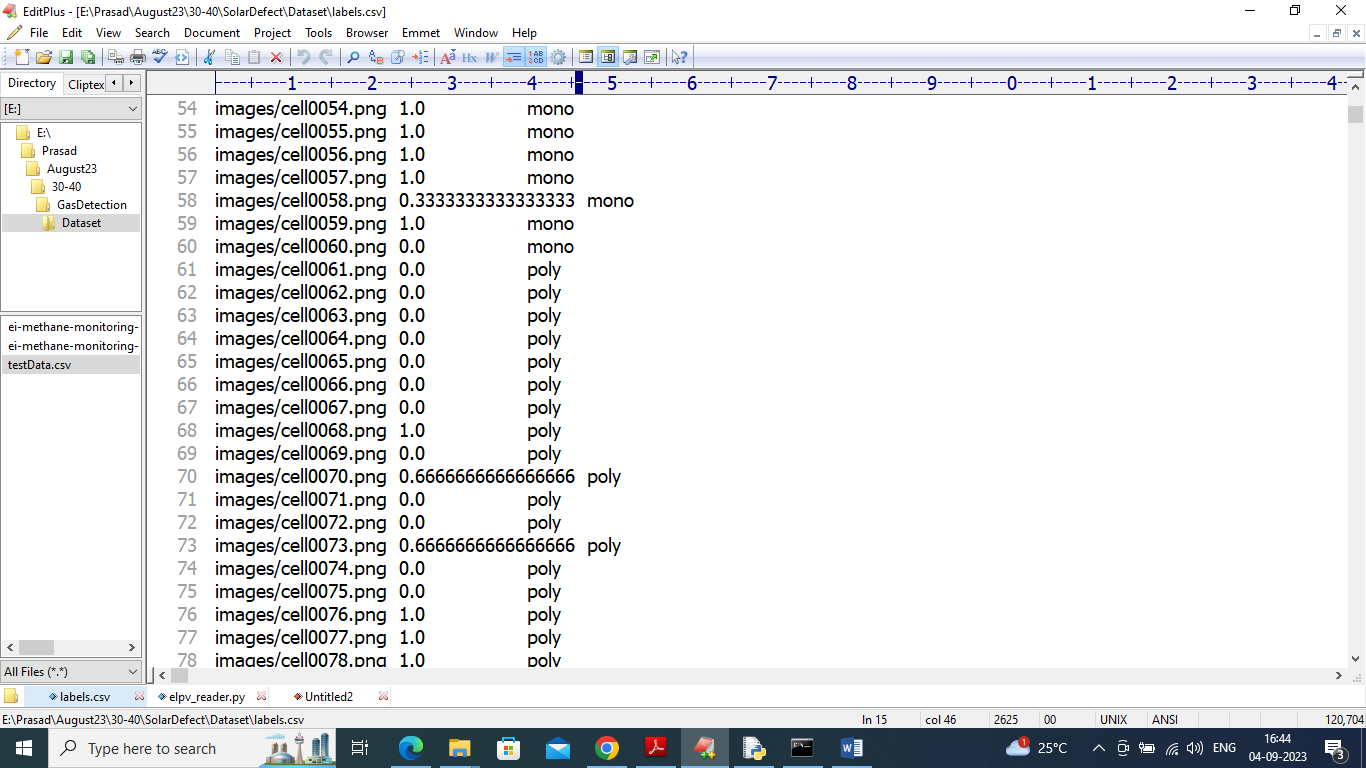
Solar Cell Surface Defect Detection Based on Optimized YOLOv5

Solar Cell devices are used to convert sun’s light into electricity and before deployment it must be check for defect and there are many existing techniques are available but its detection accuracy is very less. So author of this paper employing modified YoloV5 algorithm for solar cell defect detection. In propose work author has processed images using 4 different enhancements techniques such as Mosaic (align all features of images equally), Mix-up (mix up image features with its labels for better prediction), HSV transform (changing colour from RGB to HSV), scaling (resizing all images equally) and Flipping (flip all parts of images in all directions). All this enhancement will make image clearer for better detection.

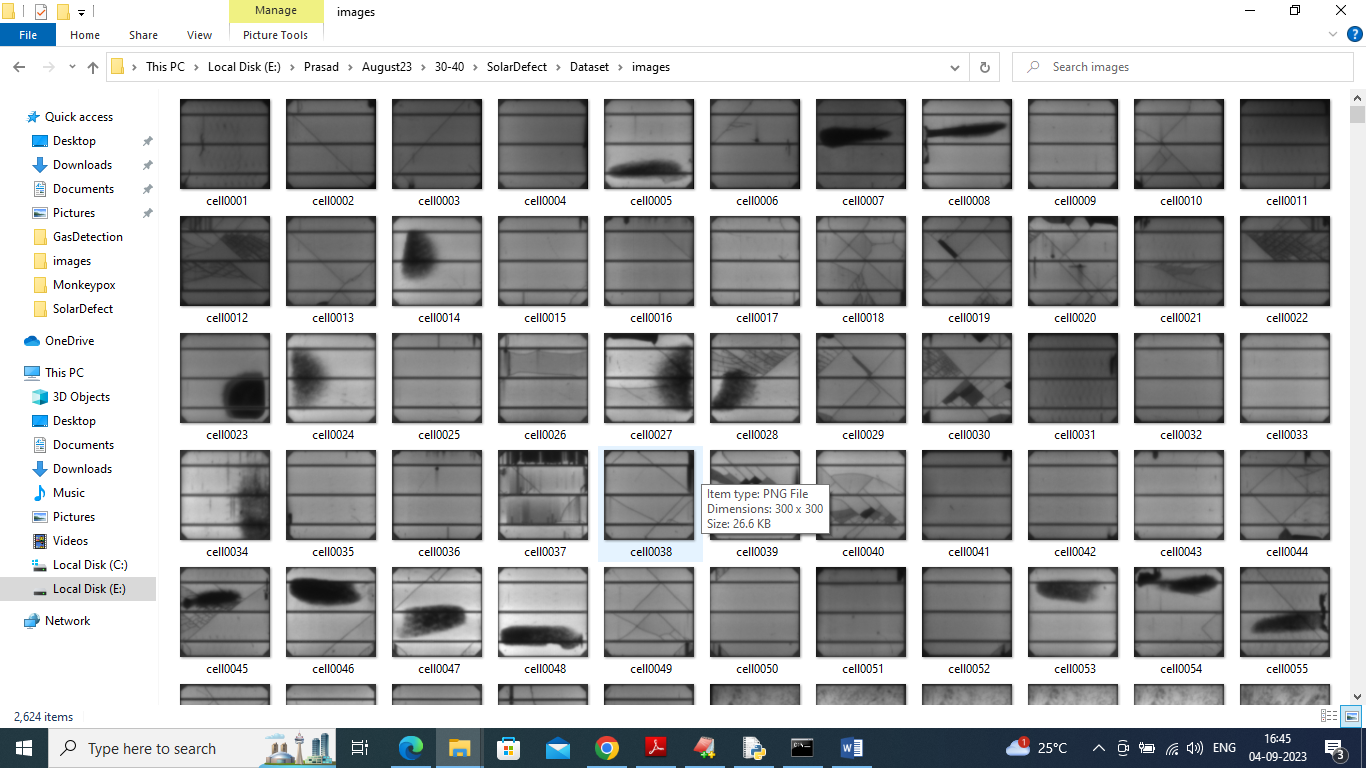
After all enhancement Yolov5 will get modified with CA attention (Channel attention model). CA attention mechanism is introduced to improve the feature extraction ability of the model; to address the problems of different target defect classification and localization concerns, the detection head in the original model is replaced with a decoupling head, which significantly improve the detection accuracy of the model without affecting the convergence speed of the model.

Modifying Yolov5 with CA and image enhancement make it improve prediction accuracy between 91 to 97%. To train propose algorithm author has used ELPV and PVEL-AD dataset where PVEL-AD dataset contains 12 different cracks or defects with bounding boxes but this dataset is not freely available to download. So we have sued ELPV dataset which contains more than 2000 images with defects as MONO and POLY. ELPV dataset does not contains any bounding boxes but it has defect probability percentage in place of bounding boxes. So by using this dataset we can classify defect as MONO or POLY and can detect defect with probability %.

