Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering (SCOPE)
B.Tech-Artificial Intelligence

Winter Semester 2022-23

April, 2023

A project report on

Waste Management System Using Drones

submitted in partial fulfillment for the JComponent project of

CSE3062 – Internet of Thing for Smart Cities
by

IBRAHIM AHMED SIDDIQUI (20BAI1189)

RATNESHWAR (20BAI1192)

ROHAN PAWAR (20BAI1201)

HARI KISHAN TAKOOR (20BAI1297)

Signature of the Candidates

Signature of the Faculty

Dr.R. Priyadarshini, AP/SCOPE

Waste Management System Using Drones

Index

Abstract:

The advancements in technology have paved the way for drones that can perform several tasks autonomously. In this project, we propose a method for waste classification using an autonomous drone that captures the video of the ground and uses deep learning model to classify waste into different categories. The video footage is analyzed through an convulation neural network model that consists of multiple layers. The model is trained on a massive dataset of labeled waste to achieve high accuracy in classification. Once the image is classified, the drone autonomously informs the relevant authorities to take action regarding waste management. By efficiently and accurately classifying waste, this autonomous drone system will help in reducing the burden on human efforts and enable better waste management practices.

Introduction

As our world struggles to manage the growing waste and pollution problems, several innovative technologies are being introduced to aid in waste management. Autonomous drones have emerged as a promising technology in waste management, with their ability to navigate, collect video footage of ground and classify waste using deep learning model. These drones are equipped with advanced sensors and cameras that can capture high-resolution video, which are then processed and analyzed to identify the different types of waste, such as plastics, metals, paper, and organic matter. Once identified, the drones inform the authorities about the types of waste and provide insights to develop better waste management strategies.

Problem Statement

As we know our surroundings are not neat and clean which leads to various problems such as blockage of drains, polluted environment, unhealthy atmosphere, unhygienic environment and other problems.
This is mainly occurred because municipality and government officials fail to clean the public places, so to solve the problem we purpose an idea by which we can monitor our surrounding through drones. The drone will help us to identify where garbage is present in our surrounding this will be done by camera module present in the drone which will classify the image based on image classification algorithms and there respective locations will be send to the government officials so that they can have the proper locations of the garbage before hand and then they can clean it. This will help us to keep our surroundings neat and clean.

Contribution

- Deep learning model (training) –

  Harikishan Takoor (20BAI1297)
- Deep learning model (testing) -

  Ratneshwar (20BAI1192)
- Simulation –

  Ibrahim(20BAI1189)
- Alert mail & Video to image -

  Rohan(20BAI1201)
- Drone configuration and flight control -

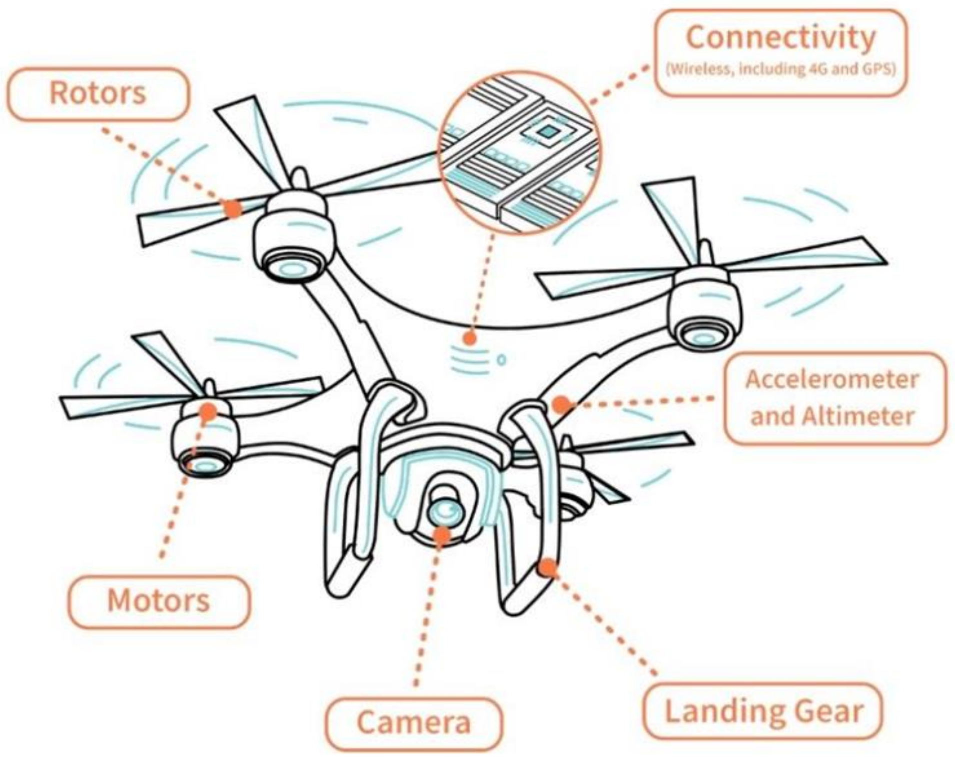
  Harikishan Takoor (20BAI1297), Ratneshwar (20BAI1192), Ibrahim(20BAI1189), Rohan(20BAI1201)

Methodology

PROPOSED METHODOLOGY –

We come up with the idea of smart drones. We will use drone simulator so that the drone will fly in the given locality and will take the video of areas in the locality. Then we will perform image classification on the images taken from the video, to identify the waste or garbage present in the locality and we will send the data to the municipality to clean the localities.
This will help to maintain a clean environment as no mentality change needs to be established on the people of society and also it will be reliable.

Design



PROPOSED ARCHITECTURE –

Drone deployment: The first step in this architecture is to deploy drones equipped with high-resolution cameras to fly over designated areas and record video of the survialance area. The drones can be manually operated or programmed to fly autonomously using GPS navigation. To cover a larger area, multiple drones can be used simultaneously.

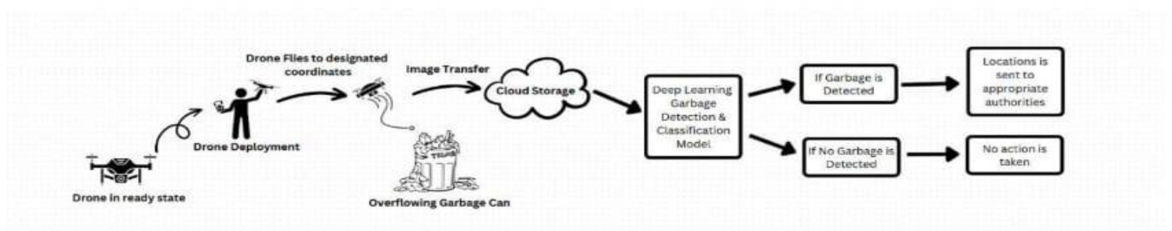
Data Transfering: The video recorded by the drones are then send to the cloud for further processing with their GPS co-ordinates. Cloud processing requires a stable internet connection and sufficient cloud resources.

Image analysis: The video is processed and every frame of video is converted to image for image classification analysis. The cloud-based system will have a Deep Learning Model that can classify the waste into different categories such as plastic, metal, glass, organic, etc. and also detect the presence of waste. The Deep Learning Model is trained using a large dataset of waste images to ensure high accuracy.

