Maynooth University
National University of Ireland Maynooth

Intelligent Systems Project
Glass Bottle Segregator
2023

Instructor

Student name	Student number
Donal Padraig Hanlon McCann	21473402
Nathan Ewnetu	21483214
Eoin James Keatley	21472754
Stephen Olaoluwa Oresanya	20426842

Dr Erivelton Nepomuceno

Contents

1. Introduction

The objective of this project is to understand and implement an intelligent system, to tackle one of the 17 sustainable Development Goals outlined in the United Nations' 2030 Agenda for Sustainable Development. Glass Recycling is a huge industry valued at 1,126.3 million dollars in 2020 [1]. This project aims to improve the efficiency of the glass recycling process in hopes to make glass recycling a more sustainable and cost-effective industry.

To achieve this the project will use the power of a neural network to classify glass bottles by their colour and sort them into the appropriate coloured bin ready for recycling. Glass is made with slightly different chemicals and manufacturing processes depending on the colour of the glass therefore sorting glass before recycling is a critical step to ensure maximum efficiency. Glass is also one of the only materials that can be 100% recycled without losing any of its properties, making it one of the most sustainable materials. This project aims to bring glass back to the forefront of the food and beverage industry, cutting down on the massive plastic bottle industry's whopping 481.6 billion bottles were used last year, of which only 9% are recycled [2].

2. Problem statement and Objectives of the project

The challenge addressed by this project is to ensure sustainable consumption and production patterns, as outlined in the UN's Sustainable Development Goal 12. Glass is a highly recyclable resource that can be reused infinitely without losing quality and does not release harmful chemicals during the recycling process, making it an attractive alternative to plastic. However, contamination is a major issue in glass recycling, as each colour of glass has a unique chemical composition and melting point. When mixed, the quality of the glass is reduced, and in some cases, it may even need to be sent to a landfill.

To address this problem, the project proposes the use of a neural network program to identify and sort glass bottles by their colour. The system will use a conveyor belt mechanism with air pressure to eliminate any loose rubbish or plastic bottles and ensure that only glass bottles remain on the belt. The neural network will be trained on a large dataset of glass bottle images of a certain number of colours and will continuously learn as the system is used in real-world applications. Anomaly detection will be used to find data points that are significantly different from the norm, triggering the general waste servos to ensure the quality of the sorted glass. The proposed solution aims to make glass a more desirable material for companies to switch to instead of plastic, thereby promoting sustainable consumption and production patterns.

3. Background

3.1 UN's 17 Sustainable Development goals:

The United Nations' 17 Sustainable Development Goals (SDGs) were set up in 2015 as a universal call to action to end poverty, protect the planet, and ensure prosperity for all. They aim to balance sustainable development's economic, social, and environmental dimensions and address issues such as poverty, hunger, access to education and healthcare, gender equality, climate change, biodiversity loss, and sustainable use of resources [3]. The SDGs are integrated and indivisible, representing a comprehensive framework for a sustainable future.

This project aligns with the above SDGs: Goal 12: Responsible Consumption and Production, Goal 13: Climate Action, and Goal 14: Life Below Water [4]. The project's aims are to promote sustainable consumption and production practices, reduce greenhouse gas emissions, and minimize the impact of waste on marine life. The successful implementation of this project can promote a circular economy, reduce the environmental impact of waste, and contribute to achieving the UN's SDGs.

3.2 Neural Networks:

3.2.1: What are Neural Networks:

Artificial Neural Networks (ANNs), also known as Neural Networks (NNs), are computing systems inspired by the structure and functionality of the human brain [5]. ANNs are designed to recognize patterns in data and learn from them, making them powerful tools for machine-learning applications.

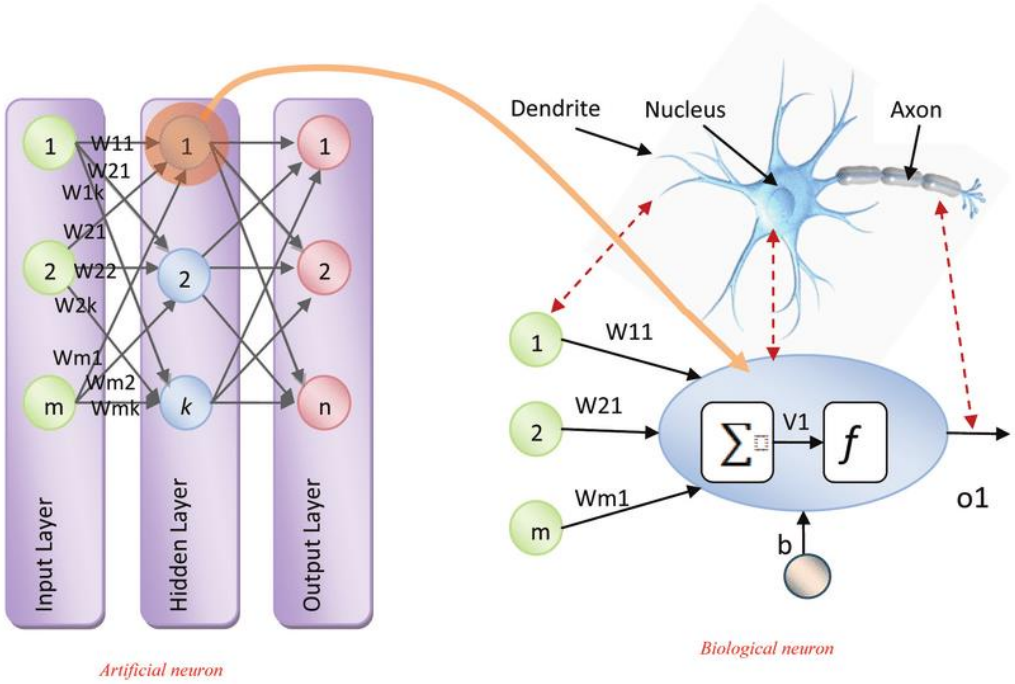


Figure 1- Differences and Similarities between an artificial and biological neuron [F1]

The basic building block of a neural network is a neuron, which receives input signals from other neurons or external sources and processes them to produce an output signal [6]. A neuron consists of an input layer, an output layer, and one or more hidden layers, which contain weights that adjust the strength of the input signals. The weighted sum of its inputs determines the output of a neuron passed through an activation function. The activation function helps to introduce nonlinearity into the model, enabling it to learn more complex patterns.

Training a neural network involves feeding input data and the corresponding output and adjusting the neurons' weights to minimize the error between the predicted and actual outputs. The process of adjusting the weights is called backpropagation, which uses an optimization algorithm such as stochastic gradient descent to find the optimal set of weights [5].

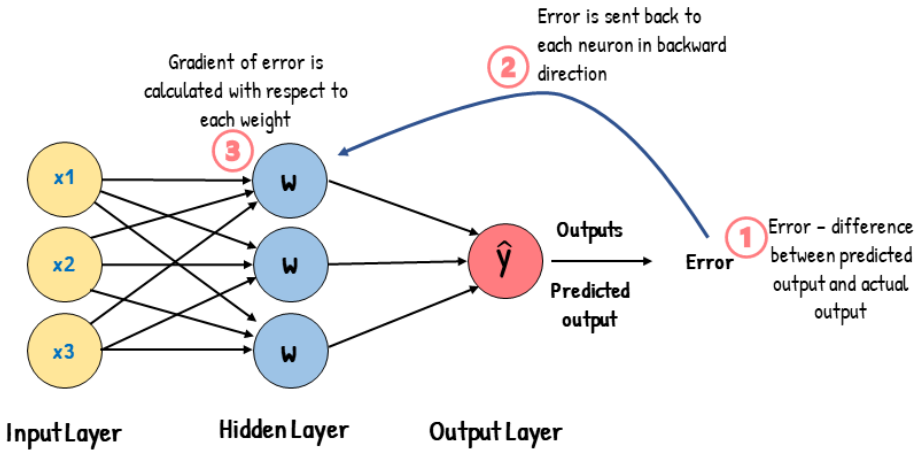


Figure 2- Example of how backpropagation works [F2].

Neural networks have shown remarkable success in various applications, such as image and speech recognition, natural language processing, and prediction tasks [7]. They are also increasingly used in healthcare, finance, and autonomous driving industries.

3.2.2 Neural Network types:

The different neural networks available to use within this projects scope are vast, however the group has identified two possible neural network structures that can be utilised to create our system. Firstly, the group identified a convolutional neural network (CNN) as a strong possibility for use within the system. This type of network is used mainly in object identification and image classification. This network has multiple layers within its structure such as convolution layers, pooling layers and fully connected layers. The convolutional layers are utilised within the extraction of features from provided input images by convoluting them with different filters. This data is then passed to the pooling layer which compresses the data down to a smaller size for use later. The fully connected layers work on the data from the pooling layers and focus on classifying the data into different categories.

