NATIONAL UNIVERSITY OF HO CHI MINH CITY
UNIVERSITY OF INFORMATION TECHNOLOGY

COMPUTER SCIENCE

PROJECT REPORT

COMPUTATIONAL THINKING

CS117.M22.KHCL

TOPIC: ROBOT IDENTIFYING AND COLLECTING PLASTIC BOTTLES AND CANS

INSTRUCTOR: **NGÔ ĐỨC THÀNH**

PROJECT MEMBERS: **BÙI QUỐC THỊNH – 20520934**
PHẠM THIỆN BẢO – 20521107
NGUYỄN HUỲNH HẢI ĐĂNG – 20521159
VŨ QUỐC THÁI BÌNH - 20521119

HO CHI MINH CITY, 6/2022

TABLE OF CONTENTS

Part 1. Introduction about the topic

1.1 Reasons for choosing the topic

In 2021, the globe expended 353 million tons of plastic rubbish, but recycled garbage amounted to just 9% of the total.

Viet Nam is one of the 20 countries with the largest amount of waste, significantly more than the global average. Approximately 19% of rubbish is destroyed, and almost 50% is buried in licensed landfills. Waste categorization is an essential and practical problem that can aid in environmental protection. We require a trash classification tool to lower the cost of categorizing trash at recycling plants, and it assists in collecting and categorizing rubbish as soon as feasible. Roboca is a robot that can auto-detect and classify two types of recycling trash - bottles and cans. It will be placed in public places such as parks, streets, playgrounds, and various other public areas.

1.2 Overview

The problem of real-time plastic bottles and can detection is related to computer vision and has numerous practical applications.

Our project aims to detect recycled garbage in real-time through a camera with a direct view, based on previous related scientific works.

The classes studied are “bottle” and “can”.

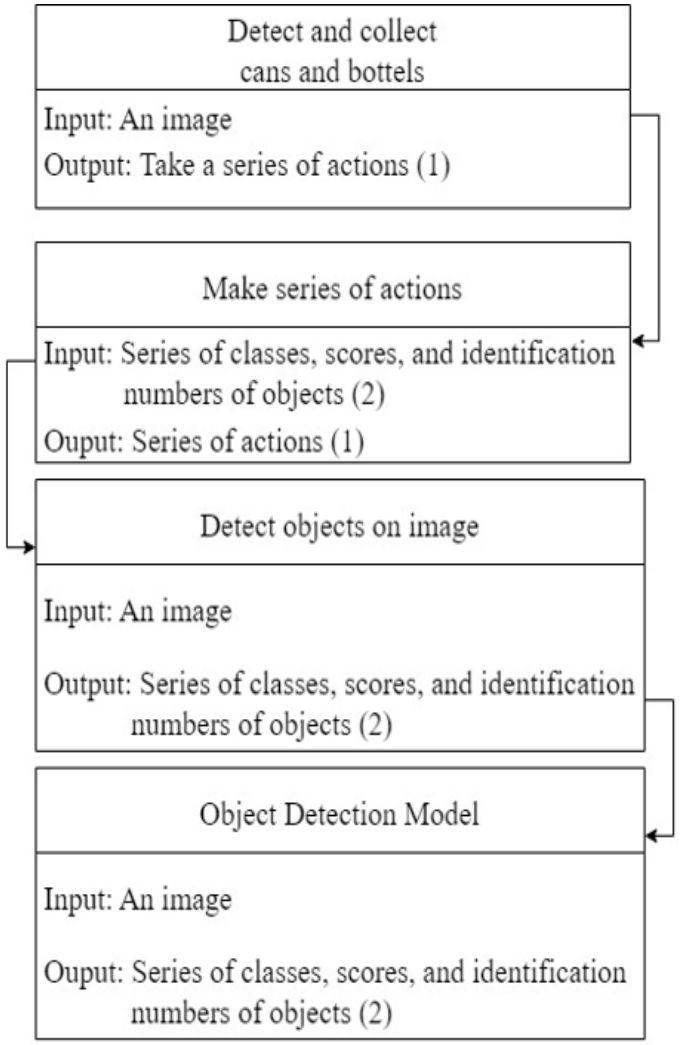
+ Input: An image that has many objects inside is extracted from a real-time camera.

+ Output: Take a series of actions for each object in the picture.

