MOON CRATERS DETECTION

PROJECT REPORT

SUMBITTED BY

S Subhitcha (20MIS1020)

Bhavaa Dharshini (20MIS1031)

Deepti Kannan (20MIS1061)

Mohammed Aadil (20MIS1099)

School Of Computer Science and Engineering

Vellore Institute of Technology

Vandalur - Kelambakkam Road, Chennai - 600 127

November - 2022

School Of Computer Science and Engineering

DECLARATION

We hereby declare that the project entitled Moon Craters Detection submitted by S Subhitcha (20MIS1020), Bhavaa Dharshini (20MIS1031), Deepti Kannan (20MIS1061), Mohammed Aadil (20MIS1099) to the School of Computer Science and Engineering, Vellore Institute of Technology, Chennai, 600 127 is a bona-fide record of the work carried out by me under the supervision of Dr. S. Geetha. We further declare that the work reported in this project, has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other Institute or University.

Place: Chennai

Signature of Candidate

Date: 19th Nov, 2022

Acknowledgement

We are obliged to give my appreciation to a number of people without whom we could not have completed this thesis successfully.

We would like to place on record my deep sense of gratitude and thanks to my internal guide Dr. S. Geetha, Vellore Institute of Technology, Chennai, whose esteemed support and immense guidance encouraged me to complete the project successfully.

We would like to thank our HoD Dr. Nisha VM, School of Computer Science and Engineering (SCOPE) and Project Co-Ordinator Dr. C. Sweetlin Hemalatha, Vellore Institute of Technology, Chennai, for their valuable support and encouragement to take up and complete this thesis.

Special mention to our dean Dr. Jagadeesh Kannan R, Associate Dean Dr. S. Geetha, School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for motivating us in every aspect of software engineering.

We thank our management of Vellore Institute of Technology, Chennai, for permitting me to use the library and laboratory resources. We also thank all the faculty members for giving me the courage and the strength that we needed to complete our goal. This acknowledgment would be incomplete without expressing the wholehearted thanks to my family and friends who motivated me during the course of my work.

S Subhitcha (20MIS1020)

Bhavaa Dharshini (20MIS1031)

Deepti Kannan (20MIS1061)

Mohammed Aadil (20MIS1099)

Abstract

YOLO v5 is a family of compound scaled object detection models trained on the COCO dataset and includes simple functionality for Test Time Augmentation (TTA), model assembling, hyperparameter evolution, and export to ONNX, CoreML and TFLite. YOLO an acronym for 'You only look once, is an object detection algorithm that divides images into a grid system. Each cell in the grid is responsible for detecting objects within itself. YOLO is one of the most famous object detection algorithms due to its speed and accuracy. For this Challenging experiment we have used YOLO V5 for the circular object (moon surface) detection using anchor boxes. To make the image clear we use image processing filters like 2D, image sharpening, and noise filters. Noises frequently corrupt pictures. Noise can be acquired during picture catch, transmission, and so on. Noise expulsion is a significant assignment in picture handling. Overall, the consequences of the commotion expulsion impact the nature of the image-handling methods.

Table Of Contents

Introduction

Yolo architecture is more like FCNN (fully convolutional neural network) and passes the image (nxn) once through the FCNN and output is (mxm) prediction. This the architecture is splitting the input image in mxm grid and for each grid generation 2 bounding boxes and class probabilities for those bounding boxes. Note that bounding box is more likely to be larger than the grid itself. Compared to other region proposal classification networks (fast RCNN) which perform detection on various region proposals and thus end up performing prediction multiple times for various regions in an image. We reframe object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities.

A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes. YOLO trains on full images and directly optimizes detection performance. This unified model has several benefits over traditional methods of object detection. First, YOLO is extremely fast. Since we frame detection as a regression problem, we don't need a complex pipeline. We simply run our neural network on a new image at test time to predict detections. As YOLO v5 is a single-stage object detector, it has three important parts like any other single-stage object detector.

1. Model Backbone
2. Model Neck
3. Model Head

1. **Model Backbone** is mainly used to extract important features from the given input image. In YOLO v5 the CSP — Cross Stage Partial Networks are used as a backbone to extract rich in informative features from an input image.

