Nat. Volatiles & Essent. Oils, 2021; 8(6): 3437-3455

NVEO
Natural Volatiles &
Essential Oils

The Yolo V5 Based Smart Cellphone Detector

Suraj Beera* , Manish Chembeti* , Naren Hrithik* , Ashish Chitturi*

(B.Tech Computer Science UG)* under the guidance of Dr. Archana Thamizharasan Department of Database Systems Vellore Institute of Technology, Vellore, Tamilnadu - 632014 – India

Abstract--Online learners engage in a variety of educational activities such as reading, writing, watching video lectures, taking online tests, and attending online meetings.

During this era of the COVID-19 pandemic the need for online activities, particularly online conferencing and learning for educational institutions& banking sector has increased in big folds and it is the only efficient and time saving method with less expenses for all kinds of work to continue.

Hence this project which is focused on analyzing attentiveness of online learners in a meeting on a video streaming platform using YOLO V5 algorithm by criteria of object detection, preferably mobile detection ,during online examinations ,in exam halls or conference rooms, in shopping malls or on the road for vehicle detection.

Index Terms—Yolo V5 Algorithm, Object detection, Video streaming.

IMPORTANT ABBREVIATIONS

The expression 'You Only Look Once' is abbreviated as YOLO. This is an algorithm for detecting and recognising different items in a photograph (in real-time). YOLO v5 is utilised to detect objects in this case. [1]

Object detection is a computer approach that searches digital pictures and videos for instances of semantic entities of a certain class (such as people, buildings, or vehicles). It has to do with image processing and computer vision. [2]

I. INTRODUCTION

THIS projectis basedon the YOLO algorithm through which we can detect the degree to which the student/person is
using mobile phone in the monitored environment in a stipulated time.

With schools and colleges being shut due to the pandemic, studies across the world are still being conducted online.
May it be a student attending school classes or an employer doing his office work.

Nonetheless, there are a number of problems about this new teaching or examination system.

1. Parents and teachers cannot monitor their all students or child
at all times during the period of classes to examine children are actually watching the video lectures or if they are simply using their mobile sallowing the video to play in the background.

2. Usage of mobiles in examination hall, during online exams and during confidential board meetings or phone prohiited areas can be resolved with this technique.

Our project aims to solve the above-mentioned problems. We plan to implement a model that could detect whether a student is actually paying attention to class or is distracted bysuch as mobile phone usage.

Now, the question arises that how do you know if one is using preferrably a mobile phone?

Well, the most obvious signs are:
1. The object which the person uses is mostly held in the hands
2. It is on a surface which is viewable by the persons eye and is
used for a detectable amount of period.

For this, we'll use the YOLO V5 algorithm.
Object detection is finding its way into a wide range of businesses, with uses ranging from personal protection to workplace productivity. [3]

Object detection and recognition are used in a variety of computer vision applications, including image retrieval, security, surveillance, autonomous vehicle systems, and machine inspection. Object identification continues to be a major roadblock. When it comes to future object detection use cases, the possibilities are endless.

II. MOTIVATION

1.TECHNICAL USABILITY:

Among the most basic challenges in computer vision is object detection. Several other subsequent computer vision tasks, including as instance segmentation, visual captioning, object tracking, and also more, are built on top of it. Walker detection, individuals counting, face detection, text detection, position detection, and numeric identification are examples of specific asset detection applications.[4]

The fast development and widespread usage of object detection systems has heightened interest in object detector accuracy and speed. The present state-of-the-art object detection works, on the other hand, are either accuracy-oriented with a huge model but high latency, or speed-oriented with a lightweight model but low accuracy. We propose a real-time object identification on mobile devices utilising the YOLO V5 framework via compression-compilation co-design in this paper.[5]

The far more recent addition to the YOLO family of devices is the **YOLOv5**. YOLO was the very first object recognition model to combine object categorization and bounding box prediction into an one end-to-end differentiable network. The Darknet framework was used to develop and manage it. YOLOv5 is the first YOLO model built with the PyTorch framework, making it more simpler and easier to use. Yet, because YOLOv5 did not make major architectural enhancements to the network in YOLOv4, it does not outperform YOLOv4 on a typical standard, the COCO dataset.

2. Increasing efficiency:
Preventing excessive or untimely mobile phone use in families, educational institutions, and companies can assist improve productivity.[6]

3. Avoiding distraction-related accidents:
People who use their phones while performing critical tasks like as driving or administering medical care can be tracked and informed to avert accidents. Similarly, assisted living arrangements for elderly individuals may be monitored.

4. Producing proof of a legal violation:
A mobile phone may be used for data theft and espionage. Mobile phones may also be restricted in exam rooms, corporate meeting rooms, theatres, government offices, embassies, courts of law, and military bases.

A mobile phone may be used for data theft and espionage. Mobile phones may also be restricted in exam rooms, business conference rooms, theatres, government offices, embassies, the courts of law, military posts, fuel outlets, hospitals, and religious locations, among other places.