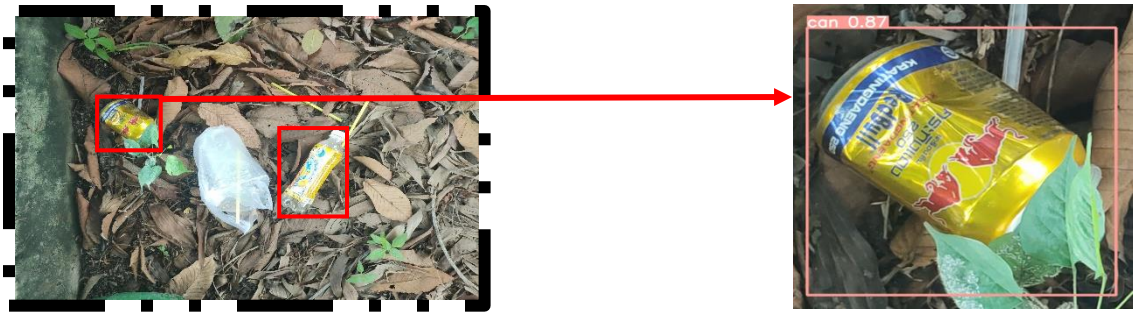
- If that object is a bottle: Pick up that bottle and place it in the bottle box.
- If that object is a can: Pick up that can and place it in the can box.
- If that object is not the above two types: Ignore it.

Part 2. Problem definition

2.1 Decomposition



Note		
No.	Name	Meaning
1	Series of actions	$[(x_1,y_1),(x_2,y_2),...,(x_n,y_n)]$ x is an integer number which 1 means collect and 0 means ignore. y is an integer number in which 0 means None, 1 means can class, and 2 implies bottle class.
2	Series of classes, scores and identification numbers of objects	$[(x_1,y_1,z_3),(x_2,y_2,z_3),...,(x_n,y_n,z_n)]$ Where x is an integer number which means None, 1 means can class, and 2 implies bottle class y is a real number being not negative which stands for the confident score z is an integer number greater than minus 1 which stands for the identification numbers of objects



Bounding box = (pc, bx, by, bh, bw, c)

pc: confidence score of class c

bx: x-center of the bounding box

by: y-center of the bounding box

bh: bounding box height

bw: bounding box width

c: class of object predicted

2.2 Constraints

It can only detect and classify plastic bottles and cans, others are ignored.

Objects can only be detected in well-lit conditions.

Objects do not stack on top of each other.

The robot can avoid humans and big obstacles ahead.

Before detecting, scale the input image to size 640x640.

Threshold 0.7 for confidence score's bounding box.

Requirements of a bounding box for each object.

Angle of view at least 1 meter away from the object and no more than 2 meters.

2.3 Abstraction

From the decomposition tree above, we can abstract unnecessary details and information.

The robot is responsible for detecting cans and plastic bottles. Then classify them, pick, and put them in the right waste basket.

Therefore, we can abstract the pick and put plastic bottles and cans in the corrected waste basket.

Now, the main problem is to detect and classify plastic bottles and cans.

2.4 Pattern recognition

After abstraction, we can see that now the problem is much easier.

This problem is similar to other object detection and object classification in the past.