The group also considered other neural network structures, such as Cognitive Neural Networks (CogNN) and Artificial Neural Networks (ANN) , before ultimately deciding on the Convolutional Neural Network (CNN) for this project.

Cognitive Neural Networks are a type of network designed to mimic the human brain's cognitive processes by incorporating elements of attention, memory, and decision-making. While CogNNs have shown promising results in tasks requiring high-level reasoning and understanding, their complexity and computational demands make them less suitable for the image processing and classification tasks in this project [8].

Artificial Neural Networks, on the other hand, are a broader class of networks encompassing several types of neural networks, including CNNs (Convolutional Neural Network) and RNNs [9]. ANNs (Artificial Neural Networks) are inspired by the structure and function of biological neural networks, and they consist of interconnected nodes, or neurons, that process information and learn patterns from input data. While ANNs can be used for various tasks, their generic nature and lack of specialized layers for specific tasks, such as image processing, make them less efficient compared to CNNs in this project [9].

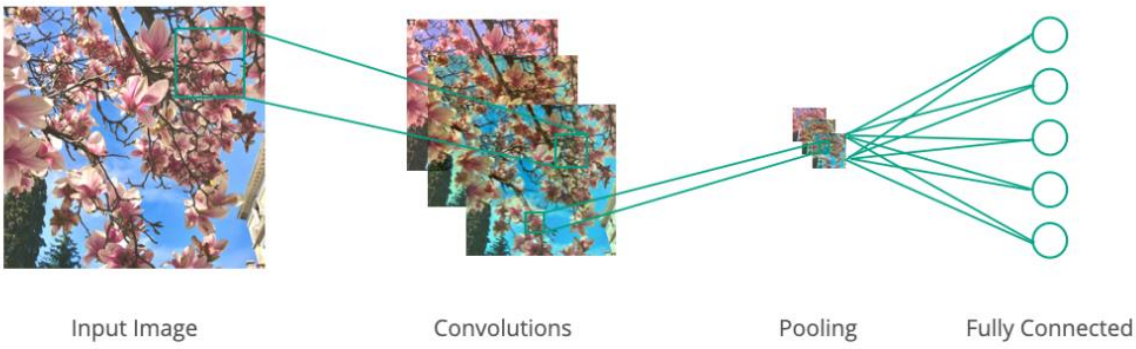


Figure 3- Example of a Convolutional Neural Network [F3]

The chosen Convolutional Neural Network offers several advantages for this project. CNNs are specifically designed for image processing tasks and have been proven to be highly effective in tasks such as object identification and image classification. The specialized layers in CNNs, such as convolution layers and pooling layers, enable the network to efficiently extract features from input images and reduce computational demands [5].

In the context of this project, the Convolutional Neural Network was deemed the best fit due to its proven effectiveness in image-related tasks, as well as its specialized layers and structure tailored for processing and classifying images as seen in *figure 3* [5]. The other neural network structures, while having their own merits, were not considered as suitable for the project's specific requirements.

3.3 History:

The history of neural networks spans several decades, from the early models proposed in the 1940s to the complex models developed today. The history of neural networks dates to the 1940s when Warren McCulloch and Walter Pitts proposed a model for artificial neurons [10]. Their model consisted of binary input signals combined in a weighted sum to produce a binary output signal. The McCulloch-Pitts neuron became the foundation for developing artificial neural networks, now a fundamental tool for machine learning applications.

In the 1950s and 1960s, researchers began to develop models for artificial neural networks, including the Perceptron model proposed by Frank Rosenblatt [11]. The Perceptron model was able to learn simple patterns, but it was limited in its ability to learn more complex patterns. This limitation led to a decline in interest in neural networks in the following decades.

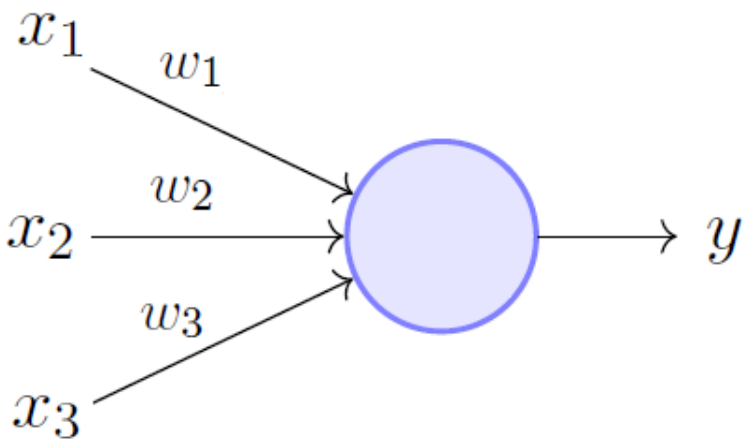


Figure 4 – The Perceptron model (Minsky-Papert 1969) [F4]

In the 1980s and 1990s, however, neural networks experienced a resurgence of interest, thanks to the development of new algorithms and the increasing availability of computational resources. The backpropagation algorithm, which allows neural networks to learn from input data by adjusting the neurons' weights, was developed in the 1980s by David Rumelhart, Geoffrey Hinton, and Ronald Williams [12]. Backpropagation enabled training neural networks with multiple layers or deep neural networks to learn more complex patterns.

In the 2000s and 2010s, deep neural networks became increasingly widespread, thanks to the availability of large datasets and powerful computing resources. Convolutional Neural Networks (CNNs), now commonly used for image recognition tasks, were introduced in the 1990s by Yann LeCun and his colleagues [13]. Recurrent Neural Networks (RNNs), which can process data sequences, were also developed in the 1990s by Sepp Hochreiter and Jürgen Schmidhuber [14]. These models have since been used in miscellaneous applications, including speech recognition, natural language processing, and autonomous driving.

While interest in neural networks declined in the 1970s and 1980s, new algorithms and computational resources led to a resurgence of interest in the 1990s and beyond. Today, neural networks are an essential tool for machine learning applications, and they continue to evolve and improve with new research and development.

3.4 Current Technology: Artificial Intelligence in Waste Disposal:

Artificial Intelligence has impacted many industries in recent years, such as the waste disposal industry. Waste disposal companies have embraced AI technologies to improve efficiency, reduce costs, and increase recycling rates [15].

One of the most known AI systems used in waste sorting is the Material Recovery Facility (MRF) [16]. MRFs use a combination of robotics and AI technologies to sort waste into various categories, including plastics, paper, and metals. MRFs can process copious amounts of trash in a short amount of time and can achieve high recycling rates [17].

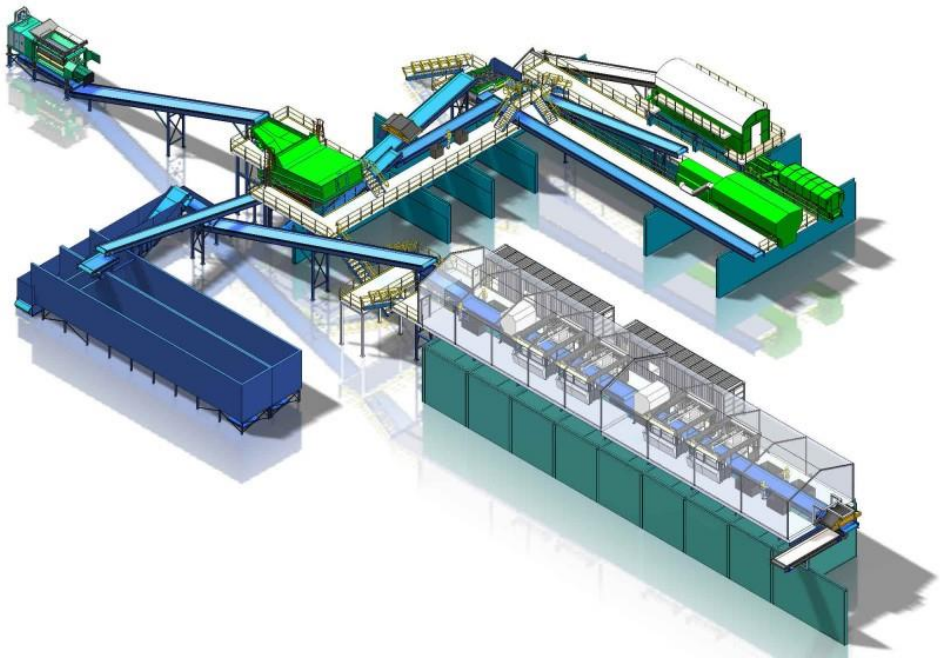


Figure 5 – Material Recovery Facility (MRF) [F5]

Another company, Covanta, has also developed an AI-powered system that uses data analytics to optimise waste-to-energy plants [18]. The system analyses data from various sources, including weather patterns and waste composition, to maximise plant efficiency and reduce emissions.