In below screen we are showing dataset images with annotation and labels



In above annotated dataset we have 3 values such as Image name, defect probability and defect class label as MONO or POLY and each image we can see inside IMAGES folder like below screen



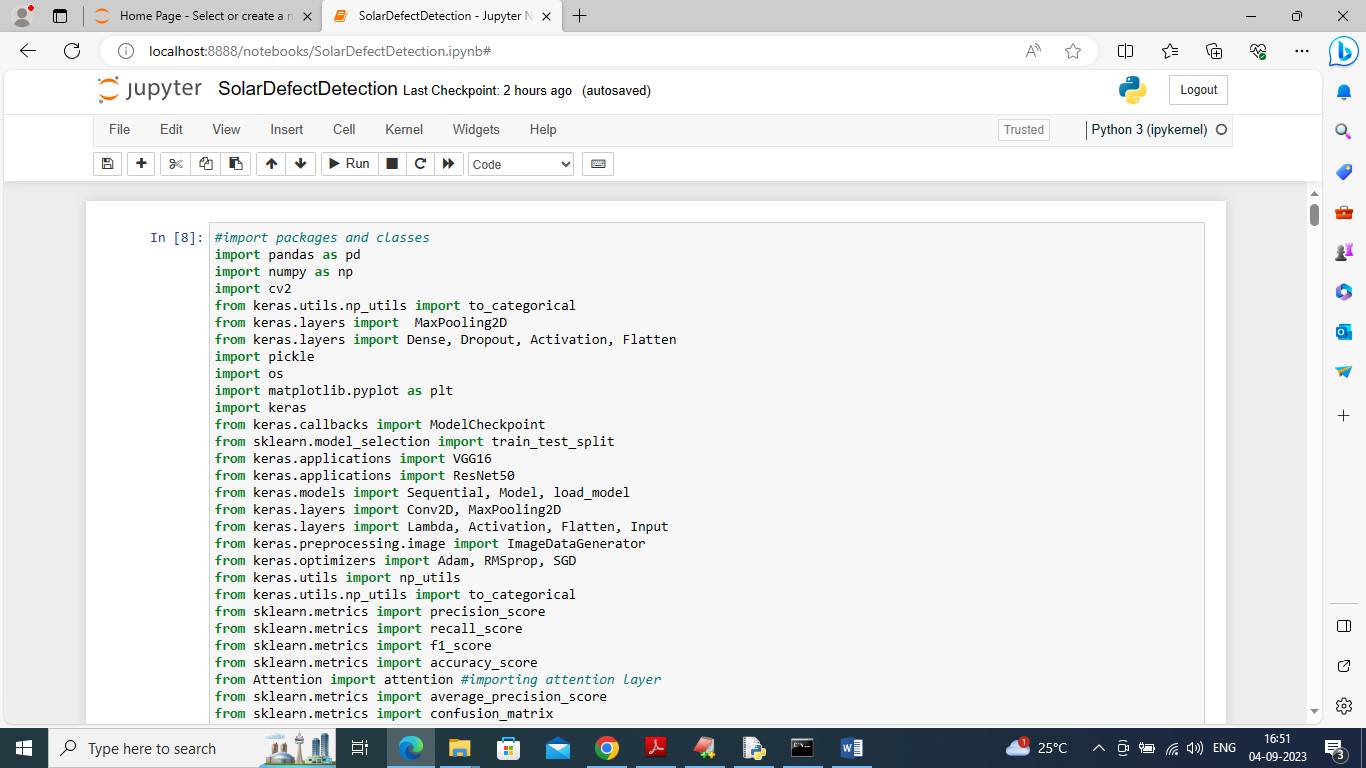
Above are the solar cell images can be used to train and test propose algorithm and this propose Optimized YoloV5 is compared with existing Faster RCNN algorithm. Each algorithm performance is evaluated in terms of accuracy, precision, recall, FSCORE, Confusion Matrix and MAP (mean absolute precision).

Extension Concept

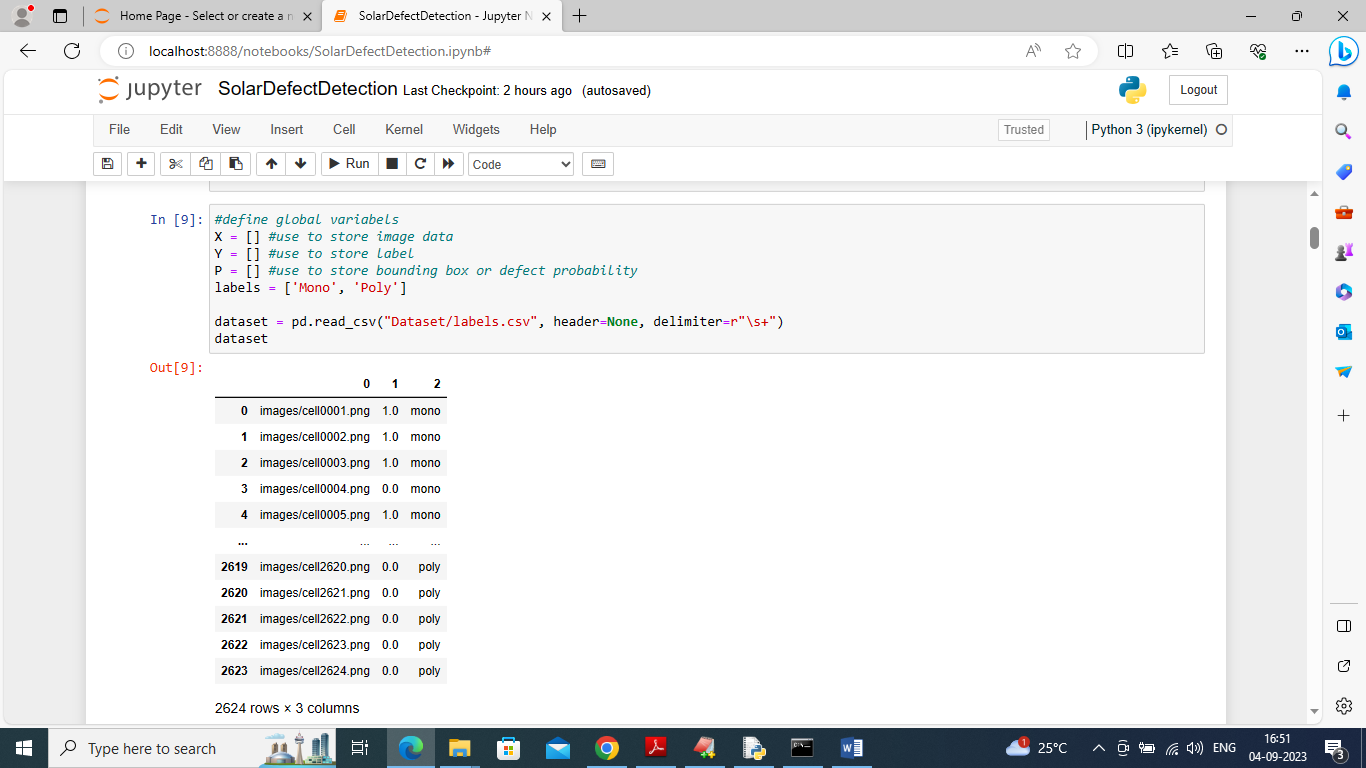
In propose work author has modified YoloV5 but not experimented with advance YOLO family other version as like Yolo6, 7 or 8. So as extension we have trained same dataset with Yolov6 and its giving better prediction accuracy compare to Optimized Yolov5.

SCREEN SHOTS

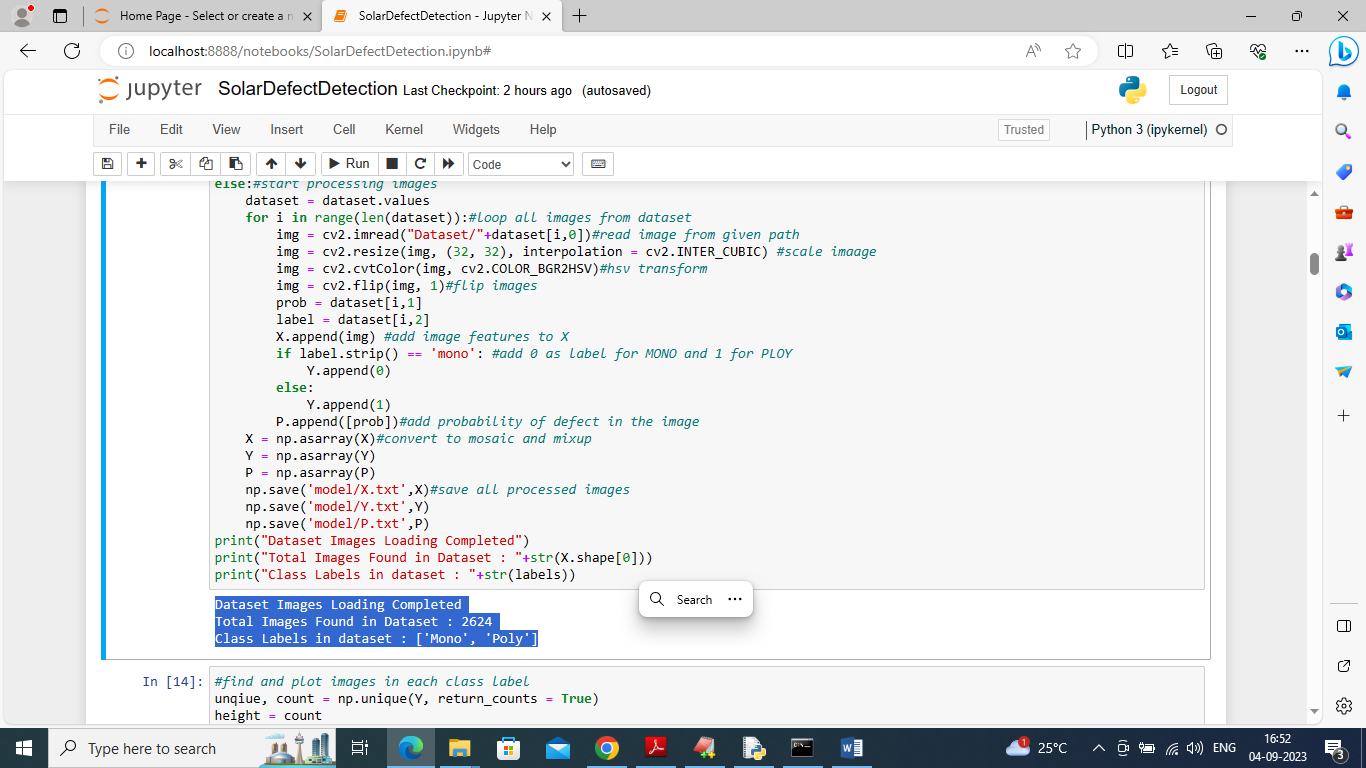
We have coded this project using JUPYTER NOTEBOOK and below are the code and output screens with blue colour comments



