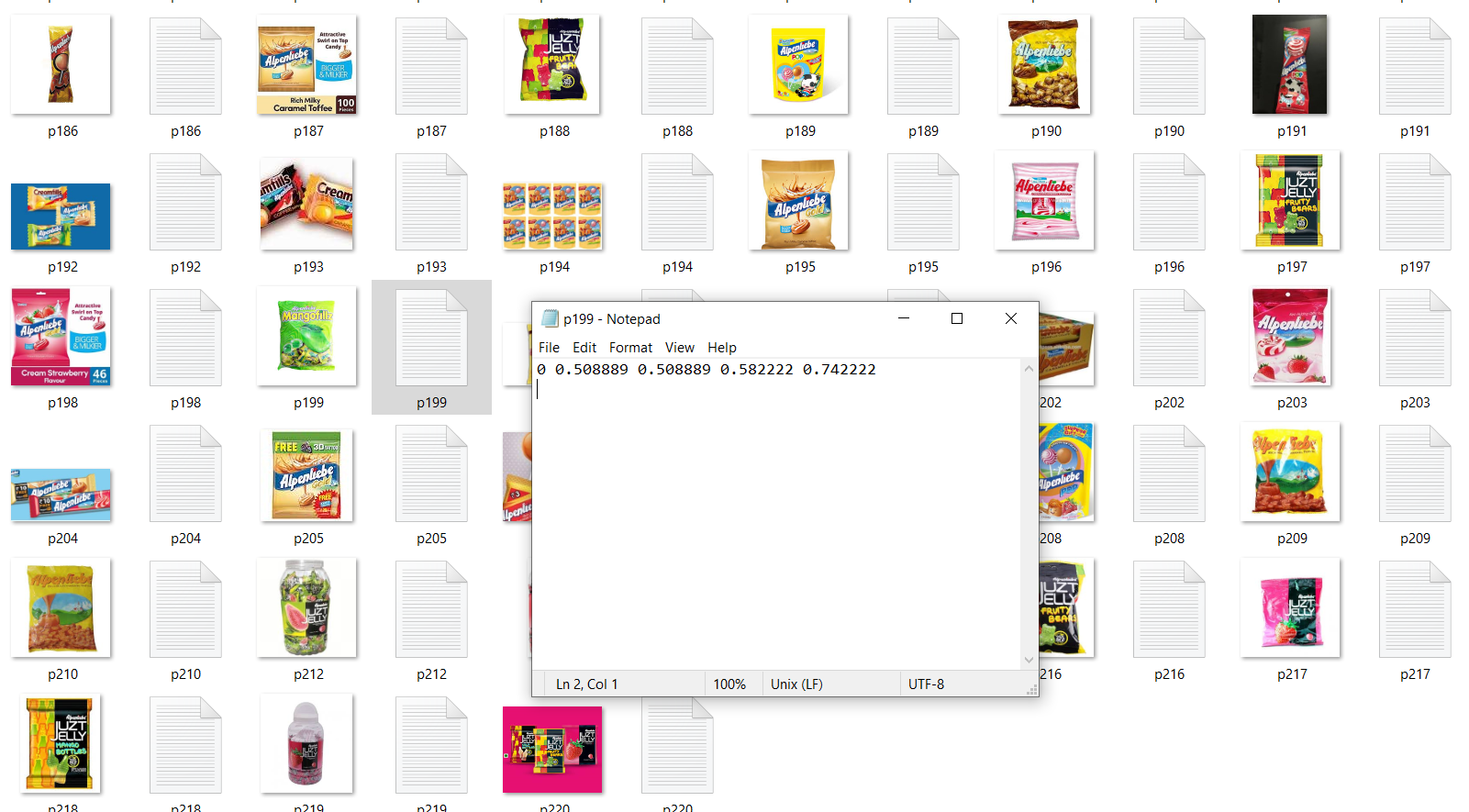
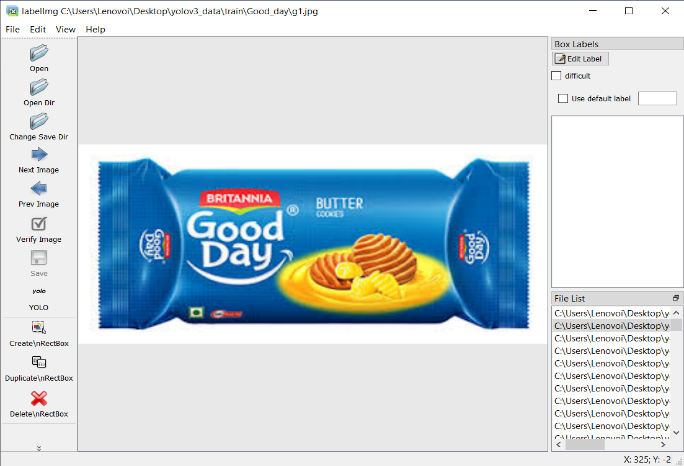
cfg file: - It contains the required information which weight file needed to perform detection.