Similarly, mobile phones may allow inmates to commit new crimes while incarcerated. Furthermore, with the rise of mobile payments, many financial transactions are now conducted via mobile phones, and therefore, losing a mobile phone may have a significant impact on an individual.

As a result, our method can assist in the detection of mobile phone abuse or theft.

In many situations, cellphone detectors are inconvenient or ineffectual.

5.Quality Education Monitoring:
Analyzing attentiveness of online learners in a meeting on a video streaming platform using YOLO V5 algorithm by criteria of object detection, preferably mobile detection ,during online examinations ,in exam halls or conference rooms, in shopping malls or on the road for vehicle detection.[7]

III. BACKGROUND AND RELATED WORK

Glenn Jocher launched YOLOv5 a few days after the YOLOv4 model was introduced on May 27, 2020. (Founder & CEO of Utralytics). YOLOv5 is developed in the Pytorch framework, and Glenn introduced the YOLOv5 Pytorch-based technique.[8]

In conclusion, the YOLO model is a speedy, compact object identification model that is incredibly efficient and accurate for its size and has been improving over time.

It's the YOLOv3 PyTorch addon.

The YOLOv3 PyTorch repository by Glenn Jocher is a natural extension of the YOLOv5 repository. Before going to operation, developers utilised the YOLOv3 PyTorch repository to convert YOLOv3 Darknet weights to PyTorch.

Many individuals commended the PyTorch branch's ease of use and expressed interest in using it for rollout.[9]

The contributions of YOLOv4 and YOLOv5 are largely to combine advances in other domains of computer vision and show that, taken together, they improve YOLO object detection.

Instead, we anticipate the classes and bounding boxes of that specific image in a single iteration of the algorithm and identify several objects using an unique neural network. The YOLO methodology is rapid in comparison to other categorization approaches.

In real time, our algorithm processes 45 frames per second. Despite making faults in localization, the YOLO algorithm can detect less false positives in the back.[10]

TheYOLO method is applicable to regression-based algorithms. We will not choose the elements of the image that are relevant to the end result.

IV OVERVIEW AND WORKING DIAGRAM

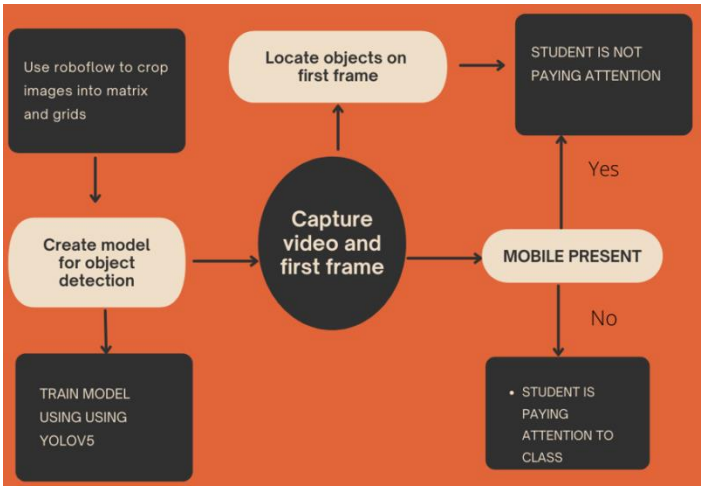

Fig 1.1 Flow chart for object detection (smartphones)

V. THE YOLO V5 ALGORITHM:

For the mobile phone detection, we first need to import all the required libraries that include torch and numpy. To begin with, we have loaded our model of YOLOv5 and tried testing whether its able to detect objects.

We also tried working with real time detections and checking the accuracy of loaded YOLO model. The next step is to actually train our model from scratch and detect whether a student is using mobile phone.

The YOLO Architecture

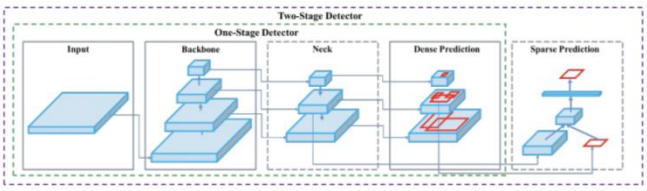

Fig 1.2 Yolo Architecture

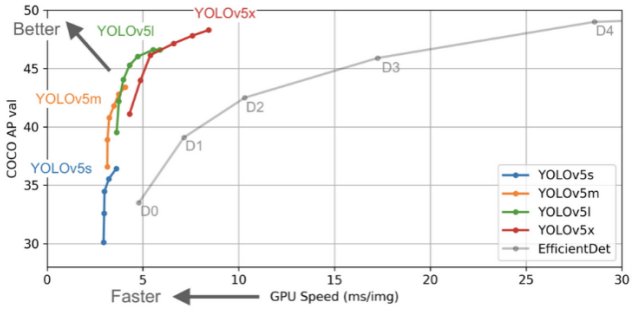

Fig 1.3YOLOv5's previous concept displays considerable potential of province object identification.