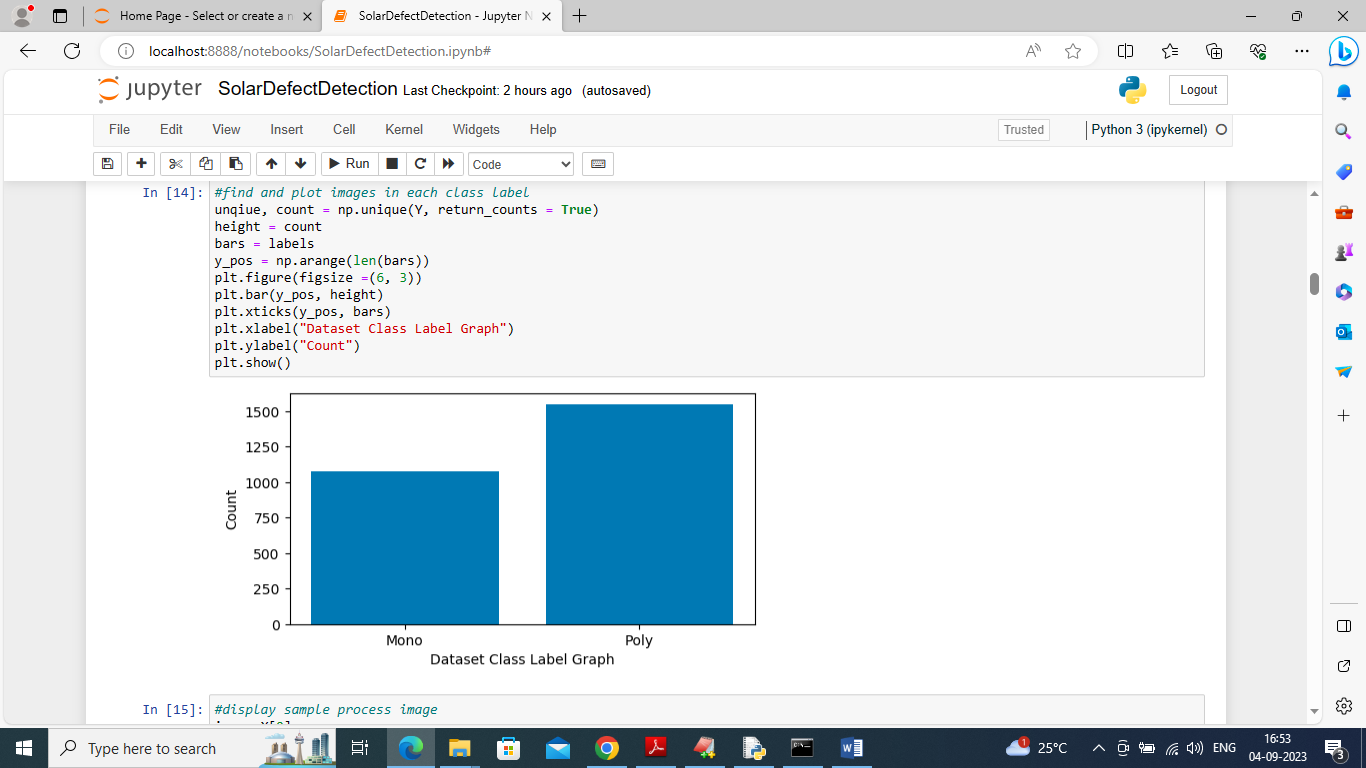
In above screen loading required python classes and packages



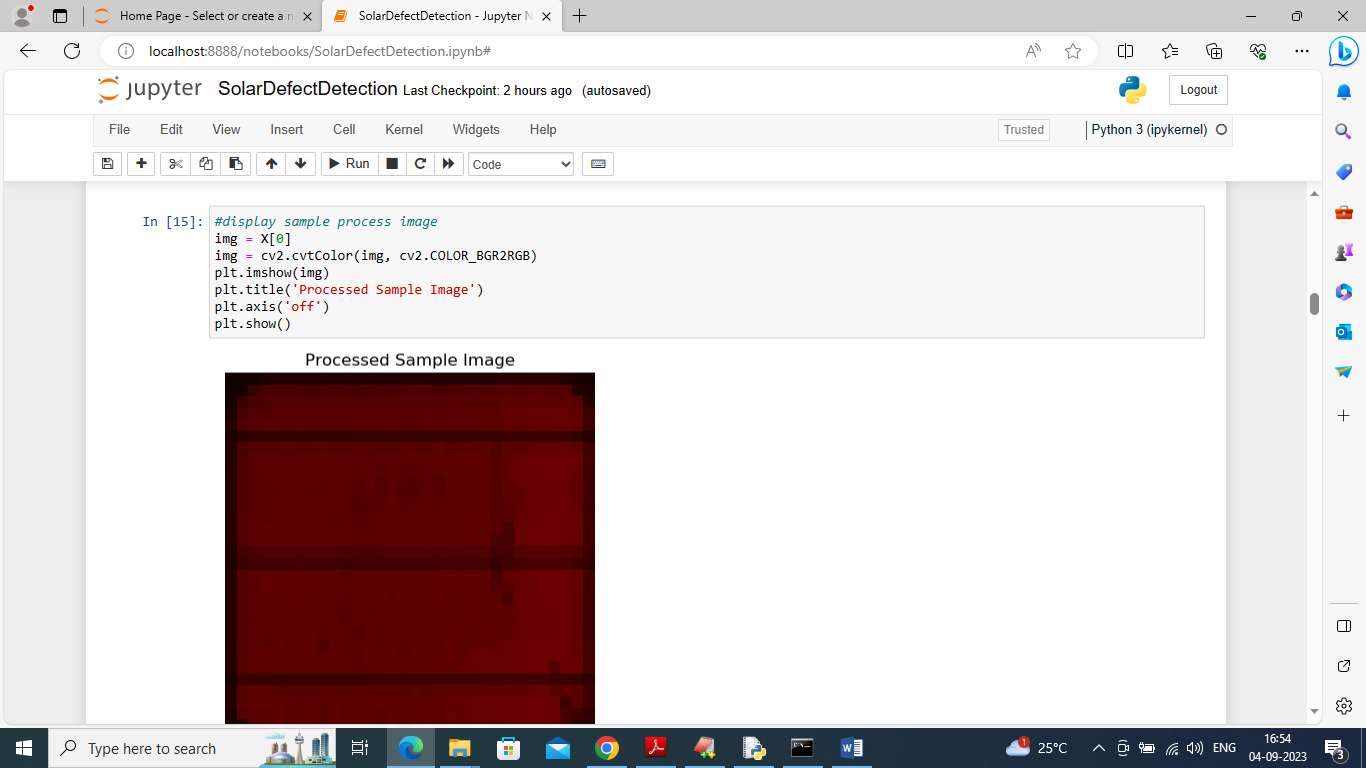
In above screen loading dataset annotation values with image name, defect probability and class labels as defect MONO or POLY



