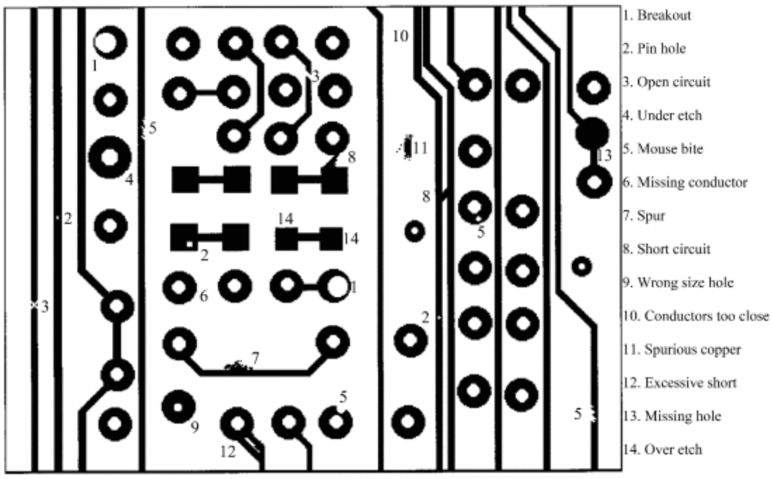
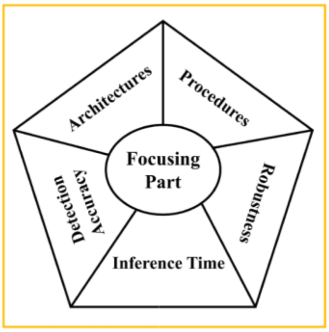
PCB DEFECT DETECTION USING VARIOUS MACHINE LEARNING MODELS

INTRODUCTION:

- In PCB manufacturing, the process can be complex and cumbersome, leading to various errors, so detection of PCB becomes an essential part of the production process.
- There are many types of defects in PCBs that occur inevitably due to mishandling or technical faults during the manufacturing process. The following picture shows different kinds of defects in PCBs.
- Six common PCB defects are included in this dataset: missing holes, mouse bites, open circuits, shorts, spurious copper, and spurs.



- We used machine learning algorithms to reduce the defects in PCBs using a custom dataset of PCB defect errors that we have used in our paper.
- To avoid model overfitting, the dataset is expanded in this study. PCB pictures are randomly clipped, turned horizontally, turned vertically, a brightness adjustment, and noise adjustment are used to expand the original dataset to 3000 pieces.
- We focused on architectures, procedures, detection accuracy, inference time, and robustness of detection algorithms in the paper.



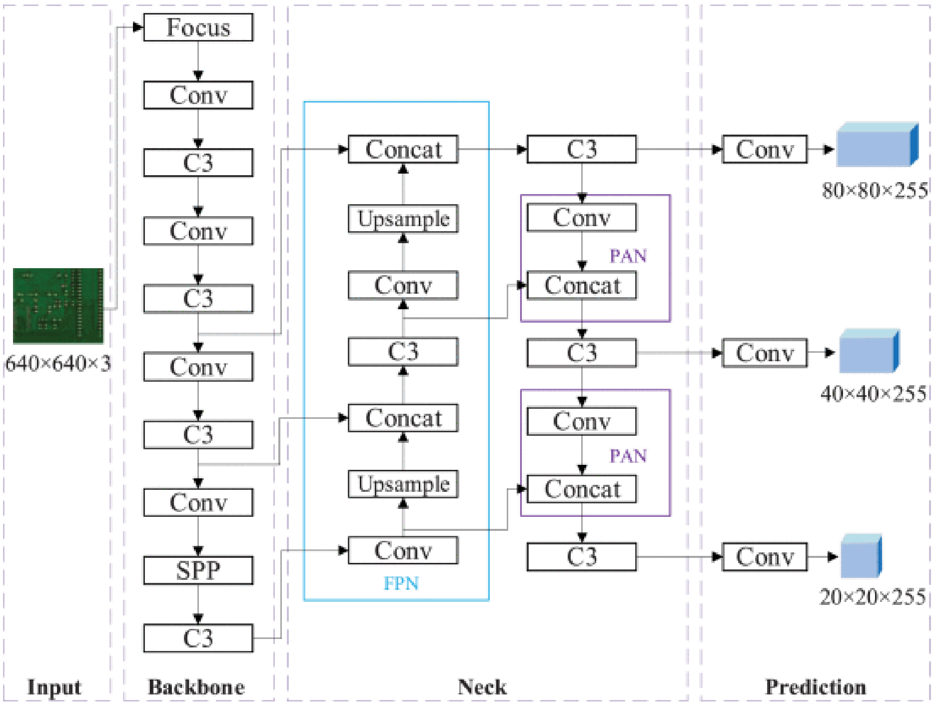
The Machine Learning Algorithms we used are yolov5s, yolov6s, yolov7, yolov8s