2. **Model Neck** is mainly used to generate feature pyramids. Feature pyramids help models to generalized well on object scaling. It helps to identify the same object with different sizes and scales. Feature pyramids are very useful and help models to perform well on unseen data. There are other models that use different types of feature pyramid techniques like FPN, BiFPN, PANet, etc. In YOLO v5 PANet is used for as neck to get feature pyramids. Understanding Feature Pyramid Networks for object detection (FPN) The model.

3. **Model Head** is mainly used to perform the final detection part. It applied anchor boxes on features and generates final output vectors with class probabilities, objectless scores, and bounding boxes. In YOLO v5 model head is the same as the previous YOLO V3 and V4 versions.

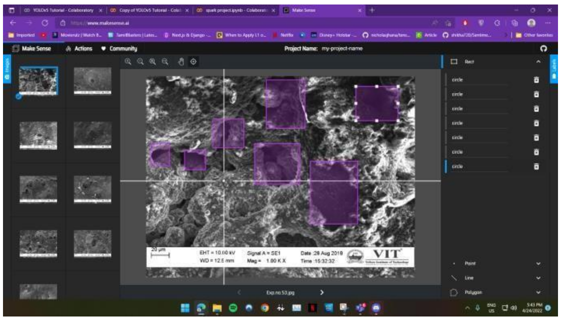
Literature Survey

Authors And Paper Title	Journal Name	Description	Result
Craters Detection on Lunar Nur Diyana Kamarudin, Siti Noormiza Makhtar, Hizrin Dayana M.Hidzir Department of Electrical and Electronic Engineering, Faculty of Engineering, Universiti Pertahanan Nasional Malaysia, 57000 Kem Sg. Besi, Sg.Besi, Kuala Lumpur	Craters Detection on Lunar	This project focused on identification of craters in terms of its characteristics and detection of these visual features of the moon to determine a safe landing site for a lunar Lander. This paper had discussed the method of employing MATLAB and image processing tool on an optical image as well as the morphological image detection fundamentals. In addition, some geometrical projection analysis in reconstructing an ellipse as a disc will be evaluated in order to obtain the orientation of the disc (crater) for autonomous optical navigation system.	This paper focused primarily on identification and detection of craters on a lunar surface. To realize these goals, an algorithm to detect craters which are the main hazardous feature on lunar is proposed. First, using the original image of craters on a moon surface, the author converts the RGB image plane to HSV image plane and analyzes only the Value parameter of a HSV plane. This output will then be used in ellipse reconstruction algorithm to get the orientation, of the disc (crater) from the camera projection. This is very useful information for the Lunar Lander as a first step before one can measure the position of the crater using the same algorithm.
Detecting Impact Craters in Planetary Images Using Machine Learning T. F. Stepinski1, Wei Ding,2 R. Vilalta3 1Dept. of Geography, Univ. of Cincinnati, OH 45221, USA. 2Dept. of Computer Science, Univ. of Massachusetts Boston, 100	Detecting Impact Craters in Planetary Images Using Machine Learning	In this Paper, they discussed two supervised machine learning techniques for crater detection algorithms (CDA): identification of craters from digital elevation models (also known as range images), and identification of craters from panchromatic images. The first class of methods rely exclusively on pattern recognition techniques to identify crater rims having circular	The algorithms presented here give reasonably accurate surveys of craters but can benefit from better training sets. They suggest that the most important challenge of CDA is to incorporate elements of transfer learning and/or machine learning to allow for efficient addition of training samples as the need arises.

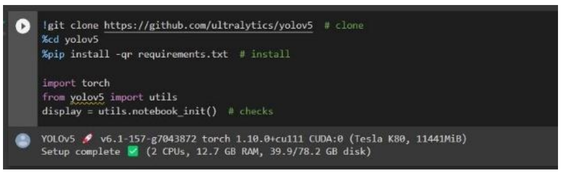
Morrissey Blvd. Boston, MA 02125-339, USA. 3Dept. of Computer Science, University of Houston, 4800 Calhoun Rd., Houston, TX 77204, USA		or elliptical features in an image. They presented applications of both techniques and demonstrate how such automated analysis has produced new knowledge.	