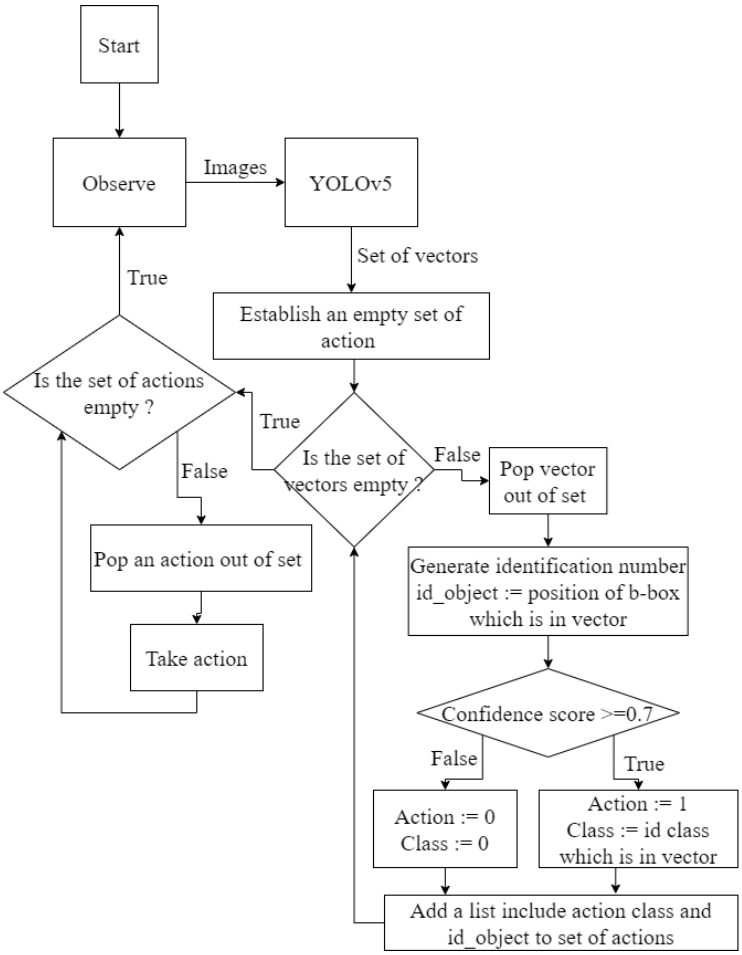
We will use the mapping technique to apply for this problem.

Part 3. Algorithms

3.1 Algorithm explanation

To solve the Object Detection problem, we need to choose a model to implement it. We used Yolov5 models to study and experiment on the above topic.

We chose these two methods because we tried before with other supporting methods such as RCNN, Fast RCNN, Faster RCNN, etc., but failed. The reason is that the image dataset prepared to serve the training of the input model is quite large as well as the training time is too long, exceeding the running time of Colab and the computer's RAM, leading to memory overflow during the learning process. Therefore, we switch to the Yolov5 model approach to match the capabilities of the computer and the prepared dataset.



3.2 Bounding box

In object detection, we often use the bounding box to describe the position of the object in the image. A bounding box is a rectangle, defined by the x coordinate value of the upper left corner of the rectangle and the y coordinate value of the bottom right corner. Another commonly used bounding box representation is (x center, y center) - the coordinate axes of the center of the bounding box, width, and height of the box.

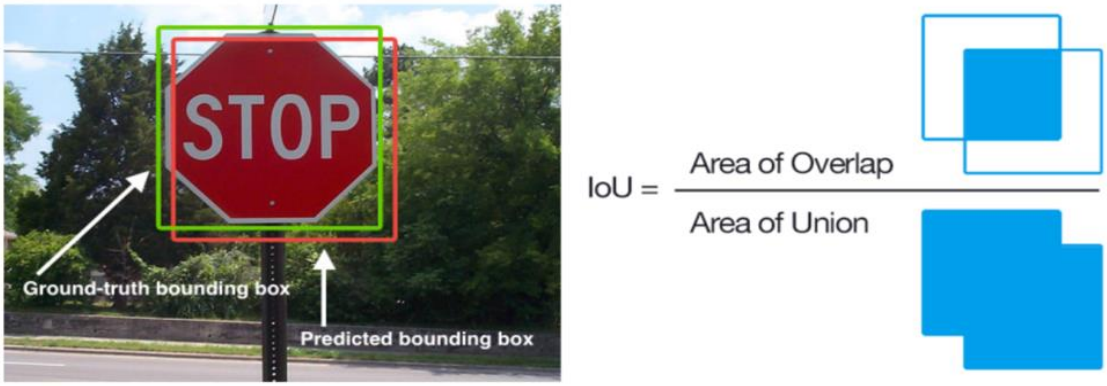
+ Predicted bounding box is the bounding box used in model detection, showing the model's object prediction.

+ Ground Truth: is the initial bounding box labeled by the user to perform training.

3.3 IoU

Intersection over Union (IOU) is an evaluation metric used to measure accuracy in object detection on a particular data set. Intersection over Union is simply a measure of evaluation. Any algorithm that provides predicted bounding boxes as output can be evaluated in IoU.

In short, it is used in evaluating whether the bounding box that predicts the object matches the actual ground truth of the object. The IoU index is in the range [0,1] and the closer the IoU is to 1, the closer the predicted bounding box is to the ground truth.



