Title: Computer Vision Based Object Classifier Robotic Arm for Recycling Plant

Prepared by: Natnael Getachew----------ATR/0382/10

Leul Mengistu----------ATR/9127/10

Abstract

Waste is unavoidable bi-product of human civilization and life. Managing waste produced from household processes to larger industries efficiently is important. Among several methods of dealing with the waste produced from different parts of the world is recycling. Recycling is known to be an efficient way to deal with wastes as it uses waste materials as input to recreate the original material or produce different material form the waste. Even though recycling is efficient way of dealing with waste materials it has several stages in its entire process that can be optimized.

Manual classification of objects in a recycling plant relies on plant personnel who visually identifies objects and manually pick up the object to sort it as the object moves on a conveyer belt. We plan a solution to the manual classification and sorting of objects by replacing them with a robotic system that identifies and classified objects using computer vision and sort them with a robotic hand. A camera will be used as a sensor for the computer vision part where the output of the computer vision system triggers the robot to make movements to sort the object on the conveyer belt in its right category.

Table of Contents

List of Figures

1. Introduction

Waste is an unavoidable by-product of most human activity. Most human activities generate waste [1]. Economic development and rising living standards in the world have led to increases in the quantity and complexity of generated waste. Developed countries generate much higher quantities of waste per capita compared to the developing countries. As the quantity of waste one nation produces increase so does the type or variety of the waste [2]. However, in some circumstances the management of even small quantities of waste is more challenging. One way of managing wastes is through recycling.

Recycling is any recovery operation by which waste materials are reprocessed into products, materials or substances for the original or other purposes.

Recycling involves altering the physical form of an object or material and making a new object from the altered material [3]. Recycling is widely known and mostly practiced waste reduction system. The overall recycling process involves sub-processes after waste generation including collection of wastes, sorting of collected waste, processing of the sorted waste and finally delivering the final recycled product.

Recycling saves resources from being squandered by reusing the resources that are already used before. Recycling 1 ton of newspaper saves the equivalent of 17 trees and producing a new aluminum can from a recycled aluminum can reduces production energy requirements by 75% [4].

In recycling plants, the collected waste will start its recycling process with classification and sorting. Classification of waste is the process of categorizing wastes based on defined categorizing rule and sorting is placing the classified waste material in its correct place.

Computer Vison based Object Classifier Robot in this project is designed to classify wastes depending on their type by using computer vison and a robot arm to sort the classified waste and optimize the recycling process.

2. Problem Statement

Recycling plants get their input materials in bulk and unorganized manner. The raw materials are classified and sorted by manual labor which is proved to be inconsistent and unreliable. Classification and sorting of the raw materials in recycling plants will reduce later costs of dyeing coloring and related costs.

We aim to build an automatic robotic system that uses computer vision to classify the materials that are input to the recycling plant organize the classified objects in their right category. We believe this will provide the solution to the problems of cost optimization and manual labor-based classification and organization of recyclable materials for recycling plants.

3. Objective

3.1. General Objective

The general aim of this project is to build a Robotic system that is used to sort objects in an automated way using a Robotic arm and computer vision.

3.2. Specific Objective

The specific aims of the project include:

- Design and build a robotic arm.
- Lifting materials using the robotic arm
- Selecting the appropriate AI model for the computer vision part of the project
- Using computer vision identify an object.
- Connect the two components so that the robotic arm can pick and place the object using the computer vision

4. Literature Review

In this project, traditional to state-of-the-art classification and sorting techniques are explored. Traditional methods that use manual sorting and mechanical sorting in which classification and sorting of materials from solid wastes are implemented using human labor or mechanical system and recent ways of computer vison techniques that are used for classification of materials for solid wastes are investigated.

4.1. Traditional Techniques

Traditional techniques used for classification of material from solid wastes are of two types, manual sorting and mechanical sorting.

4.1.1. Mechanical Sorting

Mechanical sorting uses material physical characteristics such as density, magnetic susceptibility or stiffness as a classification and sorting of materials from the whole waste [5]. Mechanical sorting before they are classified and sorted, they are shredded into smaller parts and then further classification and sorting will be applied on the shredded waste using air separators ballistic separators, and magnets [6].

4.1.2. Manual Sorting

Manual sorting uses human labor for classification and sorting of materials from a waste for recycling purpose. In developing countries manual labor is cost effective compared to automated systems [7]. It is believed that the unique ability of humans to recognize materials in a very short time allows manual sorting for co-mingled recovery of material efficiency of 95% [8]. Although humans have this unique ability of very fast recognition of materials which is used in classification of material with in a waste for recycling, developed nations have restricted manual sorting due to limiting factors such as human fatigue and exposition to hazardous materials.

4.1.3. Automated technique

Automated technique of classification and sorting of waste materials uses artificial intelligence with computer vision for classification of material. Computer vision is a type of artificial intelligence that allows computers systems to extract useful information from digital images, videos, and other visual inputs, as well as to take actions or make recommendations based on that data. If AI allows computers to think, computer vision allows them to see, observe, and comprehend the world around them [9].

Computer vision aims to teach robots how to see and perceive pictures in the same manner that humans do. This is no simple undertaking, given the intricacy of our eyes, let alone the processing our brains do with these pictures [9].

4.2. Real-time object detection

Real-time object detection is an ongoing area of research in computer vision. It is the process of detecting or discovering objects in real-time, such as a car, a person, a water bottle, and other items, using quick inference while retaining a high degree of accuracy [10]. It's employed in a variety of applications in the current culture, including security, self-driving automobiles, industrial quality control, and surveillance.

There are different ways of doing object detection such as:

- Viola jones object detection
- Scale-invariant features transform (SIFT)Histogram of oriented gradients (HOG) features
- Single Shot MultiBox Detector (SSD)

Viola jones object detection

The Viola-Jones object detection framework was introduced by Paul Viola and Michael Jones in 2001 as an object detection framework. It's a method that isn't based on neural networks which means we have to first define the features using methods such as the support vector machine technique for classification. Although it may be trained to recognize several object classes, the challenge of face detection was the driving force behind it.

Selecting Haar-like features, constructing an integral picture, running AdaBoost training, and creating classifier cascades are the four primary phases in the Viola-Jones approach [11].

The method examines numerous tiny subregions of a picture and attempts to discover a face by looking for certain traits in each subregion. Because a picture might have numerous faces of varied sizes, it must verify many distinct locations and scales. To detect faces, Viola and Jones employed Haar-like characteristics. This method was not used for this project because we can get much faster and more reliable results from other algorithms such as Convolutional neural networks or MobileNet architecture which are discussed in a later section.

Scale-invariant features transform (SIFT)

SIFT is an algorithm that is commonly used for detecting features within images. invented by David Lowe in 1999 It is often used because like the title suggests, it works for objects in images of multiple scales, making it practical to use. this is also a method that isn't based on neural networks. The way SIFT works is that it takes key points from another algorithm and finds patches of pixels around those given key points [12]. These patches consist of pixel gradients within the image. These gradients are then used to create orientation histograms - also known as key point descriptors [13]. These orientation histograms contain important aspects of the patches of pixels. These descriptors achieve robustness against contrast changes and rotations.

4.2.1. Single Shot MultiBox Detector (SSD)

SSD is a neural network-based object detection approach. which means we don’t have to define features. in this approach because they are based on convolutional neural networks. Before we proceed with the concept of SSD let's see what a convolution neural network is.

Convolutional Neural Networks

A convolutional neural network (CNN) is a widely used image analysis and classification tool [14]. These neural networks simulate the human brain to recreate the learning process.

These networks are a big part of computer vision, and they're a good approach to provide machines with identifying skills. CNNs vary from traditional neural networks in that they contain convolutional layers. These convolutional layers can recognize patterns in a picture. There are a specific number of filters or kernels that must be specified and defined for each convolutional

layer. The top layer uses basic filters to extract characteristics and shapes, whereas the deeper layers are more precise and are used to recognize particular objects or regions in a picture [15].

Filters, also known as kernels, are made up of n by m matrices that are randomly initialized. When a filter is given an input, it slides over the input over an n by m group of pixels until it has passed through all of the pixels in the picture. This is referred to as convolving.

Convolution is the process of multiplying individual pixels in an image with their corresponding pixels in a filter and then adding the results until the full set of pixels is summed up. The resultant pixel is then placed in its proper spot inside the convolved picture.

For example, we can see below a 5 by 5 image matrix being convolved with a 3 by 3 kernel or filter.

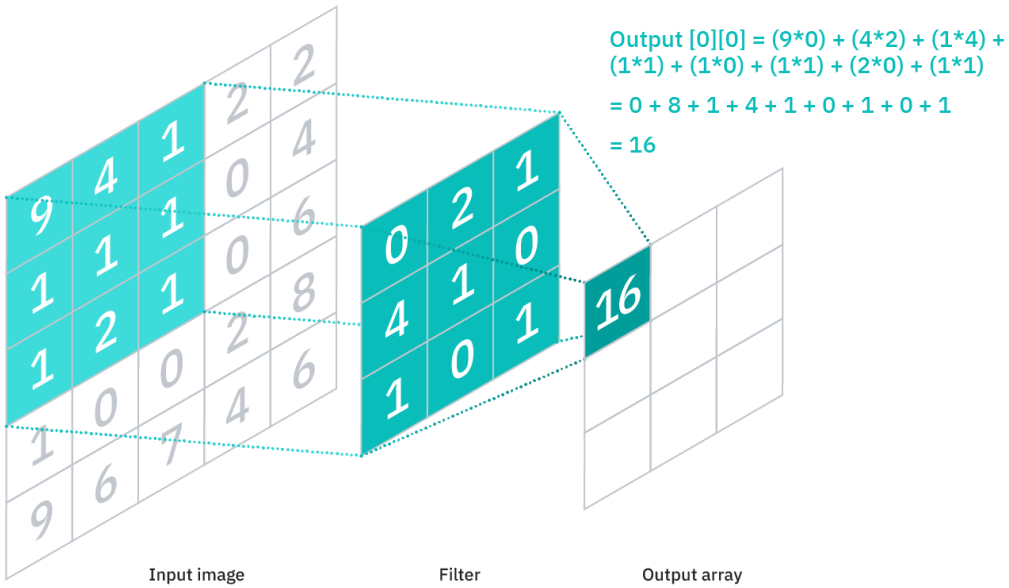


Figure 1: Convolution of 5 x 5 matrix with a 3 by 3 kernel

The network can recognize patterns and characteristics inside pictures as a consequence of convolution. After the image goes through every layer in the network, the object should be identified and classified [15].

The SSD architecture is a convolution network that learns to predict bounding box locations and classify these locations in one pass. Hence, SSD can be trained end-to-end. The SSD network consists of base architecture followed by several convolution layers [17].