Steps to Generate cfg for your project is described below

Classes file: - It contains the name of trained object in the form of indexes which is used to display names based on index of detection.

OpenCV: - It provides the platform to yolov3 to detect object.

To generate all the required files for our model firstly we have to train our model for we require lots of images after that we have to label all the images so that these images can be used by model to train itself.



training images annotate images using LabelImg

**Why we used Yolov3?**

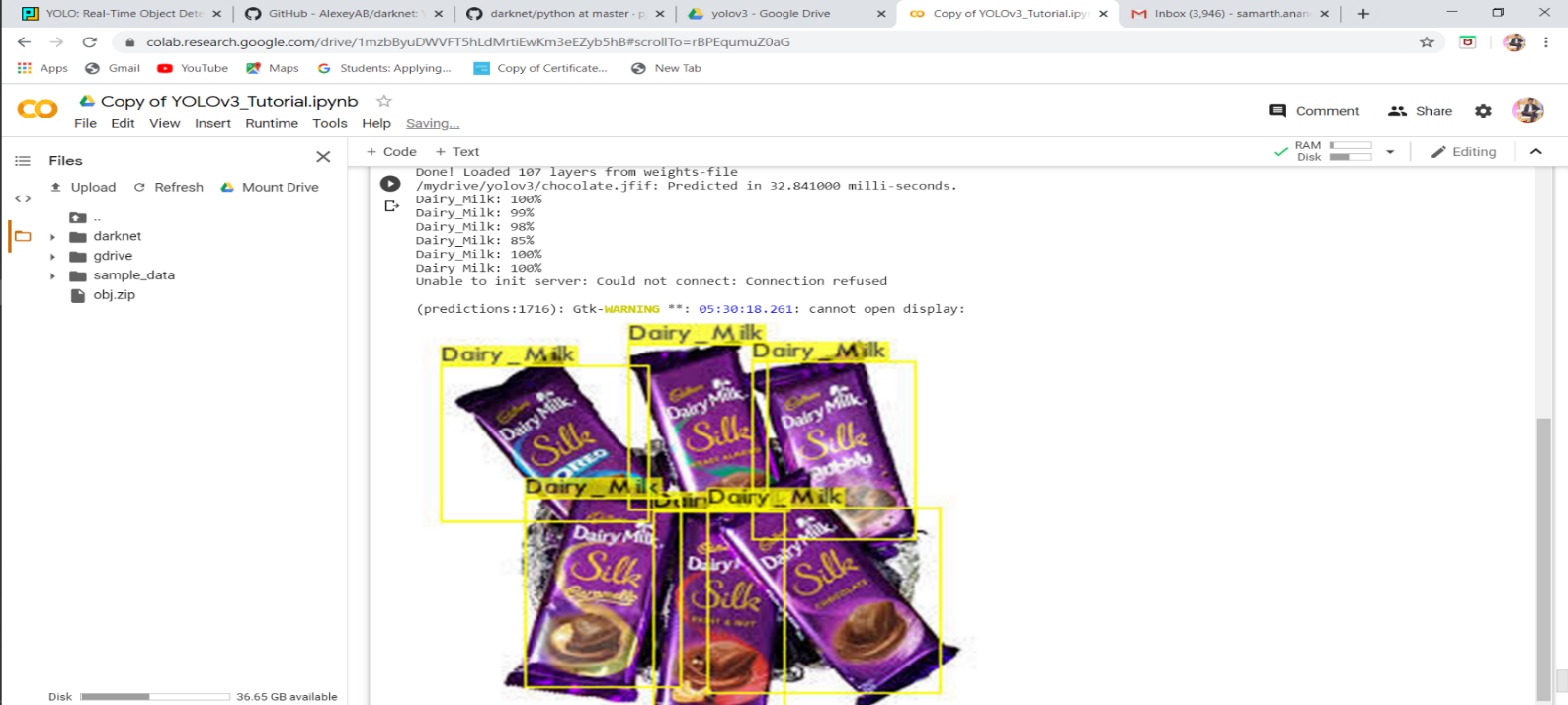
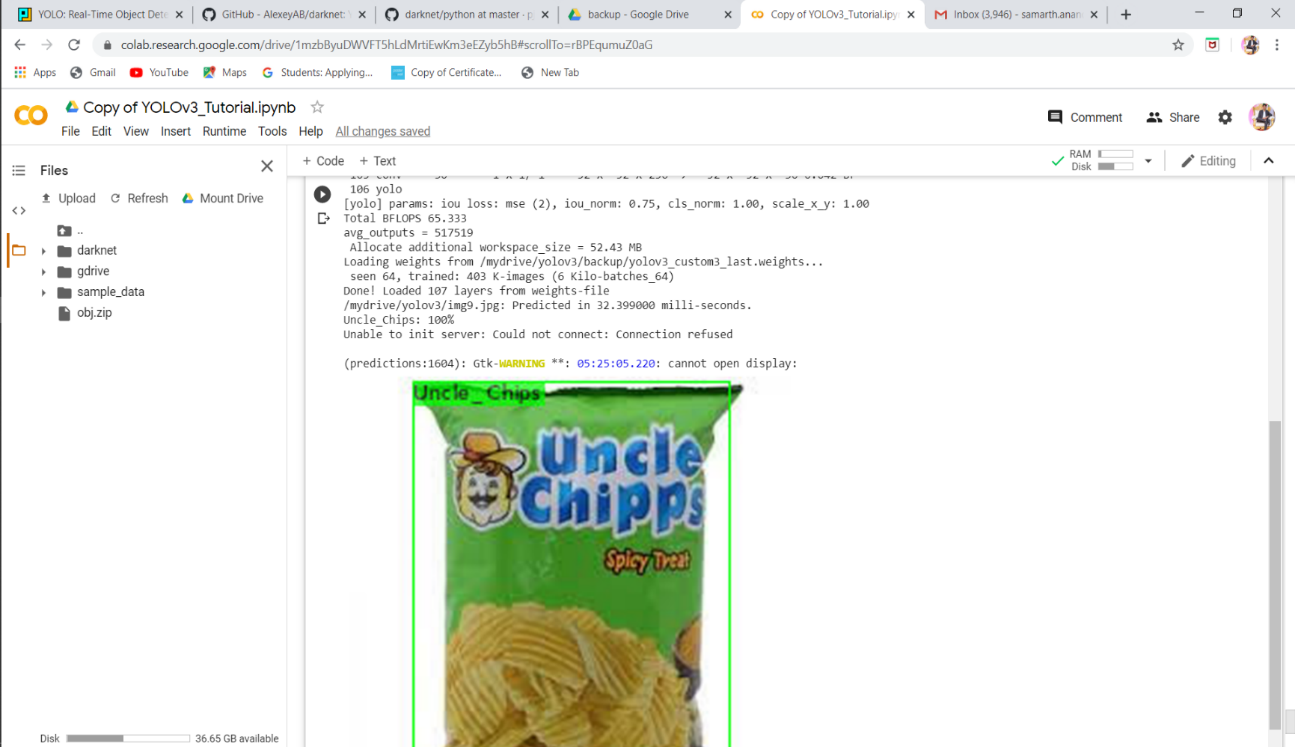
We used yolov3 for custom object detection due to high prediction speed (approx. 20 milliseconds) and high accuracy in object detection.