Waste Detection: Based on the classification results, the waste can be classified and detected. Now using this, the appropriate coordinates of the areas where waste was detected is further passed as an output from the DL model.

Data analytics: The data collected from the drone and cloud-based system can be used to generate insights on the waste generation patterns, waste composition, and recycling rates. These insights can be used to optimize the waste management process and make intelligent decisions. For example, the data can be used to identify areas with high waste generation rates and prioritize waste management efforts accordingly.

Maintenance and monitoring: The drones and the cloud-based system should be regularly maintained and monitored to ensure optimal performance and reliability. Any issues or faults should be promptly addressed to minimize downtime and ensure the smooth functioning of the system. Regular maintenance and monitoring can help extend the lifespan of the equipment and reduce the risk of costly repairs.


Drone in ready state
Drone Deployment
Drone Flies to designated coordinates
Overflowing Garbage Can
Image Transfer
Cloud Storage
Deep Learning Garbage Detection & Classification Model
If Garbage is Detected
If No Garbage is Detected
Locations is sent to appropriate authorities
No action is taken

Waste management System using Drones
System Architecture

Communication Model -

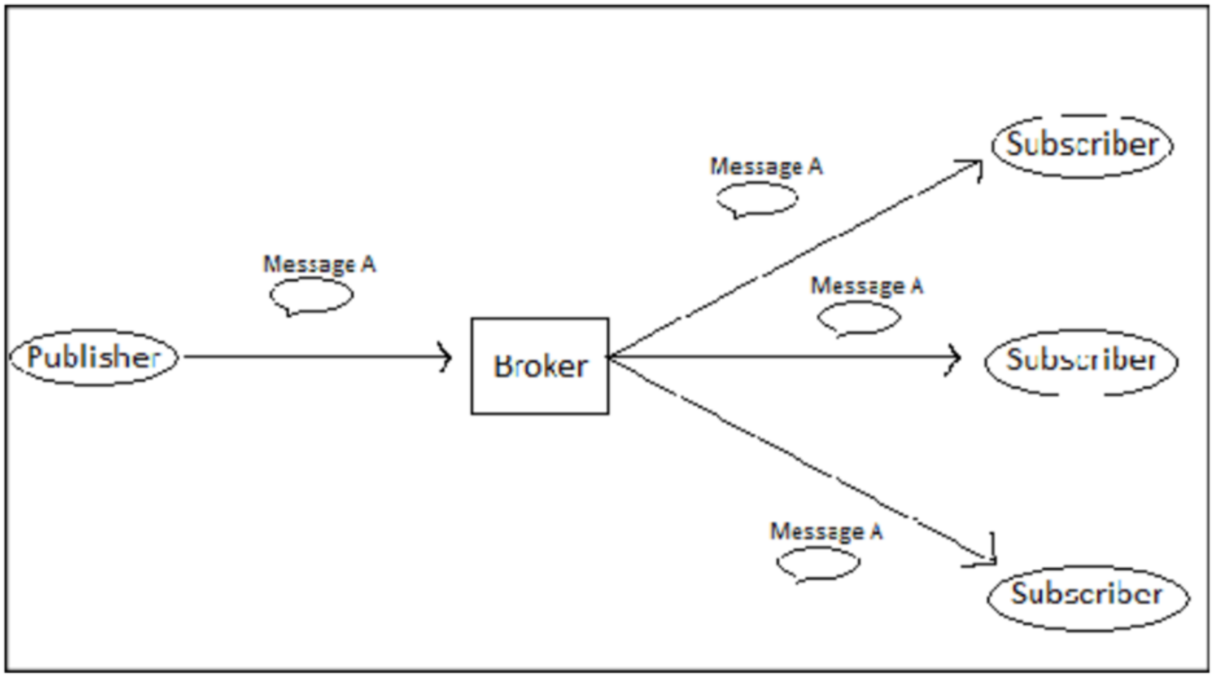
Publisher-Subscriber

This model comprises three entities: Publishers, Brokers, and Consumers.

· **Publishers** are the source of data. It sends the data to the topic which are managed by the broker. They are not aware of consumers.

· **Consumers** subscribe to the topics which are managed by the broker.

· Hence, **Brokers** responsibility is to accept data from publishers and send it to the appropriate consumers. The broker only has the information regarding the consumer to which a particular topic belongs to which the publisher is unaware of.



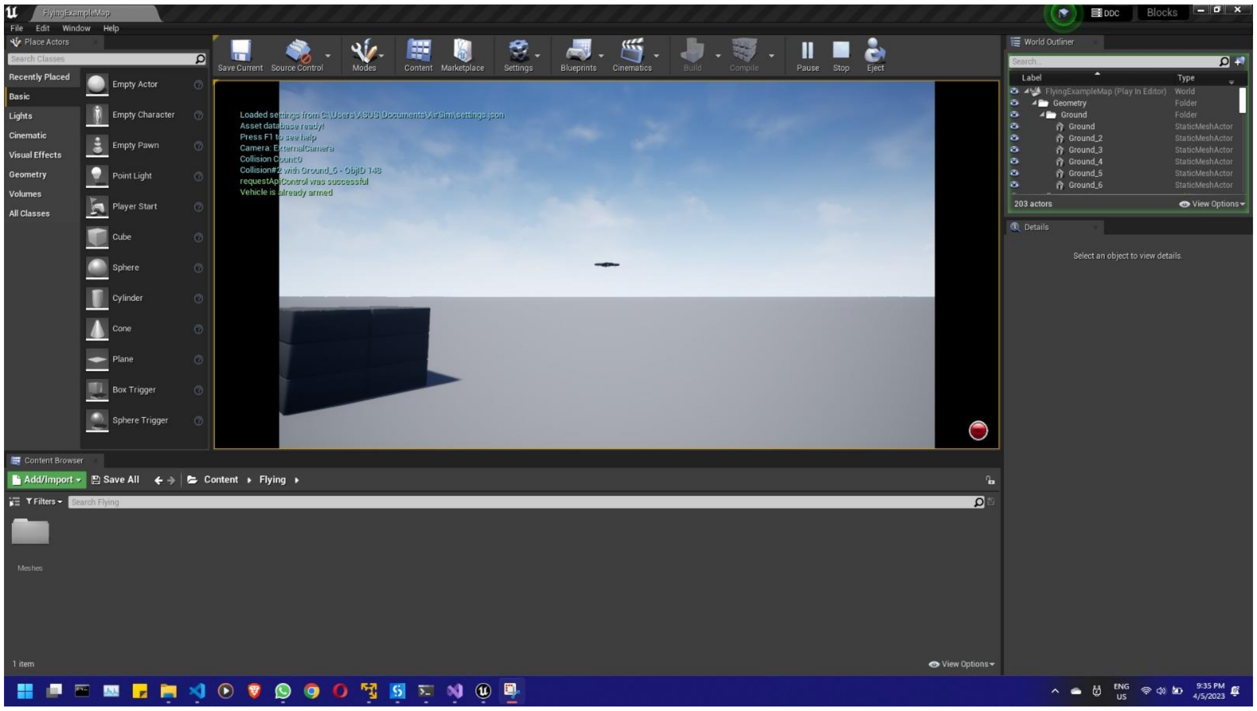
Use of this model in Drones:

The Publisher can be any component on the drone that generates data, such as a sensor or a flight controller. The Subscribers can be other components on the drone, such as an autopilot or a camera, or they can be external devices, such as a ground station or a remote controller.