The architecture is as shown below

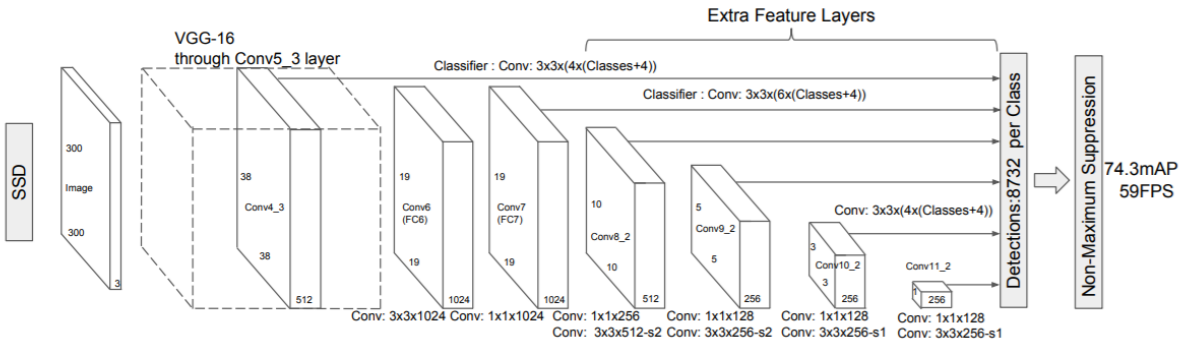


Figure 2: SSD model architecture

The SSD architecture has three main components that are future extraction, the detection heads for the detection box, and non-max suppression [17].

SSD typically extracts features using an Auxiliary network (base network). The method in the figure above employs VGG to extract feature maps. VGG16 (also called OxfordNet) is a convolutional neural network architecture named after the Visual Geometry Group from Oxford, which developed it, and the 16 in VGG16 refers to it having 16 layers that have weights [18]. However, the last several layers of VGG, such as max pool, FC, and Softmax, are ignored, and the VGG output is utilized as feature maps on which detections are made [17].

In SSD More convolution layers are added in which the intermediate tensors are kept so that a stack of feature maps with a variety of sizes are generated to make a detection. Let us assume, that we have a feature layer of size n x m and we have c channels. Then the convolution (mostly 3 x 3) is applied on the n x m x c feature layer. So, there are k bounding boxes possibilities for each location of the objects recognized, each with a probability score [17].

Finally, Non-max suppression is employed to ensure that an item is surrounded by just one bounded box. It accomplishes this by: At the start, any bounding boxes surrounding objects with a probability less than a given threshold, say 0.7, are removed. Then, for each item, the box with the highest probability factor is identified and all other boxes are suppressed so we save the one

with the highest probability factor. As a result, there is just one bounded box around a single recognized item [17].

4.2.2. YoloV5

Object detection is a concept of locating objects in an image [20] and object detection is able to provide valuable information for semantic understanding of images and videos, and is related to many applications, including image classification, human behavior analysis, face recognition and autonomous driving [21]. To detect and classify recyclable wastes we used deep learning mechanism where layers of neural networks are used to detect and classify the type of the recyclable wastes found in an image.

YOLO V5 also known as “You only look once” is an object detection algorithm that divides an image into grid system in which each grid is responsible to detect an object within itself. YOLO V5 is a single stage object detector meaning it is part of a class of object detection models which are one-stage, that is models which skip the region proposal stage of two-stage models and run detection directly over a dense sampling of locations. YOLO V5 does faster inferencing with lower performance because of being one-stage model.

YOLO V5 has mainly three parts, these include:

- Model Backbone
- Model Neck
- Model Head

Model Backbone

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.

Model Neck

Model Neck is mainly used to generate feature pyramids. Feature pyramids help models to work properly 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. YOLO V5 uses PANet as neck to get feature pyramids.

Model Head

The 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, scores for being an object, and bounding boxes.

Activation Functions

An activation function in a neural network defines how the weighted sum of the input is transformed into an output from a node or nodes in a layer of the network. YOLO V5 uses Leaky ReLU activation function in middle/hidden layers and the sigmoid activation function in the final detection layer.

4.3.HSV COLOR SPACE

Color space is a mathematical model to represent color information as three or four different color components. Color space explains how the colors are represented and specify the components of color space accurately to learn how each color spectrum looks like [19]. Hue is an angle from 0 to 360 degrees usually 0 is red, 60 degrees is yellow, 120 degrees is green, 180 degrees is cyan, 240 degrees is blue and 300 degrees is magenta. Hue indicates the type of color (such as red, blue or yellow) or the hue of the color where the color is found in the color spectrum. Red, yellow and purple are words that indicate hue. The second component of HSV color space is saturation. the saturation of a color is a measure of how pure the color is. Saturation is usually valued from 0 to 1 and shows a grayish color value where 0 indicates gray and 1 indicates pure primary color. The third component of HSV is value or intensity, which is a measure of how much brightness is a color or how much light comes from a color. value can be valued from 0 to 100%.

4.4. MQTT

MQTT is very useful for connections with remote client where a small code size is needed, or internet bandwidth is very low. For example, it has been used in sensors communicating to a broker using satellite link or in a range of home automation and small devices. Also, ideal for mobile applications because of its low power usage, small size, minimized data packets and well distribution of information to one or many receivers.

MQTT Protocol was first designed in 1999, but with the growth of the IoT, and the need to communicate between low-powered devices, MQTT has recently found its market. MQTT was designed with ability to run in an embedded environment where it would reliably and effectively provide an avenue for communication.

MQTT protocol uses a publish/subscribe architecture whereas HTTP uses its request/response architecture. MQTT protocol is event driven and enables messages to be pushed to clients. The heart of MQTT protocol is the MQTT broker, and it is responsible for dispatching messages between the senders and receivers. Every client that publishes a message to the broker includes a topic into that message. The topic is routing information for a broker. Each client that wants to receive messages need to subscribe to a certain topic and broker delivers all messages with the matching topic to a particular client. The clients don't need to know each other, they only communicate using the topic over MQTT broker.

MQTT uses the client/server model. Every device that is connected to a server, using TCP known as (broker) message in MQTT is a discrete chunk of data and it is ambiguous for the broker. Therefore, MQTT is a message-oriented protocol. The address that the message published to it is called topic. The Device may subscribe to more than one topic, and it receives all messages that are published to these topics. The broker is a central device between the spoke model and the mentioned hub. The main MQTT broker responsibilities are processing the communication between MQTT clients and distributing the messages between them based on their interested topics [22]. The broker can deal with thousands of connected devices at the same time. Upon receiving the message, the broker must search and find all the devices that own a subscription to this topic [23].

A. MQTT Client: MQTT client may be any of IoT object that sends or receive data, not just devices. Any device can be a client like microcontroller or the server. The MQTT client type depends on its role in the system whether it is a subscriber or a publisher [23].

B. MQTT Broker: The broker is a central device between the spoke model and the mentioned hub. The main MQTT broker responsibilities are processing the communication between MQTT clients and distributing the messages between them based on their interested topics. The broker can deal with thousands of connected devices at the same time. Upon receiving the message, the broker must search and find all the devices that own a subscription to this topic [23].

MQTT architecture contains three components. Those are a publisher, a broker, and a subscriber. The device that is interested in specific topics register on it as a subscriber to be informed when the publishers are publishing their topics by the broker. The publisher transfers the information to the subscribers via the broker (the interested entities). It is working as a generator of interested data, and then, the authorization of the subscribers and the publishers are checked by the broker to realize the associated security issues [24].

5. Methodology

There are a lot of methods or approaches for object detection some mentioned in the previous section. For this project object detection, we are detecting currently plastic bottles and bottle caps as the main object to be detected for recycling. Starting with a previously untrained neural network model and training it from scratch requires a lot of resources such as a lot of labeled data and training time. So, to reduce these resources we used transfer learning.

Transfer learning is a concept in machine learning where we use the knowledge acquired while solving some problem and use this knowledge to solve other related problems [17]. it uses pre-trained models. For example, knowledge gained while learning to recognize dogs could apply when trying to recognize different breeds of dogs.

The pre-trained model we used for this project is the SSD MobileNet V2 FPNLite 320x320 which is found in the TensorFlow model zoo available pre-trained models.

5.1. Object Detection and Classification

5.1.1. SSD MobileNet

When we look at this model architecture it is a combination of the SSD neural network architecture and MobileNet architecture [18].

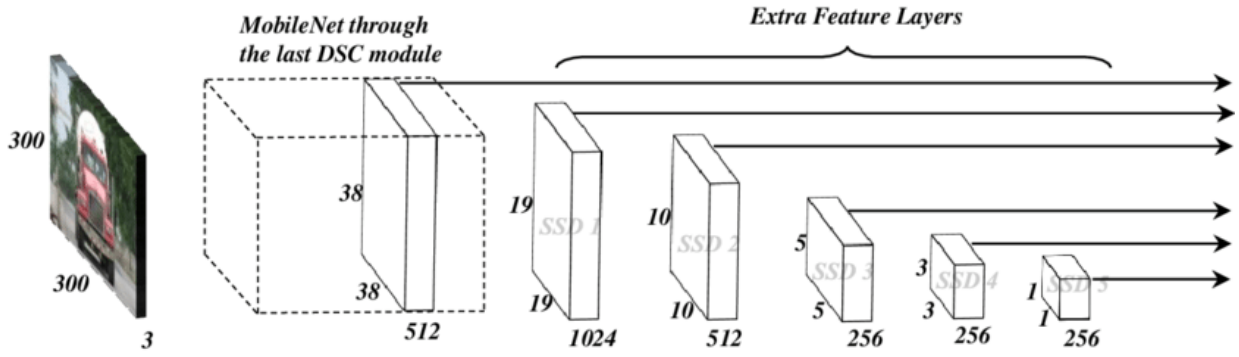
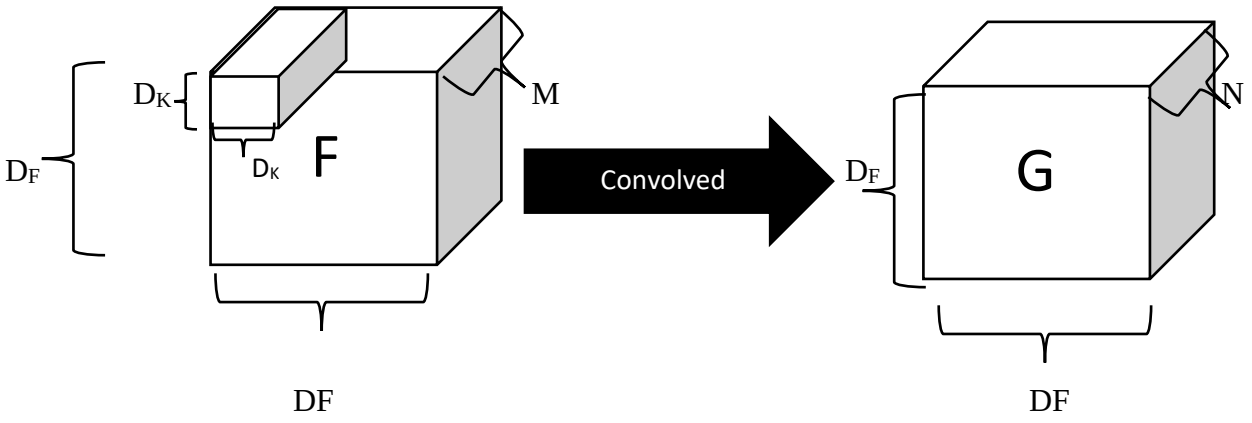


Figure 3: SSD MobileNet architecture

In SSD architecture the base network used in the original paper was VGG as mentioned in the section above but here for SSD MobileNet the base network is replaced by MobileNet [18]. which is an efficient architecture over VGG because MobileNet uses Depthwise separable convolution layers the purpose of this architecture is same as VGG which is to extract features from an image but since MobileNet uses Depthwise separable convolution layers is more efficient will so how in a later section.

