

**<Project UID: 46>**

**<Traffic Sign Detection using YOLO>**

**<Annie D'Souza, Priyanshi Gupta>**

**-By Shrey Gupta (190100112)**

## Introduction to YOLO

YOLO is an abbreviation for the term ‘You Only Look Once’. This is an algorithm that detects and recognizes various objects in a picture (in real-time). Object detection in YOLO is done as a regression problem and provides the class probabilities of the detected images.

YOLO algorithm employs convolutional neural networks (CNN) to detect objects in real-time. As the name suggests, the algorithm requires only a single forward propagation through a neural network to detect objects. This means that prediction in the entire image is done in a single algorithm run. The CNN is used to predict various class probabilities and bounding boxes simultaneously.

## Why the YOLO algorithm is important

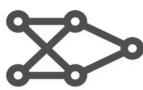
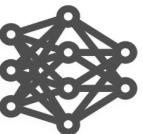
YOLO algorithm is important because of the following reasons:

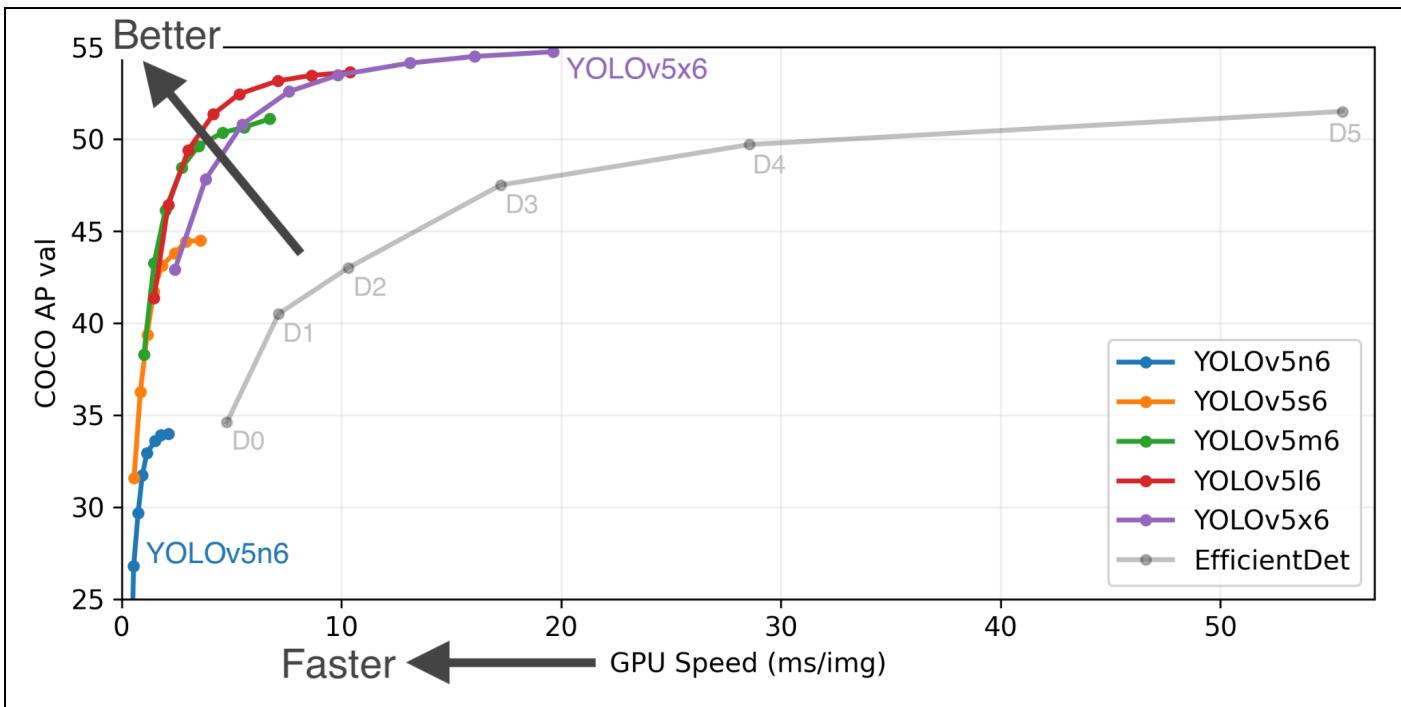
- **Speed:** This algorithm improves the speed of detection because it can predict objects in real-time.
- **High accuracy:** YOLO is a predictive technique that provides accurate results with minimal background errors.
- **Learning capabilities:** The algorithm has excellent learning capabilities that enable it to learn the representations of objects and apply them in object detection.

## YOLOv5

YOLOv5 is a family of compound-scaled object detection models trained on the COCO dataset, and includes simple functionality for Test Time Augmentation (TTA), model ensembling, hyperparameter evolution, and export to ONNX, CoreML and TFLite.

## Different Models in YOLOv5

				
Nano	Small	Medium	Large	XLarge
YOLOv5n	YOLOv5s	YOLOv5m	YOLOv5l	YOLOv5x
4 MB <sub>FP16</sub> 6.3 ms <sub>V100</sub> 28.4 mAP <sub>COCO</sub>	14 MB <sub>FP16</sub> 6.4 ms <sub>V100</sub> 37.2 mAP <sub>COCO</sub>	41 MB <sub>FP16</sub> 8.2 ms <sub>V100</sub> 45.2 mAP <sub>COCO</sub>	89 MB <sub>FP16</sub> 10.1 ms <sub>V100</sub> 48.8 mAP <sub>COCO</sub>	166 MB <sub>FP16</sub> 12.1 ms <sub>V100</sub> 50.7 mAP <sub>COCO</sub>



We can see that nano (YOLOv5n) model has the fastest computation time but has a comparatively low mAP while xLarge (YOLOv5x) model has best mAP but takes the longest time to train.

## Metrics

**Precision** measures how accurate is your predictions. i.e. the percentage of your predictions are correct.

**Recall** measures how good you find all the positives. For example, we can find 80% of the possible positive cases in our top K predictions.