WORKING:

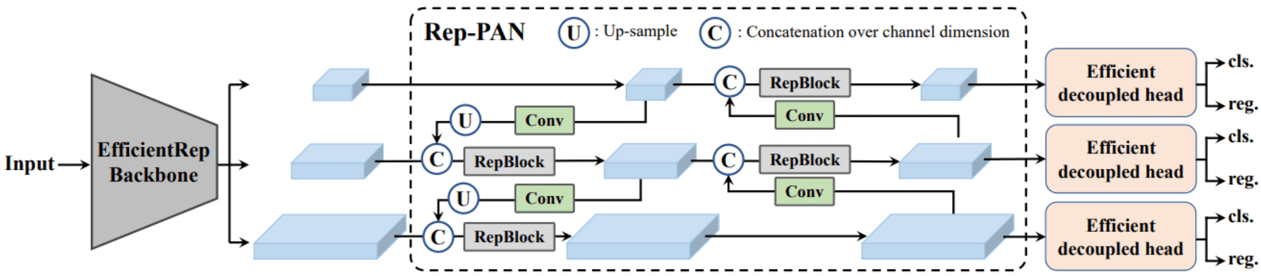
1. Collected a dataset of images. The dataset has a variety of PCBs, both with and without defects. The images are taken in a variety of lighting conditions and from different angles.
2. Labelled the images. Each image in the dataset is labelled with the type and location of any defects.
3. Training the YOLOv5 /YOLOv6/YOLOv7/YOLOv8 model. The YOLOv5 model can be trained using the labelled images. The training process can take several hours or even days, depending on the size of the dataset and the hardware used.
4. Evaluated the model. Once the model is trained, it can be evaluated on a set of test images. The evaluation results will show how well the model is able to detect defects.
5. Deployed the model. Once the model is evaluated and found to be satisfactory, it can be deployed for use in production. The model can be deployed on a variety of devices, including computers, mobile devices, and embedded systems.

ARCHITECTURES:

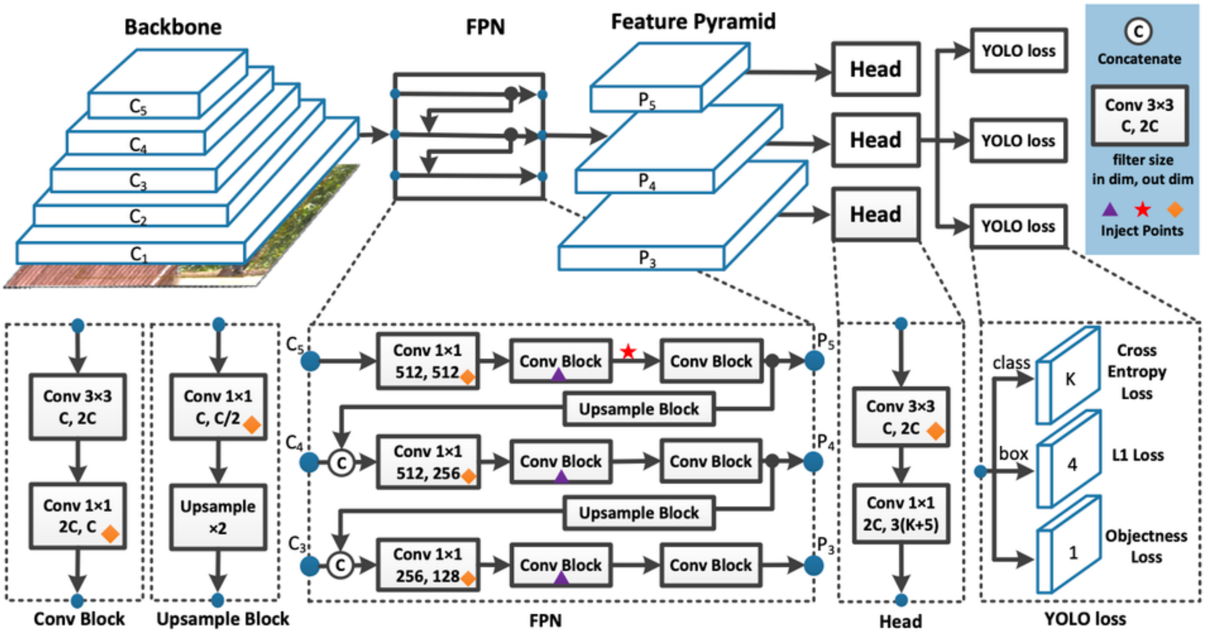
YOLOV5s: YOLOv5s uses CSPDarknet as the backbone for feature extraction from images1. It has 7.3 million parameters and achieves 37.3 mAP on the COCO validation dataset2. It can run at 140 FPS on a Tesla V100 GPU3.