Depthwise separable convolution

We have seen what and how convolution neural networks work including the convolution operation know let's calculate the cost so we can see how the Depthwise separable convolution might more efficient computation cost wise.



To calculate the computation required we can look at the number of multiplication operations since multiplication is an expensive operation compared to addition. For a single convolution operation, say from the above figure we have an input image F with a dimension of $D_F x D_F \ x \ M$ where M is the number of channels for example in RGB image M is 3 and $D_F x D_F$ is the input image diminutions (width and height). This input is convolved by N kernel and the output dimension after convolution is $D_F x D_F \ x \ N$ in the N kernels single kernel dimension is $D_k \ x D_k x M$ and when we convolve the input with the N kernels or filters ($F * K$) where k has dimension $D_k x D_k x M x N$.

So, the computation cost is the number of multiplications which is calculated by multiplying the number of elements in the kernel, for N number of kernels and the position taken by the kernel which is the dimension of the input.

$$Cost_{\ CNN} = \ D_k x D_k x M x N x D_F x D_F \ = D_F^2 x D_k^2 x \ M \ x \ N$$

In Depthwise separable convolution the process of convolution is separated into a Depthwise and

Pointwise convolution

Depthwise convolution is applied over a single channel rather than M channels so the kernel dimension will be $D_k \ x D_k x 1$ given an n channel the output will be $D_k \ x \ D_k x \ 1 \ x \ M$

In pointwise convolution we use a $1x1$ kernel which will be convolved with the depthwise convolution and the dimension of this convolution will be $1x1xMxN$.

The computation cost will be the sum of Depthwise convolution and pointwise convolution.

$$Cost_{DSC} = Depthwise\ convolution\ cost + Pointwise\ concvolution$$

$$Cost_{DSC} = D_k\ x\ D_k x\ M\ x\ D_F x D_F + M\ x N\ x\ D_F x D_F = M x D_F^2 + (D_k^2 x\ N)$$

So, comparing the two costs by taking the ratio

$$R = \frac{Cost_{DSC}}{Cost_{CNN}} = \frac{1}{N} + \frac{1}{D_k^2}$$

Let's take an assumption that N=100 and DK = 512 the ratio becomes 0.01004 this means that DSC in this example if consider $T = 1/R \approx 100$ times less multiplication than CNN there is a thread off of accuracy but its less than one or two percent.

In MobileNet they use this Depth-Wise Separable convolution and to combat the accuracy loss they used batch normalization and the Rectified Linear Unit (ReLU) activation function. The activation function is used to add nonlinearity to the neural network.

5.2 Implementation

5.2.1. TensorFlow

The implementation for detecting the plastic bottle and other plastic material components was made using TensorFlow version 2.7.0.

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML, and developers easily build and deploy ML-powered applications.

TensorFlow has an object detection API that makes it easy for constructing, train, and deploying object detection models. In TensorFlow object detection API, there are a variety of pre-trained models that are trained on different datasets such as the coco (common object in context) data set, the KITTI dataset, and the open images dataset.

The roadmap we took for the training of custom object detection using the TensorFlow framework was

- Preparing the dataset.
- Train and evaluation SSD MobileNet model.
- Make inferencing using images and video.

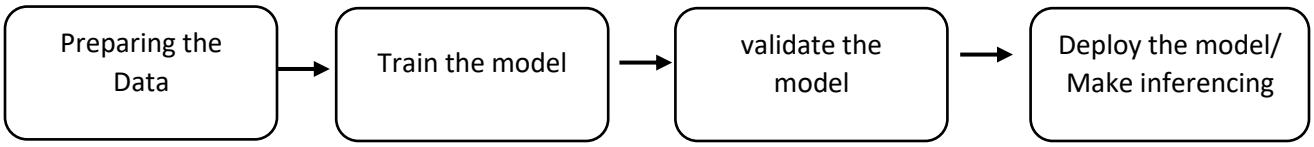


Figure 4: Object detection pipeline

Preparing Data

There are common data set used for the project of recycling that is Trashnet dataset which includes 2527 images:501 glass, 594 paper, 403 cardboard, 482 plastic, 410 metal, 137 trash but for the implementation of our project we took three sets of data for training three different custom object detection model.

Dataset one: consisted of 501 glasses, 594 paper, 403 cardboard, 482 plastic and 410 metal images form the Trashnet dataset.

Dataset two: consisted of 278 plastic bottles and plastic bottle caps, the bottle caps were labeled using their color such as blue cap or red cap etc.

Dataset three: consisted of the same data set as dataset two the difference is that in dataset three the plastic bottle caps are not labeled based on their color.

The dataset was dividing it for training and evaluation of the models 80% for training and 20% for evaluation, we then used the labeling tool called LablelImg which is a free, open-source tool for graphically labeling images. It's written in Python and uses QT for its graphical interface. It's an easy, free way to label a few hundred images. After labeling the images the labeling process looks like the image shown below this will create an XML file consisting of the XY coordinates

of the object that is drown bounding box as well as the object name, file location, and other attributes this will then be used for training the model.

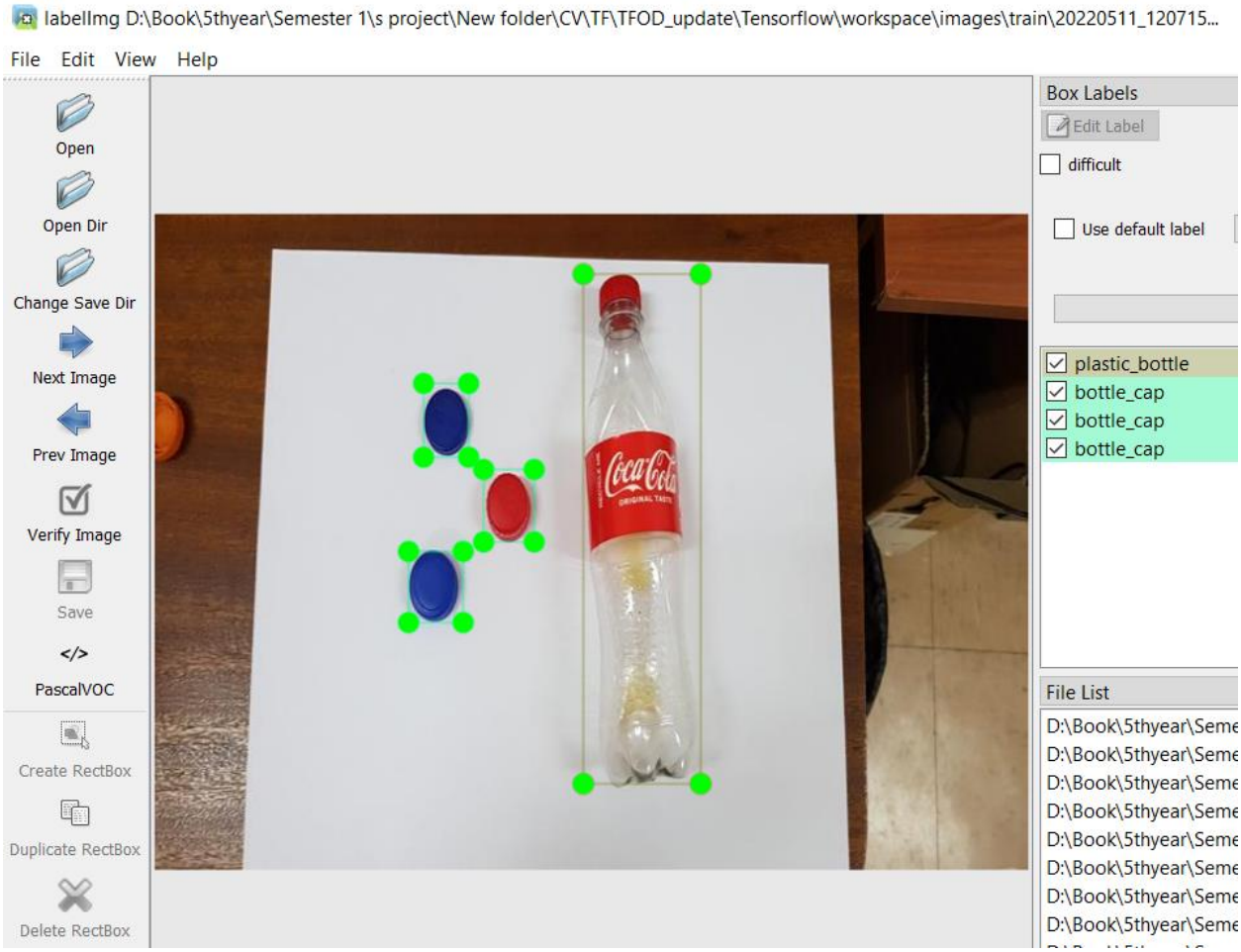


Figure 5: labelling of collected image using LablelImg software.

5.2.2. YOLOV5

This project is about classifying and sorting or ordering of recycled materials from wastes. Since the classification part is done through the analysis of images, object detection plays the important role in the detection and classification of recyclables from the real time video feed.

First to detect and classify the recyclable waste from video frames we used mainly two different methods and used the better model for classification. The models used are YOLO V5 and SSD model from the TensorFlow.

We used YOLO V5 as one of our object detecting model as it is fast for real time purpose and it doesn't require a lot of processing power for inferencing. The first thing done was annotating all the images taken with camera with different labels using Makesence.ai which is an online platform for annotating images. The second thing done was training the YOLO V5 model with the dataset prepared. While doing the labeling, the raw images taken with camera were divided into two groups for training and validating purpose and the data proportion while grouping the data was 8:2 (Eight to two) for training and validating respectively. We used different images for different trainings and the average images taken to train the model was 100 images.

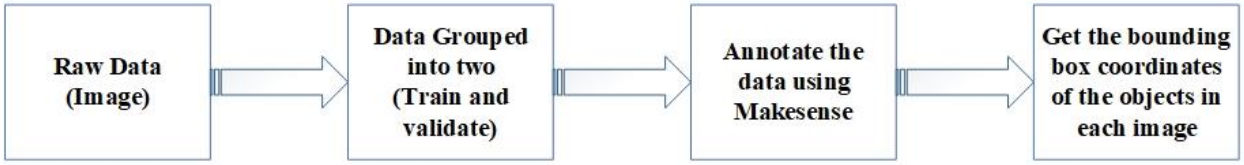


Figure 6: Steps of annotation and labeling the dataset.