Another example of AI in waste disposal is using robotic arms in waste collection [19]. The Robotic Waste Collection System, developed by the European Union's Horizon 2020 project, uses robotic arms to collect and transport waste bins to the collection vehicle. The system has sensors and cameras that enable the robotic arms to identify and locate the waste bins. The robotic arms then use machine learning algorithms to pick up and transport the waste bins to the collection vehicle [20]. This system has been shown to reduce the risk of injury to waste collectors and improve the efficiency of waste collection processes.

Using AI in waste sorting, collection, and landfill management has made waste disposal more sustainable and environmentally friendly. As AI technologies continue to evolve, we expect to see further advancements in the field of waste disposal [21].

4. Lit Review:

Recent studies have demonstrated the potential of machine learning, particularly neural networks, to improve waste management processes [22]. A study published in the Journal of Cleaner Production in 2021 showed that a machine-learning model could accurately identify and sort different types of plastic waste, including high-density polyethylene (HDPE), polyethylene terephthalate (PET), and polystyrene (PS), with an accuracy of up to 98% [22]. This approach could significantly improve recycling efficiency, reduce landfill waste, and lower the environmental impact of waste management.

Another study published in the Journal of Environmental Management in 2020 used a deep convolutional neural network (CNN) to classify mixed waste items based on their material composition [23]. The CNN organized waste items such as glass, paper, and plastic with an overall accuracy of 94.5%. The authors suggested that this approach could be instrumental in waste management facilities, where sorting mixed waste is challenging and time-consuming.

Similar to these studies, our proposed project aims to use a neural network to identify and sort waste materials. However, our project focuses on glass bottles rather than plastic or mixed waste. Studies on machine learning applications in waste management have highlighted the potential of these techniques for optimizing municipal solid waste collection [24]. One such study used a support vector machine algorithm to predict waste generation in different neighbourhoods based on population density, land use, and demographic data, achieving a correlation coefficient of 0.95 [24].

The method of glass segregation currently in use is manual sorting by colour, with clear, green, and brown being the primary divisions [25]. This segregation is vital due to the unique chemical compositions and melting points of different glass varieties. However, manual sorting is both time-consuming and prone to errors, leading to possible contamination of the recycled glass. In addressing this issue, a study by Lijuan Li [25] utilized a convolutional neural network (CNN) to classify glass based on its colour and shape, achieving an accuracy of 98.5%.

Another study by Pranab Mandal [27] used a deep neural network to categorize glass based on its colour and chemical composition, attaining an accuracy of 95%. Both studies underscore the potential of neural networks in the recycling industry. In addition to colour, other properties of glass can also be used for classification. For instance, a study by Cheng Lei [24] used a neural network to classify glass based on its fluorescence properties. The neural network could accurately classify glass into four categories based on its fluorescence spectrum, demonstrating the potential for using additional properties to improve glass classification.

Machine learning has also been used to optimize waste-to-energy processes, contributing to a more sustainable waste management system [28]. Additionally, robotic waste collection systems have been developed for smart cities, further demonstrating the potential of machine learning and automation in waste management [28,29].

In conclusion, our project has the potential to improve the efficiency of glass bottle sorting and contribute to a more sustainable future. By sorting glass bottles based on their type and colour, we can reduce the amount of waste that ends up in landfills and ensure that materials are properly recycled or reused. The success of neural networks in previous studies supports the feasibility and potential effectiveness of our proposed approach.

5. Implementation:

5.1 Overview:

To develop our CNN glass colour segregator, we used Jupyter Notebook as our Python virtual environment and wrote all the code using TensorFlow, Keras API and OpenCV libraries. The following steps were taken to develop the project:

5.2 Dataset Preparation:

Data set preparation is a crucial step in building a successful machine learning model, especially for image classification tasks. We collected a dataset of glass bottle images of different colours, including green, clear, and brown, from various sources. They were put into a data directory and split into separate files based of colour. The dataset was then cleaned.

```
data_dir = r'C:\Nathans Projects\EE297_Final\data'
```

```
image_exts = ['jpeg','jpg', 'bmp', 'png']
```

```
for image_class in os.listdir(data_dir):
    for image in os.listdir(os.path.join(data_dir, image_class)):
        image_path = os.path.join(data_dir, image_class, image)
        try:
            img = cv2.imread(image_path)
            tip = imghdr.what(image_path)
            if tip not in image_exts:
                print('Image not in ext list {}'.format(image_path))
                os.remove(image_path)
        except Exception as e:
            print('Issue with image {}'.format(image_path))
            os.remove(image_path)
```

Figure 6 – Code used to get rid of bad images from data set

Figure 7 – Sample from data set

The code in *figure 6* code ensures that the data set only contains high-quality images in the specified format, which can improve the performance of the machine learning model. *Figure 8* is an example of the bad images getting remove. However, it is also important to ensure that the data set is diverse and representative of the real-world scenarios that the model will be applied to. To achieve a diverse and representative data set, additional steps such as data augmentation, balancing class distributions, and collecting more data may be necessary.

```
Image not in ext list C:\Nathans Projects\EE297_Final\data\brown_glass\192px.svg
Image not in ext list C:\Nathans Projects\EE297_Final\data\brown_glass\beershopall2.png
Image not in ext list C:\Nathans Projects\EE297_Final\data\brown_glass\vector.svg
```

Figure 8 – Example of the output code of the faulty images from data set getting removed from the directory.

5.3 Data Loading & Pre-processing:

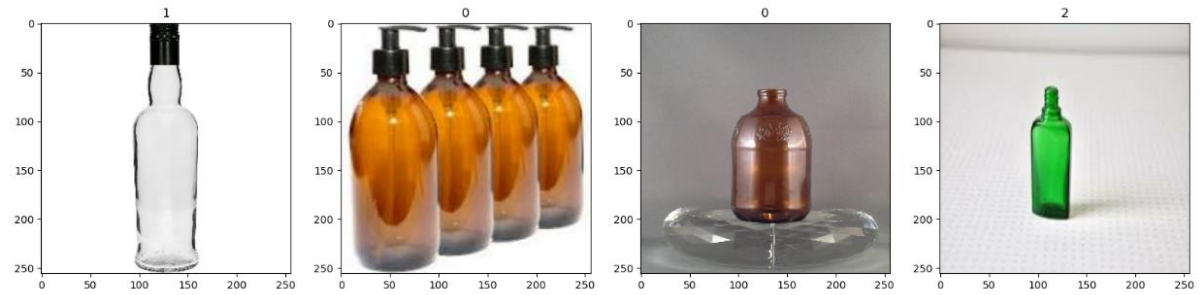


Figure 9 – Output of function to see if the system recognises different colours.

To see if our data set can be loaded in, we us Keras API to create a data iterator and load in a batch of images and their labels. This helps us visualise and confirm our dataset can be loaded in correctly and that the labels match the corresponding images. As seen in *figure 9* the brown bottles are labelled as 0 and the clear bottles as 1 and the green as 2.

We used OpenCV to pre-process the images by applying various transformations such as resizing, cropping, and normalization to improve the model's performance. We resized our images to speed up the process to 256 x 256 pixels to reduce the training time.

5.4 Setting up Training and Evaluation:

The directories for training and evaluation were then created by making an algorithm to randomly split the dataset into training and evaluation directories. Subdirectories were also created in the evaluation and training directories to ensure the images were still split by colour. The percentages for the training and evaluation split were set at 80% for the training and 20% for the evaluation. The more images the training set had, the more sufficient the network could become.

Figure 10 – The new train and evaluation directories made.

5.5 Data Augmentation:

Data augmentation is a technique used to artificially increase the size of the dataset by applying transformations to the existing data, such as rotation, zoom, flip, and shift. This helps to increase the diversity of the data and prevent overfitting, which occurs when the model performs well on the

training data but poorly on the testing data. In this step, we use the ImageDataGenerator class from Keras to perform data augmentation.

```
train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)

val_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(train_dir,target_size=(256, 256),batch_size=32,class_mode='categorical')

val_generator = val_datagen.flow_from_directory(val_dir,target_size=(256, 256),batch_size=32,class_mode='categorical')
```

Figure 11 – Data Augmentation

5.6 Model Training:

In this step we trained our model using the augmented dataset. Our model is trained using the Adam optimiser and categorical cross-entropy loss function. The model consists of three sets of convolutional and max pooling layers, followed by a flatten layer, two dense layers, and a SoftMax output layer. The first convolutional layer has 16 filters, the second convolutional layer has 32 filters, and the third convolutional layer has 64 filters. The dense layers have 256 and 3 neurons, respectively.