The very first version of YOLOv5 is extremely fast, reactive, and simple to use, addS a new PyTorch training and deployment mechanism that improves the high tech for object detectors. Additionally, YOLOv5 is extremely user-friendly and arrives "ready to use" on customizable objects straight out of the box.

Object recognition with excellent performance is achieved using YOLO (You Only Look Once) models. YOLO splits a picture into grids, each of which identifies things inside its own boundaries. They may be used to detect objects in real time using data streams.[11]

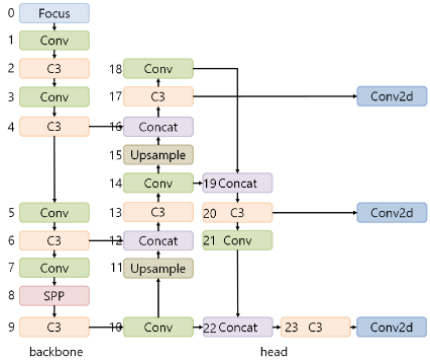

Fig 1.4 Yolo V5 Architecture

VI. MOBILE PHONE DETECTION

We have divided the work in two phases. One is the integrated mobile phone detection model which we will be using with our eye and yawn model to make predictions. Second is our independent mobile phone detection model that gives us a score. For both we have made use of Yolov5.

YOLOv5 is a collection of object detection topologies and modeling techniques which was before on the COCO dataset that reflects Ultralytics accessible research exploring future vision AI technologies, combining knowledge gained and success factors acquired from millions of hours of work.[12]

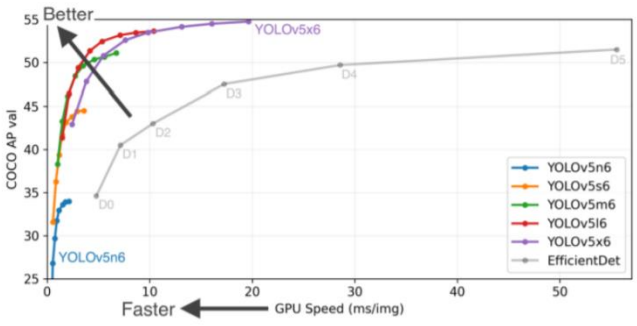

Fig 1.5 Integrated Mobile Detection Model -Coco names

```
['person', 'bicycle', 'car', 'motorcycle', 'airplane', 'bus', 'train', 'truck', 'boat', 'traffic light', 'fire hydrant', 'st
reet sign', 'stop sign', 'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'cow', 'elephant', 'bear', 'zebr
a', 'giraffe', 'hat', 'backpack', 'umbrella', 'shoe', 'eye glasses', 'handbag', 'tie', 'suitcase', 'frisbee', 'skis', 'snowb
oard', 'sports ball', 'kite', 'baseball bat', 'baseball glove', 'skateboard', 'surfboard', 'tennis racket', 'bottle', 'plat
e', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl', 'banana', 'apple', 'sandwich', 'orange', 'broccoli', 'carrot', 'h
ot dog', 'pizza', 'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed', 'mirror', 'dining table', 'cell phone', 'desk',
'toilet', 'door', 'tv', 'laptop', 'mouse', 'remote', 'keyboard', 'cell phone', 'microwave', 'oven', 'toaster', 'sink', 'refr
igerator', 'blender', 'book', 'clock', 'vase', 'scissors', 'teddy bear', 'hair drier', 'toothbrush', 'hair brush']
```

VII Working, Implementation and Training our model:

```
net = cv2.dnn.readNet("yolov3.weights", "yolov3.cfg")
classes = []
with open("coco.names", "r") as f:
    classes = [line.strip() for line in f.readlines()]
layer_names = net.getLayerNames()

output_layers = [layer_names[i - 1] for i in net.getUnconnectedOutLayers()]
```

Fig 1.6 Training model code

```
currentframe=0
step=10
frames_count=5
framescaptured=0
steps=10
```

```
if currentframe > (step*frame_per_second):
    currentframe = 0
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    cv2.imwrite("C:/Users/Manan/Desktop/Projects/AI PROJECTS/S
    print("SS taken")
    cv2.imshow('img', frame)

    framescaptured+=1
    img_counter+=1

    if framescaptured > frames_count-1:
        ret = False
    if cv2.waitKey(1)==ord('q'):
        break
currentframe += 1
```

Fig 1.7 Capturing images:

2. Independent Mobile Phone Detection with score:

Software Requirement.