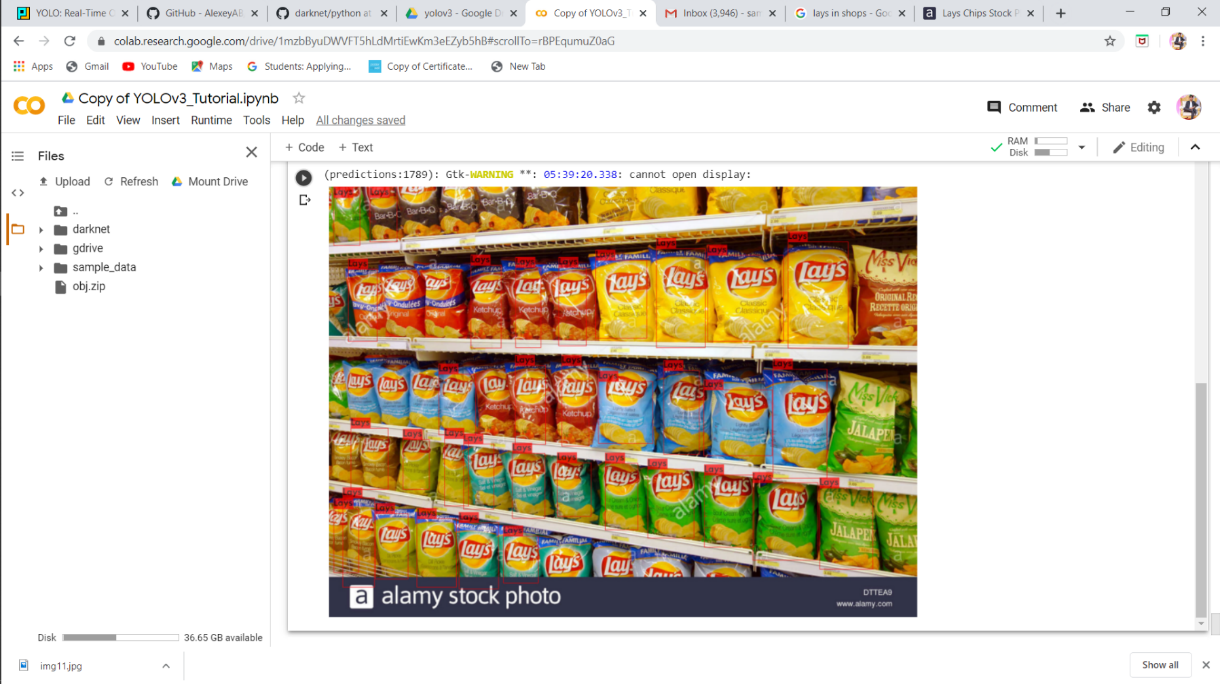
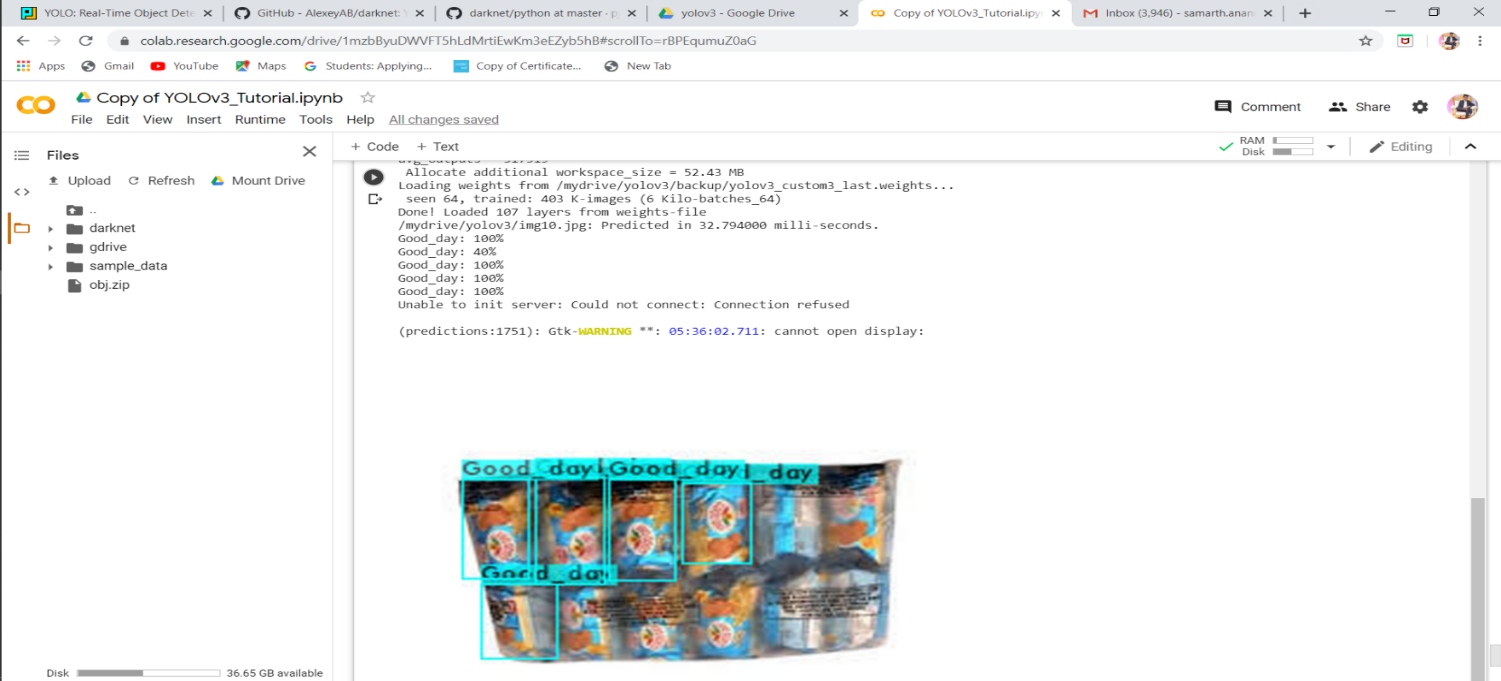