These parameters were selected from trial and error. The first layer had 16 filters as the input images are not too complex. The second layer had twice the amount of the first layer as its common practice because it allows for more complex features to be learned. The same is true for the third layer as it is again twice the size of the second layer. The dense layer was gotten through tuning. It is a high number because it has multiple layers preceding it and therefore many features which are extracted need to be learned. The output layer has 3 neurons which corresponds to the output amount. The choice of the Adam optimizer and categorical cross-entropy loss function are standard choices for multi-class classification problems.

```
18 model = Sequential()
19
20 model.add(Conv2D(16, (3,3), 1, activation='relu', input_shape=(256,256,3)))
21 model.add(MaxPooling2D())
22 model.add(Conv2D(32, (3,3), 1, activation='relu'))
23 model.add(MaxPooling2D())
24 model.add(Conv2D(64, (3,3), 1, activation='relu'))
25 model.add(MaxPooling2D())
26 model.add(Flatten())
27 model.add(Dense(256, activation='relu'))
28 model.add(Dense(3, activation='softmax'))
29
30 model.compile(optimizer=Adam(), loss=CategoricalCrossentropy(), metrics=['accuracy'])
31
32 logdir='logs'
33 tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=logdir)
34
35 hist = model.fit(train_generator,epochs=20,validation_data=val_generator, callbacks=[tensorboard_callback])
```

Figure 12 – Training of the model

The *model.compile()* function compiles the model using the specified optimizer, loss function, and accuracy as the evaluation metric. The *model.fit()* function is then called to train the model for 20 epochs using the *train_generator* as the training data and *val_generator* as the validation data. During training, the *tensorboard_callback* is used to log the training progress to the TensorBoard for analysis.

The accuracy, evaluation accuracy, loss and valuation loss can all be obtained from the code if *figure 12*.

5.7 Evaluation:

The next stage is to assess the model's performance after training. The loss and accuracy metrics of the training and validation datasets are frequently examined during evaluation. The loss metric represents how well the model can minimize the error between its predictions and the actual values, while the accuracy metric represents how often the model is correct.

During training, the model's loss and accuracy on both the training and validation datasets are recorded after each epoch. This information can be seen using graphs.

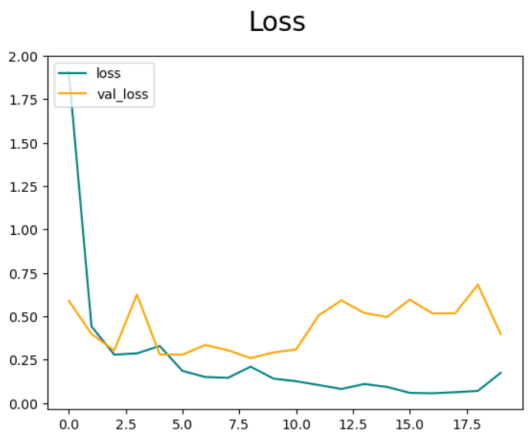


Figure 13 – Loss

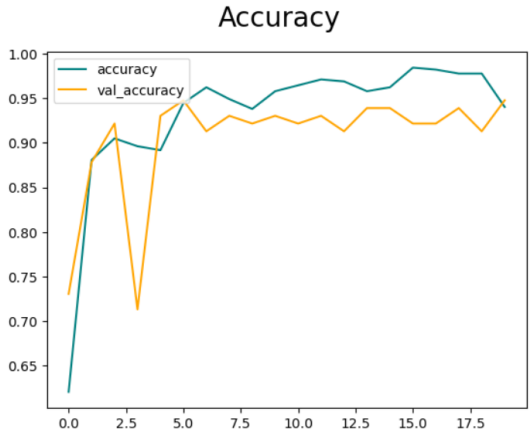


Figure 14 - Accuracy

The loss curve shows how the loss (i.e., error) on the training and validation datasets changes throughout training. Ideally, the loss should decrease steadily on the training and validation datasets, indicating that the model is learning from the data and generalizing well to new examples. An indication of overfitting arises in our model. This means the model may have become too specialized for the training data and needs to generalize better to new examples. The difference between these losses begin to decrease at the final epochs meaning our model should be ok. The loss value is a continuous value that can range from 0 to infinity. Ideally, the loss is often clipped to a maximum value (e.g., 2), so that it does not become too large.

Similarly, the accuracy curve shows how the accuracy of the training and validation datasets changes throughout training. It is ranges from 0 to one on the y axis as it measures the percentage of correctly predicted labels out of all the samples in the dataset. Ideally, the accuracy should increase steadily on both the training and validation datasets, indicating that the model is improving at making correct predictions. Although some early epochs may have large drops, our model achieves this steady increase rate. Our evaluation and accuracy are similar, meaning our system is balanced. This similarity means our model is learning from the dataset, and its performance is going well.

5.8 Testing:

After training and saving the CNN model for the glass colour segregator, we can evaluate its precision on images not included in the training or validation datasets. The model's predictions were validated using an image that was downloaded from the internet. We first pre-processed the image using the same procedures as in the data preparation stage to achieve this. The trained model was then used to predict an image after pre-processing. The projected class for the provided image is the one with the highest probability. To assess how accurately the model predicted the class of the image, we compared it to the actual class of the image.

Figure 15 – image used to test.

```python
import tensorflow as tf
from tensorflow.keras.preprocessing import image
import numpy as np

# Load the image from file
img_path =  r'C:\Users\natew\Downloads\download (3).jpg'
img = image.load_img(img_path, target_size=(256, 256))
img = image.img_to_array(img)
img = np.expand_dims(img, axis=0)

# Normalize the image
img /= 255.0

# Make a prediction with the model
prediction = model.predict(img)
print(prediction)

# Print the predicted class
if prediction[0][2] > 0.33:
    print('The bottle is green with probability {:.2f}%'.format(prediction[0][2]*100))
elif prediction[0][0] >0.33:
    print('The bottle is brown with probability {:.2f}%'.format(prediction[0][0]*100))
elif prediction[0][1] > 0.33:
    print('The bottle is clear with probability {:.2f}%'.format(prediction[0][1]*100))
```

Figure 16 – Test image code

```
1/1 [==============================] - 0s 436ms/step
[[9.9951839e-01 4.8067234e-04 9.1990051e-07]]
The bottle is brown with probability 99.95%
```

Figure 17 - Output

The first line "1/1 [==============================] - 0s 436ms/step" indicates that one input image was processed in one step, with a processing time of 436ms. The second line represents the output of prediction value which shows us the predicted values for each of the three classes. The first number is the probability for the brown class which is here 99.95%, the second number is for the green class which lies at 0.04806% and the final number represents the clear class which lies at 0.0000919%.

5.9.1 Deployment:

With our model working accurately all we needed was to implement it onto an Arduino. To do this we copied the code from *figure 16* and added a few lines onto it.

```
# Connect to the Arduino Uno
ser = serial.Serial('COM5', 9600)
```

Figure 18 – connects to the Arduino.

The Serial class is used to establish a communication between the Arduino and the computer as seen in *figure 18.* The trained model is then loaded into memory from the specified file path. The script then enters an infinite loop, which continuously captures an image from the webcam, resizes it, converts it to a NumPy array, and makes a prediction using the trained model.

```
# Define the labels for each class
labels = ['brown', 'green', 'clear']
while True:

        # Capture an image from the webcam
        cap = cv2.VideoCapture(0)
        ret, frame = cap.read()
        cap.release()
        # Resize the image
        img = Image.fromarray(frame)
        img = img.resize((256, 256))
        # Convert the image to a numpy array
        x = np.array(img)
        # Expand the dimensions of the array to match the model's input shape
        x = np.expand_dims(x, axis=0)
        # Make the prediction
        prediction = model.predict(x)

        # Get the predicted class label
        predicted_class = labels[np.argmax(prediction)]
```

Figure 19 – The model making a prediction based of loop of images it is taking.

Based on the predicted class label, a signal is sent to the Arduino to activate the actuator and flash the corresponding LED light. If the predicted class label is 'brown', the actuator is activated for 1.5 seconds, if it is 'green', the actuator is activated for 2.5 seconds, and if it is 'clear', no actuator is activated. The predicted class label and corresponding probability are also printed to the console.

```
        # Send a signal to the Arduino based on the prediction
        if predicted_class == 'brown':
            # Brown bottle detected, activate the actuator and flash the brown LED
            ser.write(b'1')
            time.sleep(1.5)
            ser.write(b'4') # stop the actuator
            print("Brown bottle detected with probability", prediction)
        elif predicted_class == 'green':
            # Green bottle detected, activate the actuator and flash the green LED
            ser.write(b'2')
            time.sleep(2.5)
            ser.write(b'4') # stop the actuator
            print("Green bottle detected with probability", prediction)
        else:
            # Clear bottle detected#
                ser.write(b'3')
                print("Clear bottle detected with probability", prediction)
```

Figure 20 – Model sending a signal down the communication line based of prediction.