- matplotlib>=3.2.2
- numpy>=1.18.5
- opencv-python>=4.1.2
- Pillow>=7.1.2

- PyYAML>=5.3.1
- requests>=2.23.0
- scipy>=1.4.1
- torch>=1.7.0
- torchvision>=0.8.1
- tqdm>=4.41.0

Using Roboflow to crop images:

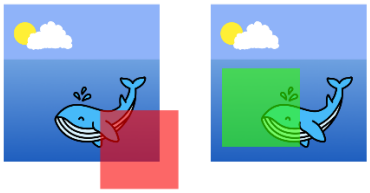

Fig 1.8 The crop on the right is valid, but the one on the left is not.

If we aim to apply a random crop to an object detection problem, we must also handle updating the bounding box. Specifically, if our newly cropped image contains an annotation that is completely outside the frame, we should drop that annotation. If the annotation is partially in frame, we need to crop the annotation to line up with the newly created edge of the frame.[13]

This is exactly how Roboflow implements a random crop when also resizing an image.

Roboflow Train is a Roboflow Auto ML product. It's the quickest and most convenient method to train and deploy a cutting-edge object detection model on your own data. We'll take care of the rest with just one click.

You'll get direct connections to a hosted inference API, a Tensorflow JS prototype you can integrate in your web application, with the on inference server you can run on end devices like the NVIDIA Jetson to query your model for forecasts using your preferred programming language once your framework is finished training

Fig 1.9 Roboflow is used for data augumentation

Roboflow enables you to easily organize, label, and prepare a high-quality dataset with your own custom data. Roboflow also makes it easy to establish an active learning pipeline, collaborate with your team on dataset improvement, and integrate directly into your model building workflow with the
roboflow pip package.

The following are the steps to train a custom Yolo V5 model:

- Prepare your surroundings.

- Create the data and folders.
- Set up the YAML configuration files.
- Train the model to recognise different types of items.

To detect things on the test datum, use your own Yolo V5 model.

A few items must be downloaded from the internet in order to train a Yolo V5 model.

The quickest way to download and set up your environment on a notebook is to use terminal commands straight from your notebook, as seen below:
GitHub's yolo V5 repository should be cloned.

Yolo V5 is based on Torch, which might be difficult to set up at times. If you're having trouble here, try switching to a Kaggle or Colab notebook: both of them function well for this installation.

Directory structure in Yolo V5

At the same level as your yolov5 folder, add a folder named data. Create a folder for photos and a folder for labels in this data folder. Make a folder for train data and another for validation data within each of them.[14]

```
|- workingdir
        |--yolov5
        |--data
            |--images
                |--train
                |--valid
            |--labels
                |--train
                |--valid
```

Fig 2.1 Yolo v5 Directory structure

Data Format for Yolo V5

The photographs

The photographs must be placed in the image folders directly. Images for training may be found in the data/pictures/train folder, while validation images can be found in the data/images/valid folder. The picture names must be simple and unique, ending in.jpg (or another format).

The Inscriptions

The labels must be stored in the data/labels/train/ or data/labels/valid/ folders. The labels file must have the same name as the picture, but with ".txt" instead of ".jpg" as the extension.

The bounding boxes must be stated one per line, with the following information on each line: the object's class number (always 0 if only one class) the bounding box's standardised centre pixel in terms of width.[15]

The bounding boxes must be provided one per line, with the class number of the item in the bounding box on that

line (always 0 if only one class)

- the width of the bounding box's standardised centre pixel
- the height of the bounding box's standardised centre pixel
- the bounding box's standardised width
- the bounding box's standardised height

The number of pixels in a picture is divided by the total number of pixels in the image to achieve standardisation. So, on an image of size (100, 100), a bounding box on pixel (10, 20) with a width of 30x40 would be standardised to (0.1, 0.2, 0.3, 0.4).

The number of bounding boxes in the label file equals the number of lines in the label file.

Now it's time for the final phase: detecting items on unseen photographs. This is done with the detect.py terminal script, which creates a new folder with outputs. You may either make photos with the bounding boxes drawn on them or create text files containing the bounding box locations.

VIII Code

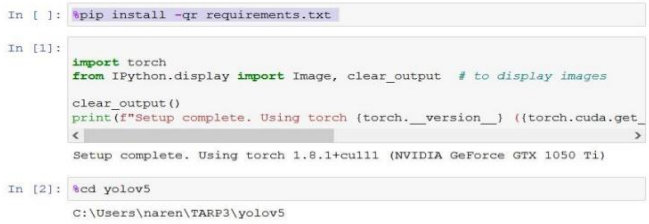

Fig 2.2 Installation code screenshot

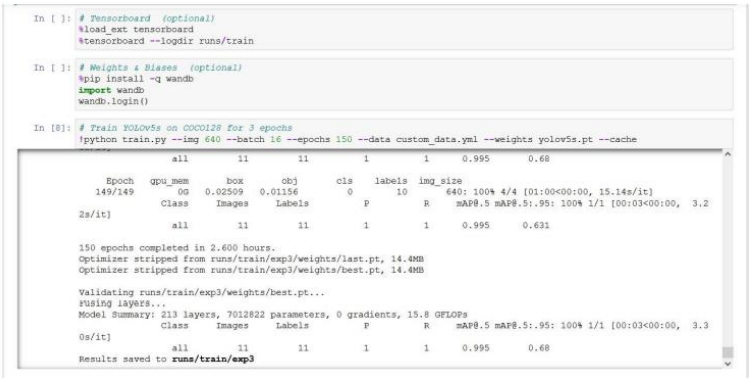

Fig 2.3 Installation code output

Detecting custom image:

detect.py executes YOLOv5 inference on a wide variety of sources, autonomously acquiring models from some of the most recent YOLOv5 release, as well as storing the results to runs/detect.