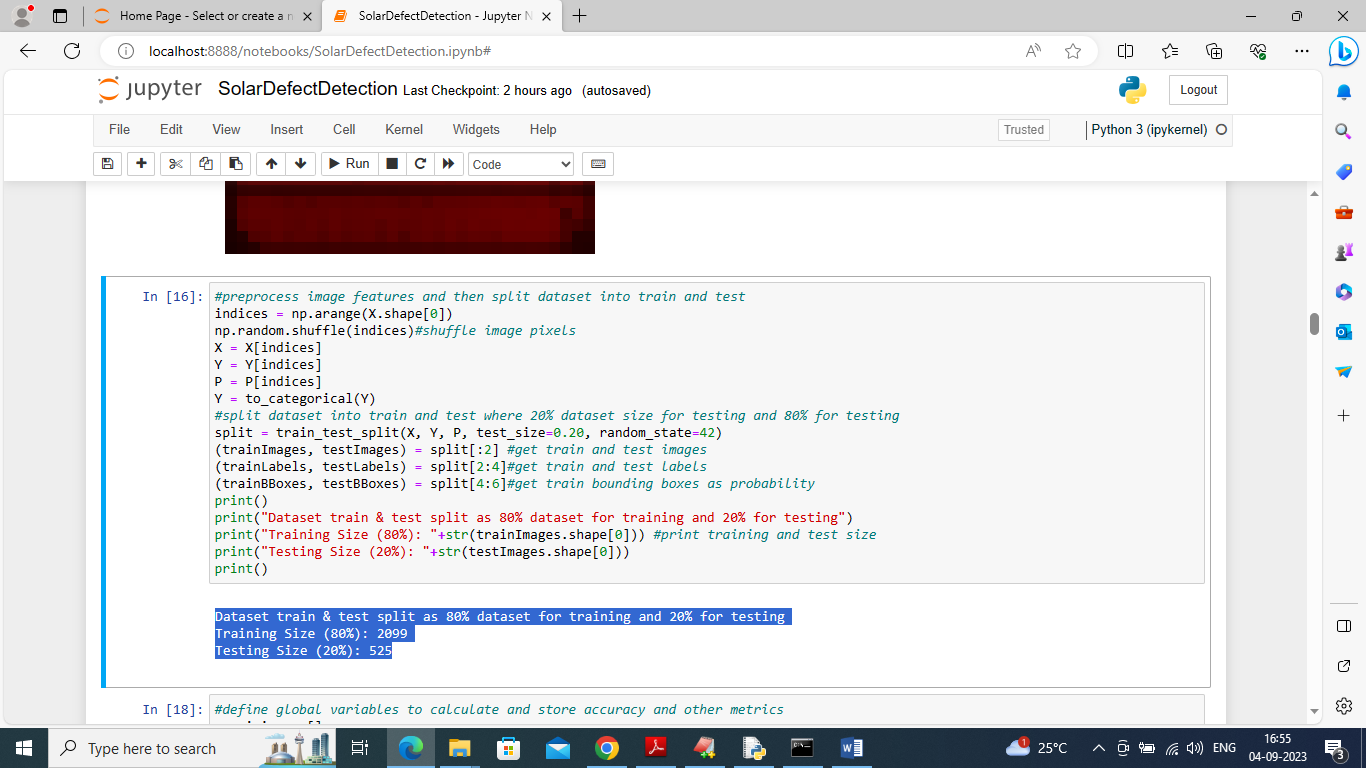
In above screen looping all values from annotation dataset and then reading each image and then applying scaling, HSV conversion, flipping and other image enhancing techniques



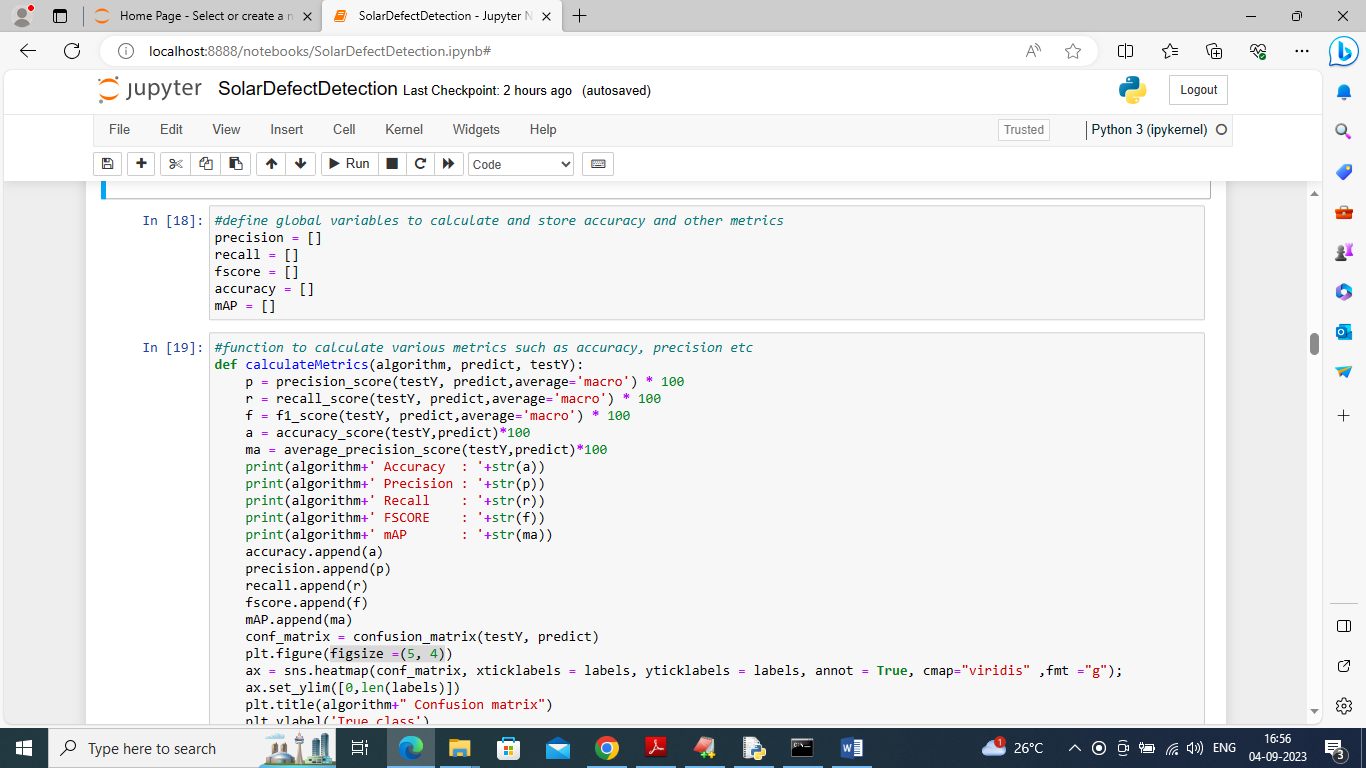
In above screen plotting graph of different defects found in dataset where x-axis represents defect NAME and y-axis represents counts



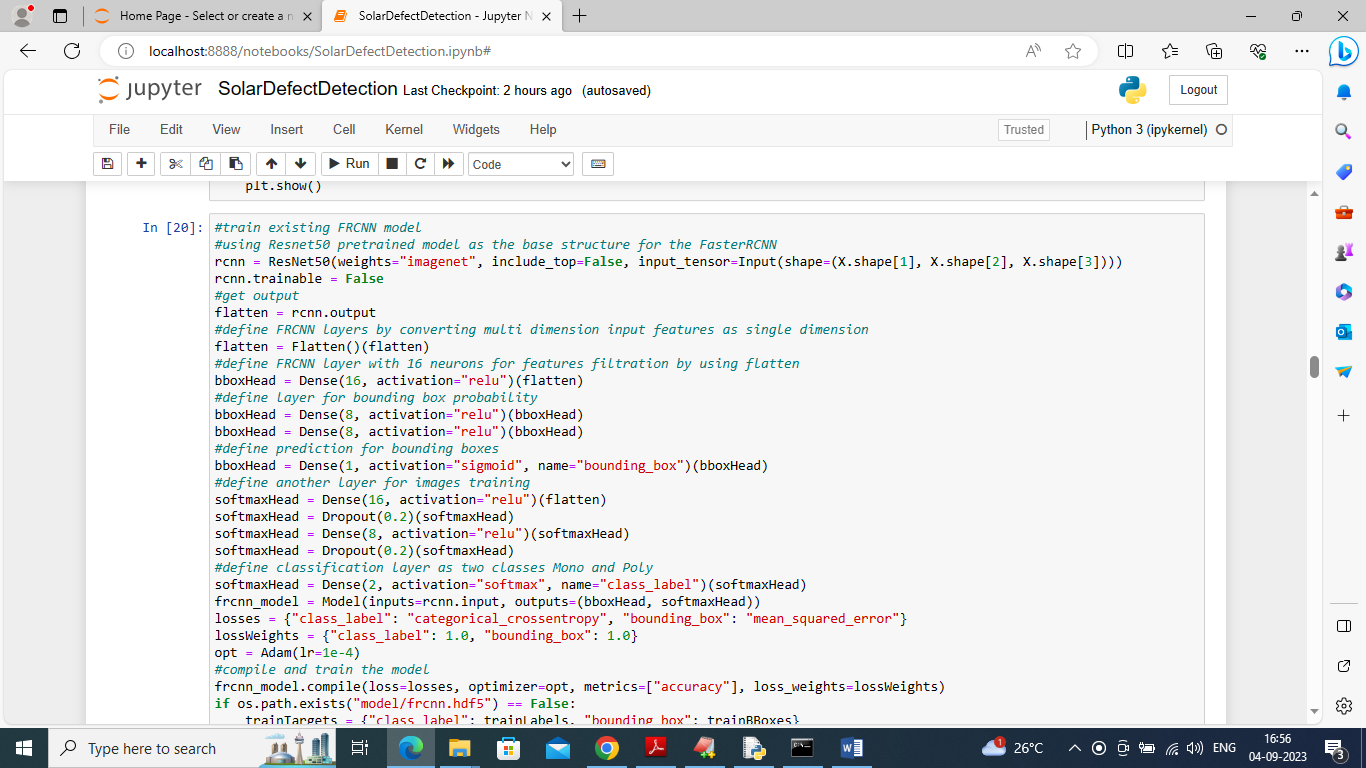
In above screen displaying processed HSV, Flip and scaled image in HSV format