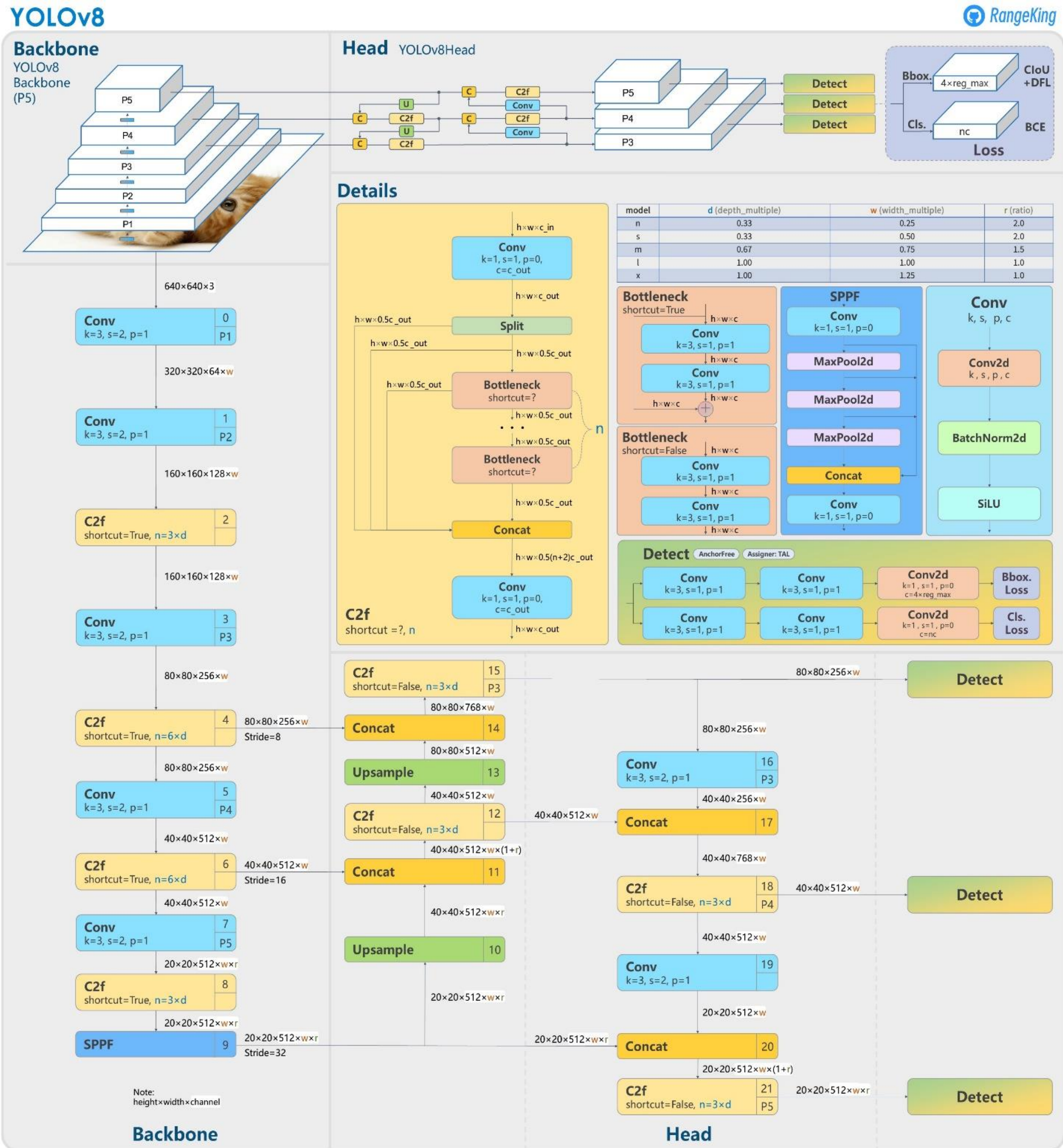
YOLOV6s: YOLOv6s uses CSPResNeXt50 as the backbone for feature extraction from images3. It has 43.2 million parameters and achieves 43.1 mAP on the COCO validation dataset2. It can run at 66 FPS on a Tesla V100 GPU3.