5.9.2 Deployment on Arduino:

The next stage of the implementation is the Arduino code for servo motor which will act as our actuator and a corresponding light so that we can see clearly what the predictions are.

```
#include <Servo.h>
int clearLed = 10;
int brownLed = 12; // pin number of the brown LED
int greenLed = 11; // pin number of the green LED
int buttonPin = 2; // pin number of the button
Servo myservo;

void setup() {
  pinMode(brownLed, OUTPUT);
  pinMode(greenLed, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  myservo.attach(3);
  Serial.begin(9600);
}
```

Figure 21 – Setup servo motor and LEDs

The program begins by setting up the pins for the brown, green, and clear LEDs, as well as the button and servo motor.

```
int signal = Serial.read();
if (signal == '1') {
  // Brown bottle detected, flash the brown LED
  myservo.write(90);
  digitalWrite(brownLed, HIGH);
  delay(1000);
  digitalWrite(brownLed, LOW);
  myservo.write(0);
} else if (signal == '2') {
  // Green bottle detected, flash the green LED
  digitalWrite(greenLed, HIGH);
  myservo.write(135);
  delay(2000);
  myservo.write(0);
  digitalWrite(greenLed, LOW);
} else if(signal == '3'){
  // Clear bottle detected, activate the actuator
  myservo.write(45);
  digitalWrite(clearLed, HIGH);
  delay(1000);
  digitalWrite(clearLed, LOW);
  myservo.write(0);
}
```

Figure 22 – Main loop

It then enters the main loop, where it waits for signals from the serial channel. If a signal of '1' is received, it activates the servo motor to open the actuator, flashes the brown LED for one second, and then returns the servo motor to its original position. If a signal of '2' is received, it activates the servo motor to a certain degree, flashes the green LED for two seconds, and then returns the servo motor to its original position. Finally, if a signal of '3' is received, it activates the servo motor to another degree, flashes the clear LED for one second, and then returns the servo motor to its original position.

5.9.3 Results:

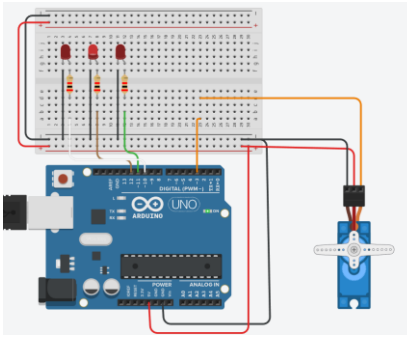


Figure 23 – Output if brown bottle detected

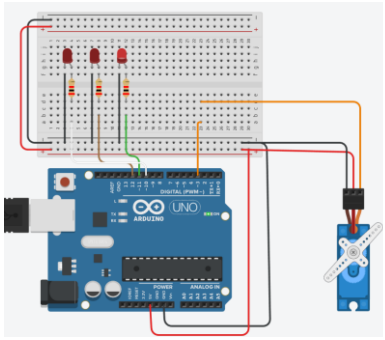


Figure 24 – Output if green bottle detected

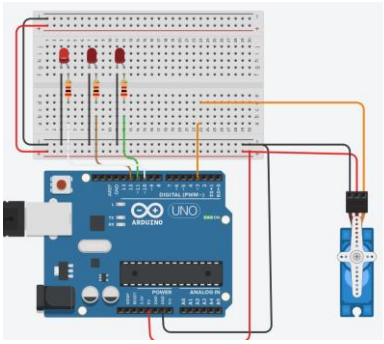


Figure 25 – Output if clear bottle detected

The figures above prove and the results from the evaluation prove that our system has been successfully implemented. The final accuracy of the system was found to be 93%.

6. Design:

6.1 Test Design:

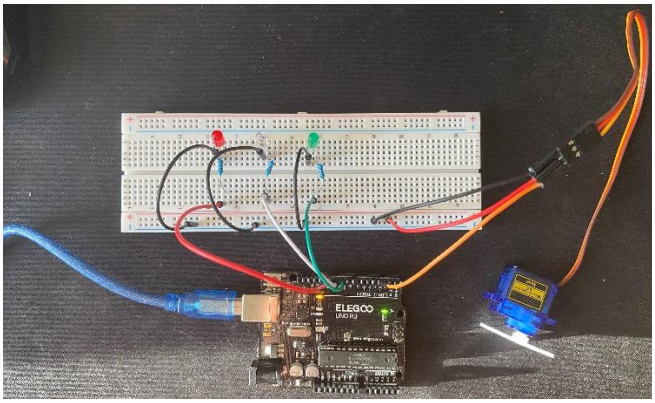


Figure 26 – Physical implementation of the test design

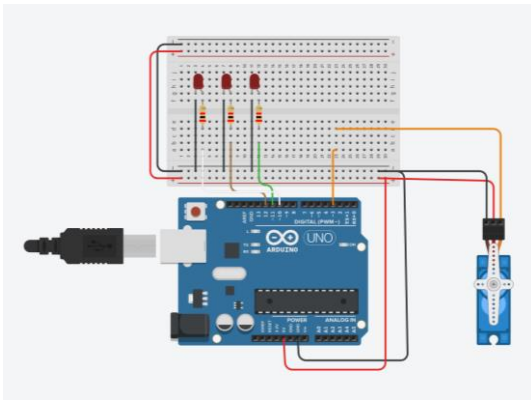


Figure 27 - Virtual implementation of the test design

The three LEDs, each with their own 2k resistor, allows us to visually confirm when each bottle is detected by the image recognition system, as each bottle type is associated with a different LED. The servo motor, which is used to activate the actuator to sort the bottles, is also an essential component of the layout. By integrating the servo motor with the system, we can effectively control the direction and extent of the movement required to sort the bottles based on the detected type.

6.2 Projected Design:

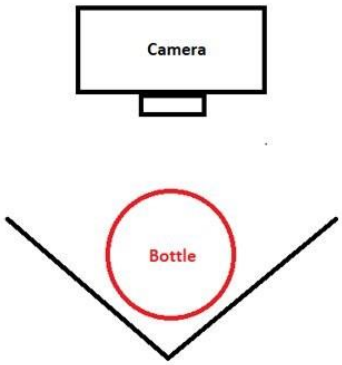


Figure 28

The system will have a small hole in the front of the unit much like a regular bottle bin. This will ensure the bottle goes in and is on its side so the camera can see as much as of the bottle as possible. This should improve the overall accuracy of the system as it has more surface area to determine what colour the bottle is. The glass bottle will drop down onto a conveyor that is in a V shape to ensure the bottle is in the correct position. Once the camera has gotten enough data from the bottle and the neural network has classified the colour of the glass bottle. The bottle will continue down the conveyor.

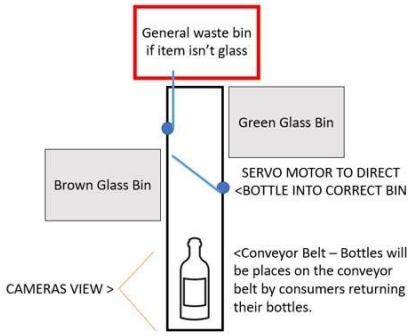


Figure 29

A servo that corresponds to the colour of the bottle will trigger an arm to extend and direct the bottle into the appropriate bin. If the system is very unsure of the colour of the bottle a general waste bin will be at the end of the conveyor to prevent contamination. It will also collect anything that happens to be on the conveyor that may not be glass.

7. Sustainability:

Sustainability is a guiding principle for this project, as the main objective is to implement an intelligent system that addresses one of the United Nations' 17 Sustainable Development Goals. In this context, sustainability refers to the ability to meet present needs without compromising the ability of future generations to meet their own needs.

This project contributes to a circular economy model, where waste is minimized, and resources are kept in use for as long as possible. The use of AI in waste management can lead to significant reductions

Figure 30 – circular economy [F6]

in greenhouse gas emissions, energy consumption, and the demand for raw materials, contributing to a more sustainable and resilient society [30].

The potential applications of AI in promoting sustainability are vast and diverse, extending beyond waste management. For instance, AI can help optimize energy consumption in buildings, improve the efficiency of renewable energy systems, and facilitate the design of sustainable cities. As AI technologies continue to evolve and improve, we expect to see further advancements in sustainability [30].