Here are their mathematical definitions:

$$Precision = \frac{TP}{TP + FP}$$

$TP$  = True positive

$TN$  = True negative

$FP$  = False positive

$FN$  = False negative

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

**mAP (mean Average Precision)** - mAP (mean average precision) is the average of AP. In some context, mAP is the average of AP for each class.. But in some context, they mean the same thing.

AP (Average precision) is a popular metric in measuring the accuracy of object detectors like Faster R-CNN, SSD, etc. Average precision computes the average precision value for recall value over 0 to 1.

## Graphs

A **PR curve** is simply a graph with Precision values on the y-axis and Recall values on the x-axis. It shows the tradeoff between precision and recall for different threshold.

A **confusion matrix** presents a table layout of the different outcomes of the prediction and results of a classification problem and helps visualize its outcomes. It plots a table of all the predicted and actual values of a classifier.

# Run 1

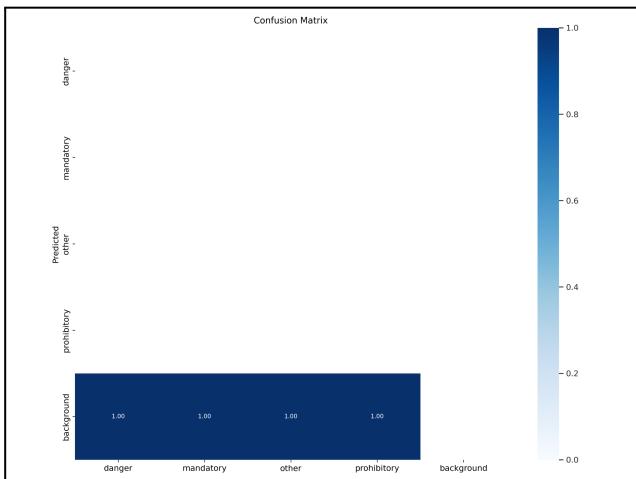
Batch Size - 16

Number of Epochs - 10

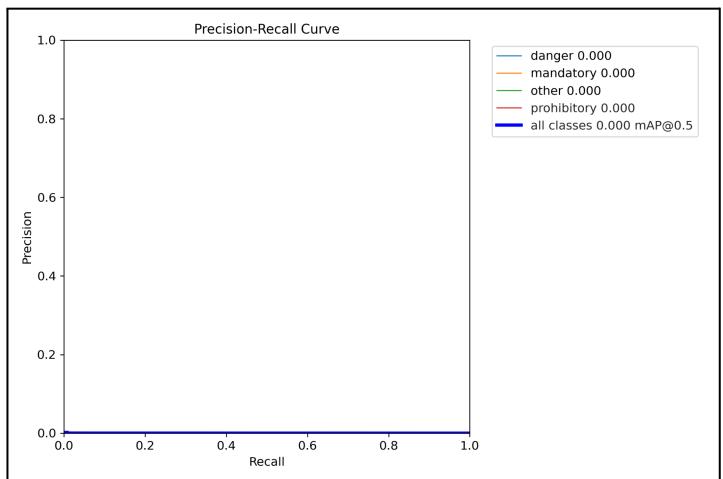
Model - yolov5n

Data Augmentation - Flip

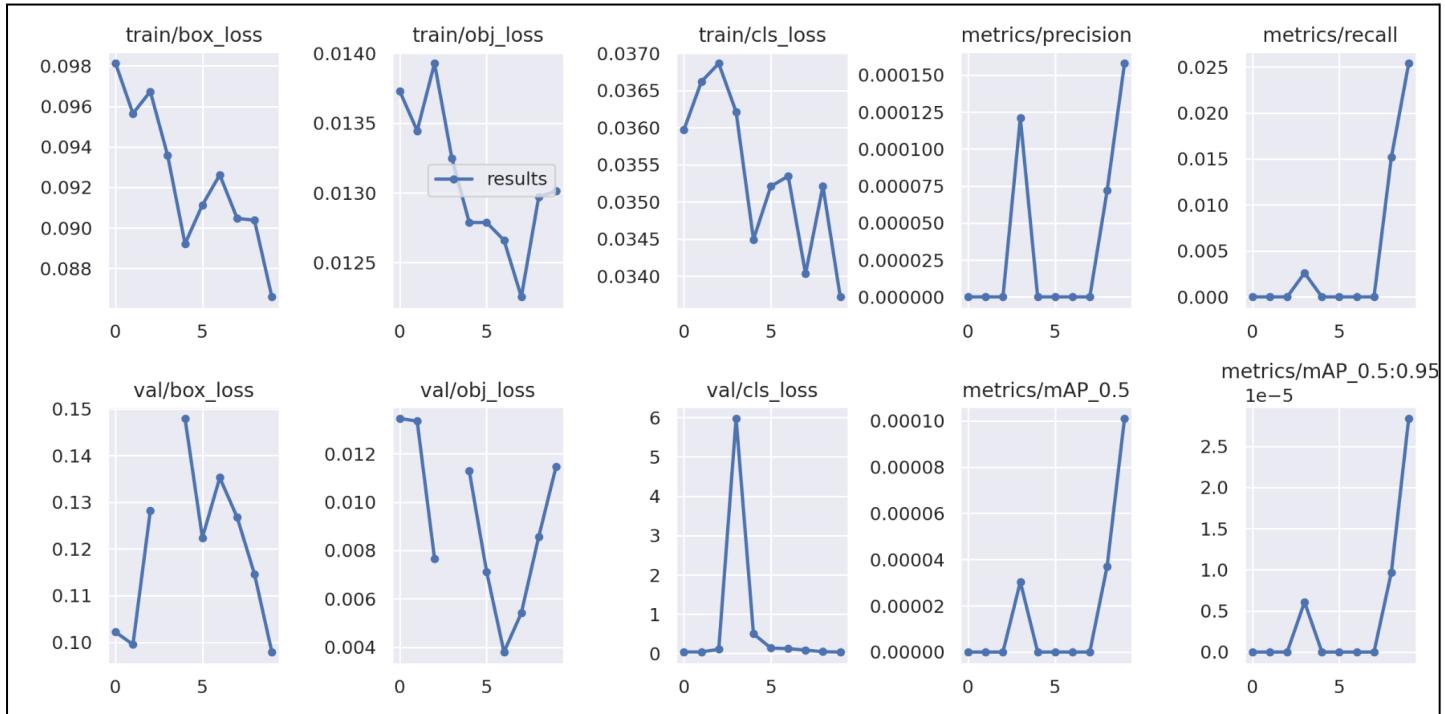
Confusion Matrix



PR curve



## Results



Maximum Precision - 0.00015788 (Recall - 0.025374)