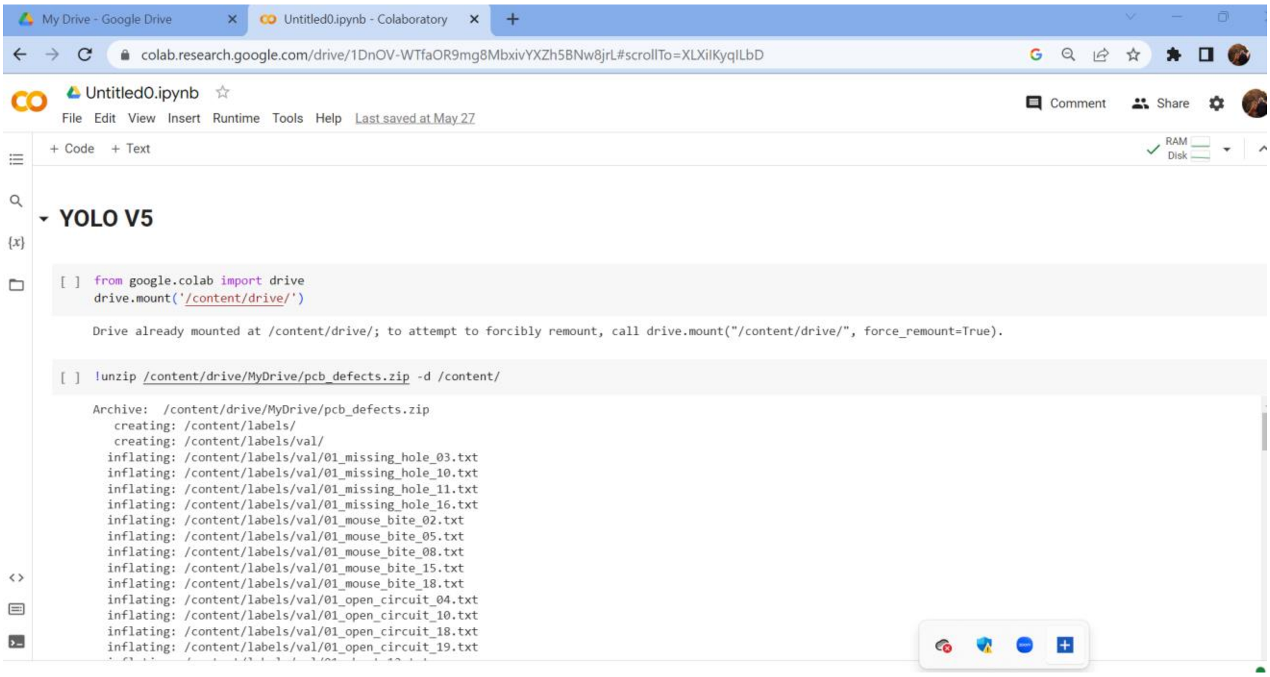
YOLOV7: YOLOv7 uses CSPResNeXt101 as the backbone for feature extraction from images4. It has 141 million parameters and achieves 56.8 mAP on the COCO validation dataset4. It can run at 56 FPS on a Tesla V100 GPU4.