The Publisher-Subscriber model in drones enables efficient and flexible communication between different components, allowing them to share information without the need for direct communication or knowledge of each other's existence. This can improve the overall performance and functionality of the drone, as well as make it easier to integrate with other devices and systems.

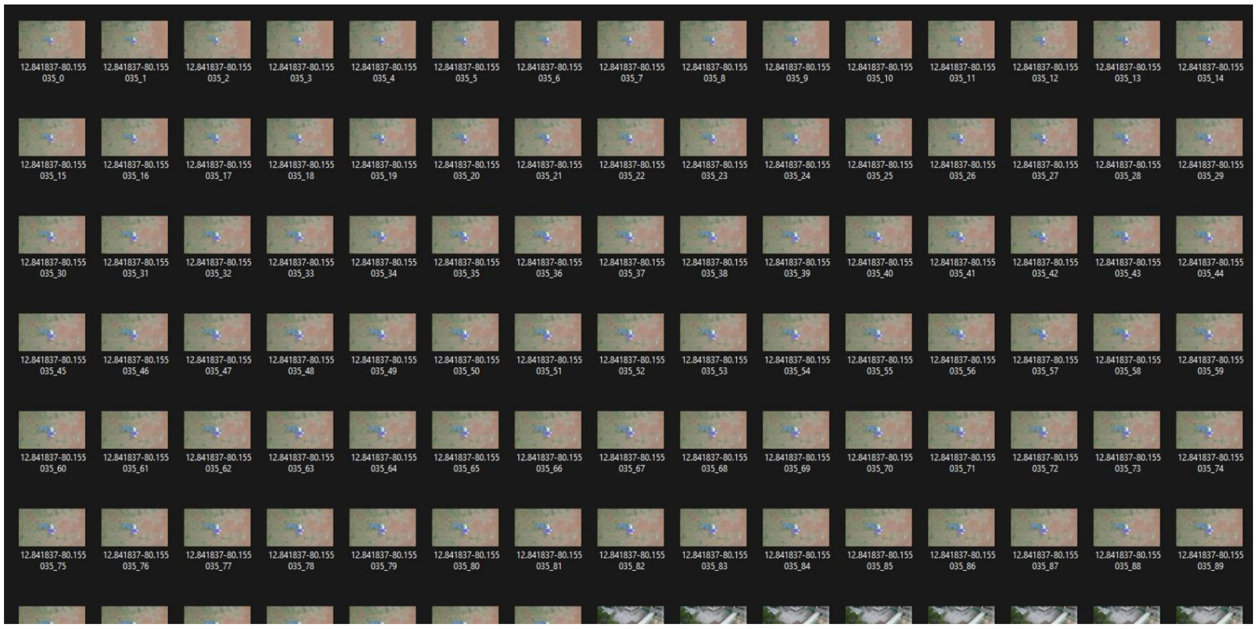
IMPLEMENTATION:

We have used AirSim drone simulator to make our drone simulator
In AirSim we can add the coordinates of the drone and we can also manage the speed of the drone



```
Loaded settings from C:\Users\ASUS\Documents\AirSim\settings.json
Asset database ready!
Press F1 to see help
Camera: ExternalCamera
Collision Count:0
Collision#2 with Ground_6 - ObjID 148
requestApiControl was successful
Vehicle is already armed
```

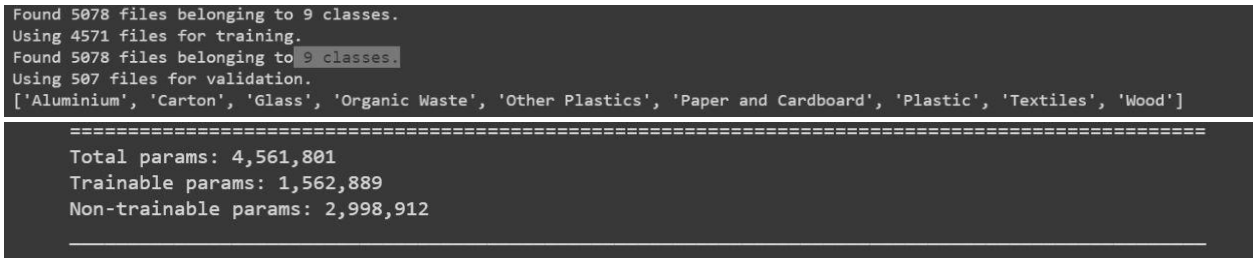
Once coordinates are set it will record the video of the environment and we are recording the video at 30fps .
Then we will extract frames from the video and it will be converted to image.



We will save the image name with its coordinate so that we can easily get the location of the waste

Then this image will be send to our deep learning model which is trained by using Mobilenetv3 pre-trained model and we have a total of 9 classes to classify different waste type.

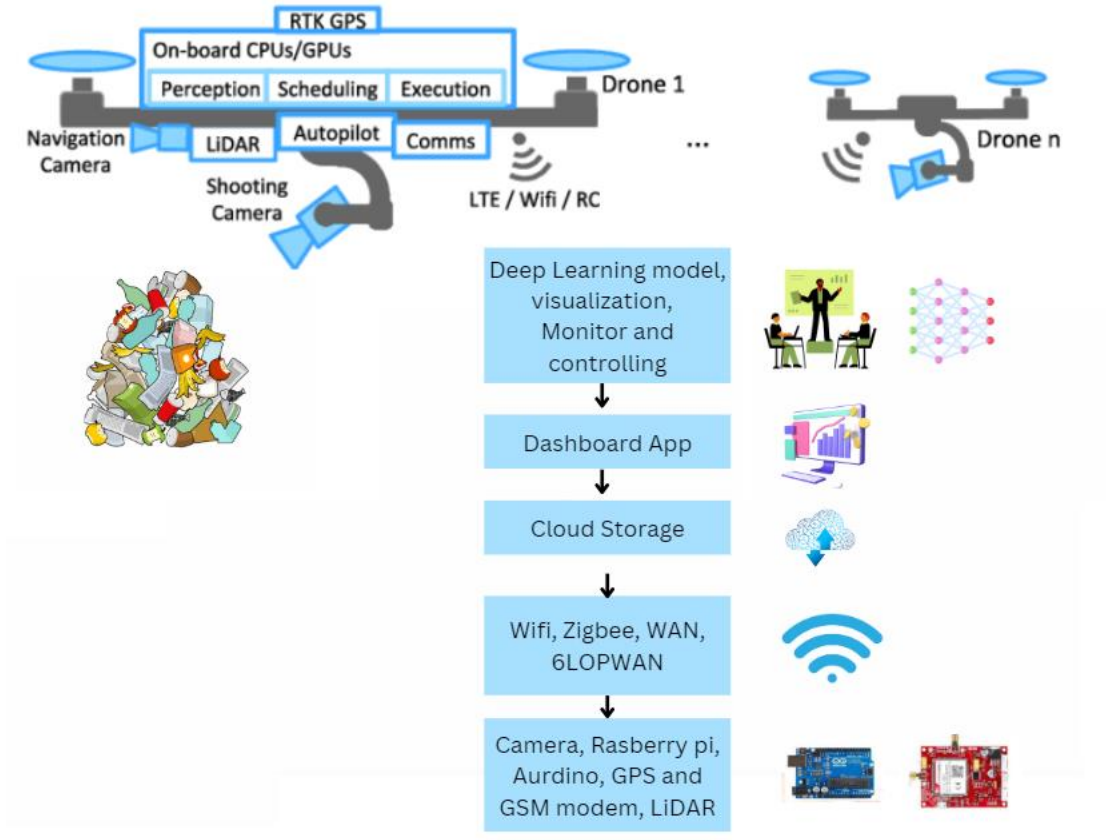
Total training image ->



```
Found 5078 files belonging to 9 classes.
Using 4571 files for training.
Found 5078 files belonging to 9 classes.
Using 507 files for validation.
['Aluminium', 'Carton', 'Glass', 'Organic Waste', 'Other Plastics', 'Paper and Cardboard', 'Plastic', 'Textiles', 'Wood']
```

```
=================================================================================================
Total params: 4,561,801
Trainable params: 1,562,889
Non-trainable params: 2,998,912
_________________________________________________________________________________________________
```