Methodology

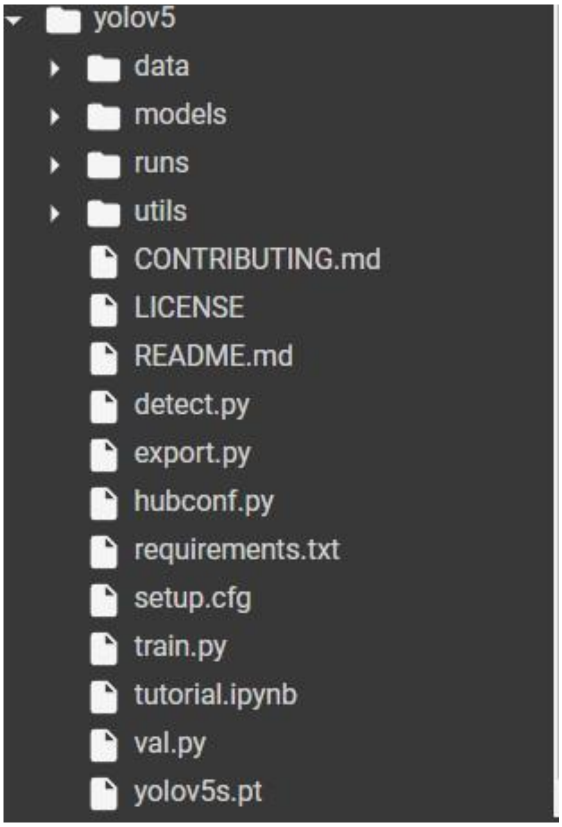
We Start with our manual annotation process using makesense.ai where we annotate each and every image's objects and export the annotations.



For this experiment the YOLO V5 predefined / pretrained is extracted and initialized.



After the execution of the previous cell, the runtime creates a library as shown in the figure below (google collab) which is nothing but the prerequisites for running our model.



As we have already Split the dataset into train and validation, we are specifying the path of train and validation located in our collab directory. The executable text which is being shown in the below figure is inside the yolov5 directory. We are modifying the number of classes and the name of the class according to our requirements which in this case is only 1 class (circular class) labelled/named as 'circle'.

coco128.yaml

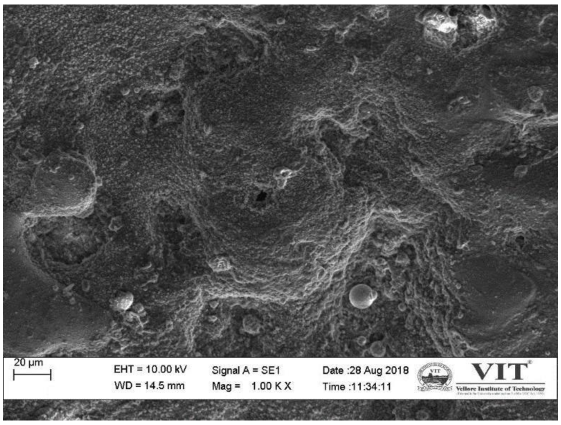
```
train: ../circles/images/train
val: ../circles/images/val


# Classes
nc: 1  # number of classes
names: ['circle']

```

The final step is training our model using YOLO V5 for a total of 150 epochs without specifying any call-backs or arguments. And parallelly checking the validation using our validation images which were defined in the beginning. The training of the images happens in a batch.

```
# Training for 150 epochs
!python train.py --img 640 --batch 16 --epochs 150 --data coco128.yaml --weights yolov5s.pt --cache
```

Dataset

The Circular Ellipse dataset has 29 images of the moon’s surface without annotations –



The images have then been split to 20 as train and the remaining 9 as validation for testing purpose. As an import phase of object detection, we have manually assigned the annotations using make sense AI and further generated the object coordinates for training our model.

Exp.no.1.txt

```
0 0.467742 0.462366 0.053763 0.071685
0 0.649866 0.703405 0.060484 0.073477
0 0.216398 0.715054 0.053763 0.064516
0 0.304435 0.260753 0.079301 0.102151
```

The associated coordinates of the objects in the first image. Four coordinates as there are 4 objects in the above image.

```
circles
    images
        train
        val
    labels
        train
        val
        train.cache
        val.cache
yolov5
```

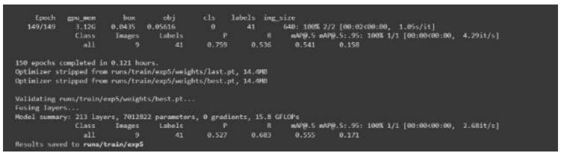
The overall data structure which includes the train validation split, labels which has the coordinates of the bounding box for both train and validation as described in the earlier images.