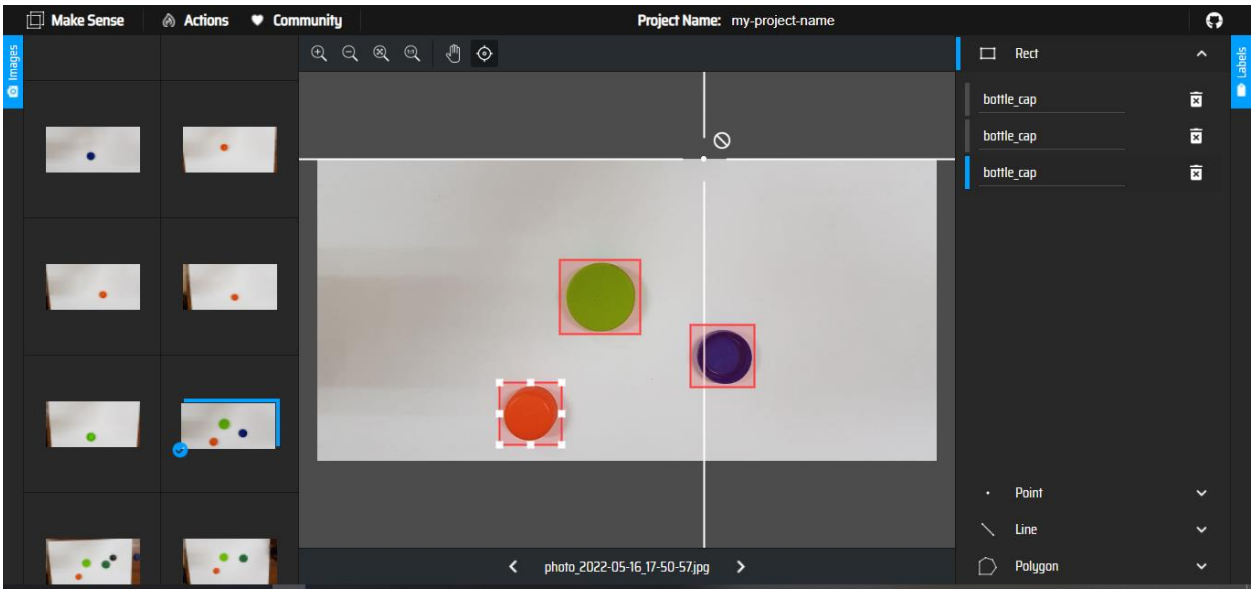


Figure 7: Labeling of images using Makesense.ai

After labeling the data the next step done was to train the YOLO V5 model on Google colab. The model was trained with a batch of 2 for 50 and 100 epochs. We trained the YOLO V5 model with different datasets to classify the waste material objects based on color especially for the bottle caps (green, blue, red, black and orange) and based on type (bottle or bottle cap).

Sorting of Classified Recyclable Waste

The robot arm that is used in this project is constructed using the Lego Mindstorm EV3. Lego Mindstorm is a hardware and software structure that is manufactured by Lego for the development of programmable robot using Lego building blocks. Lego Mindstorm EV3 is a third generation Mindstorms that was released in 2013. Lego Mindstorm EV3 comes in two editions as Home edition and Educational edition. In this project Lego Mindstorm EV3 Educational edition kit is used. Lego Mindstorm EV3 has a total of 601pieces with ev3 brick and other parts.

5.2.3. Robot Arm Implementation

The brain of the Lego Mindstorm EV3 is the ev3 brick. Ev3 brick of Lego Mindstorm EV3 has the following specifications:

- 300MHz 32-bit Arm based Microcontroller.
- 16MB flash memory and 64MB of RAM.
- Linux operating system.
- USB port and Bluetooth Interface.
- 4 input and 4 output ports.
- Speaker and 6AA battery.

Lego Mindstorm Ev3 also have two large servo motors and one small servo motors with others sensors like touch sensor, color sensor, gyro sensor and ultrasonic sensor. For this project ev3 brick, two large servo motor, one small servo motor and touch sensor are used with other building block parts of the Lego Mindstorm EV3 are used.

Large servo motors of Lego Mindstorm EV3 uses a tachometer generated feedback signal for control. It has a running and stall torque of 20 N.cm and 40 N.cm respectively with a rotational

speed of 160 – 170 rpm. Small servo motor of The Lego Mindstorm EV3 also uses tachometer generated feedback signal for control. It has running and stall torque of 8 N.cm and 11 N.cm respectively with a rotational speed of 240 – 250 rpm. Both large and small servo motors have one degree of precision.

The two large servo motors are used to rotate the body of the robot arm while the small servo motor is used to control the gripper of the robot arm. The design of the robotic arm is implemented using a Lego Digital Designer software which has all pieces of the Lego Mindstorm Ev3. One of the large servo motors is used to rotate the base while the other large motor is used to rotate the elbow. Using the Lego Digital Designer, the robot arm is designed by assembling each part of the robot arm. The figure below illustrates the Lego Digital Designer assembly process.

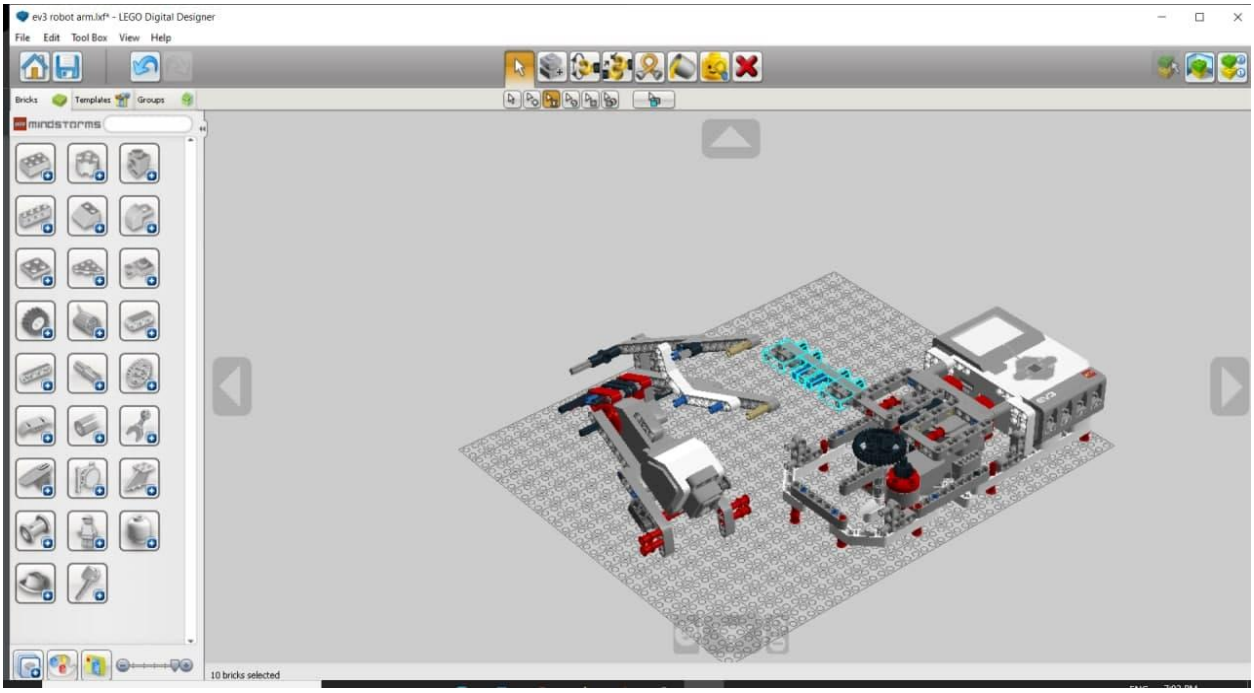


Figure 8: Lego Digital Designer

The program to control the robot arm python3 was used with the pybricks library. First the ev3 brick is initialized then the base servo motor is initialized and during initialization the port to which the motor is connected on the brick and gear teeth numbers of the gears that are connected

to the base servo motor are passed as an argument. The elbow motor is initialized in the same way the base motor is initialized. The small servo motor that is used as a gripper controller is initialized by passing the port number as an argument. Touch sensor is also initialized. The touch sensor is used to restrict the rotation of the base motor. Then after initializing all the important parts of the robot arm then using the pick, place and release method which target positions are passed as an argument to the robot becomes functional to pick and place an object.