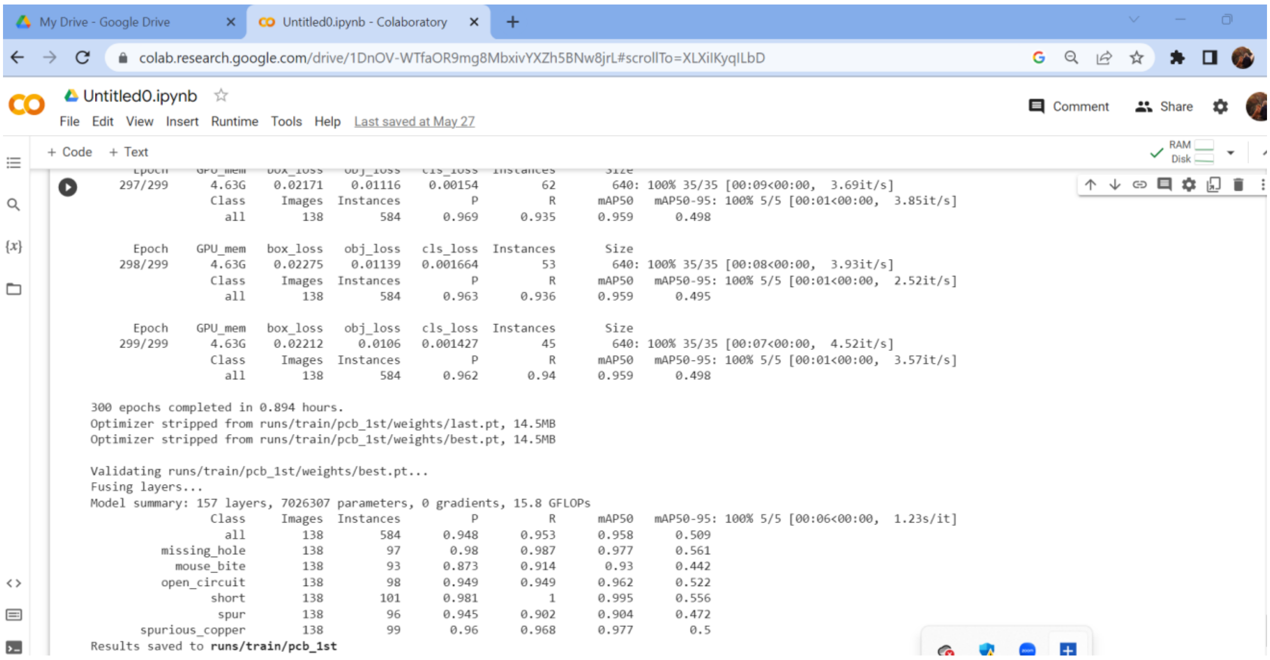


<u>YOLO8s:</u> YOLOv8s uses CSPResNeXt101 + SPP + PANet as the backbone for feature extraction from images5. It has 141 million parameters and achieves 58.4 map on the COCO validation dataset5. It can run at 230 FPS on a RTX 4090 GPU5.



IMPLEMENTATION:

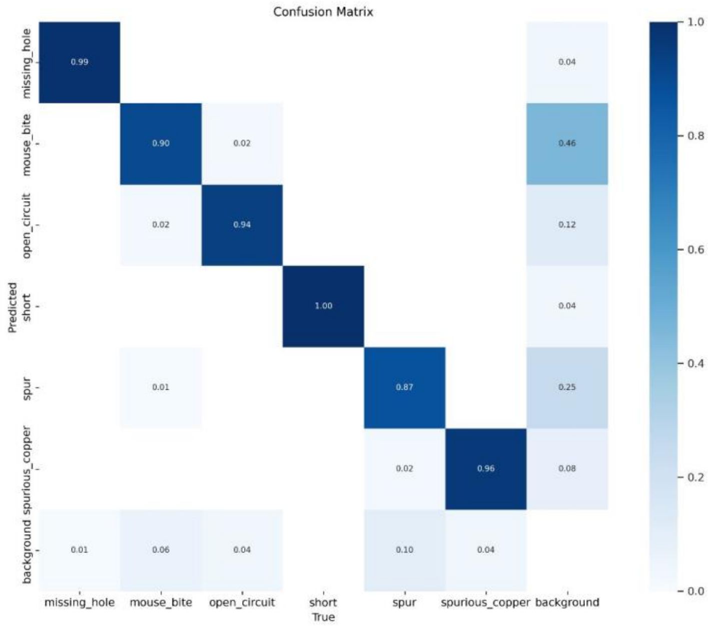





```
      Epoch    GPU_mem   box_loss   obj_loss   cls_loss  Instances       Size
    297/299      4.63G    0.02171    0.01116    0.00154         62        640: 100% 35/35 [00:09<00:00,  3.69it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 5/5 [00:01<00:00,  3.85it/s]
                   all        138        584      0.969      0.935      0.959      0.498

      Epoch    GPU_mem   box_loss   obj_loss   cls_loss  Instances       Size
    298/299      4.63G    0.02275    0.01139   0.001664         53        640: 100% 35/35 [00:08<00:00,  3.93it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 5/5 [00:01<00:00,  2.52it/s]
                   all        138        584      0.963      0.936      0.959      0.495

      Epoch    GPU_mem   box_loss   obj_loss   cls_loss  Instances       Size
    299/299      4.63G    0.02212     0.0106   0.001427         45        640: 100% 35/35 [00:07<00:00,  4.52it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 5/5 [00:01<00:00,  3.57it/s]
                   all        138        584      0.962       0.94      0.959      0.498

300 epochs completed in 0.894 hours.
Optimizer stripped from runs/train/pcb_1st/weights/last.pt, 14.5MB
Optimizer stripped from runs/train/pcb_1st/weights/best.pt, 14.5MB

Validating runs/train/pcb_1st/weights/best.pt...
Fusing layers...
Model summary: 157 layers, 7026307 parameters, 0 gradients, 15.8 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 5/5 [00:06<00:00,  1.23s/it]
                   all        138        584      0.948      0.953      0.958      0.509
          missing_hole        138         97       0.98      0.987      0.977      0.561
            mouse_bite        138         93      0.873      0.914       0.93      0.442
          open_circuit        138         98      0.949      0.949      0.962      0.522
                 short        138        101      0.981          1      0.995      0.556
                  spur        138         96      0.945      0.902      0.904      0.472
       spurious_copper        138         99       0.96      0.968      0.977        0.5
Results saved to runs/train/pcb_1st
```

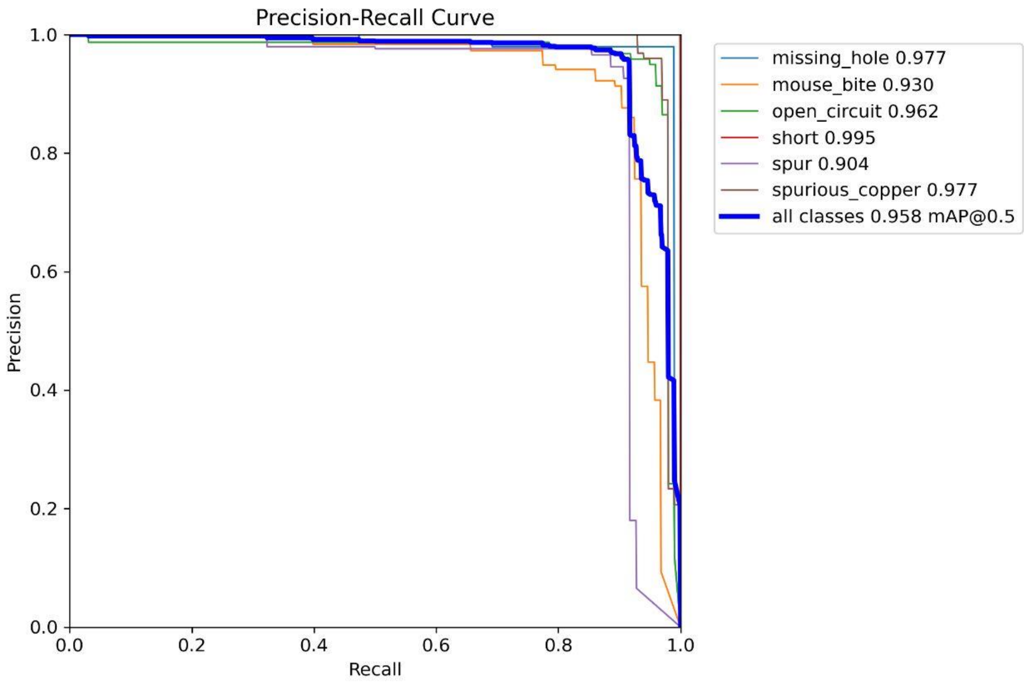
RESULTS:

YOLOV5:



Confusion Matrix:

Precision-Recall Curve:



---

yolov5s- 0.95 map@0.5

yolov7- 0.94 map@0.5

yolov8s- 0.94 map@0.5

Yolov6s- 0.93 map@0.5