Layered Architecture: -



Above is a five layered architecture diagram for our project. Each layered can be described as -

Sensors –

Camera sensor to capture video of the survialance area.

GPS sensor to map the video to their respective coordinates

Communication –

Contains both large range (Wifi , WAN, 6LOPWAN) and short range (Zigbee) communication protocols.

Middleware –

Cloud layer for further processing.

Application –

Consist of a dashboard containing graphs and charts

Business –

Transforming data into meaningful information for Dicision support system.

Hardware and Software Components

Hardware Components:

Rasberry Pi –

Raspberry Pi is defined as a minicomputer the size of a credit card that is interoperable with any input and output hardware device like a monitor, a television, a mouse, or a keyboard effectively converting the set-up into a full-fledged PC at a low cost.

Camera Module –

The Raspberry Pi Camera Board is a custom designed add-on module for Raspberry Pi hardware. It attaches to Raspberry Pi hardware through a custom CSI interface. The sensor has 5-megapixel native resolution in still capture mode. In video mode it supports capture resolutions up to 1080p at 30 frames per second.

GPS SENSORS –

Raspberry Pi GPS Module is built with CP2102 as USB to UART Bridge chip, which is stable and faster. There is also an L80-39 GPS chip inside the chip. The L80-39 is with 66 search channels and 22 simultaneous tracking channels, which can help communicate satellite with UART or USB.

Software Components:

1. Computer vision software: This software analyses the video footage and applies code to extract each frame and convert it into image dataset.

2. GPS/Navigation software: This software helps in guiding the drone to specific locations where waste management is required.

3. Communication software: This software is responsible for processing data, sending and receiving the data such as images, videos, and other important mission-related data to and from the connected devices or servers.

4. Deep learning model: This is specialized model that improve the detection capabilities of the drone as it collects more data.

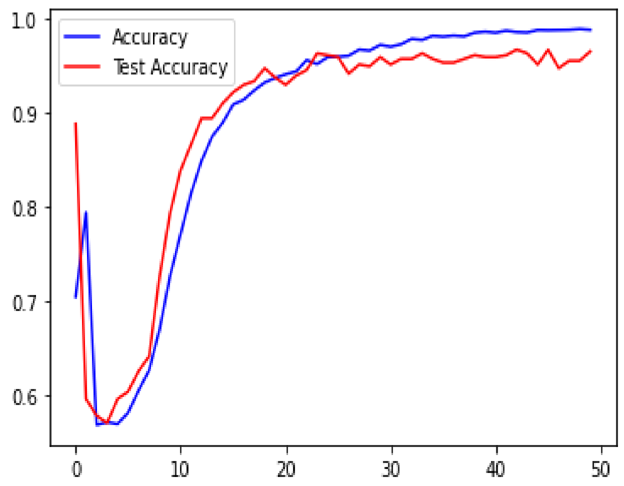
5. Operating system: The drone's operating system manages the resources and hardware of the drone, and ensures smooth functioning of all the software components.

6. Control software: This software controls the flight of the drone, including takeoff, landing, and maneuvering in flight.

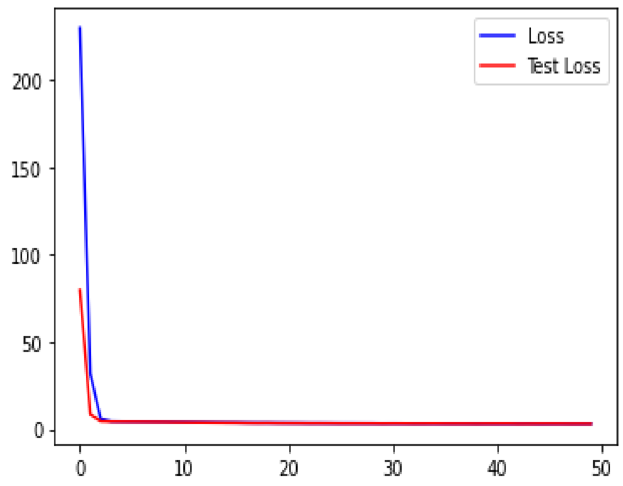
7. Data storage software: This software is responsible for storing the video footage, waste detection information, and other mission data.