7.1 Greener Solution:

A way to make our solution greener is to consider using eco-friendly materials in the construction of the physical components of our system. For example, we could use biodegradable or recycled plastic for the casing of the machine or opt for natural materials like wood. Additionally, we could use renewable energy sources to power our system such as solar panels or a wind turbine.

Another way to make our project eco-friendlier is to consider the end-of-life of the components. When the project is no longer in use or needs to be replaced, we can ensure that the components are disposed of in a responsible manner. This can be done by recycling the materials or properly disposing of them through electronic waste programs.

Other ways we can prioritize sustainability and environmental friendliness in this project are:

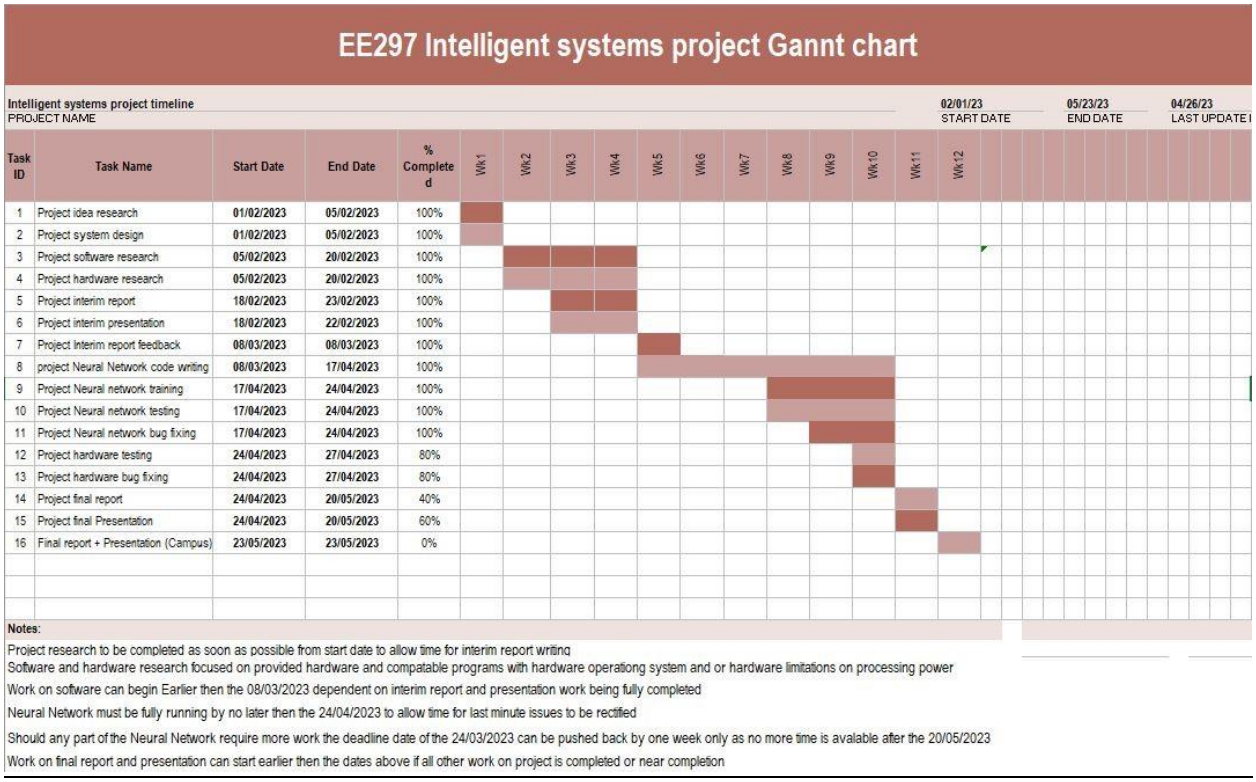
Energy Efficiency: We could design our AI systems and hardware components to be energy efficient. This could be achieved by optimizing the algorithms used and selecting energy-efficient hardware components.

Life Cycle Consideration: We should design our system to be durable and long-lasting. This reduces the need for frequent replacements, thereby reducing the manufacturing and waste disposal impacts.

This can significantly reduce the carbon footprint of the project. By incorporating these green strategies, we can ensure our project is not only technologically advanced but also environmentally responsible.

8. Work plan and Gannt chart

8.1 Gannt Chart



EE297 Intelligent systems project Gannt chart

Intelligent systems project timeline — PROJECT NAME — START DATE 02/01/23 — END DATE 05/23/23 — LAST UPDATE 04/26/23

Task ID	Task Name	Start Date	End Date	% Complete d	Wk 1	Wk 2	Wk 3	Wk 4	Wk 5	Wk 6	Wk 7	Wk 8	Wk 9	Wk 10	Wk 11	Wk 12
1	Project idea research	01/02/2023	05/02/2023	100%												
2	Project system design	01/02/2023	05/02/2023	100%												
3	Project software research	05/02/2023	20/02/2023	100%												
4	Project hardware research	05/02/2023	20/02/2023	100%												
5	Project interim report	18/02/2023	23/02/2023	100%												
6	Project interim presentation	18/02/2023	22/02/2023	100%												
7	Project Interim report feedback	08/03/2023	08/03/2023	100%												
8	project Neural Network code writing	08/03/2023	17/04/2023	100%												
9	Project Neural network training	17/04/2023	24/04/2023	100%												
10	Project Neural network testing	17/04/2023	24/04/2023	100%												
11	Project Neural network bug fixing	17/04/2023	24/04/2023	100%												
12	Project hardware testing	24/04/2023	27/04/2023	80%												
13	Project hardware bug fixing	24/04/2023	27/04/2023	80%												
14	Project final report	24/04/2023	20/05/2023	40%												
15	Project final Presentation	24/04/2023	20/05/2023	60%												
16	Final report + Presentation (Campus)	23/05/2023	23/05/2023	0%												

Notes:

Project research to be completed as soon as possible from start date to allow time for interim report writing
Software and hardware research focused on provided hardware and compatable programs with hardware operationg system and or hardware limitations on processing power
Work on software can begin Earlier then the 08/03/2023 dependent on interim report and presentation work being fully completed
Neural Network must be fully running by no later then the 24/04/2023 to allow time for last minute issues to be rectified
Should any part of the Neural Network require more work the deadline date of the 24/03/2023 can be pushed back by one week only as no more time is avalable after the 20/05/2023
Work on final report and presentation can start earlier then the dates above if all other work on project is completed or near completion

8.2 Project progress in relation to Gannt chart timeline

1. Research into a project idea was completed within week one. This research into a design for the system was also completed it included researching the feasibility of the idea, availability of components to the group, technical specifications of the overall system and difficulty of building the system.
2. From week two until week 4 the group researched the several types of neural network structures. The group picked a Convolutional network as the neural network within week three.
3. During week three the group began drafting the interim report as well as creating the accompanying presentation. By the end of week three the interim report was completed and during the beginning of week four it was proofread alongside the PowerPoint.
4. During week four the group presented the PowerPoint to Dr Nepomuceno and the other groups, initial feedback was received on the presentation.
5. Feedback was received on the interim report during week 5, the feedback was reviewed as a group during the Wednesday lab session. Work on writing of the code for the neural network began in week 5. The group ran into difficulties immediately with installing the required software onto the supplied raspberry pi. These difficulties continued until week 7. it was decided after the Thursday lab session to switch from using the raspberry pi to a laptop for ease of use with programs that cannot run on the raspberry pi.
6. During week 8 the group solved the issues with the software and programs not working correctly. During this time, the group also began to write the code for the neural network using Anaconda prompt alongside Jupyter lab. This allowed the group to get the project timeline back on track.
7. Due to the issues in weeks 5 and 6 the group decided that an extra week was needed to have a neural network completed, trained and tested. It was decided then that the group would work fully on the neural network and to not begin any testing or bug fixing until the network

was fully built. This caused issues later as many bugs were present in the code as well as in the dataset. These ranged from broken images to errors within the code.

8. By the end of week 10, a large amount of the bugs causing the issues that occurred while training the neural network were resolved and the neural network was successfully trained and tested to an accuracy of approximately 90 to 95% for any unknown image of a glass bottle presented to the system.
9. During the final week of lectures a quick practice of the final presentation was performed during the Wednesday morning lab session, the group was able to present a draft PowerPoint to Dr Nepomuceno, who later provided feedback allowing the group to fix any issues and errors in the report. This also allowed the group to see where there was a lack of clarity or information on the slides and what needed to be more focused on. The group then returned to work on the final draft of the presentation.
10. During the weeks 10 through to 12 the group was in constant communication to ensure there was no overlap of work on one section of the presentation or report, this was done to ensure all parts received attention and to help reduce the workload on any one member of the group. The group believed that by dividing the work up it would provide a better and more accurate report on the work performed throughout the year.