Hình 13.10: Ví dụ về IoU [11]

3.4 Confidence score

Confidence score is the probability that model object detection predicts that object. The value is intended to determine whether the model correctly detected the object, as well as whether the model's prediction was effective. Through the value of the confidence score, we can adjust the training model, align the IOU value, accordingly, prepare more datasets, etc.

3.5 YOLO model

In the past few years, object detection has become one of the important topics of deep learning because of its high applicability, easy-to-prepare data, and numerous application results. New object detection algorithms can perform seemingly real-time tasks, even faster than humans, without sacrificing accuracy. Among them, YOLO - You Only Look Once may not be the best algorithm, but it is the fastest in the class of object detection models. The versions of this model all have very significant improvements after each version.

Object Detection algorithms are divided into two main groups:

- The family of RCNN (Region-Based Convolutional Neural Networks) models for solving problems of positioning and object recognition.
- The family of YOLO (You Only Look Once) models for object recognition are designed to recognize objects in real time.

The YOLO architecture includes base networks are convolution networks that perform feature extraction. The back part is the Extra Layers applied to detect objects on the feature map of the base network.

YOLO performs the following steps:

- Step 1: Divide the image into G×G grid cells.
- Step 2: For each grid cell, run a CNN that predicts the bounding boxes in that cell. The object's center of gravity will be found in the grids and if it is in any grid cell, the grid cell containing the object's center of gravity will be responsible for finding that object.
- Step 3: Run non-max suppression algorithm.

Steps of non-max-suppression:

- Step 1: First, we will find a way to reduce the number of bounding boxes by filtering out all bounding boxes that have a probability of containing an object less than a certain threshold (threshold), usually chosen as 0.5.
- Step 2: For intersecting bounding boxes, non-max suppression will select a bounding box with the highest probability of containing the object. Then calculate the IoU interference index with the remaining bounding boxes. If this index is greater than the threshold, it means that the ratio of 2 bounding boxes is very high. We will remove the lower probability bounding box and keep the highest probability bounding box. Finally, we obtain a unique bounding box for an object.

3.6 YOLOv5 model

Yolov5 is the product of author Glenn Jocher - researcher, and CEO of Ultralytics. This is an organization that aims to help AI learners in general and technology enthusiasts, in particular, have access to machine learning models in a simpler, more efficient, and intuitive way.

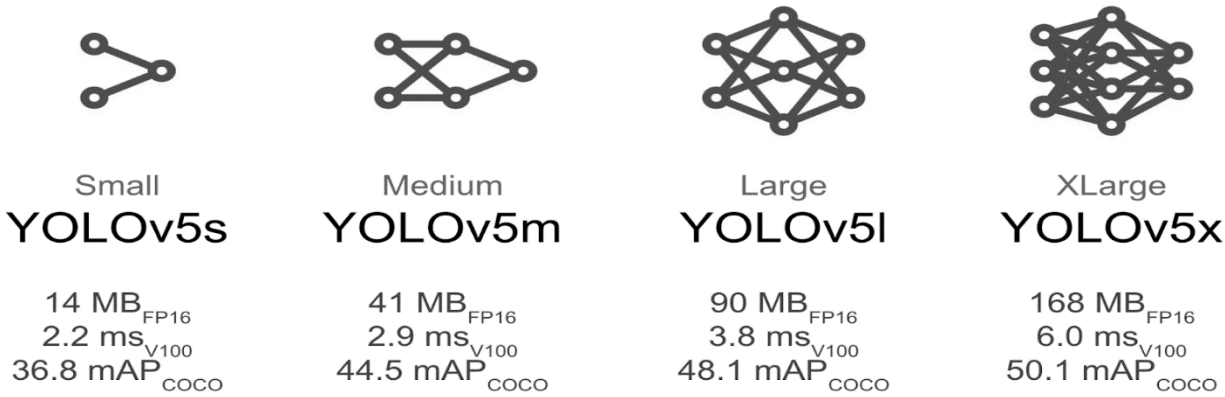
The model is also based on Yolo architecture and uses algorithm optimization strategy in a convolutional neural network, such as customizing anchor box size with each dataset type, using mosaic data augmentation (each input image is a combination of 4 images to make the context of the image richer), CSPNet (keeping part of the information from the previous layers, while reducing the complexity of the model), ...