Graph: -

Accuracy curve



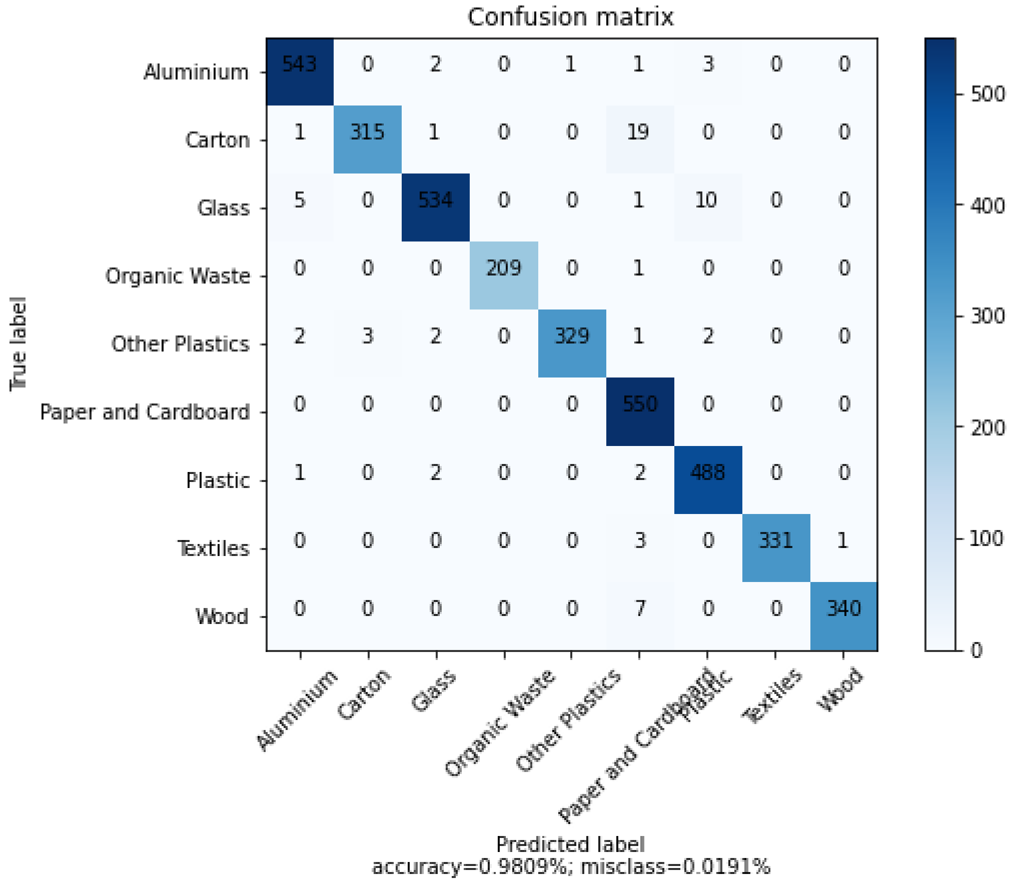
Training accuracy v/s Testing accuracy



Training loss v/s Testing loss

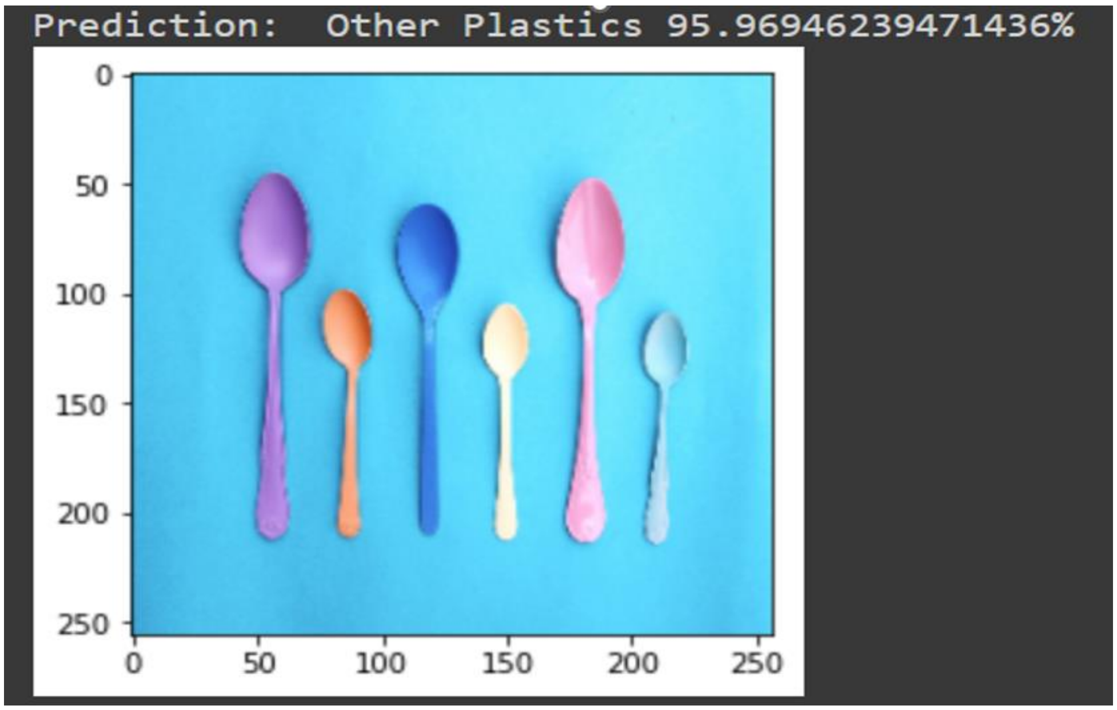
Performance

Confusion Matrix -



	Aluminium	Carton	Glass	Organic Waste	Other Plastics	Paper and Cardboard	Plastic	Textiles	Wood
Aluminium	543	0	2	0	1	1	3	0	0
Carton	1	315	1	0	0	19	0	0	0
Glass	5	0	534	0	0	1	10	0	0
Organic Waste	0	0	0	209	0	1	0	0	0
Other Plastics	2	3	2	0	329	1	2	0	0
Paper and Cardboard	0	0	0	0	0	550	0	0	0
Plastic	1	0	2	0	0	2	488	0	0
Textiles	0	0	0	0	0	3	0	331	1
Wood	0	0	0	0	0	7	0	0	340

PREDICTION IMAGE –

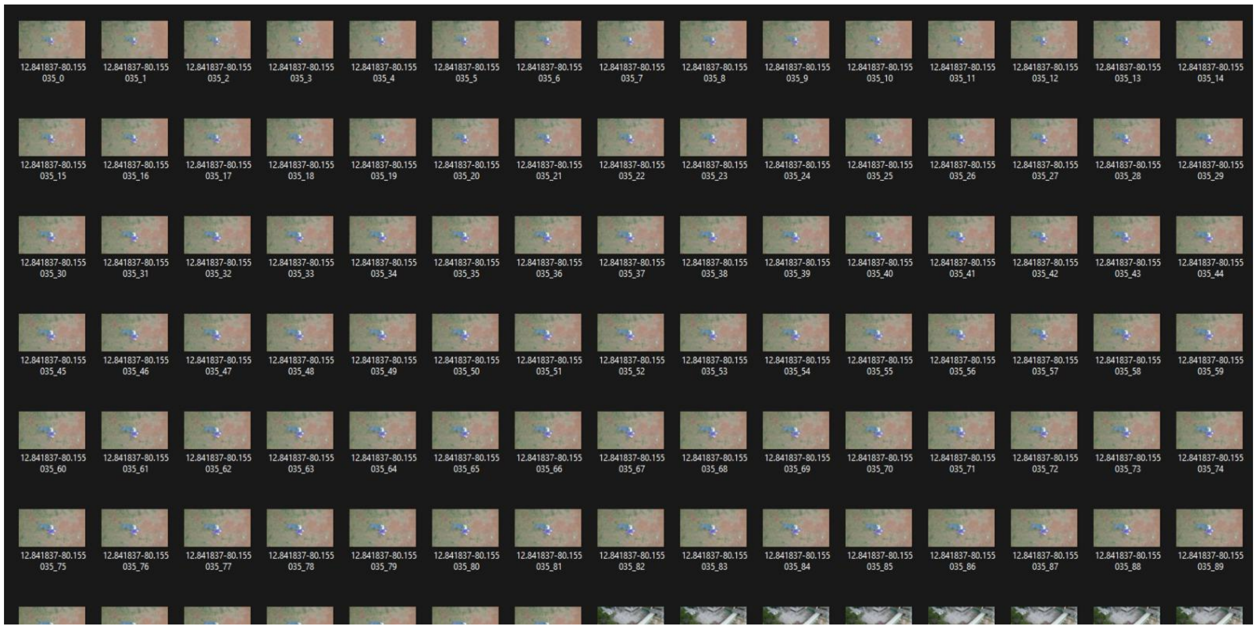


```
Prediction:  Other Plastics 95.96946239471436%
```