**Training:**

The training was done using Google Colab as it provides free GPU for faster and efficient training of the network. After pre-processing datasets i.e. generating label file for each image, both images and their respective label files need to be kept together. The cfg file was used for training configurations which include three yolo layers. As for higher accuracy, each object is to be trained for at least 2000 iterations. So, the dataset was trained for 6000 iterations as 3(total classes) \* 2000 = 6000. The values of batch and subdivisions needs to set 64 and 8 respectively for optimal training speed of the model. The height and width values were set at 416 each for optimum speed and better accuracy of custom object detection model. The number of filters used in the convolution layer were set to 24 as the value depends on total number of classes as, filters = (classes + 5)\*3. The total amount of time required to train the network with the above configurations was more than 14+ hours. The weights file thus generated after 6000 iterations were used to carry out object detections and analysing the performance of the model.

Some Sample Images testing during our project:





**Reference:**

[1] Official\_YOLO\_website (<https://pjreddie.com/darknet/yolo/>)

[2] Andrew Ng’s YOLO explanation (<https://www.youtube.com/watch?v=9s_FpMpdYW8>)

[3] Joseph Redmon, Ali Farhadi Paper (<https://pjreddie.com/media/files/papers/YOLOv3.pdf>)

[4] Towards Data Science Article “What’s new in YOLOv3?”

(<https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b>)