Techniques Used

Image processing filters like 2D, image sharpening and noise filters are used. We also used Python and its libraries like Matplotlib, NumPy, OpenCV etc. for this project. Google Colab IDE environment is used. For Computer Vision we will explore algorithms like YOLO.

Experimental Result

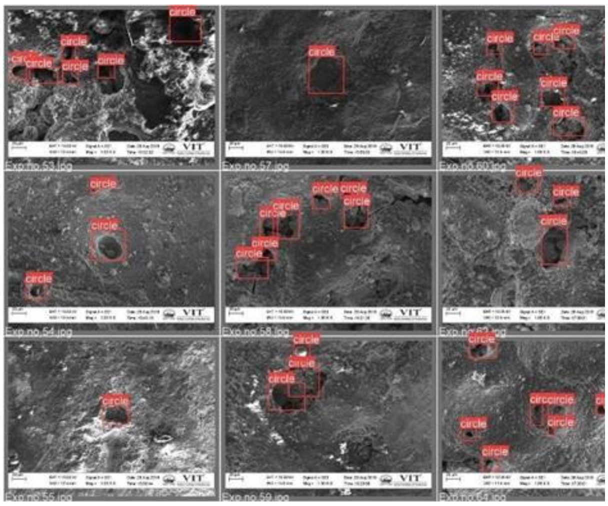
To evaluate object detection models like R-CNN and YOLO, the mean average precision (mAP) is used. The mAP compares the ground-truth bounding box to the detected box and returns a score. The higher the score, the more accurate the model is in its detections. So, for our model the mAP value is 0.555 which is 55%. Along with the loss recorded was 0.171 which is really low for a object detection model. The below image further shows all the other metrics relevant.



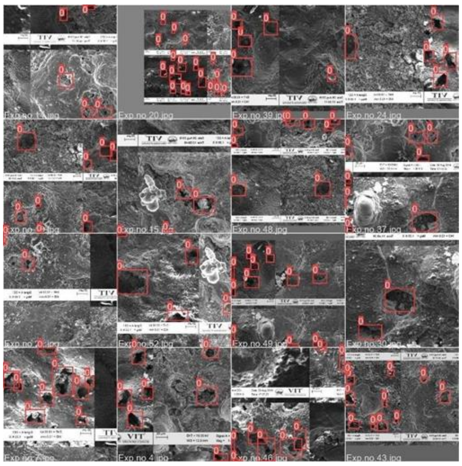
```
     Epoch   gpu_mem       box       obj       cls    labels  img_size
   149/149     3.126    0.0435   0.05616         0        41       640: 100% 2/2 [00:02<00:00,  1.05s/it]
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 1/1 [00:00<00:00,  4.29it/s]
                 all          9         41      0.759      0.536      0.541      0.158

150 epochs completed in 0.121 hours.
Optimizer stripped from runs/train/exp5/weights/last.pt, 14.4MB
Optimizer stripped from runs/train/exp5/weights/best.pt, 14.4MB

Validating runs/train/exp5/weights/best.pt...
Fusing layers...
Model summary: 213 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 1/1 [00:00<00:00,  2.68it/s]
                 all          9         41      0.527      0.683      0.555      0.171
Results saved to runs/train/exp5
```