Results

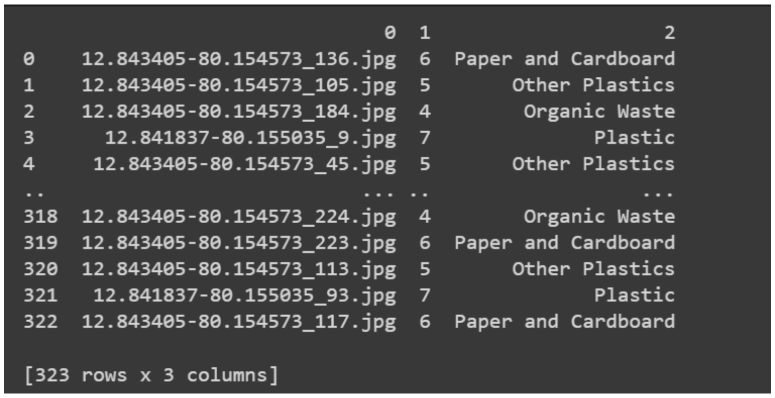
Testing data ->

We will test the 323 frames which got from our drone’s video –

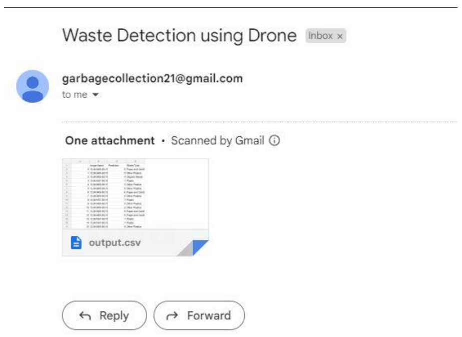


Testing results ->

Once we get the testing result we will save it as a csv file and then we will send the csv file to the cleanliness authorities through mail.



```
                             0  1                     2
0    12.843405-80.154573_136.jpg  6  Paper and Cardboard
1    12.843405-80.154573_105.jpg  5       Other Plastics
2    12.843405-80.154573_184.jpg  4        Organic Waste
3      12.841837-80.155035_9.jpg  7              Plastic
4     12.843405-80.154573_45.jpg  5       Other Plastics
..                           ... ..                  ...
318  12.843405-80.154573_224.jpg  4        Organic Waste
319  12.843405-80.154573_223.jpg  6  Paper and Cardboard
320  12.843405-80.154573_113.jpg  5       Other Plastics
321   12.841837-80.155035_93.jpg  7              Plastic
322  12.843405-80.154573_117.jpg  6  Paper and Cardboard

[323 rows x 3 columns]
```



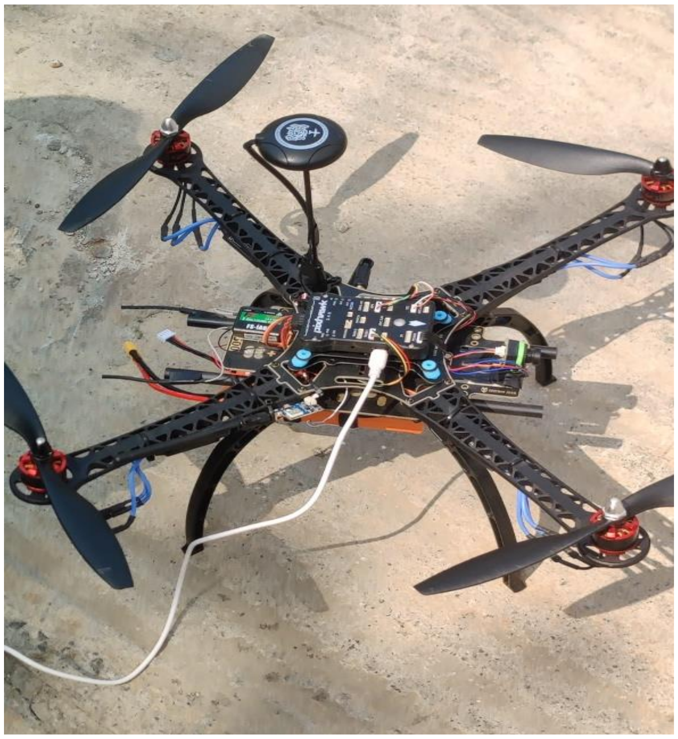
Waste Detection using Drone Inbox

garbagecollection21@gmail.com
to me

One attachment • Scanned by Gmail

output.csv

Drone image –

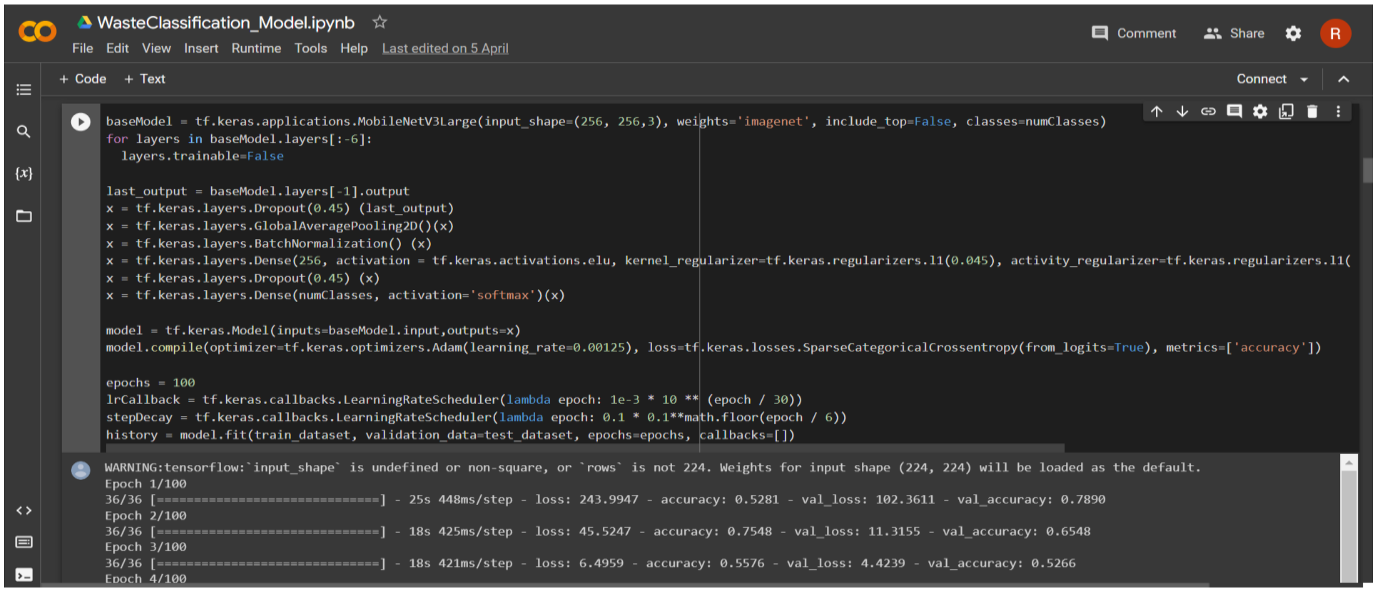


Conclusion

In conclusion, the proposed architecture for solid waste management through drones and cloud analysis offers a scalable, efficient, and cost-effective solution for managing waste. It can help reduce the environmental impact of waste and contribute to the sustainable development of our communities. However, the implementation of this architecture requires significant investment in hardware, software, and cloud resources, as well as regulatory approvals and community support.