Example inference sources are:

python detect.py --source 1 # mycam
file.jpg # image
file.mp4 # video path/

directory path/*.jpg
glob Detecting
objects in a recorded video(using1.mp4).

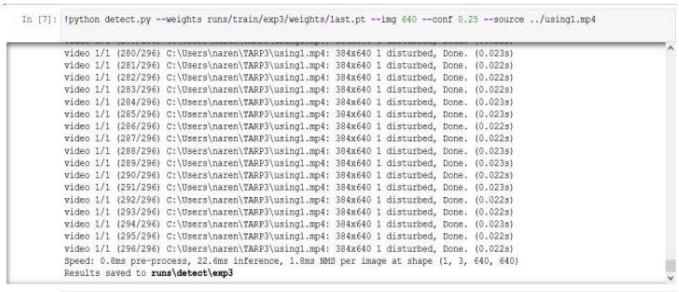

Fig 2.4 Detecting custom image:

Detecting objects in an existing image folder(using mobile phone)

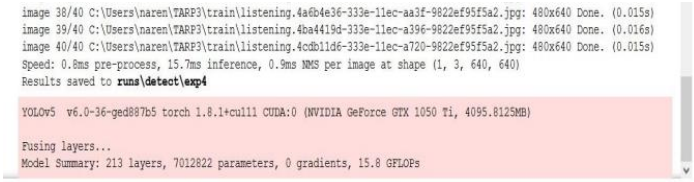

Fig 2.5 Detecting objects in an existing image folder(using mobile phone)

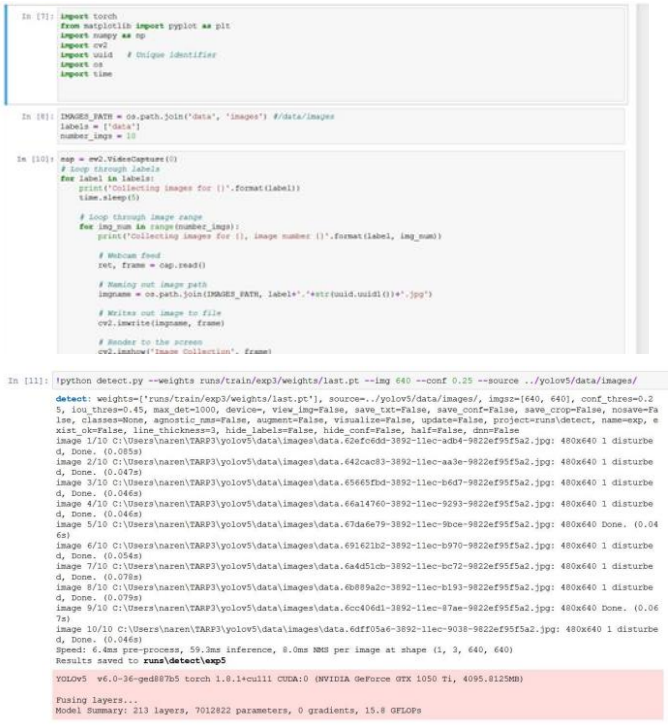

Fig 2.6 Processed result delivered to folder

IXRESULTS

Metrics calculated in Mobile phone detection model.

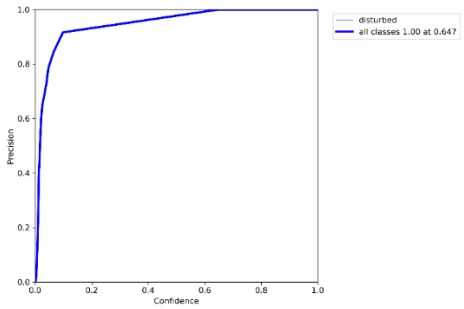

Fig 2.7 Precision Graph

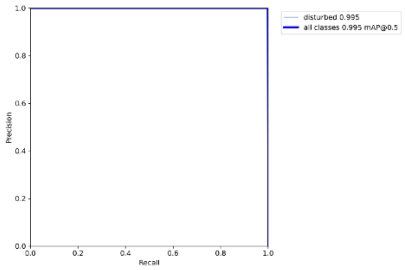

Fig 2.8 Precision and Recall Graph

Recall and Confidence Graph

Recall is a measure that indicates how many correct positive predictions were produced out of all possible positive forecasts. Unlike precision, which only considers the right positive forecasts out of all true positives, recall considers the positive predictions that were overlooked.[16]