Maximum Recall - 0.025374 (Precision - 0.00015788)

Maximum mAP - 0.00010095

# Run 2

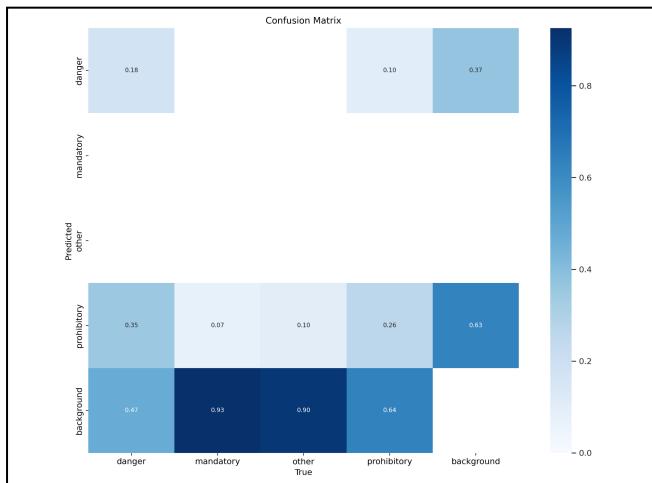
Batch Size - 16

Number of Epochs - 40

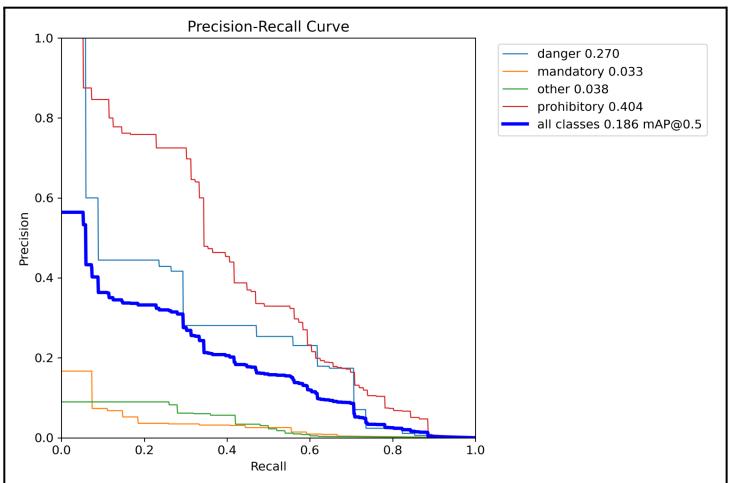
Model - yolov5s

Data Augmentation - Flip

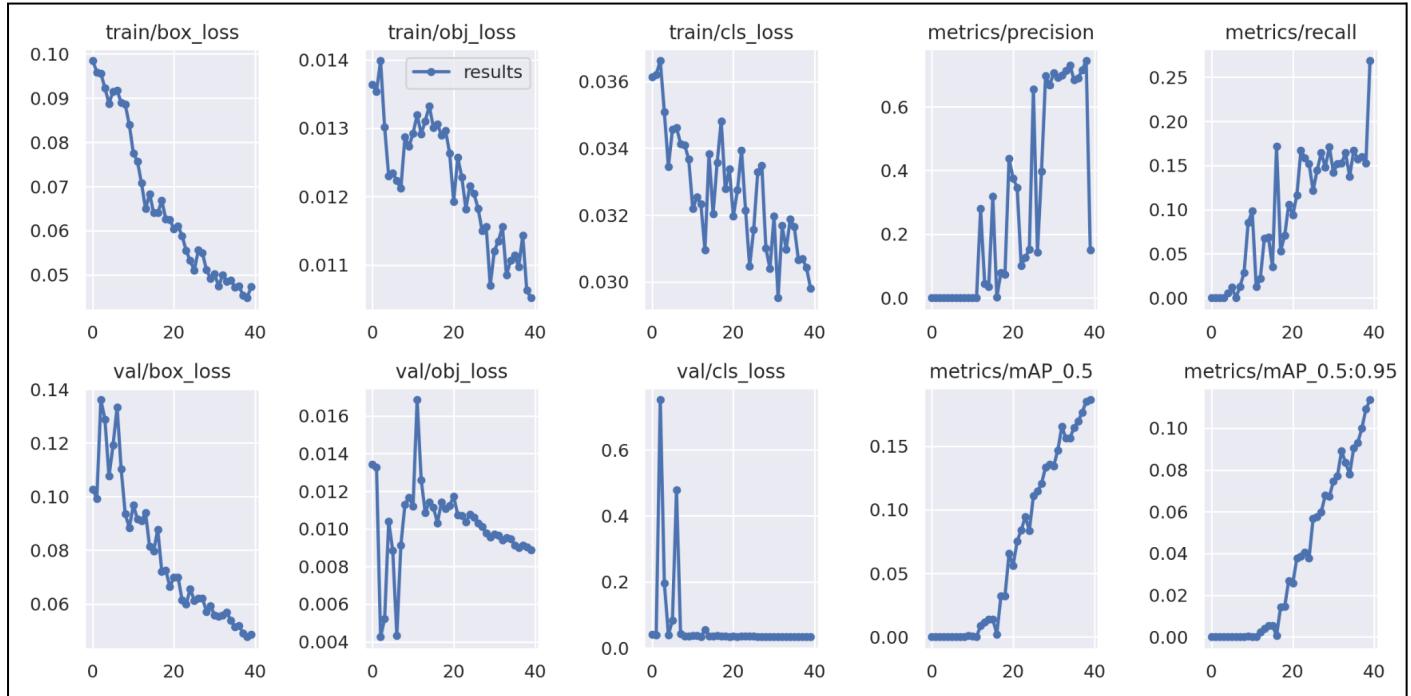
Confusion Matrix



PR curve



## Results



Maximum Precision - 0.74441 (Recall - 0.15257)