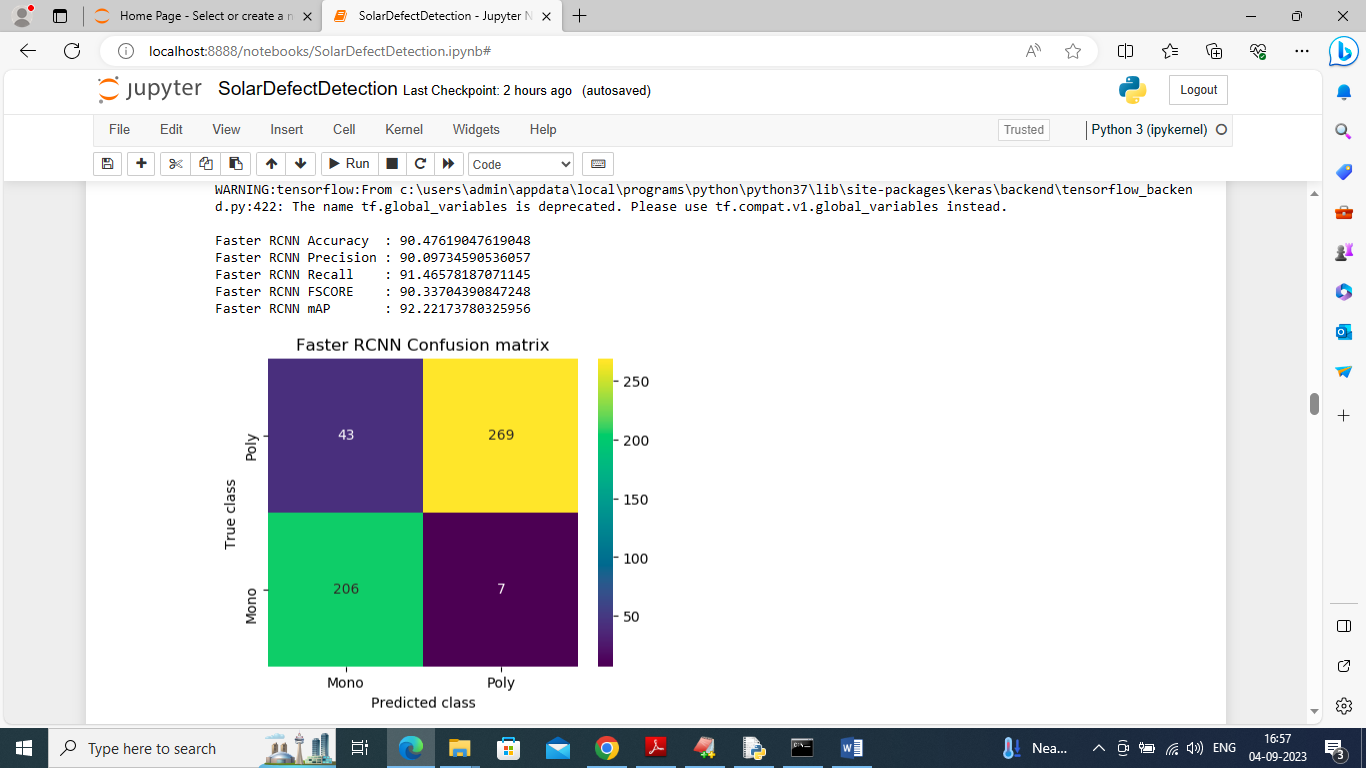
In above screen shuffling and splitting dataset into train and test where 80% dataset for training and 20 for testing and for training we are using image features, class labels and defect probability



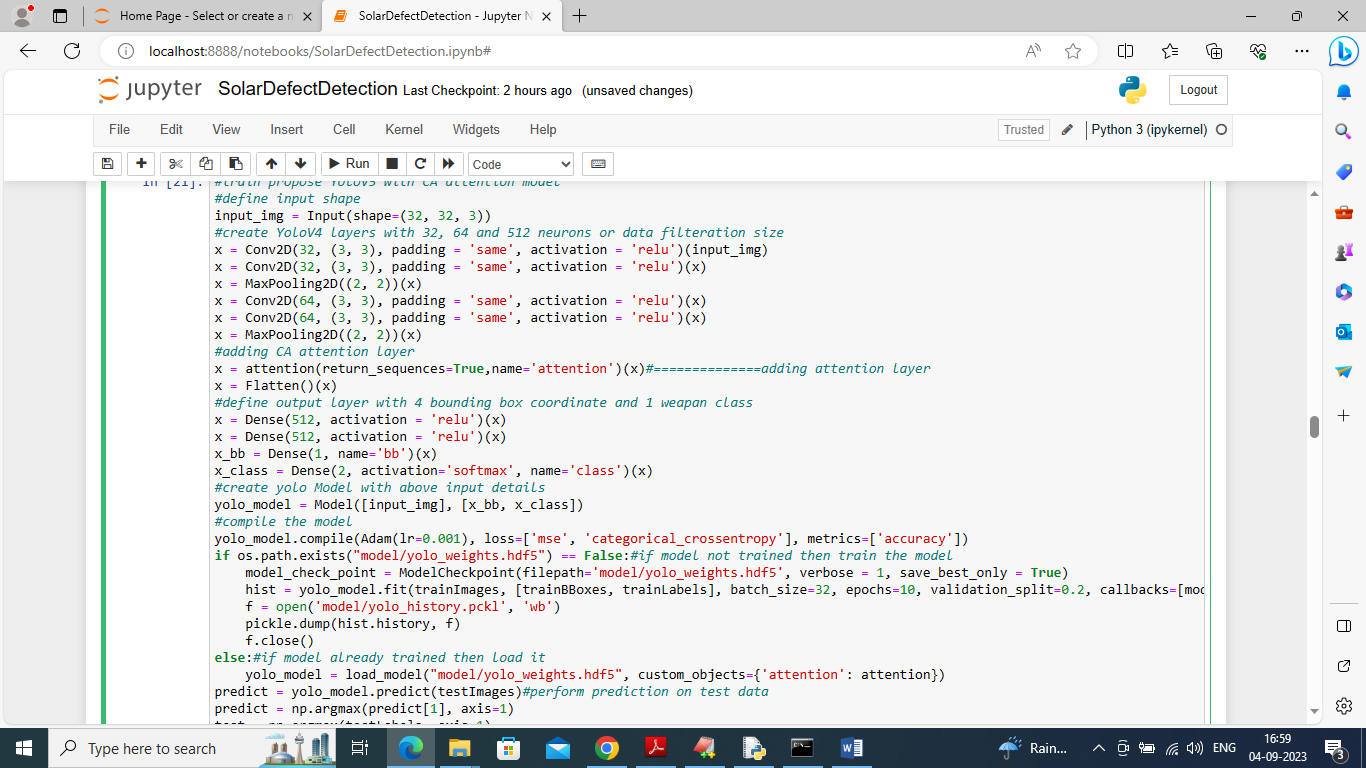
In above screen defining function to calculate accuracy and other metrics