8.3 Group work plan

PROJECT NAME	EE297 INTELLIGENT SYSTEMS PROJECT			Group	C
PROJECT DELIVERABLE	Glass bottle sorting system				
SCOPE STATEMENT	Develop a system for sorting glass that covers multiple UN sustainable development goals				
START DATE	01/02/2023	END DATE	23/05/2023	OVERALL PROGRESS	85% as of 26/04/2023

Task No.	TASKS TO COMPLETE	ASSIGNED TO	START DATE	END DATE	DURATION in days	STATUS

1	**Project idea research** Researching the UN sustainable development goals and finding goals that the group can implement a solution to, as well as research into	Everyone	01/02/2023	05/02/2023	4	complete
2	**Project system design** Design a system that will cover the requirements of the project and use easily available components. Researching what ways, the glass can be manipulated, moved and processed	Everyone	01/02/2023	05/02/2023	4	complete
3	**Project software research** Researching what software is available for use in building a neural network. Finding solutions to the hardware/software problems found while researching.	Eoin Keatley Donal McCann, Stephen Oresanya	05/02/2023	20/02/2023	15	complete
4	**Project hardware research** Researching how to integrate hardware with the software of the neural network to provide a mechanical or visual output for the system. Finding components needed for the system and researching the best implementation of the components	Nathan Ewnetu, Stephen Oresanya	05/02/2023	20/02/2023	15	complete
5	**Project interim report** Writing the interim report and ensuring all required information is available in the report. Proofreading the report to ensure correct spellings, information and formatting	Everyone	20/02/2023	23/02/2023	4	complete
6	**Project interim presentation** Creating the interim presentation, formatting the slides in the correct order, checking information in slides, proofreading slides and practicing presentation.	Everyone	18/03/2023	22/02/2023	4	Complete
7	**Project interim feedback** The reading of the feedback on both presentation and report. Understanding errors and ensuring to correct errors in later versions of final report and presentation	Everyone	08/03/2023	08/03/2023	1	Complete
8	**Project neural network code writing** Writing neural network code as well as finding an efficient solution, if possible, to the code problems. Writing, commenting and debugging code required for neural network.	(Originally Eoin but errors to with raspberry pi) Nathan Ewnetu Donal McCann (for future works)	08/03/2023	24/04/2023 (Overrun due to issues with software)	47	Complete (1 week over date)
9	**Project neural network training** Creating a dataset for the neural network to read from and supervising the software to ensure no issues occur during the training of the system.	Nathan Ewnetu Donal McCann	17/04/2023	24/04/2023	7	Complete
10	**Project neural network testing** Testing the network with unseen test images, images source rom internet as well as images taken of glass bottles by group members	Everyone (All members provided test samples and tested the network)	17/04/2023	24/04/2023	7	Complete

11	**Project neural network bug fixing** Finding any issues with the neural network and its code, comparing results to ensure no errors. If errors detected fixing them and ensuring neural network works as intended	Nathan Ewnetu, Stephen Oresaya	17/04/2023	24/04/2023	7	Complete
12	**Project hardware testing** Testing the hardware after connection to software of system. Ensuring servos and any other sensors function correctly	Nathan Ewnetu, Eoin Keatley	24/04/2023	27/04/2023	4	80% complete
13	**Project hardware bug fixing** Testing the hardware to ensure the system's sensors and actuators respond correctly to the input provided. Fixing issues that appear during testing	Nathan Ewnetu, Donal McCann, Stephen Oresaya	24/04/2023	27/04/2023	4	80% complete
14	**Project final report** Writing of the final report for submission, formatting the text images and graphs, adding titles and subtitles. Proof reading of reports contents, spelling fixing and grammar checks. Checking no issues reoccur from interim report, following the mistakes and fixing them from interim feedback.	Everyone	24/04/2023	20/05/2023	26	40% complete
15	**Project final presentation** Writing of the final presentation, filling in relevant information, formatting slides into a readable format. Practicing the presentation. Ensuring feedback from interim presentation and practice final presentation is implemented.	Everyone	24/04/2023	20/05/2023	26	60% complete
16	**Final report + presentation(campus)** Presenting the final presentation for grading as well as answering questions on presentation and report for interview.	Everyone	23/05/2023	23/05/2023		

9. Future work

Future work that the group can carry out to improve the system that the group designed is as follows.

- Implement an extra layer of software to compare and check if the object detected is a glass product or if it is another type of waste.
- Increase the accuracy further by rewriting the neural networks layers and structure to increase the accuracy from training and decrease the loss from the training of the neural network.
- Update the dataset with more images of glass bottles and add in a general waste dataset for use with the extra layer of software for detecting if the object is glass or general waste.

The group has worked on a solution to ensure that the glass bins do not become mixed with other types of rubbish that may end up entering the system due to user error or intentionally placed into the system. For this solution the group built a separate neural network like the currently operational system. This system used a dataset containing a large variety of images ranging from cardboard objects to metals, plastics and other objects. The Group hoped to have this system integrated into the

final systems architecture, however due to time and complexity constraints the group only achieved a separate instance for testing of this system. Through testing of the system, the following results were achieved for the loss and accuracy after ten Epochs.

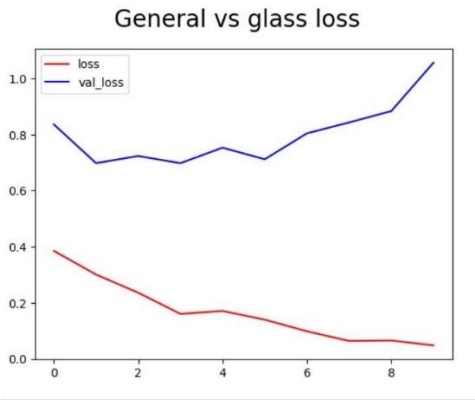


Figure 31 – Loss

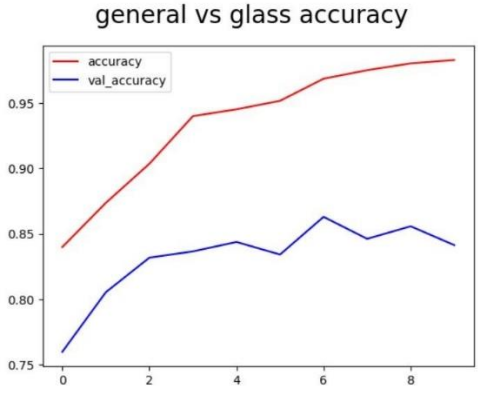


Figure 32 - Accuracy

As seen above the accuracy of this network is high as well as having a minimal loss value for the data. This helps to give a proof of concept for implementing this in the future as an upgrade for the system. Through this possible future upgrade to the system the group believes that the issues presented from user error or intentional misuse can be avoided as this upgrade would reduce the chance of objects entering the incorrect bins and contaminating the glass in those bins.

The group can also in the future, optimise the neural network further by adding in extra hidden layers to the system to help with improving the overall accuracy, speed and efficiency of the code used so far. The group believes that through testing and researching further a higher accuracy can be achieved, the group also believes that with a higher accuracy system the overall power consumption and processing time can also be reduced Furter. The aim of reducing the power consumption and processing time will help with implementing a green solution that is both environmentally friendly to manufacture, as well as run as low power consumption will help achieve the goal above.

Through updating and expanding the dataset the group believes that it can allow for a higher accuracy system. Given enough runtime and data to process the group believes that a high accuracy system can be achieved. Using a larger dataset, the system would be able to classify a larger variety of objects and a larger amount of camera angles in relation to the object. The group believes that by increasing the dataset, the issues presented from improperly inserted objects or damaged objects can be reduced to as low a chance as possible.

10. References:

[1] Pulidindi, K. and Prakash, A. (2021) *Recycled glass market outlook and statistics 2021-2027*, *Global Market Insights Inc.* Available at: https://www.gminsights.com/industry-analysis/recycled-glass-market#:~:text=Recycled%20Glass%20Market%20size%20was,on%20the%20furnace%20during%20manufacturing

[2] Habits of Waste. (2021) *Platic Bottle Stats*, Available at:https://habitsofwaste.org/call-to-action/plasticbottles/#:~:text=Plastic%20Bottle%20Facts&text=That's%2040%20billion%20per%20month,plastic%20is%20wasted%20each%20year

[3] United Nations. (2015). Transforming our world: the 2030 Agenda for Sustainable Development. Retrieved from https://www.un.org/sustainabledevelopment/development-agenda/

[4] United Nations. (n.d.). Goal 12: Responsible Consumption and Production. Retrieved from https://www.un.org/sustainabledevelopment/sustainable-consumption-production/

[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, May 2015

[6] J. Schmidhuber, "Deep learning in neural networks: An overview," Neural Networks, vol. 61, pp. 85-117, May 2015

[7] J. Deng et al., "Imagenet: A large-scale hierarchical image database," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 248-255, June 2009

[8] D. Kleyko, E. Osipov, "Towards Cognitive Neural Networks", 2015 IEEE International Conference on Data Science and Data Intensive Systems, pp. 69-76, 2015.