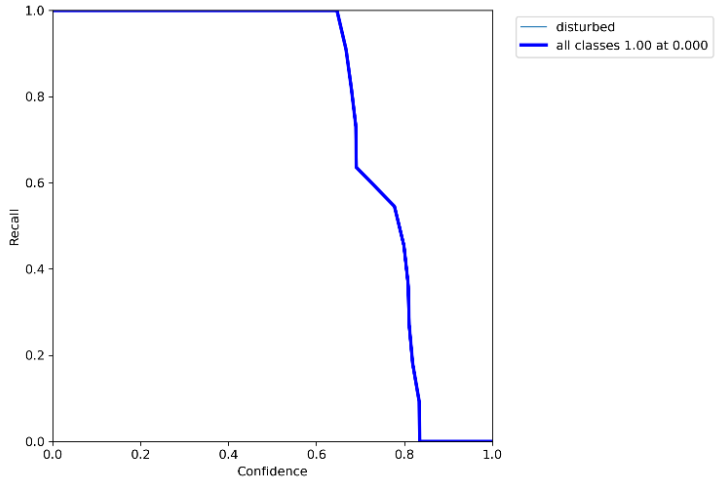

Fig 2.9 Recall Graph

Prediction and Confidence Graph

The accuracy of the results. This is the range in which the population mean is expected to fall. The range of values within which the "actual" god's-own-truth finding is obtained is termed as a confidence interval.

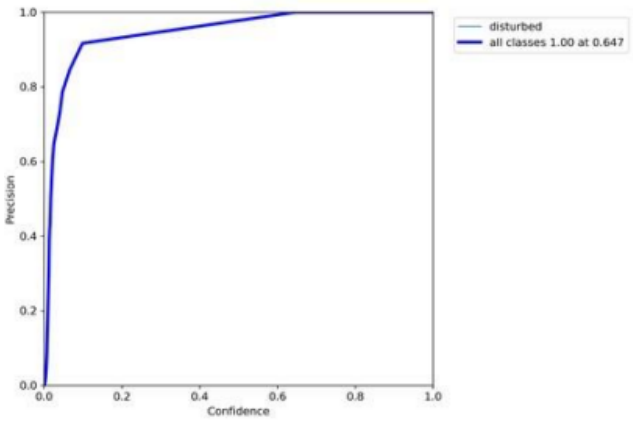

Fig 3.0 Prediction and Confidence Graph

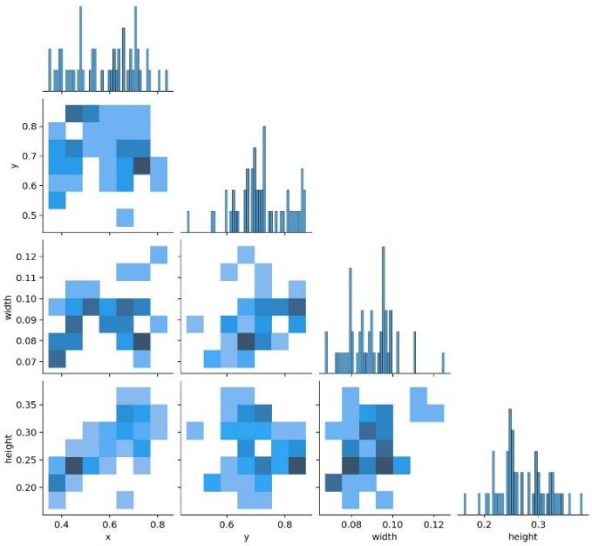

Fig 3.1 Labels correlogram

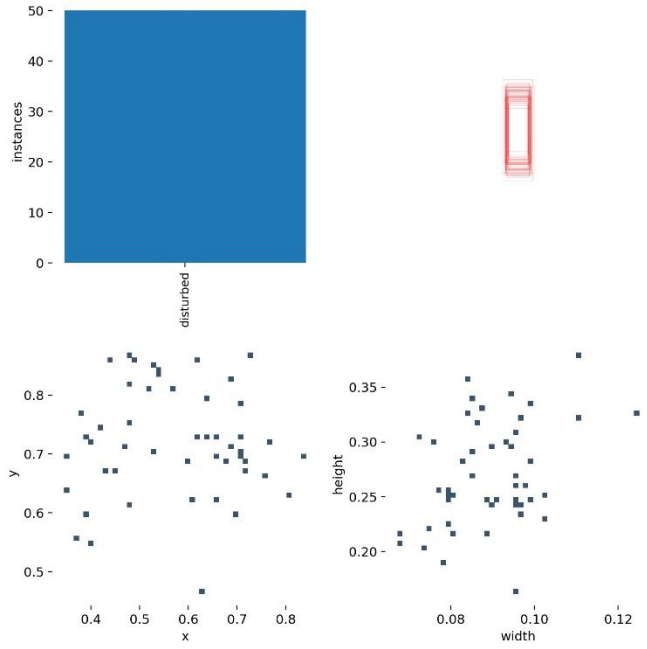

Fig 3.2 Labels and its boxes

Graphs can either have positive correlation, negative correlation or no correlation. [17]

Positive correlation suggests that as one variable rises, another rises as well. A negative correlation suggests that when one variable rises, another falls. They have a negative connection.