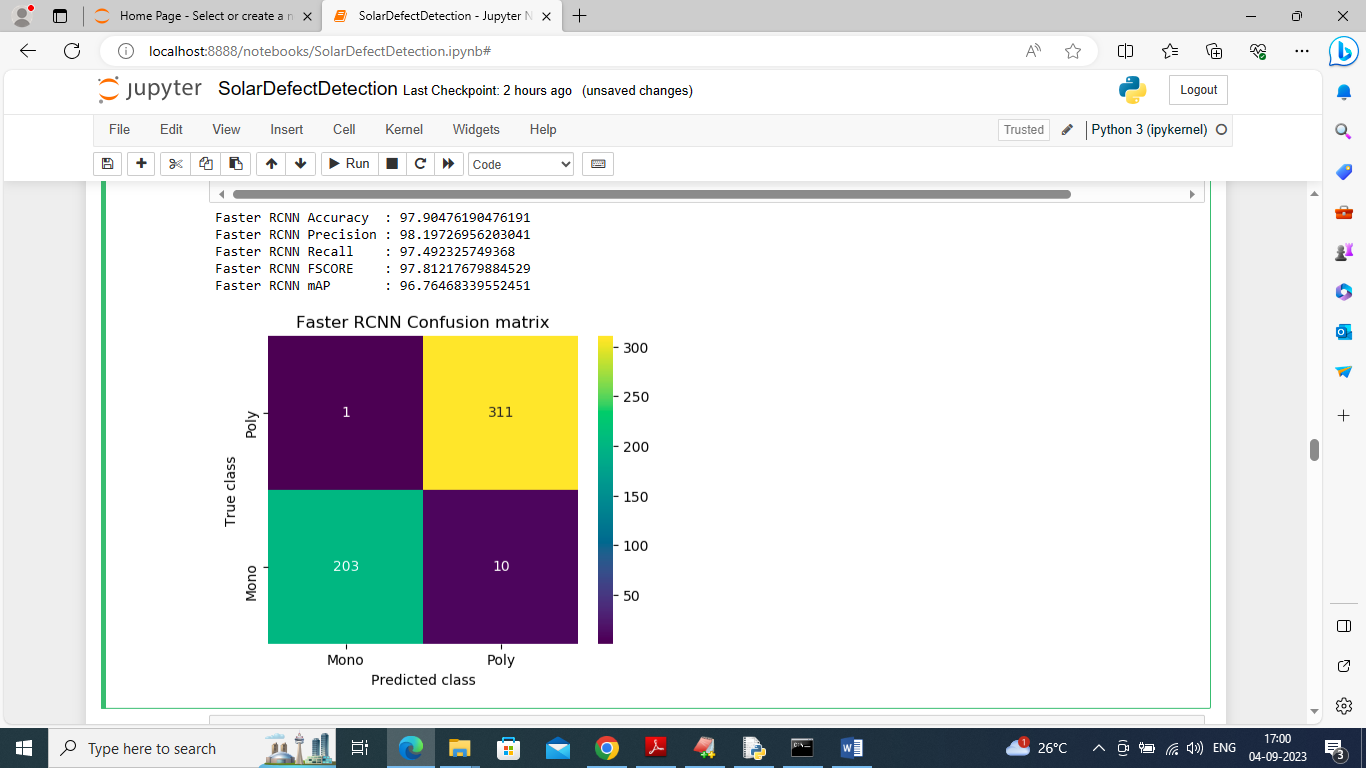
In above screen training existing FRCNN algorithm and after executing this block will get below output



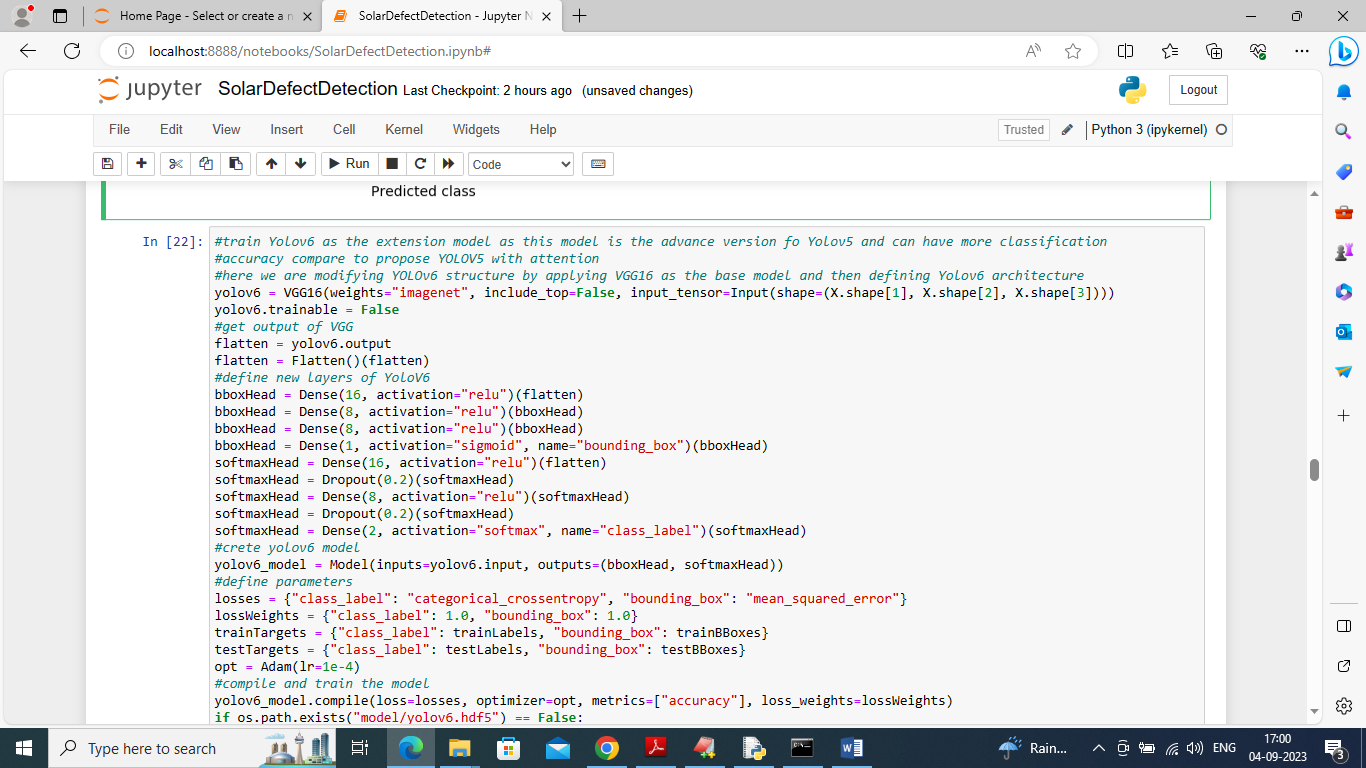
In above screen Faster RCNN got 90% accuracy and can see other metrics like precision, recall, FSCORE and MAP values. In confusion matrix graph x-axis represents Predicted Labels and y-axis represents True Labels where yellow and green boxes contains correct prediction count and blue boxes contains incorrect prediction count



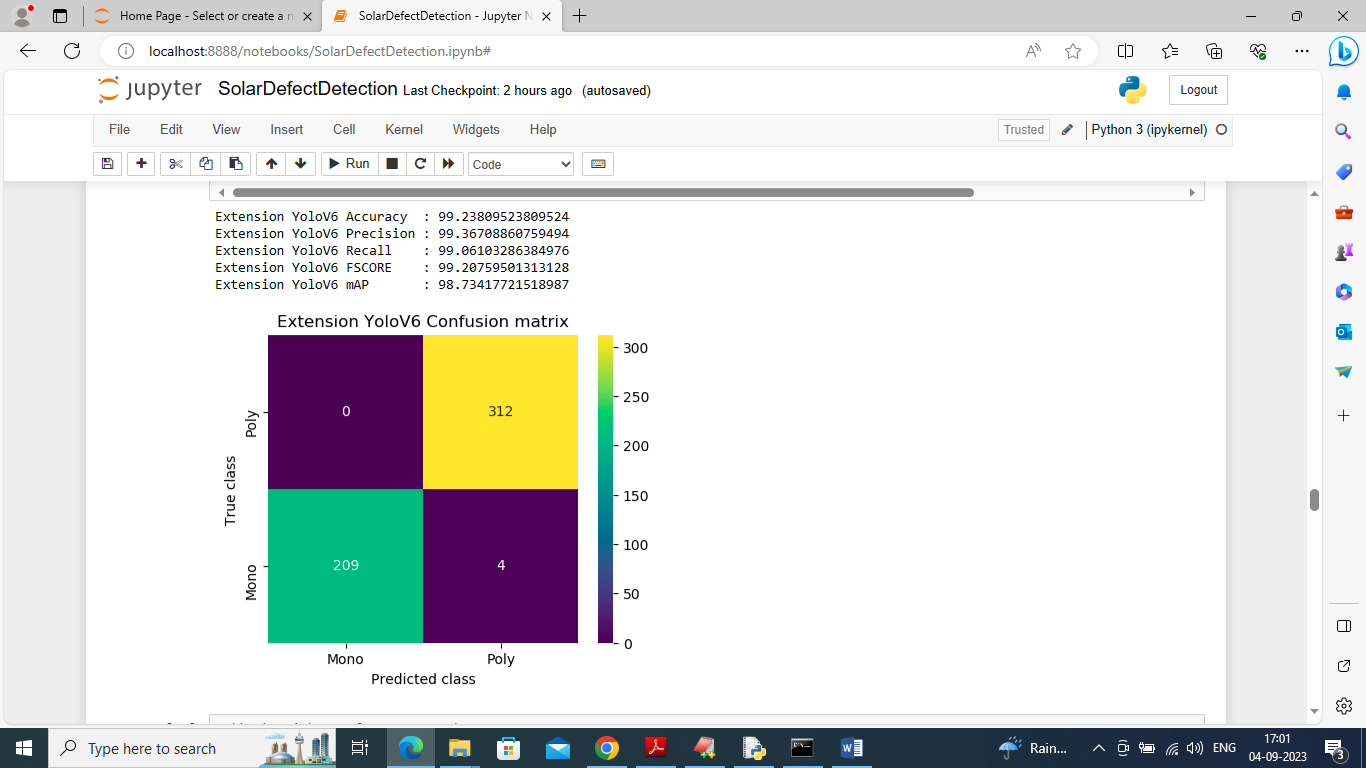
In above screen defining propose optimized YOLOv5 with CA attention layer and after executing above block will get below output