In addition, the model uses PyTorch and is released on GitHub. Initially, when it was released, it was controversial due to the lack of significant improvement over Yolov4 resulting in no official scientific paper for this model. However, the larger Pytorch user community than Darknet leads to the use of this framework which makes it easy to install and integrate on IoT devices. Therefore, it is accepted by everyone to this day.

The structure of Yolov5 includes 4 main parts: Input, Backbone, Neck, and Head.

- Input: mainly contains data pre-processing, including mosaic data augmentation.
- Backbone: CSPDarknet53 => This new structure helps to increase the learning capacity of the CNN network, reduce the computational volume, and reduce the memory cost.
- Neck: avoid loss of information during bottom-up, and top-down during layer reconstruction.
- Head: Using Transformer encoder block, increase the ability to distinguish features, and predict class and bounding box. Consists of 2 floors:
  + Dense Prediction: helps predict the entire model, locate the bounding boxes => find the area that is likely to be an object.
  + Sparse Prediction: areas that are likely to continue to be predicted => return the final prediction result.

Choosing a pretrained model to start training will make the training faster, the model can then be further trained to fit the actual data set or used directly in machine learning problems.



Part 4. Evaluation

4.1 Precision – Recall

Based on a threshold confidence score during training (threshold) to determine true detection, false detection. Usually choose 0.5.

- True Positive (TP): IoU greater than or equal to the threshold, is a correct detection.
- False Positive (FP): IoU is less than threshold, is a wrong detection.
- False Negative (FN): case where the ground truth does not have a predicted bounding box.

Precision is the ratio of True Positive (TP) predictions to the total number of positive predictions.

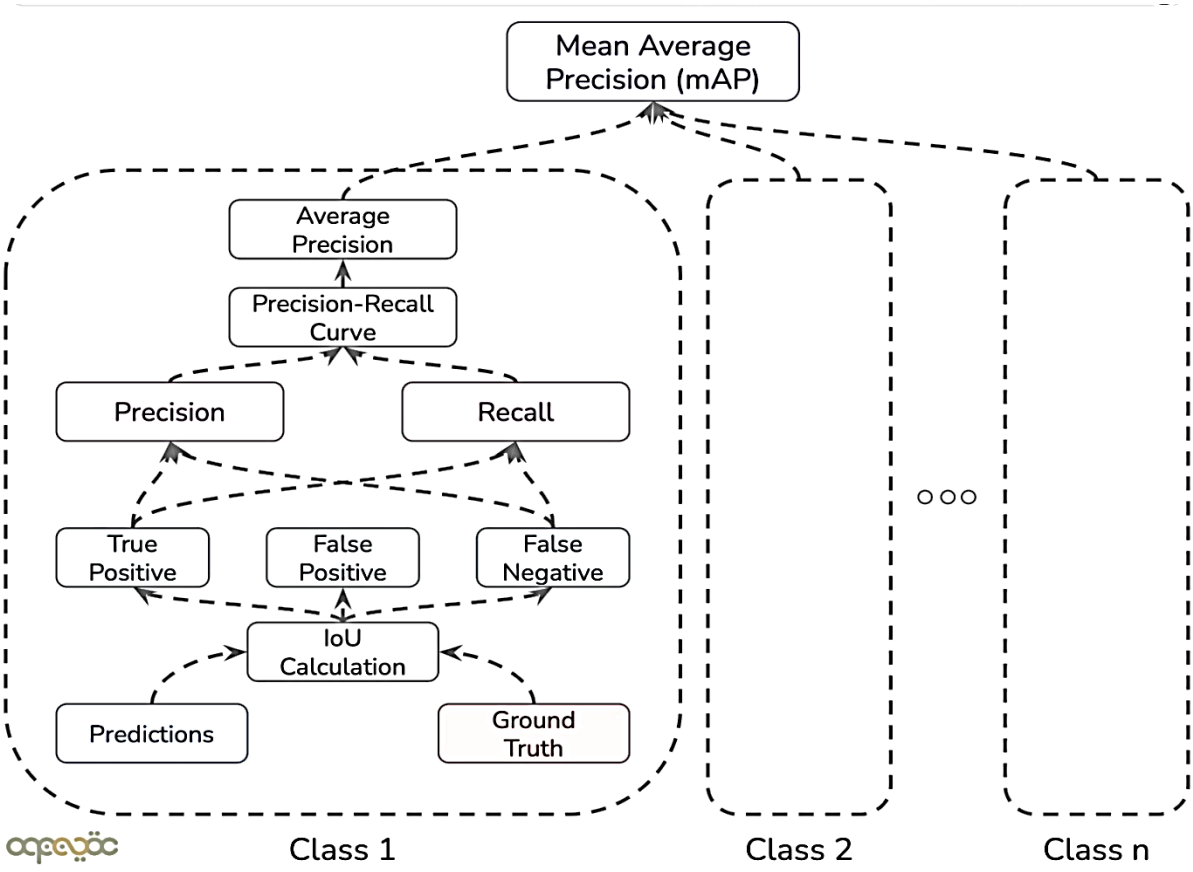
Recall is the ratio of the number of predicted True Positives out of the actual positives.