Maximum Recall - 0.26876 (Precision - 0.14971)

Maximum mAP - 0.18645

# Run 3

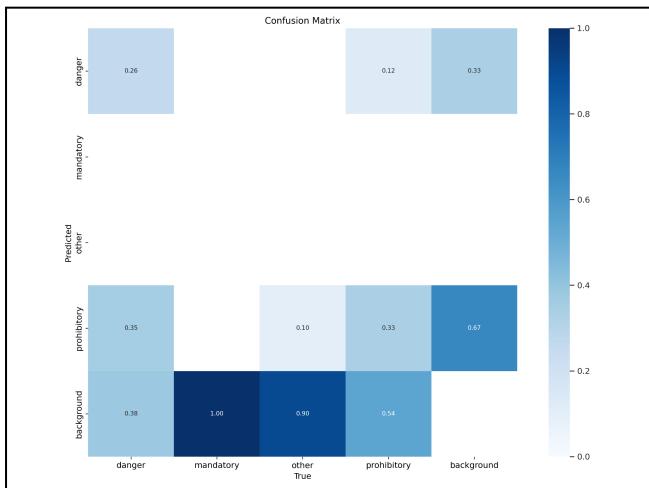
Batch Size - 16

Number of Epochs - 45

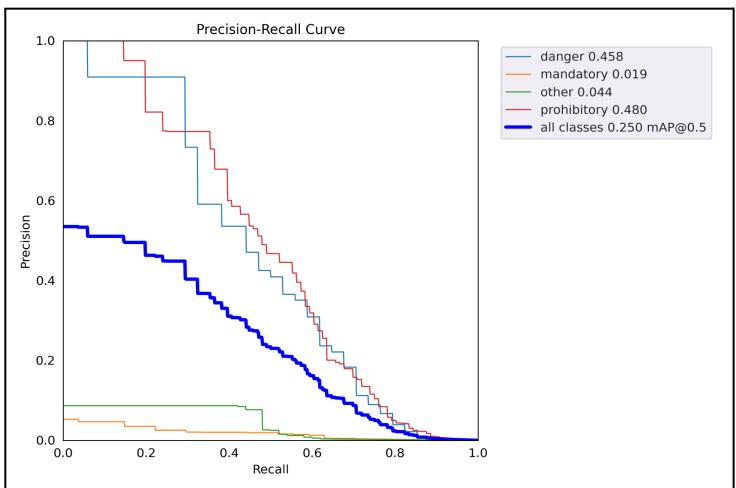
Model - yolov5s

Data Augmentation - Flip

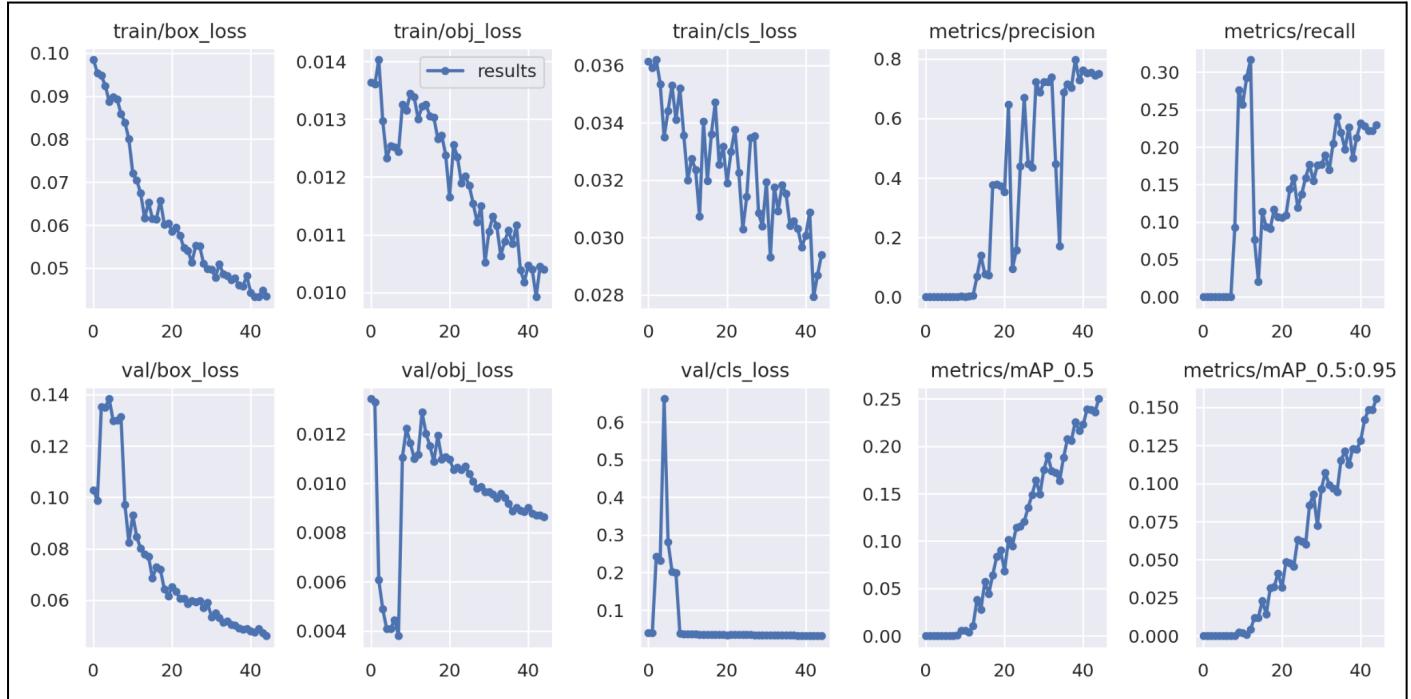
Confusion Matrix



PR curve



## Results



Maximum Precision - 0.79753 (Recall - 0.18505)

Maximum Recall - 0.31665 (Precision - 0.0035093)