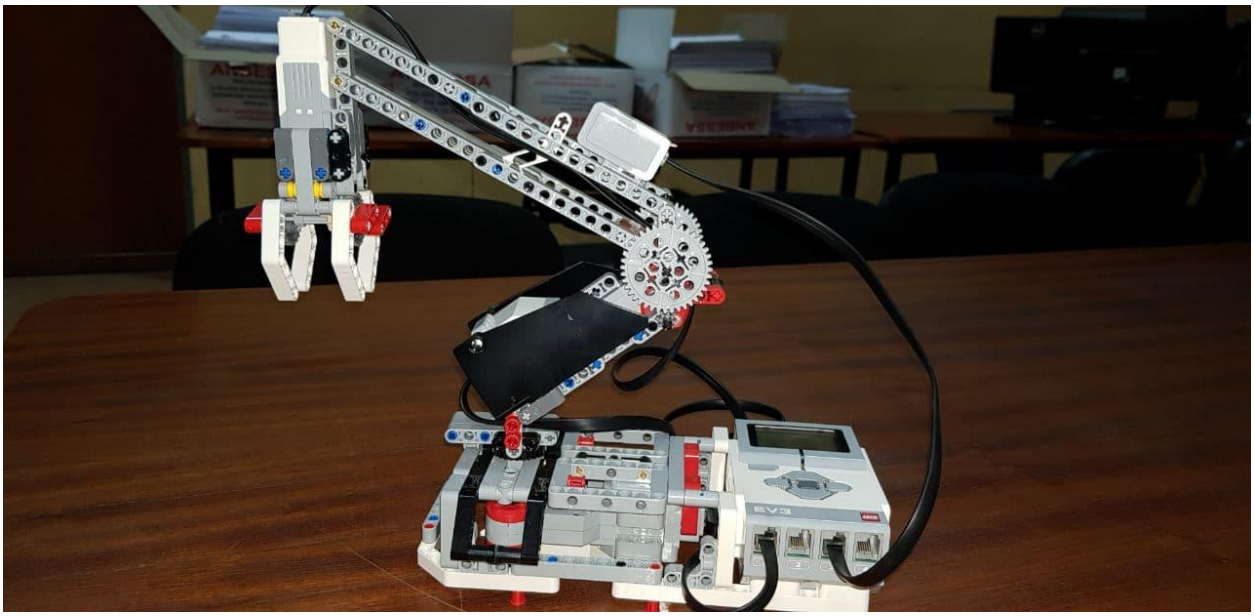


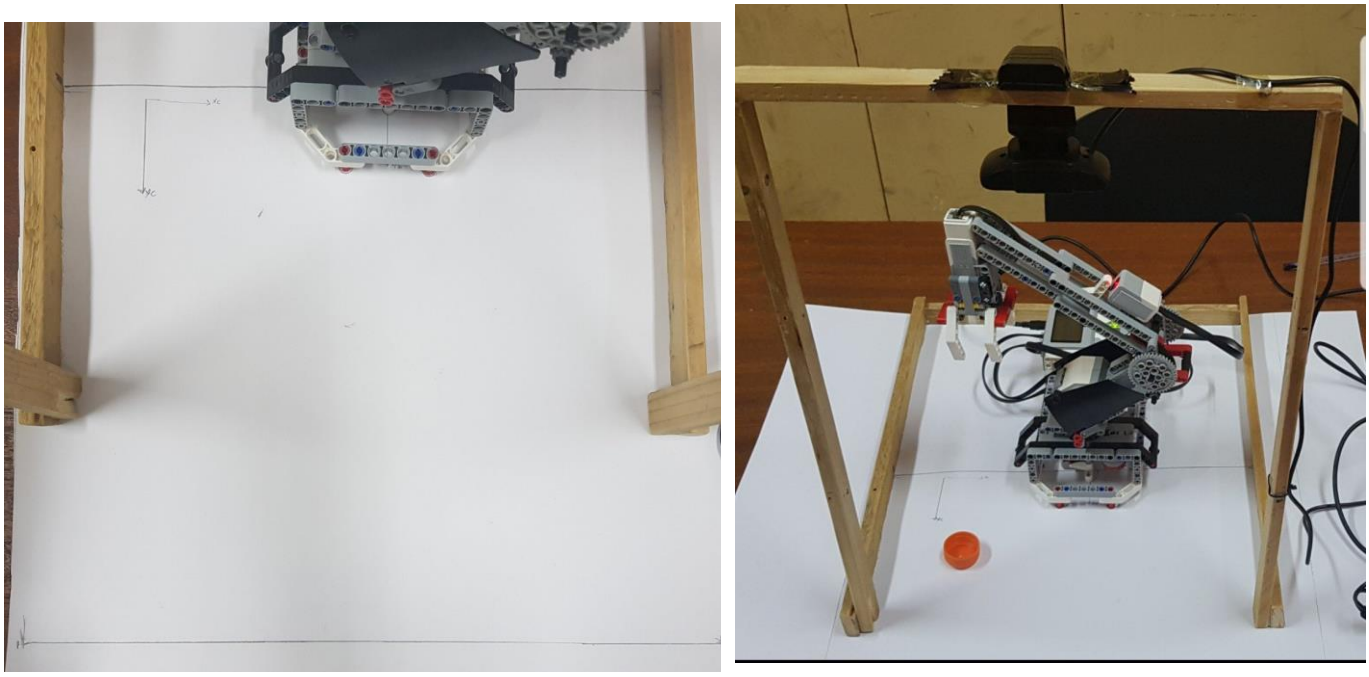
Figure 9: Lego Mindstorm EV3 made robot hand

5.2.4 Sorting with Robot Arm

The Ev3 robotic arm has a very week processing power for this reason the ai model wasn't able to work on the ev3 robotic arm so, we had to work out a way in which the AI model can run on the computer and it can send the detected object location coordinate in terms of base angle rotation to the robotic arm so that the robotic arm can pick and sort the object.

First, we made the robot arm pick and place objects independently by kipping the motors other than the base motor do a routine movement that is to pick and place once the base motor rotation angle is known.

Figure 10: Setup of the project work space.



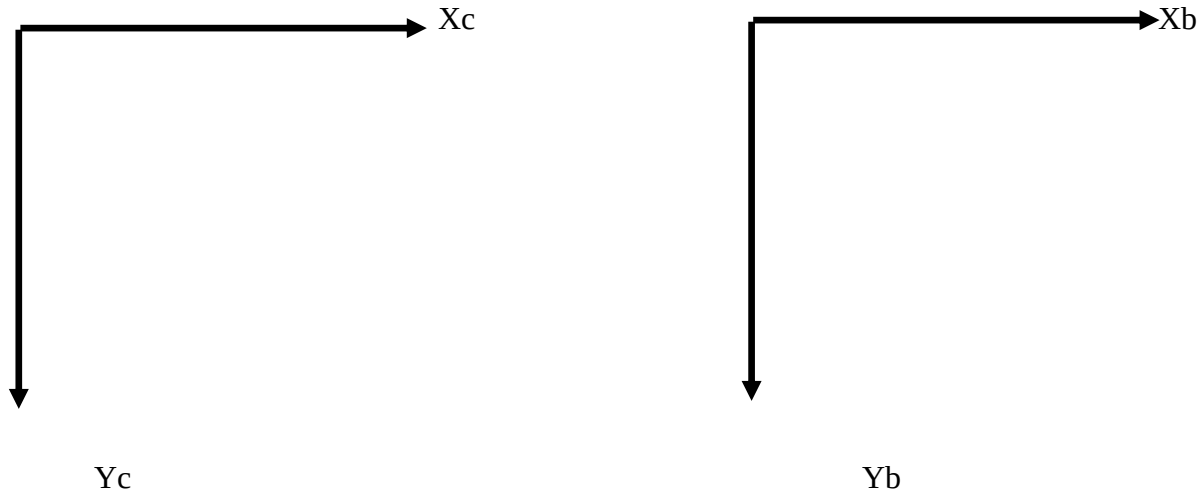


Figure 11: Coordinates for the camera frame (Xc,Yc) and for the base rotator of the robot(Xb,Yb).

In the Setup of the project work space for the robot arm to go and pick an object as seen by the computer camera we needed coordinate systems. We started by labeling the two-coordinate axis, one for the base motor of the robot that is the base rotation axis Xb and Yb the other coordinate we defined was the camera frame of reference. The camera used for detection and gathering of some of the dataset was a webcam called Miincam which was placed parallel to the surface of base. We labeled the origin of the camera frame (Xc = 0, Yc =0). which is the up-left corner of the camera frame, we noted this point as origin point. The X axis of the camera that is the Xc increases from left to right and the Y axis of the camera frame that's Yc increase from up to down from origin of the camera. The value of the axis is in pixel format for the camera's frame and the frame of our work space was 640 by 480 pixels. In order to get the position of the object in centimeter these values were converted. To convert them into centimeter we measured the distance of the frame in both x and y direction on the view plane of the camera and divided it over the pixel distance of the fame. This gave us an approximate value of Xc and Yc values in centimeter.

PIXEL _TO_CM of the Width is $\frac{38\text{ cm}}{640\text{ pixels}} = 0.059$

PIXEL _TO_CM of the height is $\frac{27.3\text{ cm}}{480\text{ pixels}} = 0.057$

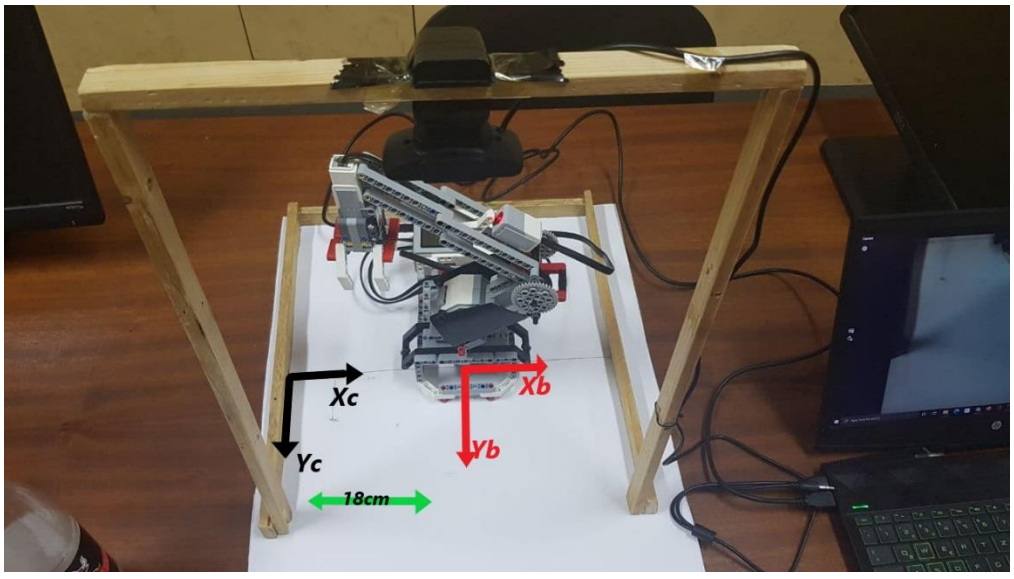
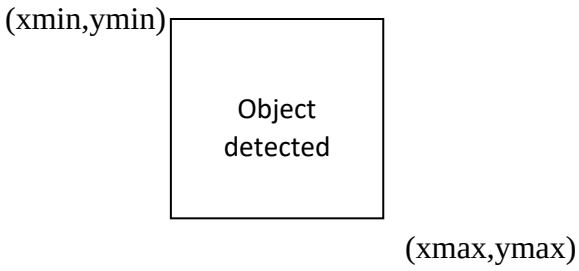


Figure 12: coordinate system for the camera and the robot.

Once we knew how to convert the pixel values to centimeters on the plane view of the camera. we used the object detection model to identify the object and center of the object from detection bounding box. The bounding box contains the xmin, ymin, xmax and ymax of each detection we can use this to find the center point of the object by:

$$\text{Xcenter} = \text{xmin} + \frac{(\text{xmax} - \text{xmin})}{2})$$

$$\text{Ycenter} = \text{ymin} + \frac{(\text{ymax} - \text{ymin})}{2}$$



Even though the center point can be converted in to centimeter values the center point calculated is in the camera frame coordinate and not in the robotic arm base coordinate. So, to solve this we have these two coordinates, one for the base of the robotic arm and another for the camera frame of reference we can use basic translation so that we can get the center point of the object in the base coordinate and use inverse kinematics to solve for base rotation so we can command the robot to move to the object and pick it up and sort it.

The translation for our work space for both the robot base frame coordinate and the camera frame reference have the same Y-axis that (Yc = Yb) and no translation is required in this axis for the x-axis translation is required since there is an offset of 18cm so Xb equals Xc minus 18cm. After calculating the Xb and Yb of the object from the robot arm base coordinate we can use inverse kinematics to solve for the angle of rotation needed for the robot to get to the object to be picked and sorted the equation we used was a simple inverse of tan calculation using Xb and Yb.

$$\theta = \arctan\left(\frac{\text{Yb}}{\text{Xb}}\right)$$

This equation is the only one needed because we took top view of the robot and kept the movement of the gripper and elbow motion of the robot arm constant. This is done for the elbow of the robot to go down when an object is located and the base motor has rotated to the object location and for the gripper to pick after the elbow of the robot reaches to the object and releases the object it has picked up first. The distance on how far the elbow goes down or up is determined by photosensor placed on the robotic arm.