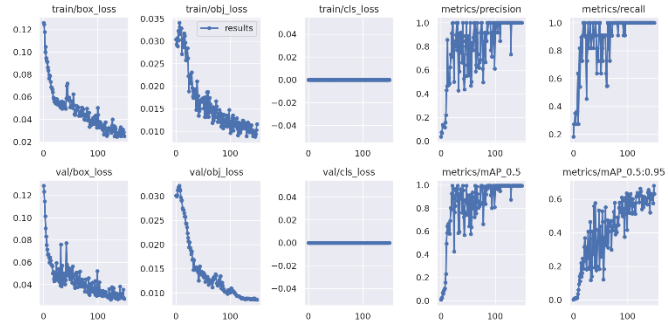

Fig 3.3 Results and Losses graph

X TRAINED DATSETS AND RESULTS:

Fig 3.4-3.9 Compilation of trained images.

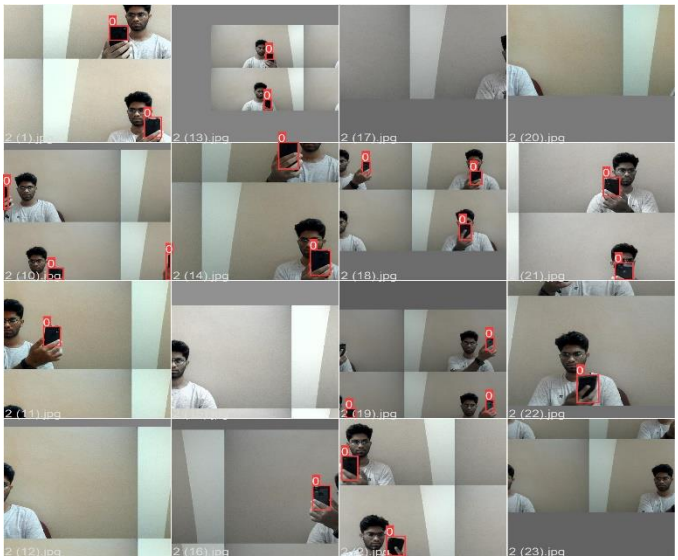

Fig 4.0 Training batch 1

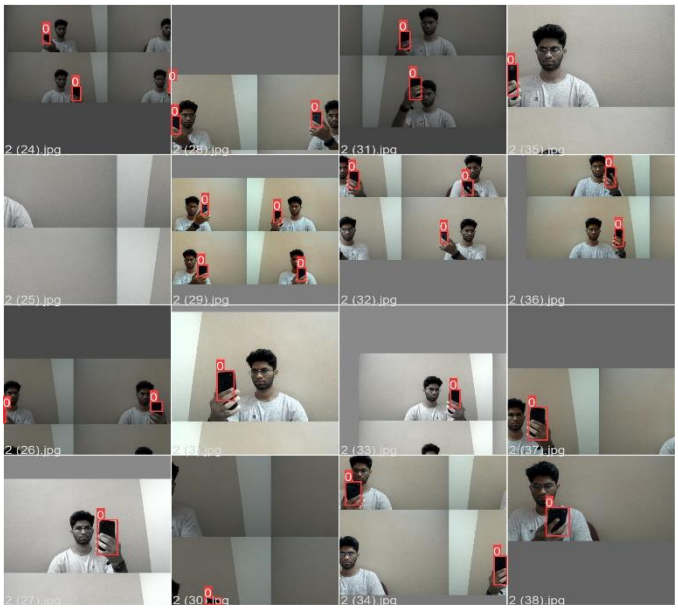

Fig 4.1Training batch 2

Validation images from the trainings

Fig 4.2 Validations batch labels

Fig 4.3 Validation Batch predictions

The mobile phone is succesfully detected.

XII YOLOv4 and YOLOv5 Training Time Comparison

The training length in YOLOv4 Darknet is determined by the number of iterations max batches (not epochs). The repository's suggestion for custom objects is 2000 x num classes. On our sample dataset, YOLOv4 Darknet takes 14 hours with this option. However, it reaches maximum validation evaluation much sooner; we witnessed maximum validation evaluation at 1300 iterations, which took around 3.5 hours.[18][19]

On the other hand, YOLOv5s trained for 14.46 minutes across 200 epochs.