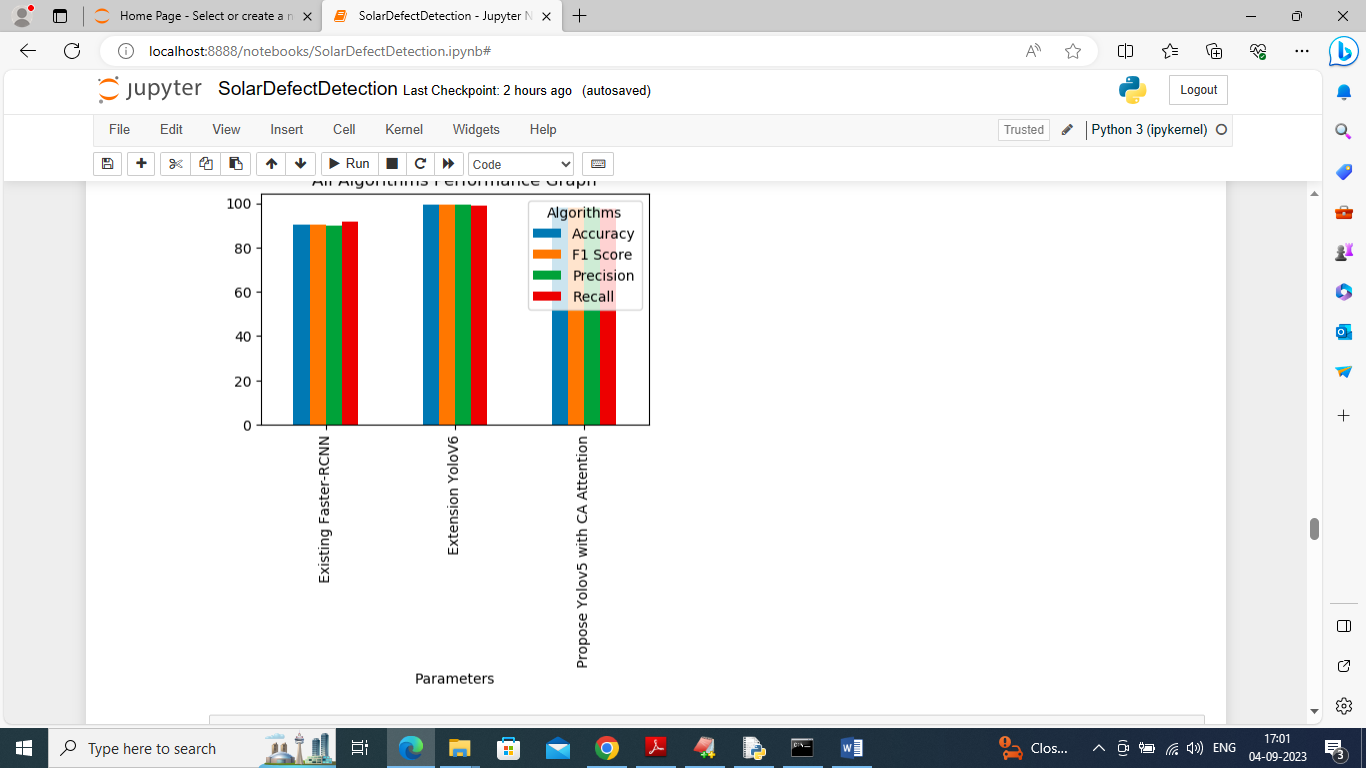
In above screen propose YoloV5 got 97% accuracy



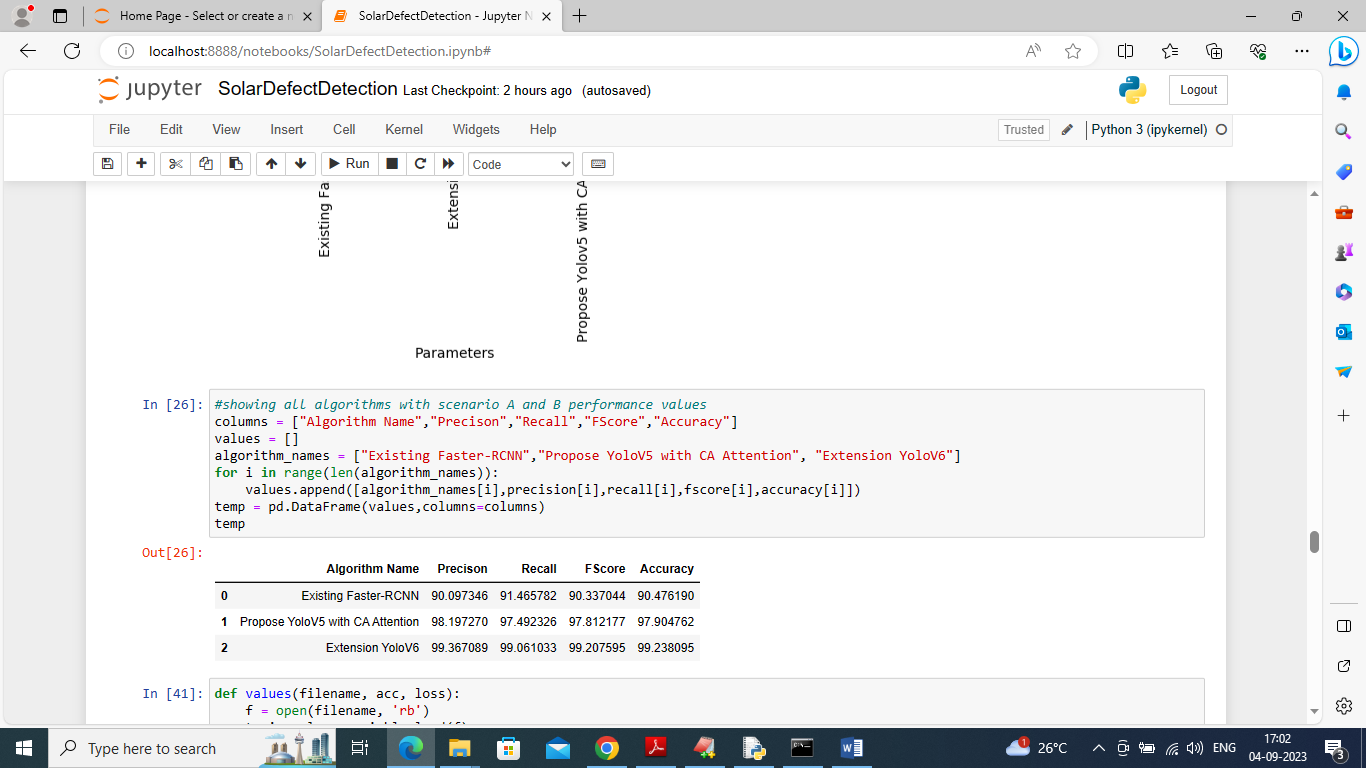
In above screen training extension Yolov6 algorithm and after execution will get below output



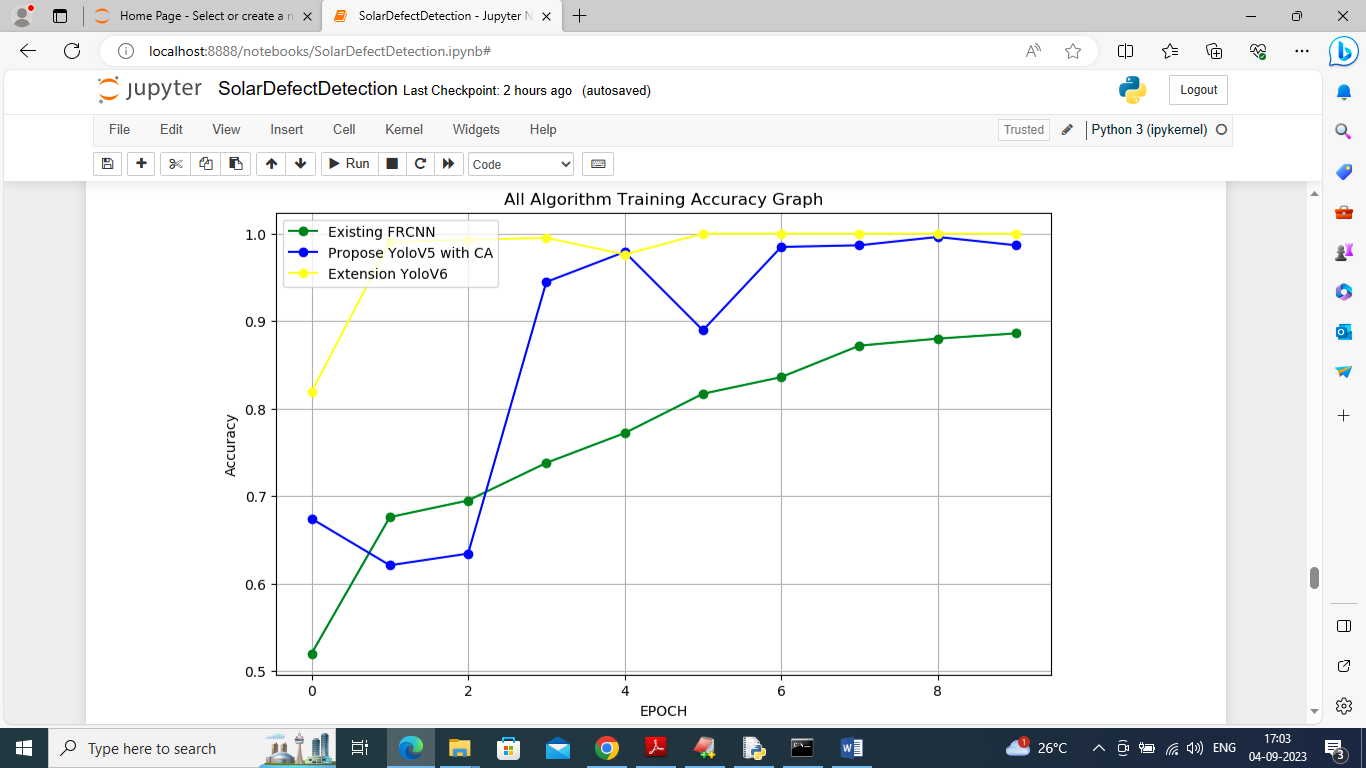
In above screen extension got 99% accuracy