The other mechanism used was to use the HSV color space to detect the color of the bottle cap for sorting. Using the object detection bounding box we found the center of the object, the detected object's frame pixel value can be used to identify the color of the cap.

Communication between the robotic arm and the Ai model that was running on the computer was done using MQTT. On the computer AI model identified the object, and after the detected object's center coordinate and the theta for robot base rotation was calculated. The computer sends or publishes the angle and color of the object to the robot using MQTT protocol the robot is subscribed to a topic that is the same as the published topic by the computer. After receiving the message, based on the information in the message the robot arm will pick and sort the object accordingly.

6 Evaluation

6.1 TensorFlow Trained Model Evaluation

First, we set up the TensorFlow training environment on our computer and using the pre-trained SSD Mobilenet Object detection model, which is trained on COCO 2017 dataset with training images scaled to 320x320 we trained our custom object detection model. The training process can be seen in the figure below.

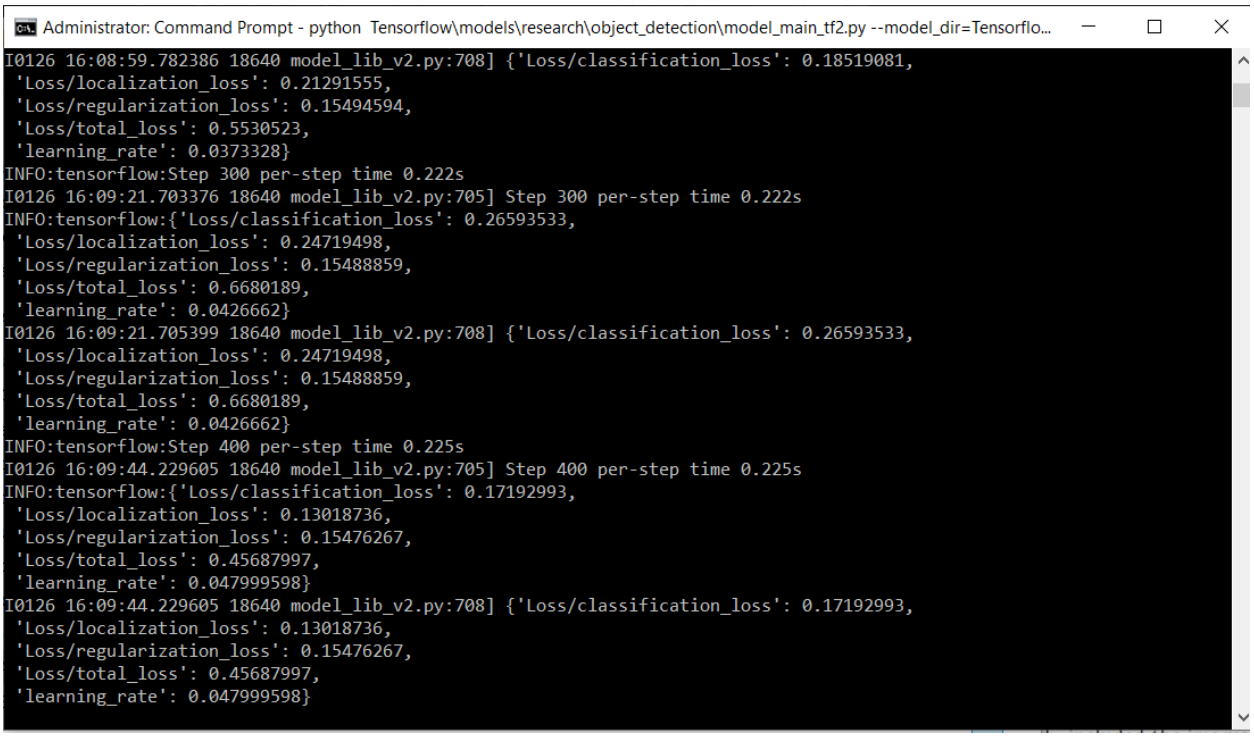


Figure 13: Training process using GPU

The training continues by adjusting the weights of the pre-trained model to reduce the loss. The loss is a matric indicating how bad the model is predicting given the training data. If it is below 1 then the model prediction is working well and once it's finished training on the dataset provided and we have a low not drastically varying loss we can then evaluate the model using the test dataset.

Using tensor board, we can see the training processes for the three dataset collections

Dataset one: this model was trained for 30 thousand steps and as we can see the although the loss for all functions including the total loss was decreasing the result fluctuates and after this may steps the learning rate was near zero meaning the model was not improving.

Figure 14: Training results of Dataset one

Dataset two: in this model the color detection for the bottle cap and the object detection for plastic bottle were trained. The model was able to detect the plastic bottle very easily but detecting the color labeled plastic bottle caps based on the color was inaccurate, the reason for this is the picture quality of the camera used and the number of collected caps were not sufficient enough for color detection. The figure below shows the losses during training.

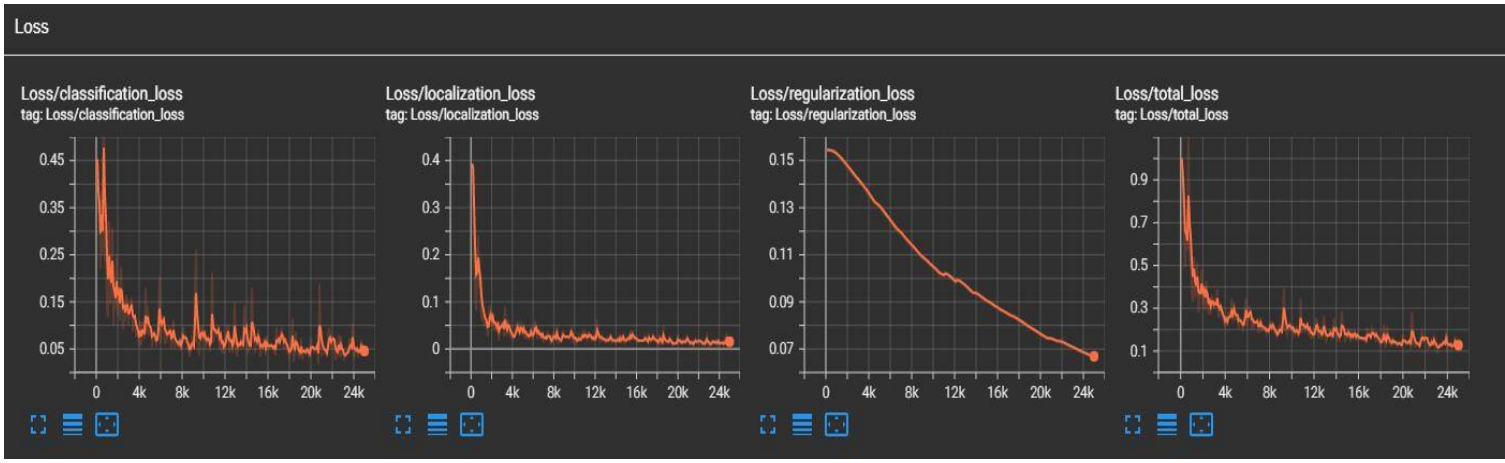


Figure 15: Training results of Dataset two

Dataset three: the final dataset was trained using the plastic bottle and the bottle cap without color detection this model was the best out of the rest because it was detecting both with high accuracy as we can see below the loss on all instance has almost gone to zero in smooth way.

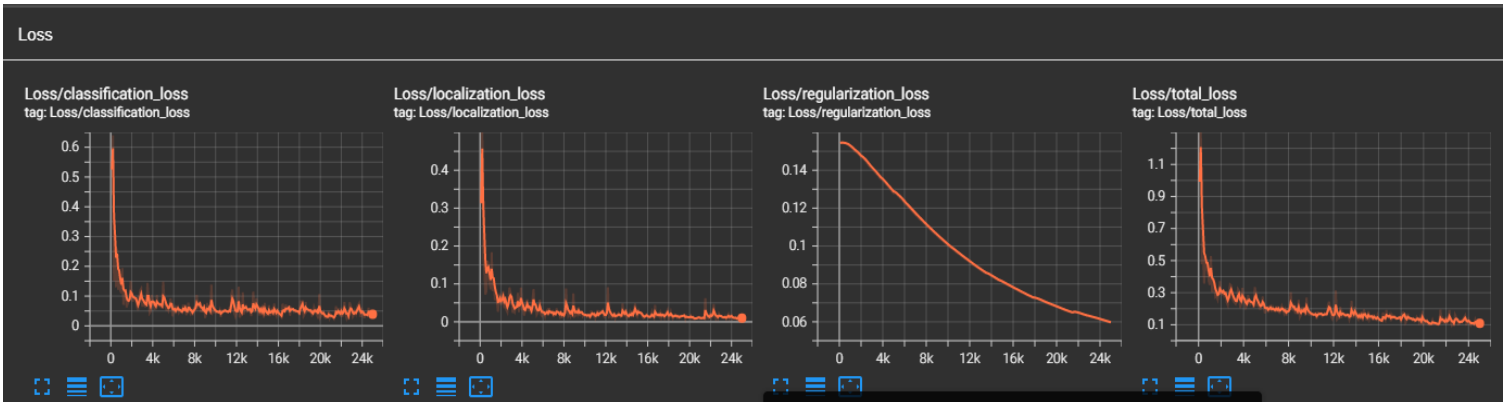


Figure 16: Training results of Dataset three

The model trained on Dataset three was used as the main mode for the object detection for color identification of the cap the HSV color space was used in relation to the object detection.

The difference between the predicted and labeled bounding boxes are computed by the localization loss, in this case, it is approximately 0 which means in our model the predicted bounding box is almost the same to the labeled one from our dataset. The classification loss reflects if the bounding box class and the expected class are the same its 0.039 for our model. The regularization loss is

created by the network's regularization function and aids the optimization algorithm's progress its 0.059 for our model. The final term is the total loss, which is the sum of the three preceding terms.

The evaluation was done using the COCO detection evaluation matric which provided the precision, recall, and loss of the trained model before talking about this let us define a few terms.

True positive (TP) is the trained model that has positively identified an object it was trained to detect in a given image.

False-positive (FP) in this case the object to be detected is in the image but the model makes the wrong detection not detecting the actual object it was trained to detect.

True negative (TN) is that the model has not detected an object because the object is not in the image.

False-negative (FN) Is when a model has not detected an object in an image but the object exists in the image.