Appendix

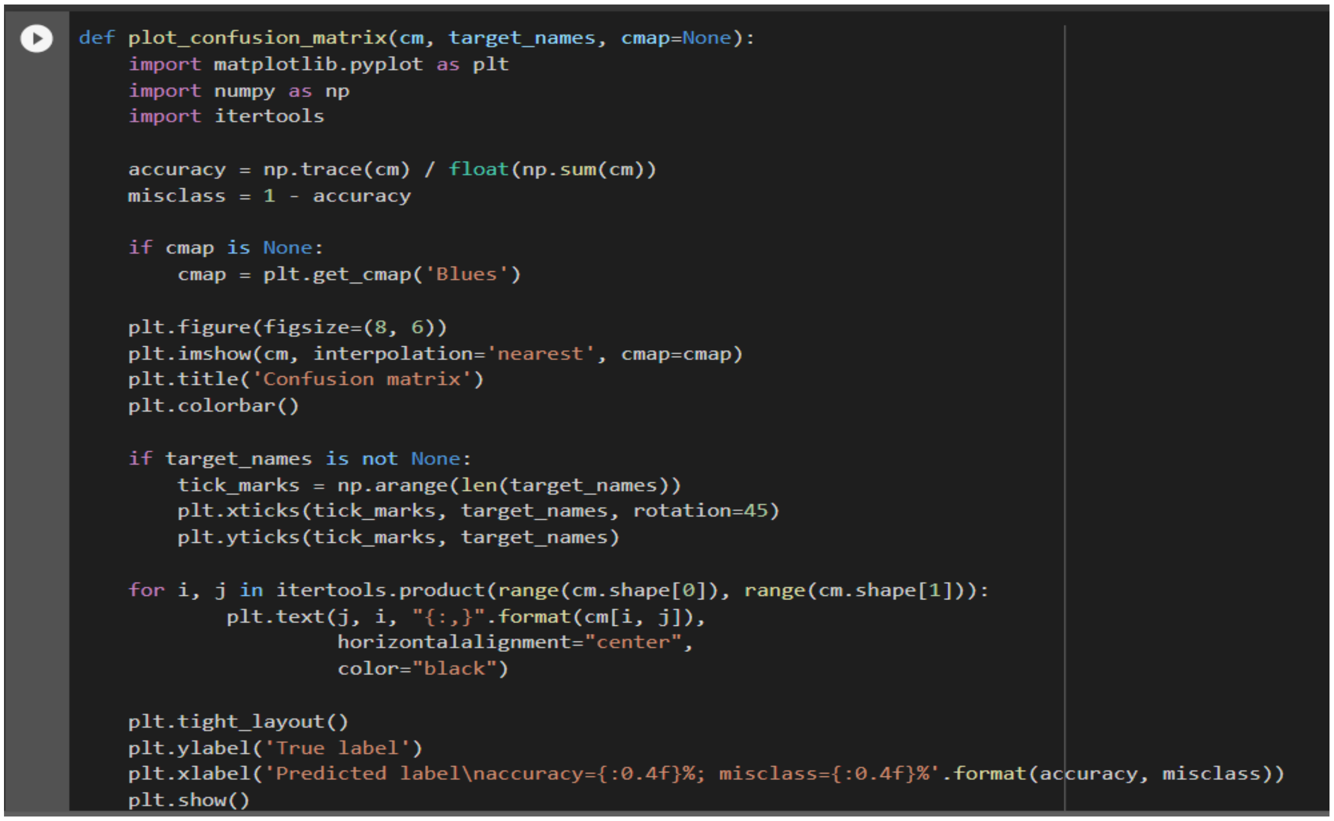
Deep Learning Model: -


WasteClassification_Model.ipynb
File Edit View Insert Runtime Tools Help Last edited on 5 April
Comment Share
+ Code + Text
Connect

```
baseModel = tf.keras.applications.MobileNetV3Large(input_shape=(256, 256,3), weights='imagenet', include_top=False, classes=numClasses)
for layers in baseModel.layers[:-6]:
  layers.trainable=False

last_output = baseModel.layers[-1].output
x = tf.keras.layers.Dropout(0.45) (last_output)
x = tf.keras.layers.GlobalAveragePooling2D()(x)
x = tf.keras.layers.BatchNormalization() (x)
x = tf.keras.layers.Dense(256, activation = tf.keras.activations.elu, kernel_regularizer=tf.keras.regularizers.l1(0.045), activity_regularizer=tf.keras.regularizers.l1(
x = tf.keras.layers.Dropout(0.45) (x)
x = tf.keras.layers.Dense(numClasses, activation='softmax')(x)

model = tf.keras.Model(inputs=baseModel.input,outputs=x)
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.00125), loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True), metrics=['accuracy'])

epochs = 100
lrCallback = tf.keras.callbacks.LearningRateScheduler(lambda epoch: 1e-3 * 10 ** (epoch / 30))
stepDecay = tf.keras.callbacks.LearningRateScheduler(lambda epoch: 0.1 * 0.1**math.floor(epoch / 6))
history = model.fit(train_dataset, validation_data=test_dataset, epochs=epochs, callbacks=[])
```

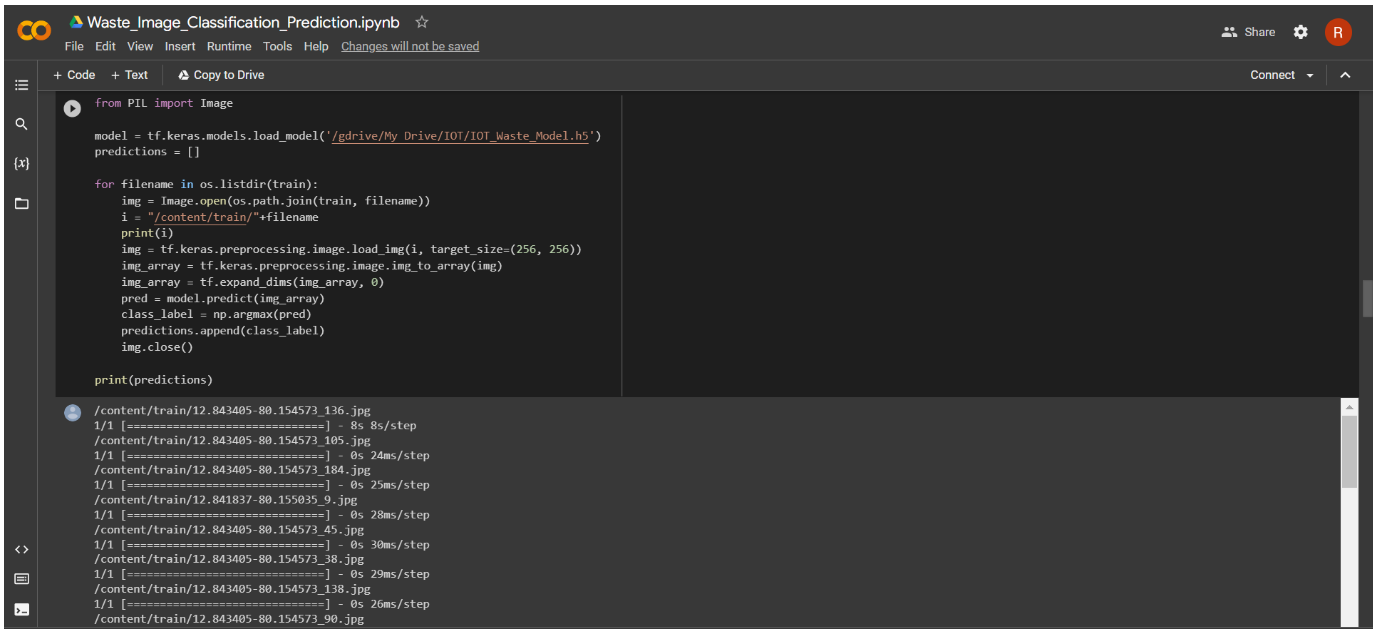
```
WARNING:tensorflow:`input_shape` is undefined or non-square, or `rows` is not 224. Weights for input shape (224, 224) will be loaded as the default.
Epoch 1/100
36/36 [==============================] - 25s 448ms/step - loss: 243.9947 - accuracy: 0.5281 - val_loss: 102.3611 - val_accuracy: 0.7890
Epoch 2/100
36/36 [==============================] - 18s 425ms/step - loss: 45.5247 - accuracy: 0.7548 - val_loss: 11.3155 - val_accuracy: 0.6548
Epoch 3/100
36/36 [==============================] - 18s 421ms/step - loss: 6.4959 - accuracy: 0.5576 - val_loss: 4.4239 - val_accuracy: 0.5266
Epoch 4/100
```