In summary, we have the formula:

Precision = TP / (TP + FP) = Number of corrected predictions / Total number of predictions

Recall = TP / (TP + FN) = Number of corrected predictions / Number of possible corrected guesses

4.2 AP



When the training is over, we will get the prediction results of each object in the image. Through the IOU calculation process to measure the prediction accuracy, we can calculate the TP, FP, and FN values. From there, it is easy to calculate the parameters of Precision and Recall. These two values are intended to draw a Precision - Recall Curve chart and apply a calculation formula to find the AP for each class.

4.3 mAP

Our Object Detection problem has one or more classes, each class we will measure AP, then average all AP values of the classes, then we find the mAP index of the model. Therefore, mAP is understood as the average value of all classes.

- mAP@.5: means average mAP when selecting IoU = 0.5

  For example: mAP@0.5 = 0.7 –> At IoU = 0.5, the AP of the model is 70%.

- mAP@ [.5:.95] means the average mAP over different IoU thresholds, from 0.5 to 0.95, 0.05 increments.

It is common to choose the IoU interval from [.5:.95] because it is difficult for the predicted bounding box to match the actual ground truth of the object, leading to the result being always wrong even though the model predicted almost the same. object accurately.

Part 5. Making the dataset

5.1 Building the dataset

To implement the topic, we have built our own dataset and have constraints, the specific information is:

Number of photos: 1267 color photos.

Dimensions: 918 x 1224 → 4000 x 3000

Objects in the picture:

- Cans and bottles of many brands.
- More than 1200 bottle labels, more than 1500 can labels.
- About 20% of bottles and about 70% of cans are slightly deformed.
- 2% cans, bottles are heavily deformed.
- Cans, bottles are not stacked, horizontal or vertical.

Brightness: The light is enough to see the object clearly, without glare.

Background:

- Road brick foundation, dry leaf yard, grass floor, roadside, under tree...
- About 30% of photos have foreign objects: garbage, cigarette packs, plastic cups, foam boxes, straws, plastic bags, logs...

Shooting angle: top-down view, about 1m away from the object, not more than 2m.

Train: 1000 photos.

Validation: 267 photos.

Test: 50 photos + 1 video (built specifically to test the effectiveness, just change the background and the object and constraints remain the same).

Example images:

Normal:

Slight deformation

Severe deformation

5.2 YOLO annotation

YOLO versions from v1 to v5 when training all require a separate annotation format for the dataset.

Purpose: Yolo annotations help show the ground truth of the objects in each image before putting it into the training model.

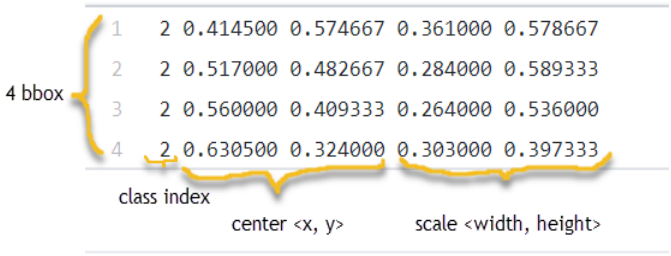
Tools used: MakeSense.AI – a website that supports labeling.