To measure TP, FP, TN, FN we use the concept of intersection over union (IoU) threshold. This threshold is calculated by dividing the intersection of predicting bounding box (Bp) and ground truth box with their union [22].

$$IOU = \frac{area(Bp\ n\ Bt)}{area(Bp\ U\ Bt)}.$$

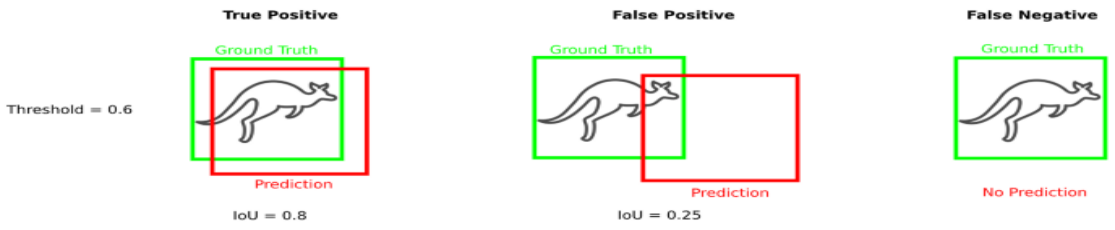


Figure 17 IoU threshold

The recall measures how good the model is in finding the positive class. It's calculated as follow:

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

The precision measures how reliable the true positive detection is or what proportion of the detection is correct it is calculated as follow:

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

Considering the above concepts for our model we got results of mean average precision and recall for different IoUs as shown below.

Figure 18: Evaluation average recall maximum and average precision of trained model

From the image we can see from the small evaluation dataset the average precision is 0.77/1 which really good result also the average recall maximum number is 0.8/1 which means the model is shown an object again and figuring out it's the true positive is around 80%. Both the values of average precision and average recall are reasonable results given the dataset.

6.2 YOLO V5 Trained Model Evaluation

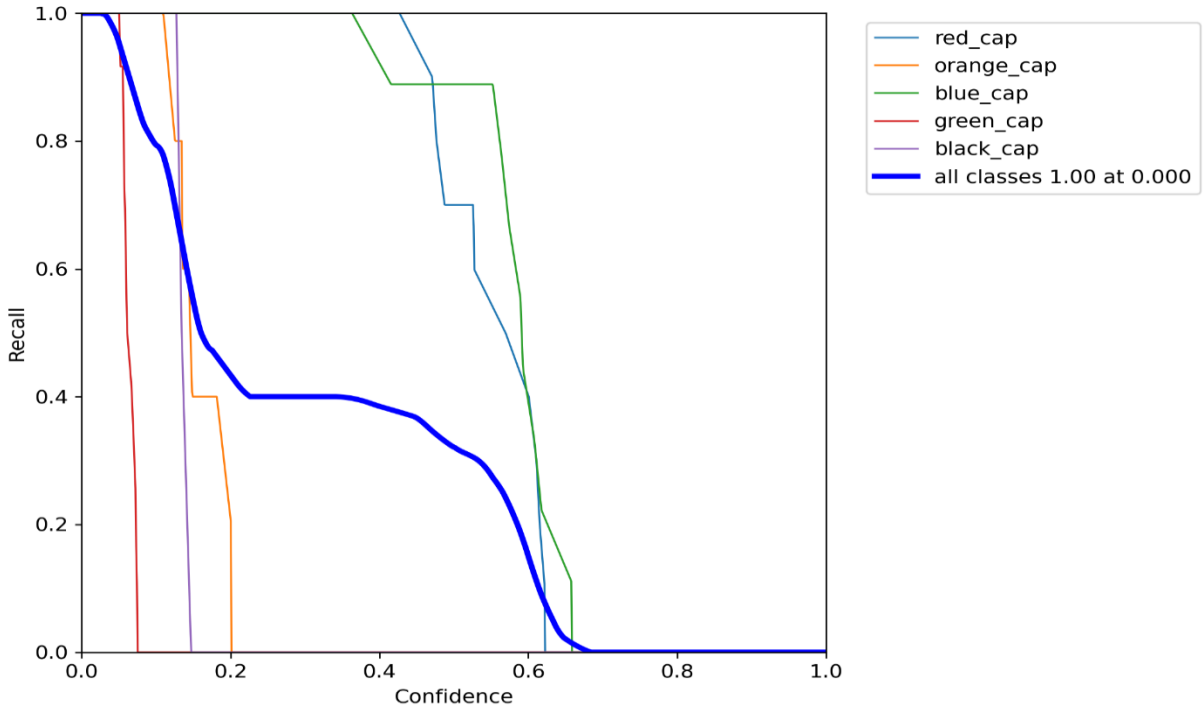


Figure 19: Recall-confidence graph for color detection of bottle cap using YOLO V5

The Confidence Score is a number between 0 and 1 that represents the likelihood that the output of a model is correct and will satisfy a user's request. The recall of the trained model is less than 0.2 or 20% with a confidence of 0.5 and above. The result we found is not satisfactory to be used as the primary model for classification.

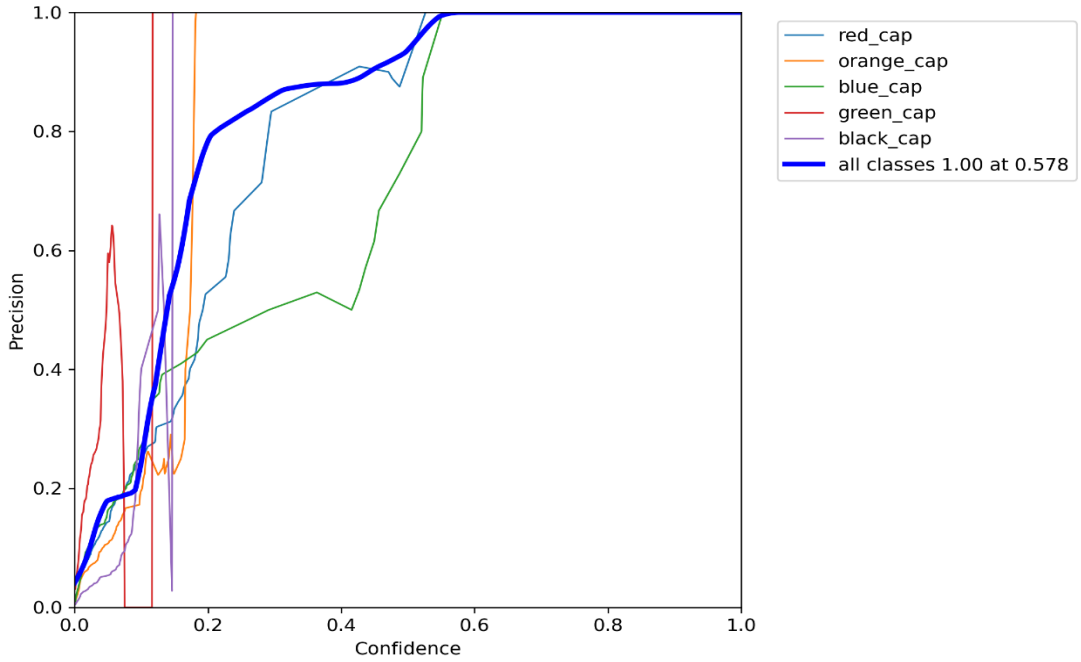


Figure 20: Precision-Confidence graph for bottle cap color detection using YOLO V5

Precision basically shows the accuracy of the model's prediction. The result of precision for the trained model is 1 for a confidence of 0.578.

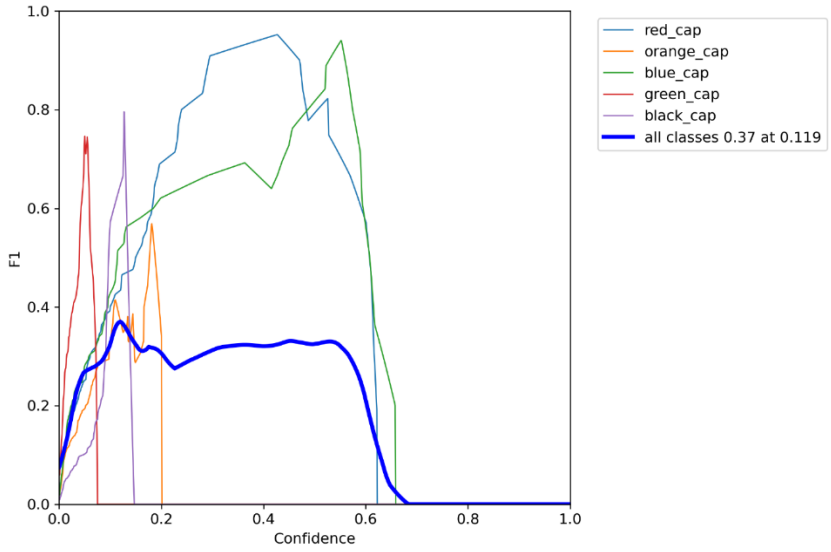


Figure 21: F1-Confidence graph for bottle cap color detection using YOLO V%

Mean average precision of the trained model is 0.69. Since we got a better performance with TensorFlow, we used TensorFlow as out primary object detecting model. YOLO V5 generally is used for real time object detection and it is preferrable for many applications because it is known to be fast at a cost of performance. For this project performance is more important than being fast. So, we used the TensorFlow model which delivered a better performance and accuracy when it comes to detecting and classifying of objects.

7 Inferencing

The last step in this phase was to use the model we have trained to do this we used OpenCV library to detect plastic bottles and bottle cap in an image as well as from our computer web came in the detecting of the object the model did recognize the object most of the time but there were cases where the model didn't detect the object right way below are some examples of detection.