Confusion Matrix Code: -



```
def plot_confusion_matrix(cm, target_names, cmap=None):
    import matplotlib.pyplot as plt
    import numpy as np
    import itertools

    accuracy = np.trace(cm) / float(np.sum(cm))
    misclass = 1 - accuracy

    if cmap is None:
        cmap = plt.get_cmap('Blues')

    plt.figure(figsize=(8, 6))
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title('Confusion matrix')
    plt.colorbar()

    if target_names is not None:
        tick_marks = np.arange(len(target_names))
        plt.xticks(tick_marks, target_names, rotation=45)
        plt.yticks(tick_marks, target_names)

    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
            plt.text(j, i, "{:,}".format(cm[i, j]),
                     horizontalalignment="center",
                     color="black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label\naccuracy={:0.4f}%; misclass={:0.4f}%'.format(accuracy, misclass))
    plt.show()
```

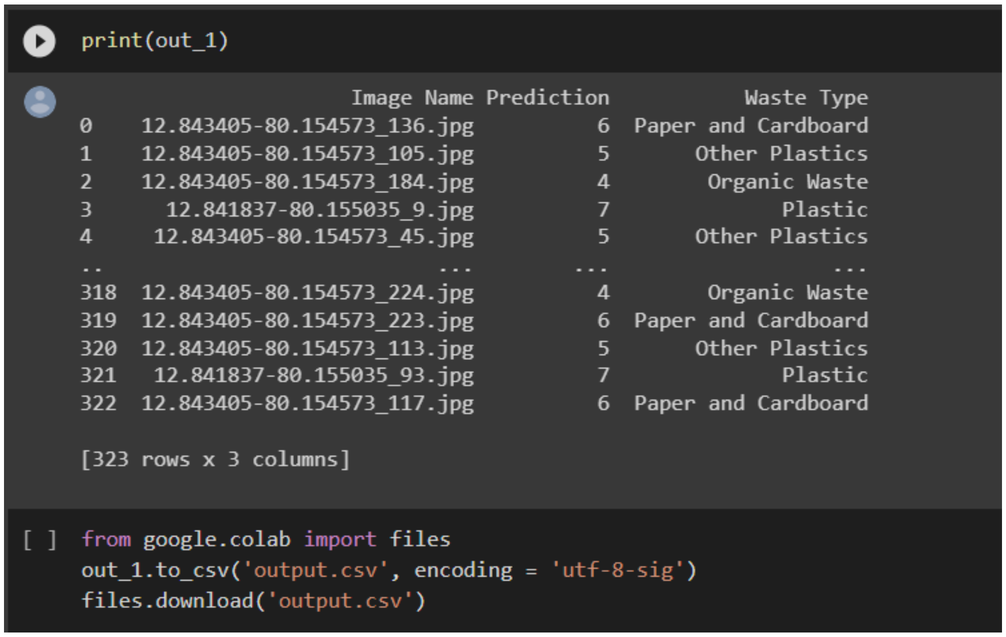
Image Classification: -


Waste_Image_Classification_Prediction.ipynb
File Edit View Insert Runtime Tools Help Changes will not be saved
Share
+ Code + Text
Copy to Drive
Connect

```python
from PIL import Image

model = tf.keras.models.load_model('/gdrive/My Drive/IOT/IOT_Waste_Model.h5')
predictions = []

for filename in os.listdir(train):
    img = Image.open(os.path.join(train, filename))
    i = "/content/train/"+filename
    print(i)
    img = tf.keras.preprocessing.image.load_img(i, target_size=(256, 256))
    img_array = tf.keras.preprocessing.image.img_to_array(img)
    img_array = tf.expand_dims(img_array, 0)
    pred = model.predict(img_array)
    class_label = np.argmax(pred)
    predictions.append(class_label)
    img.close()

print(predictions)
```

```
/content/train/12.843405-80.154573_136.jpg
1/1 [==============================] - 8s 8s/step
/content/train/12.843405-80.154573_105.jpg
1/1 [==============================] - 0s 24ms/step
/content/train/12.843405-80.154573_184.jpg
1/1 [==============================] - 0s 25ms/step
/content/train/12.841837-80.155035_9.jpg
1/1 [==============================] - 0s 28ms/step
/content/train/12.843405-80.154573_45.jpg
1/1 [==============================] - 0s 30ms/step
/content/train/12.843405-80.154573_38.jpg
1/1 [==============================] - 0s 29ms/step
/content/train/12.843405-80.154573_138.jpg
1/1 [==============================] - 0s 26ms/step
/content/train/12.843405-80.154573_90.jpg
```

Final Output and storing it in CSV File: -



```python
print(out_1)
```

```
                   Image Name Prediction          Waste Type
0    12.843405-80.154573_136.jpg          6  Paper and Cardboard
1    12.843405-80.154573_105.jpg          5       Other Plastics
2    12.843405-80.154573_184.jpg          4        Organic Waste
3      12.841837-80.155035_9.jpg          7              Plastic
4     12.843405-80.154573_45.jpg          5       Other Plastics
..                           ...        ...                  ...
318  12.843405-80.154573_224.jpg          4        Organic Waste
319  12.843405-80.154573_223.jpg          6  Paper and Cardboard
320  12.843405-80.154573_113.jpg          5       Other Plastics
321   12.841837-80.155035_93.jpg          7              Plastic
322  12.843405-80.154573_117.jpg          6  Paper and Cardboard

[323 rows x 3 columns]
```

```python
from google.colab import files
out_1.to_csv('output.csv', encoding = 'utf-8-sig')
files.download('output.csv')
```