Maximum mAP - 0.2501

# Run 4

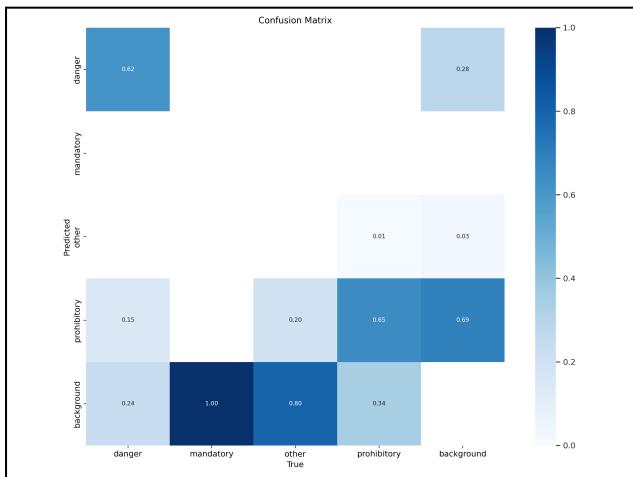
Batch Size - 16

Number of Epochs - 40

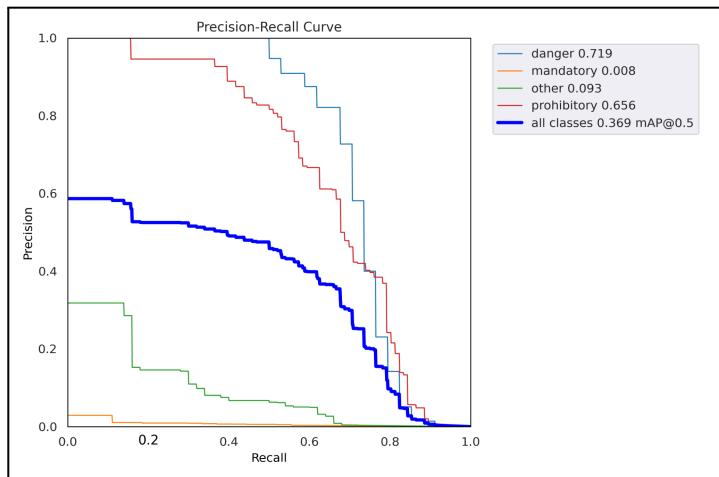
Model - yolov5n

Data Augmentation - Crop (upto 75%)

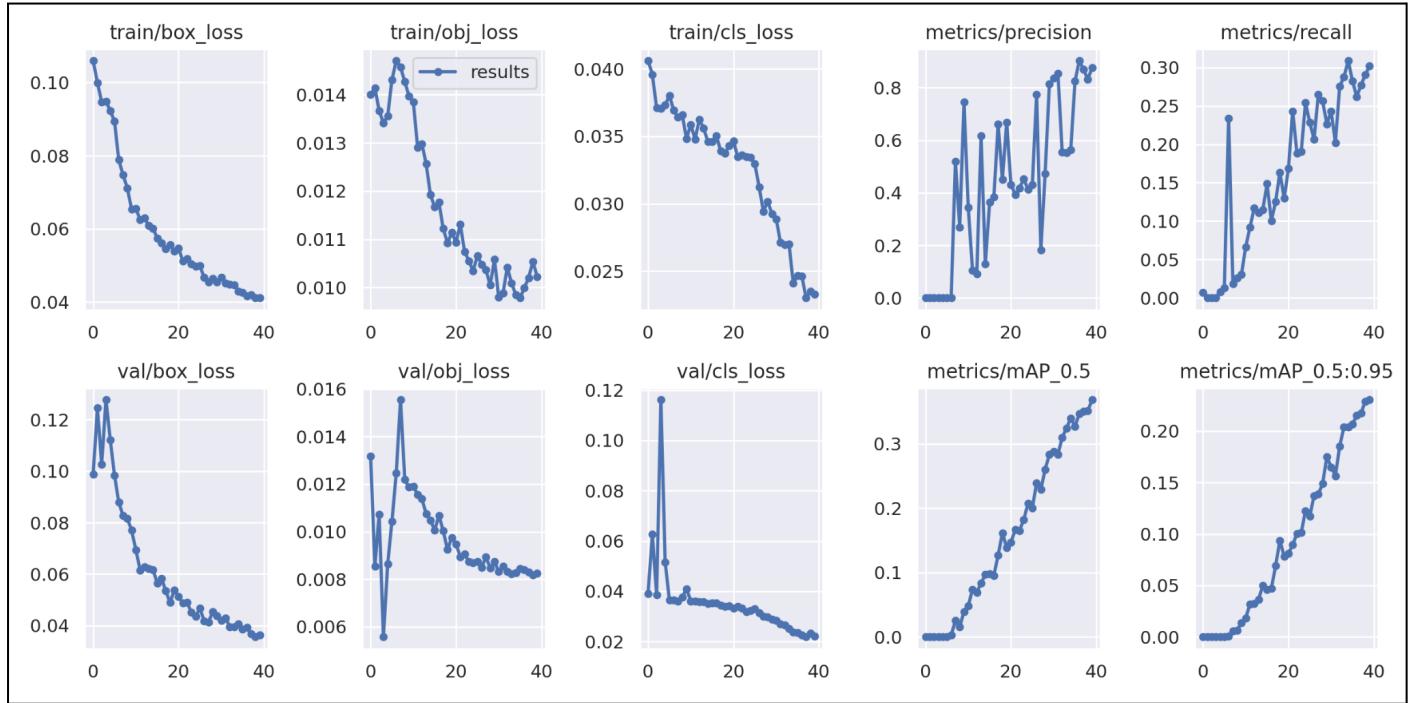
Confusion Matrix



PR curve



## Results



Maximum Precision - 0.90271 (Recall - 0.26177)

Maximum Recall - 0.56415 (Precision - 0.30882)