The generated annotations of the images

Batch of 9

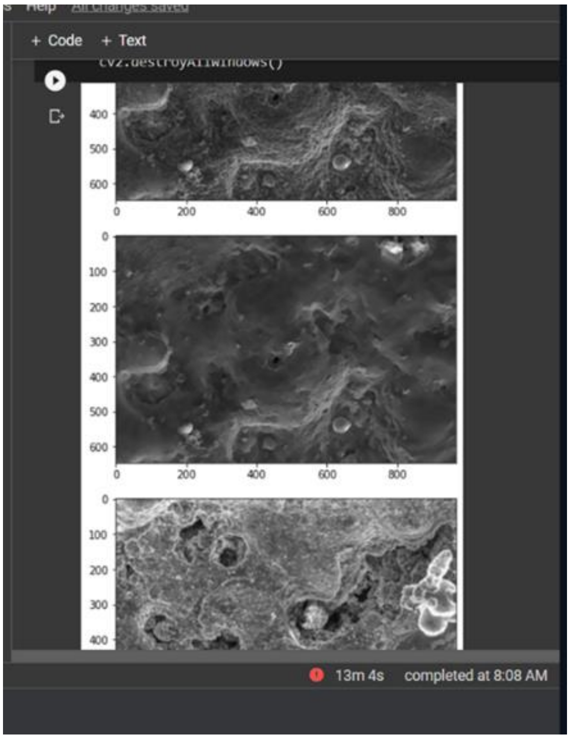


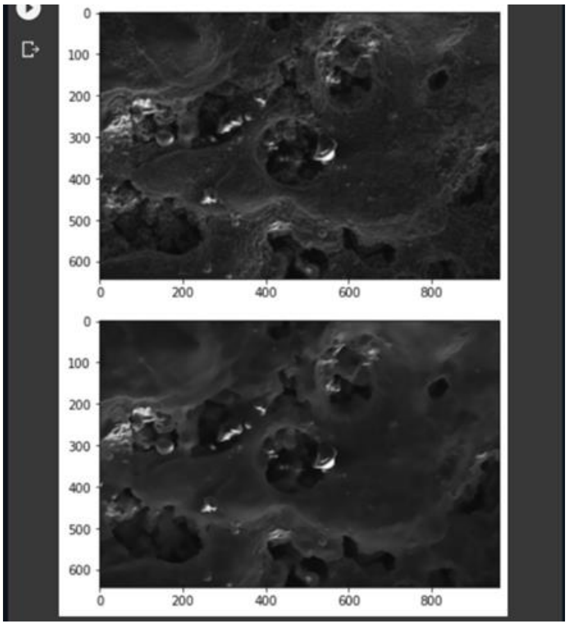
Batch of 16



For all the above images the red bounding box depicts the detected circular object and does it with high precision.

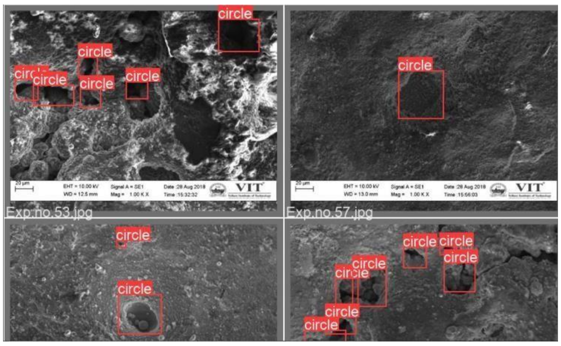
The generated images after applying image processing (noise filters)