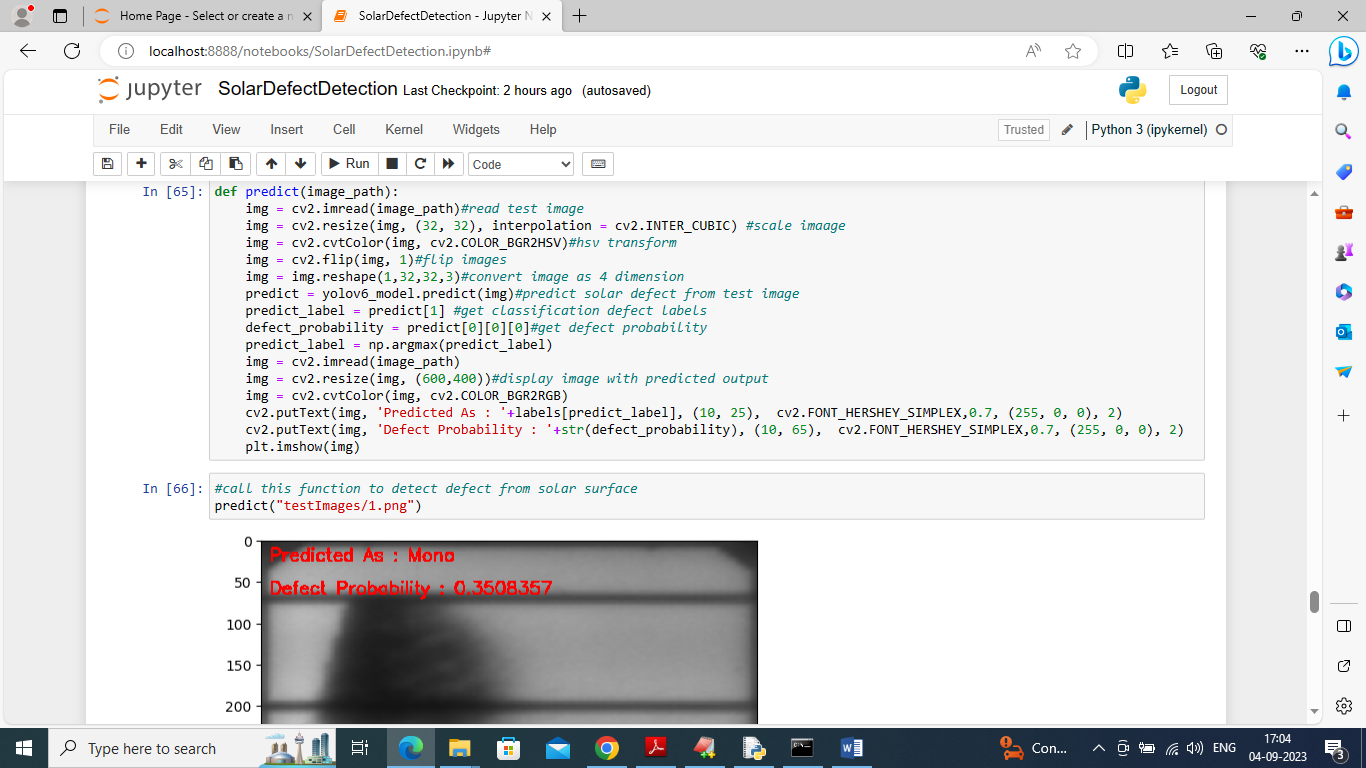
In above screen displaying performance of all algorithms where x-axis represents algorithm names and y-axis represents accuracy and other metrics in different colour bars and in all algorithms extension got high accuracy



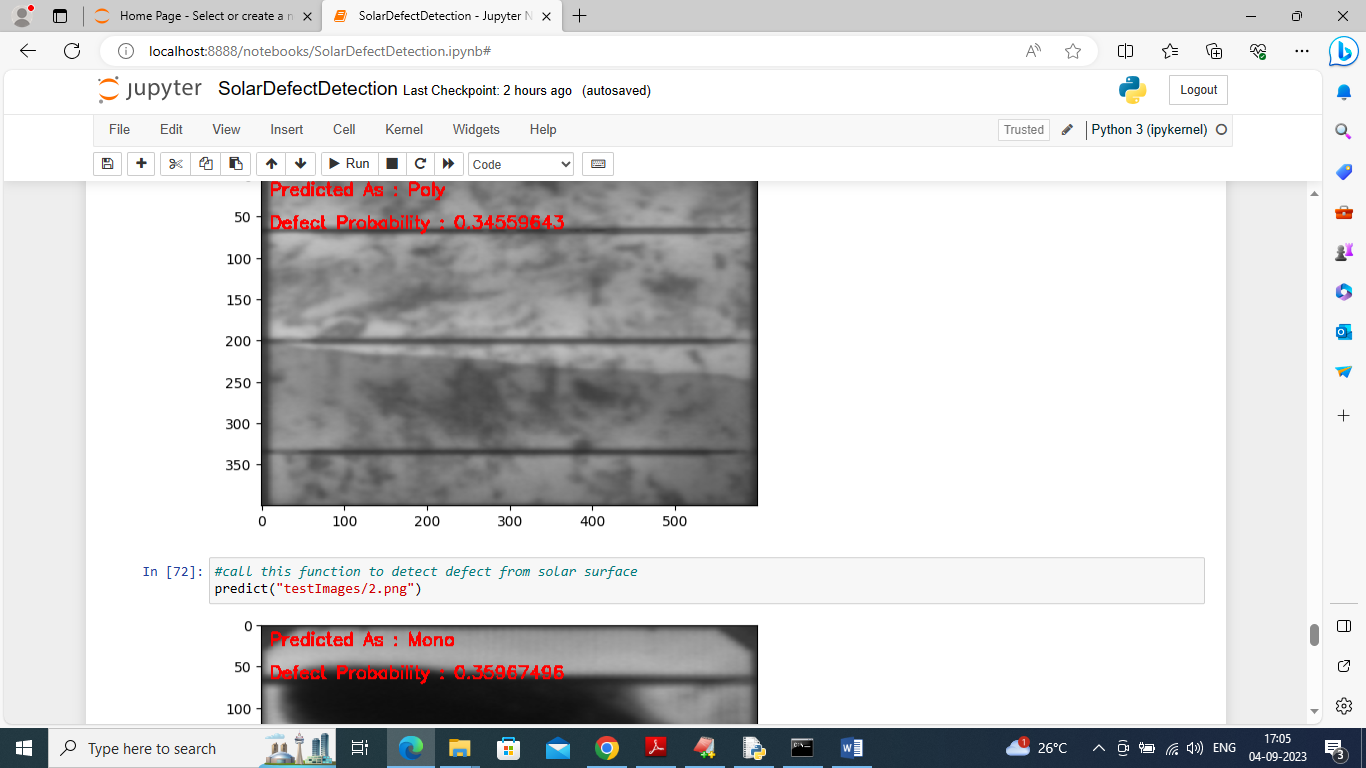
In above screen displaying all algorithm performance in tabular format



In above training graph x-axis represents training epoch and y-axis represents training accuracy where yellow line is for extension Yolov6, blue line for propose Yolov5 and green line for existing Faster RCNN



In above screen defining prediction function to predict defect using extension model as Yolov6 and in image in red colour text we can see defect name as ‘MONO and defect probability as0.35%’



In above screen can see defect type and detected probability for other images