Maximum mAP - 0.36848

# Run 5

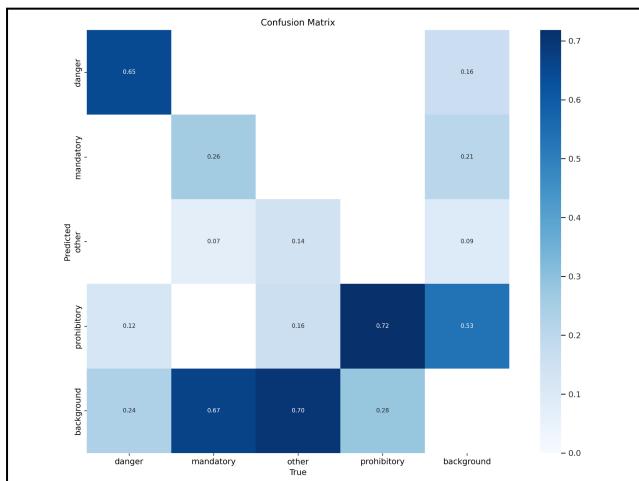
Batch Size - 16

Number of Epochs - 40

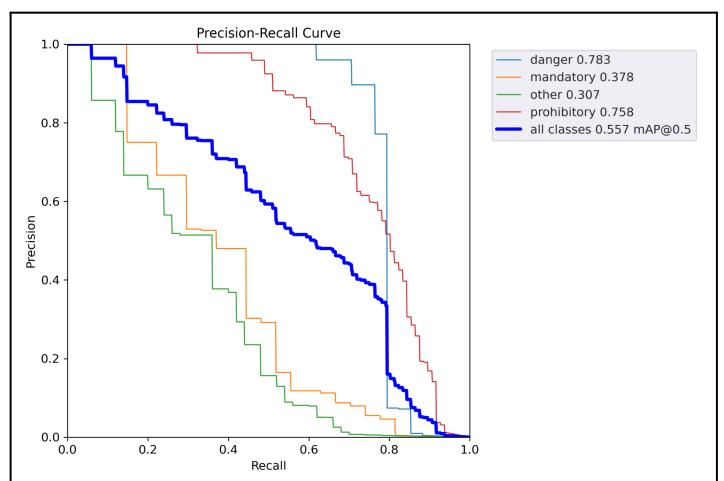
Model - yolov5s

Data Augmentation - Crop (upto 75%)

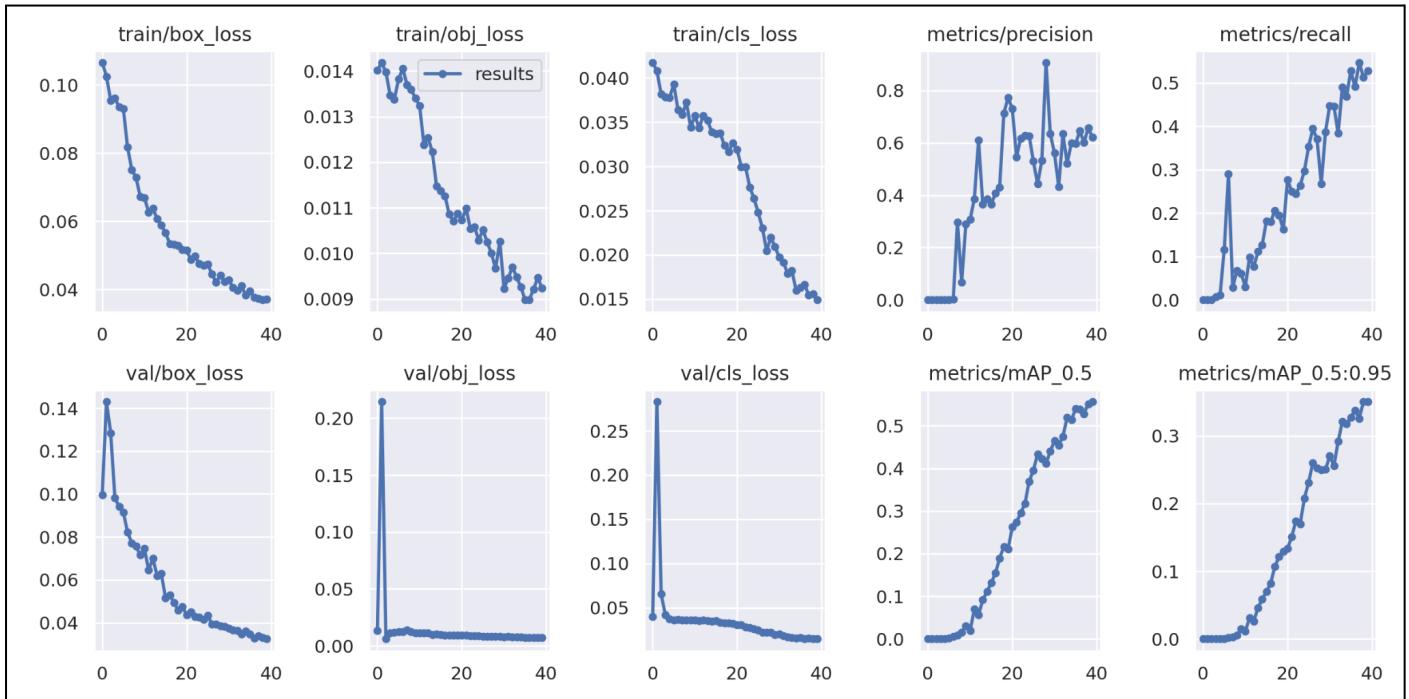
Confusion Matrix



PR curve



## Results



Maximum Precision - 0.90646 (Recall - 0.26685)

Maximum Recall - 0.60181 (Precision - 0.54672)