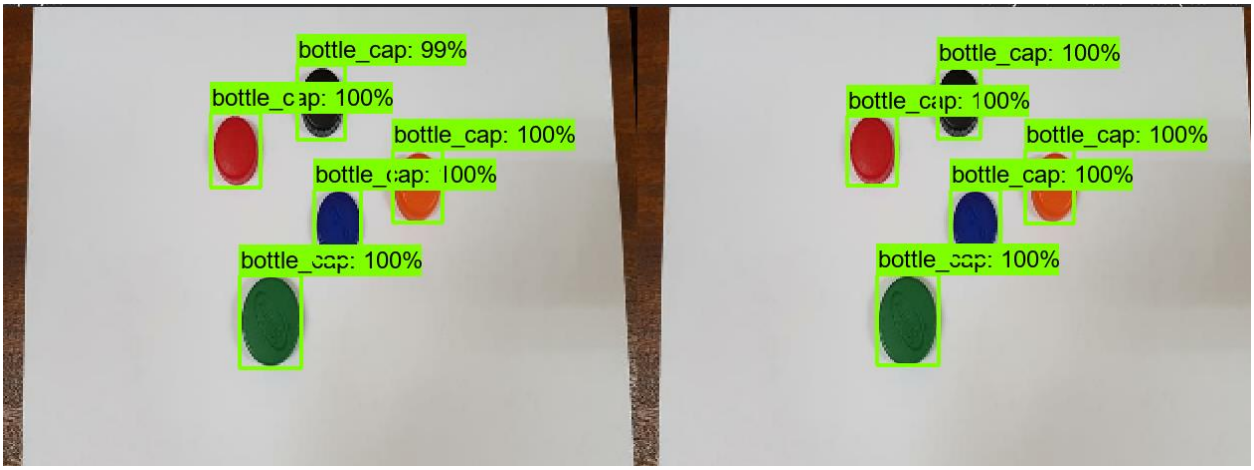
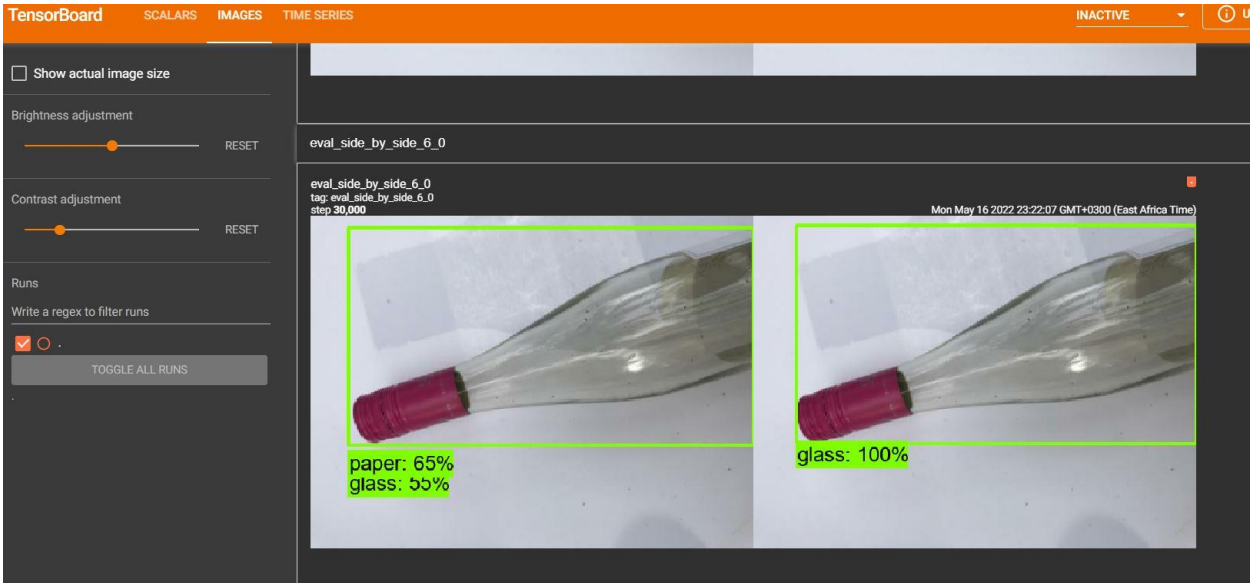
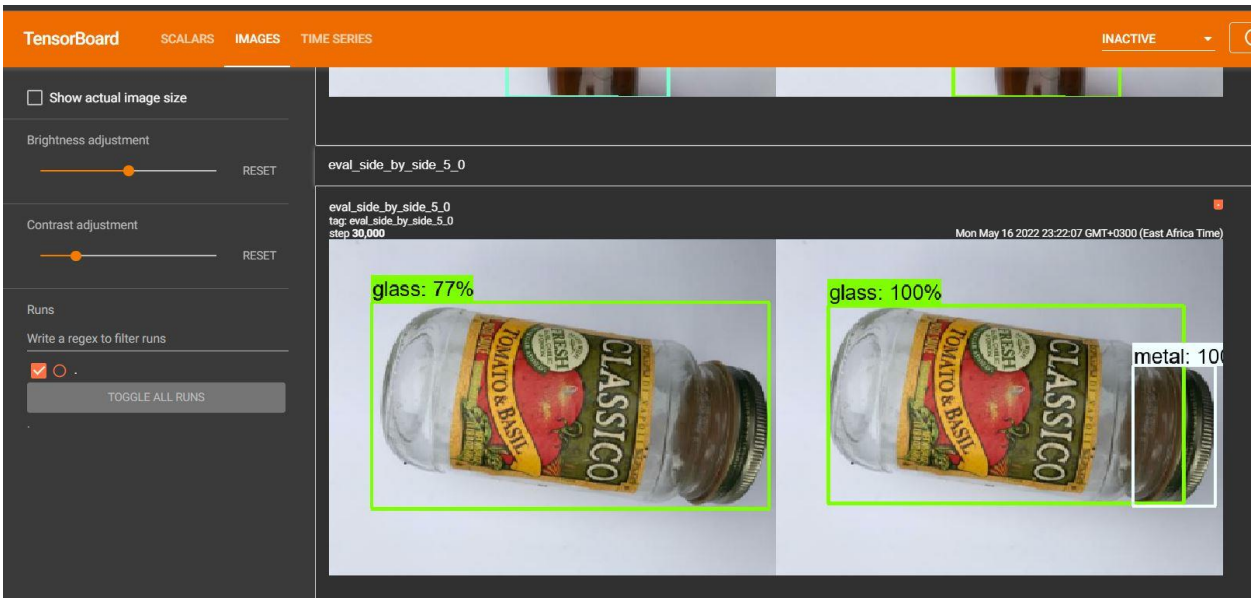


Figure 22: Inferencing Bottle Caps with TensorFlow Trained Model

Below are images from model trained on dataset three

We also tried to the model for dataset one the result is from evaluation because of the model inaccuracy real time detection was getting lot of detection wrong.





8 Conclusion

Considering the number of the dataset for our model, which was sufficient to train the model, in general, had an overall good performance. We are able to classify plastic bottles and bottle caps with the AI model trained with our own data. This model can be improved by collecting and labeling more images of recycling material with different background lighting and different depth in relation to the camera.

We are able to sort plastic bottle caps with the robot hand based on the color of the plastic bottle caps.

The robot arm gripper that is used now cannot pick and place materials like plastic bottles and other medium sized materials thus the gripper of the robot arm can be replaced with a better gripper that can handle to pick and place medium sized materials.

9 Future work

In regards to object detection, our future work will be collecting and labeling more images of recycling material so we can build a smarter Artificial-Intelligent model that will help the robotic arm also work with other object detection models that might yield better results.

Changing the gripper of the robotic arm with a 3D printed gripper is also part of the future work planned in this project.

Using better camera for inferencing can highly increase the performance of the models. For the future we plan to use better camera for better performance.

10 List of Symbols and Acronyms

AI = Artificial Intelligence

N.cm = Newton Centimeters

Rpm = Revolutions per minute

11 References

[1] Brunner, P. H., & Rechberger, H. (2014). Waste to energy—key element for sustainable waste management.
Waste Management, 37, 3-12.

[2] Vergara, S. E., & Tchobanoglous, G. (2012). Municipal Solid Waste and the Environment: A Global Perspective.

[3] Rotter. V. Susanne, Material Recycling; Teaching Materials IHE Netherlands. 2009.

[4] Sandra. Levinne, Solid Waste management Privatization Procedural Manual; Waste Reduction and Recycling. 2000.

[5] Grzegorzek, M., Schwerbel, D., Balthasar, D. & Paulus, D. (2011) Automatic sorting of alluminium alloys based on spectroscopy measures.

[6] Chandrappa, R. & Das, D. B. (2012) Solid Waste Management. Environmental Science and Engineering. 1. Aufl. edition. Berlin, Heidelberg, Springer-Verlag. Available from: http://ebooks.ciando.com/book/index.cfm/bok_id/373359.

[7] Wahab, D. A., Abidin, A. & Azhari, C. H. (2007) Recycling trends in the plastics manufacturing and recycling companies in Malaysia. Journal of Applied Sciences. 7 (7), 1030-1035.

[8] Tchobanoglous, G. & Kreith, F. (2002) Handbook of solid waste management. Second edition edition. New York, McGraw-Hill.

[9] “What is Computer Vision?” IBM, 20-Oct-2020. [Online]. Available: https://www.ibm.com/topics/computer-vision. [Accessed: 27-Jan-2022].

[10] W. Tarimo, M. M.Sabra and S. Hendre, "Real-Time Deep Learning-Based Object Detection Framework," 2020 IEEE Symposium Series on Computational Intelligence (SSCI), 2020, pp. 1829-1836, doi: 10.1109/SSCI47803.2020.9308493.

[11] Peleshko and K. Soroka, "Research of usage of Haar-like features and AdaBoost algorithm in Viola-Jones method of object detection," 2013 12th International Conference on the Experience of Designing and Application of CAD Systems in Microelectronics (CADSM), 2013, pp. 284-286.

[12] Anant Choksuriwong, Bruno Emile, Helene Laurent, and Christophe Rosenberger. Comparative study of global invariant descriptors for object recognition. Journal of Electronic imaging, 17(2):023015, 2008

[13] Andrey Nikishaev. Feature extraction and similar image search with opencv for newbies, Sep 2019

[14] Yinghao Chu, Chen Huang, Xiaodan Xie, Bohai Tan, Shyam Kamal, and Xiaogang 29 Xiong. Multilayer hybrid deep-learning method for waste classification and recycling. Computational Intelligence and Neuroscience, 2018, 2018

[15] V. Verma, H. Chauhan, D. Gupta and S. Gupta, "Different Convolution Neural Network Based Models for Garbage Detection: A Review," 2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE), 2020, pp. 117-121, doi: 10.1109/WIECON-ECE52138.2020.9397961.

[16] “Convolutional Neural Networks” IBM, 20-Oct-2020. [Online]. Available: https://www.ibm.com/topics/computer-vision. [Accessed: 27-Jan-2022].

[17] Liu, Wei. "SSD: Single shot multibox detector". Computer Vision – ECCV 2016. European Conference on Computer Vision. Lecture Notes in Computer Science. Vol. 9905. pp. 21–37. arXiv:1512.02325. doi:10.1007/978-3-319-46448-0_2. ISBN 978-3-319-46447-3. S2CID 2141740

[18] H. Wang, "Garbage Recognition and Classification System Based on Convolutional Neural Network VGG16," 2020 3rd International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE), 2020, pp. 252-255, doi: 10.1109/AEMCSE50948.2020.00061.

[19] Ganesan, Sathish B S, Shaik, Khamar Basha, Kalist V 2015 Neural Network based SOM for Multispectral Image Segmentation in RGB and HSV Color Space 2015 *International Conference on Circuit, Power and Computing Technologies* (ICCPCT)

[20] P. F. Felzenszwalb, R. B. Girshick, D. Mcallester, and D. Ramanan, “Object detection with discriminatively trained part-based models,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 9, p. 1627, 2010.

[21] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” in ACM MM, 2014.

[22] Durkop, Lars, Bjorn Czybik, and Jurgen Jasperneite. "Performance evaluation of M2M protocols over cellular networks in a lab environment." Intelligence in Next Generation Networks (ICIN), 2015 18th International Conference on. IEEE, 2015.

[23] Grgiü, Krešimir, Ivan Špeh, and Ivan Heÿi. "A web-based IoT solution for monitoring data using MQTT protocol." Smart Systems and Technologies (SST), International Conference on. IEEE, 2016.

[24] M. B. Yassein, M. Q. Shatnawi, S. Aljwarneh and R. Al-Hatmi, "Internet of Things: Survey and open issues of MQTT protocol," 2017 International Conference on Engineering & MIS (ICEMIS), Monastir, 2017, pp. 1-6.