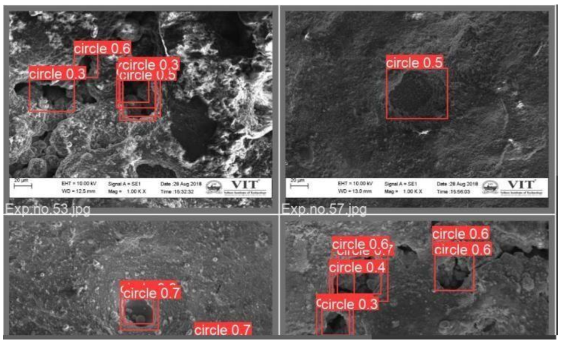


Conclusion And Future Work

Actual images with annotations



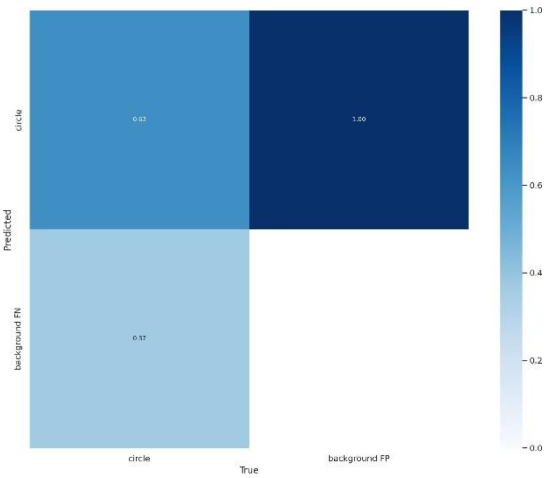
Predicted objects



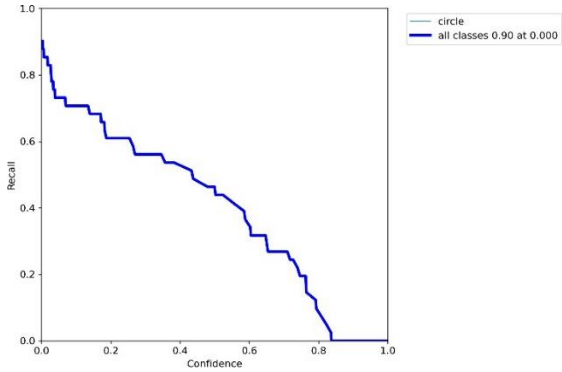
As the predicted boxes are more of less overlapping the annotated boxes, The IOU is high which says the model is well trained for detecting the desired objects.

Statistics of the Model

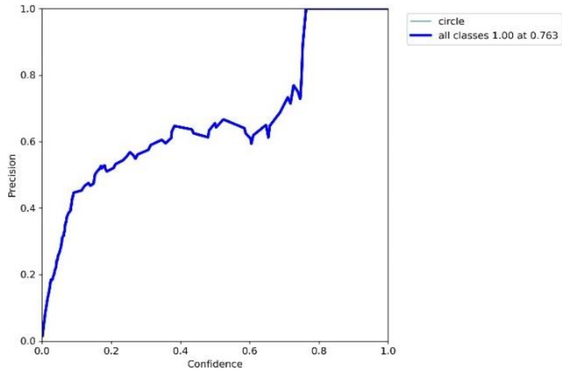
a) Confusion matrix



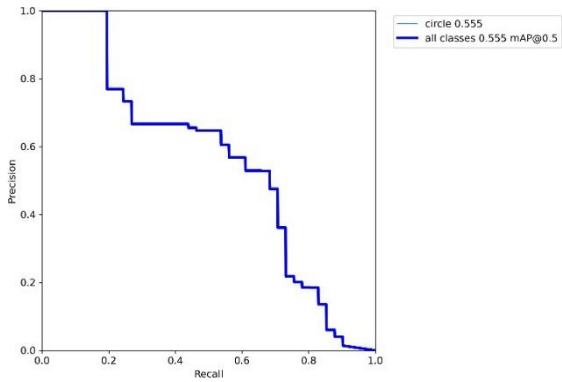
b) Recall vs Confidence graph



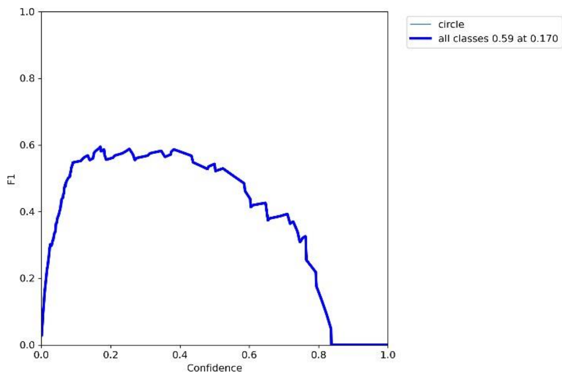
c) Precision vs Confidence graph



d) Precision vs Recall graph



a) F1 vs Confidence graph



The statistics shows that with every increasing epoch the model trains better and gives a higher overall precision making the YOLO the best in the business out of all the available object detection models. For the future work, these detections can be done by converting the real SAR image to usable image to find the number of craters.

References

1. https://pytorch.org/hub/ultralytics_yolov5/#:~:text=YOLOv5%20%F0%9F%9A%80%20is%20a%20family,Model
2. https://github.com/ultralytics/yolov53
3. https://towardsdatascience.com/the-practicalguide-for-object-detection-with-yolov5algorithm-74c04aac4843
4. https://medium.com/axinc-ai/yolov5-thelatest-model-for-object-detectionb13320ec516b
5. https://www.makesense.ai/