Maximum mAP - 0.55667

# Run 6

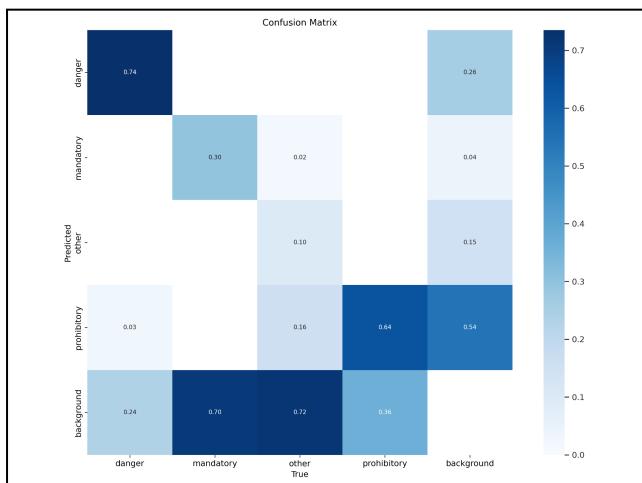
Batch Size - 16

Number of Epochs - 70

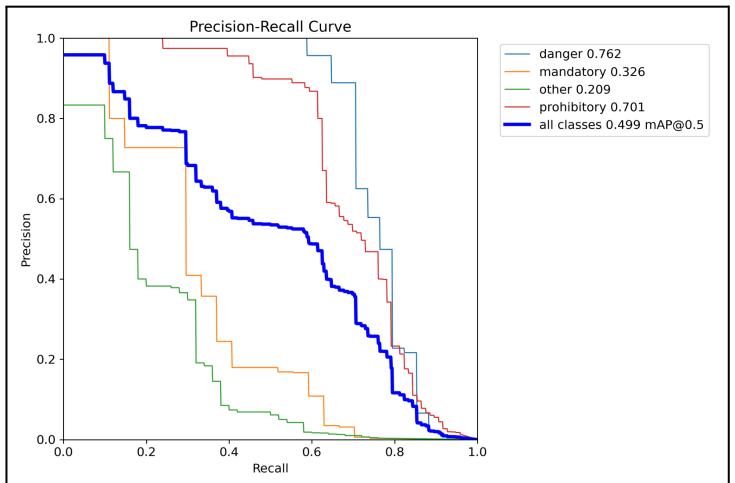
Model - yolov5n

Data Augmentation - Crop (upto 75%)

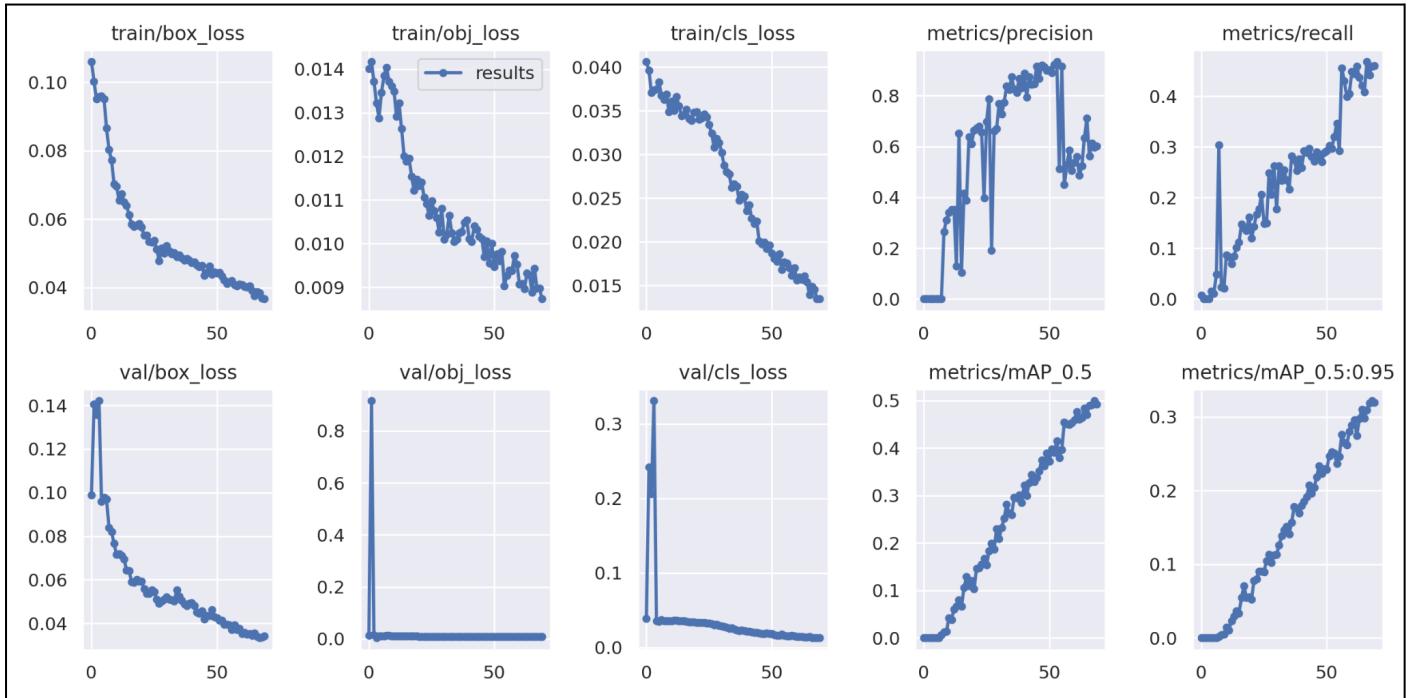
Confusion Matrix



PR curve



## Results



Maximum Precision - 0.93376 (Recall - 0.3197)

Maximum Recall - 0.56174 (Precision - 0.4678)