Fig 4.4 YOLOv4 and YOLOv5 Training Time Comparison

XIII Conclusion

The first version of YOLOv5 is highly fast, responsive, and easy to use. While YOLOv5 does not provide a new model topology to the YOLO family, it does add an additional PyTorch training and deployment framework that upgrades the advancements for object detectors. Nevertheless, YOLOv5 is
extremely adaptive and it is usually "out of the box" ready to use on bespoke objects.

Basically we are using roboflow to crop the images into matrices and further into grids and then creating our own customised dataset using YOLO v5 for mobile detection and training it using the Yolov5 algorithm.This gives a very high accuracy for moble detection.

The YOLOv5 model is implemented in Pytorch, as compared to previous improvements that used the DarkNet technology. This streamlines the model's interpretation, training, and deployment.[20]

YOLO v5 offers a tremendous advantage of run speed. The smaller YOLO v5 model is 2.5 times better than the bigger YOLO v5 model, and it identifies tiny things better. Ultralytics has done a brilliant job with its open - sourced YOLO v5 model, which is easy to train and deploy for inference.

Hence using our own customised dataset and YOLO v5 algorithm, we are able to detect mobiles with a very high accuracy thereby

XIV REFERENCES

[1] Joseph Redmon, Santosh Divvala, Ross Girshick, “You
Only Look Once: Unified, Real-Time Object Detection”,The IEEE Conference on Computer Vision.

[2] S. Ren, K. He, R. B. Girshick, and J. Sun, “Faster R-CNN: towards real-time object detection with region proposal networks,” CoRR, vol. abs/1506.01497, 2015. [Online]. Available: http://arxiv.org/abs/1506. 01497 2

[3] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger,” arXiv preprint, 2017. 3, 4

[4] X. Zhang, X. Zhou, M. Lin, and J. Sun, “Shufflenet: An extremely efficient convolutional neural network for mobile devices,” arXiv preprint arXiv:1707.01083, 2017. 3, 7

[5]How to Train YOLOv5 On a Custom Dataset
Jacob Solawetz, Joseph Nelson
https://blog.roboflow.com/how-to-train-yolov5-on-a-custom-dataset/

[6] Training a Custom Object Detection Model With Yolo-V5
Prabhat Kumar Sahu
https://medium.com/analytics-vidhya/training-a-custom-object-detection-model-with-yolo-v5-aa9974c07088

[7]How to Train A Custom Object Detection Model with YOLO v5 ;Jacob Solawetz
https://towardsdatascience.com/how-to-train-a-custom-object-detection-model-with-yolo-v5-917e9ce13208
[8]Wang, Chien-Yao (2021). "Scaled-YOLOv5: Scaling Cross Stage Partial Network". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). arXiv:2011.08036. Bibcode:2020arXiv201108036W.

[9]Zhang, Shifeng (2018). "Single-Shot Refinement Neural Network for Object Detection". Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4203–4212.arXiv:1711.06897. Bibcode:2017arXiv171106897Z.

[10] DEVELOPERS CORNER
Guide to Yolov5 for Real-Time Object Detection
https://analyticsindiamag.com/yolov5/

[11] Shafiee, Mohammad Javad, et al. "Fast YOLO: A fast you only look once system for real-time embedded object detection in video." arXiv preprint arXiv:1709.05943 (2017).

[12] Simony, M., Milzy, S., Amendey, K., & Gross, H. M. (2018). Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops (pp. 0-0).

[13] Liu, C., Tao, Y., Liang, J., Li, K., & Chen, Y. (2018, December). Object detection based on YOLO network. In 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC) (pp. 799-803). IEEE.

[14] Huang, Z., Wang, J., Fu, X., Yu, T., Guo, Y., & Wang, R. (2020). DC-SPP-YOLO: Dense connection and spatial pyramid pooling based YOLO for object detection. Information Sciences, 522, 241-258.

[15] Fang, W., Wang, L., & Ren, P. (2019). Tinier-YOLO: A real-time object detection method for constrained environments. IEEE Access, 8, 1935-1944.

[16] Ahmad, T., Ma, Y., Yahya, M., Ahmad, B., & Nazir, S. (2020). Object detection through modified YOLO neural network. Scientific Programming, 2020.

[17] Yin, Y., Li, H., & Fu, W. (2020). Faster-YOLO: An accurate and faster object detection method. Digital Signal Processing, 102, 102756.

[18] Adarsh, P., Rathi, P., & Kumar, M. (2020, March). YOLO v3-Tiny: Object Detection and Recognition using one stage improved model. In 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS) (pp. 687-694). IEEE.

[19] Lu, Y., Zhang, L., & Xie, W. (2020, August). YOLO-compact: An Efficient YOLO Network for Single Category Real-time Object Detection. In 2020 Chinese Control And Decision Conference (CCDC) (pp. 1931-1936). IEEE.

[20] Ding, C., Wang, S., Liu, N., Xu, K., Wang, Y., & Liang, Y. (2019, February). REQ-YOLO: A resource-aware, efficient quantization framework for object detection on FPGAs. In Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (pp. 33-42).