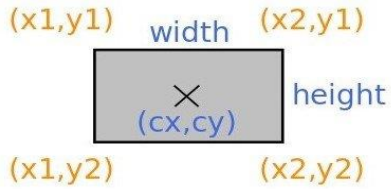
The content of the file is in txt format, showing the following parameters:

<id-class> <center-x> <center-y> <width> <height>

- id-class: Integer from 0 to number of classes - 1. Each integer corresponds to 1 class.
- center-x: x center of the bounding box.
- center-y: y center of the bounding box.
- width: The width of the bounding box.
- height: The height of the bounding box.

The values center-x, center-y, width, height are all normalized to the range [0, 1]. The purpose of creating the above values is to help scale the size of the object compared to the image before entering the learning model.





Part 6. Implementation

6.1 Result

Result on Validation set:

- Apply Pretrain YOLOv5s:

```
Model summary: 290 layers, 20856975 parameters, 0 gradients, 48.0 GFLOPs
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 17/17
                 all        266        650      0.976      0.948       0.98      0.855
              bottle        266        185          1      0.895      0.973      0.854
                 can        266        465      0.952          1      0.986      0.856
Results saved to runs/train/exp
```

- Apply Pretrain YOLOv5m:

```
 Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 34/34 [00
   all        266        650      0.953      0.968      0.983      0.855
bottle        266        185       0.97      0.935      0.978      0.852
   can        266        465      0.937          1      0.988      0.859
```

Result on Test custom:

- Apply Pretrain YOLOv5s:

```
test: Scanning '/content/drive/MyDrive/ComputerVision_Project/Test/labels.cache' images and labels
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 2/2 [
                 all         50         97      0.911      0.867      0.939      0.748
              bottle         50         50       0.95       0.82      0.942      0.742
                 can         50         47      0.872      0.915      0.936      0.754
```

- Apply Pretrain YOLOv5m:

```
test: Scanning '/content/drive/MyDrive/ComputerVision_Project/Test/labels.cache' images and labels
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 2/2 [
                 all         50         97      0.901      0.913      0.958      0.807
              bottle         50         50      0.934      0.848      0.968      0.814
                 can         50         47      0.868      0.979      0.947      0.799
Speed: 0.2ms pre-process, 10.8ms inference, 1.7ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp2
```

6.2 Demo

You can click here to see the demo of this solution.

6.3 Conclusion

To summarize, the can and plastic bottle identification demonstrated an allowable output, and the trained model may be used in the real-time video with no difficulty. The solution can handle occluded and tiny things rather well. Nonetheless, in certain cases of recognizing cans and tiny bottles, the model is still muddled up between the two and displays incorrect responses even though it can detect them. Our team will continue to master and bridge the current challenges.

Part 7. Preferences

[1] Redmon, J. and Farhadi, A. (2018) YOLOv3: An Incremental Improvement. Computer Science, arXiv: 1804.02767.

http://arxiv.org/abs/1804.02767

[2] YOLO you only look once real time object detection explained - Manish Chablani

[3] YOLO object detection YOLO - forum machine learning cơ bản

[4] YOLO, YOLOv2 - Jonathan hui

[5] You Only Look Once: Unified, Real-Time Object Detection - Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi

[6] YOLO9000: Better, Faster, Stronger - Joseph Redmon, Ali Farhadi

[7] YOLOv5 Improved YOLOv5 Based_on Transformer Prediction Head for Object - ICCVW_2021

https://openaccess.thecvf.com/content/ICCV2021W/VisDrone/papers/Zhu_TPH-YOLOv5_Improved_YOLOv5_Based_on_Transformer_Prediction_Head_for_Object_ICCVW_2021_paper.pdf?fbclid=IwAR1xVO_v_m57tgToewuQ7F33NE3rhiPIVT7JPbMoEcdy40Ol0JMzP1oTHGE

[8] Stanford University: Cheatsheet convolutional neural networks

https://stanford.edu/~shervine/l/vi/teaching/cs-230/cheatsheet-convolutional-neural-networks#:~:text=T%E1%BA%A7ng%20t%C3%ADch%20ch%E1%BA%ADp%20(CONV)%20T%E1%BA%A7ng,feature%20map%20hay%20activation%20map

[9] A DEEP LEARNING OBJECT DETECTION METHOD FOR AN EFFICIENT CLUSTERS INITIALIZATION

https://arxiv.org/pdf/2104.13634.pdf