Maximum mAP - 0.49938

# Run 7

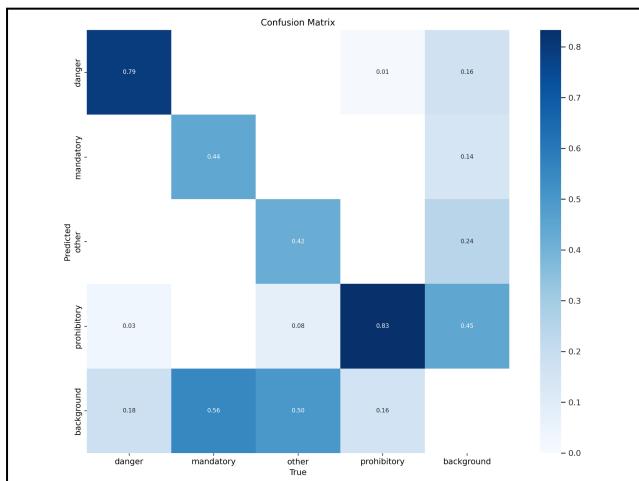
Batch Size - 16

Number of Epochs - 100

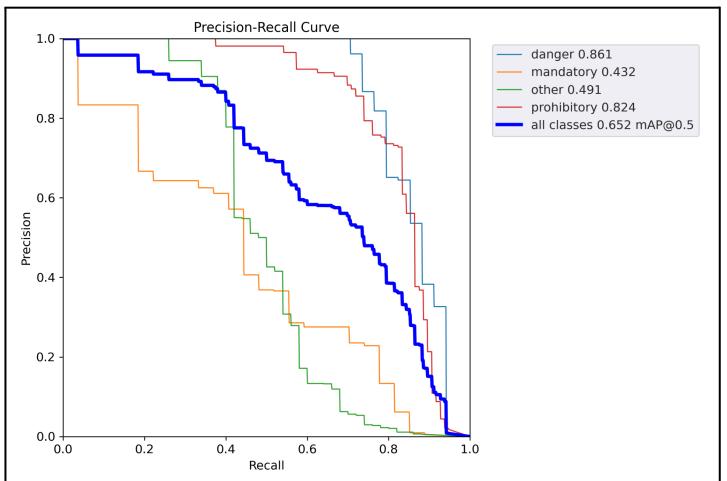
Model - yolov5n

Data Augmentation - Crop (upto 75%)

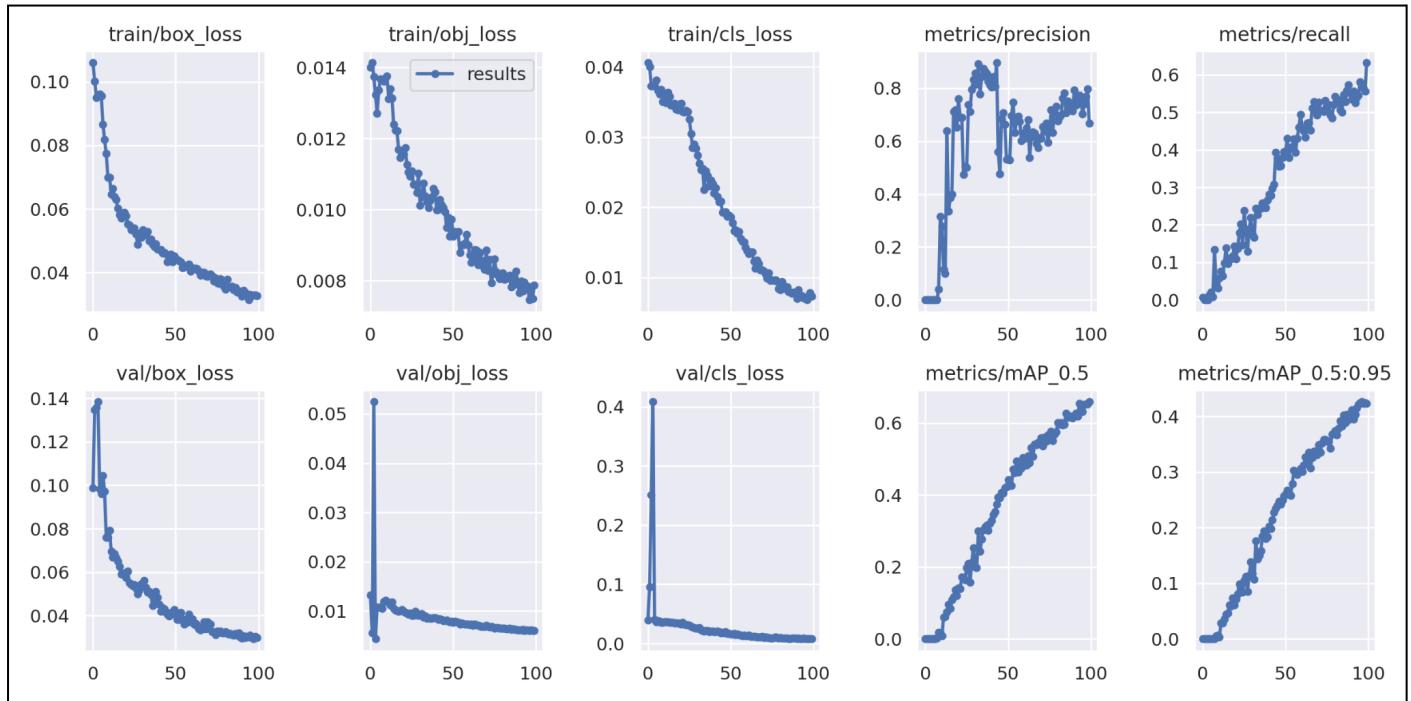
Confusion Matrix



PR curve



## Results



Maximum Precision - 0.89664 (Recall - 0.30714)

Maximum Recall - 0.66718 (Precision - 0.63247)

Maximum mAP - 0.6592

## Observations

1. Per epoch training time for YOLOv5s was around 7minutes whereas for YOLOv5n was 4minutes
2. As the epochs increases the maximum precision and recall values increases for YOLOv5s model
3. For YOLOv5n model maximum precision is around 0.93 for 70 epochs with recall as 0.3
4. Maximum Precision for YOLOv5s model was around 0.9 for 40 epochs with recall as 0.27
5. YOLO performed better on set with data augmentation as cropped rather than flip augmentation
6. Maximum Recall for YOLOv5s model around 0.6 with precision as 0.54 for 40 epochs
7. For YOLOv5n maximum recall was around 0.67 with precision as 0.63 for 100 epochs
8. Best precision is 0.93376 with Recall - 0.3197. Therefore F1 score is 0.238
9. Best recall is 0.66718 with Precision - 0.63247. Therefore F1 score is 0.324
10. Best mAP score is 0.6592

## Challenges

The biggest challenge was the limited kernel time of colab notebook. In colab notebook the kernel gets reset automatically if it idle for few minutes or it is running for more than 6 hours without manually hovering over the cells. So quite a few times I had to retrain the model just because the kernel got reset

## Conclusion

1. The best mAP score is 0.6592
2. Best recall is 0.66718 with Precision - 0.63247. Therefore F1 score is 0.324
3. The best model is YOLOv5n with 100 epochs, 16 batch size and crop data augmentation

## References

[mAP \(mean Average Precision\) for Object Detection | by Jonathan Hui | Medium](#)  
[Introduction to YOLO Algorithm for Object Detection | Engineering Education \(EngEd\) Program | Section](#)