[9] A. Zell, "Simulation Neuronaler Netze", Oldenbourg Wissenschaftsverlag, 1994

[10] W. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," Bulletin of Mathematical Biophysics, vol. 5, no. 4, pp. 115-133, Dec. 1943

[11] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," Psychological Review, vol. 65, no. 6, pp. 386-408, Nov. 1958

[12] D. Rumelhart, G. Hinton, and R. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, no. 6088, pp. 533-536, Oct. 1986

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, Nov. 1997

[15] G. K. Schindler and M. K. Schindler, "Artificial intelligence: a framework for the future," Business Horizons, vol. 62, no. 1, pp. 1-11, 2019.

[16] M. R. Aziz et al., "Material Recovery Facility: An overview of the technology and the sustainability aspect," in 2019 International Conference on Robotics, Automation and Sciences (ICORAS), 2019, pp. 1-6.

[17] C. F. Scheinberg and L. Jeong, "Waste and recycling innovation using artificial intelligence," in § 2019 International Solid Waste Association World Congress, 2019, pp. 1-8.

[18] J. L. Welter, "Data analytics in the waste-to-energy industry," Waste Management, vol. 76, pp. 347-356, 2018.

[19] M. B. Delipinar et al., "Robotic Waste Collection System," Procedia Computer Science, vol. 169, pp. 109-116, 2020.

[20] M. W. Ahmed et al., "Development of a Robotic Waste Collection System for Smart Cities," in 2019 International Conference on Robotics, Automation and Sciences (ICORAS), 2019, pp. 1-6.

[21] A. Banerjee and R. K. Pandey, "Artificial Intelligence in Waste Management: A Review," International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 9, no. 4S, pp. 390-394, 2020.

[22] Andersen, P.H. and Åberg, S. (2021). Testing the waters: Translating MNE technology in a base-of-the-pyramid context. *Journal of Cleaner Production*, 281, p.125195.

[23] Leong, W.-H., Teh, S.-Y., Hossain, M.M., Nadarajaw, T., Zabidi-Hussin, Z., Chin, S.-Y., Lai, K.-S. and Lim, S.-H.E. (2020). Application, monitoring and adverse effects in pesticide use: The importance of

reinforcement of Good Agricultural Practices (GAPs). *Journal of Environmental Management*, 260, p.109987.

[24] Wang, S., Wang, J., Zhao, S. and Yang, S. (2019). Information publicity and resident's waste separation behaviour: An empirical study based on the norm activation model. *Waste Management*, [online] 87, pp.33–42.

[25] Li, H., Li, Y., Luo, Q., Li, B., & Li, L. (2018). Glass recognition using deep learning. Journal of Physics: Conference Series, 1032(1), 012047

[26] Mandal, P., Gogoi, M., & Chowdhury, A. (2019). Glass classification using deep learning. Journal of Electronic Imaging, 28(2), 023006.

[27] Cheng, L., Xie, J., Tang, Y., Wang, L., & He, H. (2021). Classification of colored glass based on fluorescence spectroscopy and deep learning. Journal of Hazardous Materials, 401, 123414.

[28] J. L. Welter, "Data analytics in the waste-to-energy industry," Waste Management, vol. 76, pp. 347-356, 2018.

[29] M. B. Delipinar et al., "Robotic Waste Collection System," Procedia Computer Science, vol. 169, pp. 109-116, 2020.

[30] V. Dignum, "Responsible Artificial Intelligence: How to Develop and Use AI in a Responsible Way," Springer International Publishing, 2019.

10.1 Figure References:

[F1] Figure 1 - M. Kalirane, "Gradient descent vs. backpropagation: What's the difference?," Analytics Vidhya, https://www.analyticsvidhya.com/blog/2023/01/gradient-descent-vs-backpropagation-whats-the-difference/

[F2] Figure 2 - Lee, C. & Benjamin, Ng & Pokharel, Shaligram. (2012). Managing Uncertain Inventory in Supply Chain with Neural Network and Radio Frequency Identification (RFID)

[F3] Figure 3 - Modulos AI. (2021, February 18). Artificial neural networks deep dive. Retrieved from https://www.modulos.ai/blog/artificial-neural-networks-deep-dive/

[F4] Figure 4 - G. Saporito, "What is a Perceptron?," Medium, https://towardsdatascience.com/what-is-a-perceptron-210a50190c3b

[F5] Figure 5 - J. 25, “Ai-powered robotic waste sorting system installed at Zanker Recycling Facility,” Recycling Product News, https://www.recyclingproductnews.com/article/27382/ai-powered-robotic-waste-sorting-system-installed-at-zanker-recycling-facility

[F6] Figure 27 - “Circular economy: Definition, importance and benefits: News: European parliament,” Circular economy: definition, importance and benefits | News | European Parliament, https://www.europarl.europa.eu/news/en/headlines/economy/20151201STO05603/circular-economy-definition-importance-and-benefits

11. Code:

```
import tensorflow as tf

import os

# Avoid OOM errors by setting GPU Memory Consumption Growth

gpus = tf.config.experimental.list_physical_devices('GPU')

for gpu in gpus:

 tf.config.experimental.set_memory_growth(gpu, True)


import cv2

Import imghdr
```

```
data_dir = r'C:\Nathans Projects\EE297_Final\data'
image_exts = ['jpeg','jpg', 'bmp', 'png']

# removes dodgy images from the data directory using
for image_class in os.listdir(data_dir):
    for image in os.listdir(os.path.join(data_dir, image_class)):
        image_path = os.path.join(data_dir, image_class, image)
        try:
            img = cv2.imread(image_path)
            tip = imghdr.what(image_path)
            if tip not in image_exts:
                print('Image not in ext list {}'.format(image_path))
                os.remove(image_path)
        except Exception as e:
            print('Issue with image {}'.format(image_path))
            #os.remove(image_path)
```

```
import numpy as np
from matplotlib import pyplot as plt
data = tf.keras.utils.image_dataset_from_directory(data_dir)
data_iterator = data.as_numpy_iterator()
batch = data_iterator.next()

#Code to show our images and there assigned number labels:
fig, ax = plt.subplots(ncols=4, figsize=(20,20))
for idx, img in enumerate(batch[0][:4]):
    ax[idx].imshow(img.astype(int))
```

```
    ax[idx].title.set_text(batch[1][idx])

data = data.map(lambda x,y: (x/255, y)) #speeds up the process to access our data from our disk

import os
import shutil
import random
# Set the path to your data directory
data_dir = r'C:\Nathans Projects\EE297_Final\data'
# Set the percentages for the train/val split
train_pct = 0.8
val_pct = 0.2
# Create directories for train and validation data
train_dir = os.path.join(data_dir, 'train')
val_dir = os.path.join(data_dir, 'val')
os.makedirs(train_dir, exist_ok=True)
os.makedirs(val_dir, exist_ok=True)



# Loop over each class directory and split the images into train and val sets
for class_name in ['brown_glass', 'green_glass', 'clear_glass']:
    # Set the path to the class directory
    class_dir = os.path.join(data_dir, class_name)

    # Get a list of all the image filenames in the class directory
    image_filenames = os.listdir(class_dir)

    # Shuffle the filenames to ensure a random split
    random.shuffle(image_filenames)
```

```
    # Determine the split index for train/val
    split_idx = int(len(image_filenames) * train_pct)

    # Split the image filenames into train and validation sets
    train_filenames = image_filenames[:split_idx]
    val_filenames = image_filenames[split_idx:]

    # Create subdirectories for each class in the train and val directories
    train_class_dir = os.path.join(train_dir, class_name)
    val_class_dir = os.path.join(val_dir, class_name)
    os.makedirs(train_class_dir, exist_ok=True)
    os.makedirs(val_class_dir, exist_ok=True)

    # Copy the train images to the train subdirectory for this class
    for filename in train_filenames:
        src_path = os.path.join(class_dir, filename)
        dst_path = os.path.join(train_class_dir, filename)
        shutil.copy(src_path, dst_path)


    # Copy the validation images to the validation subdirectory for this class
    for filename in val_filenames:
        src_path = os.path.join(class_dir, filename)
        dst_path = os.path.join(val_class_dir, filename)
        shutil.copy(src_path, dst_path)


from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout
from tensorflow.keras.losses import CategoricalCrossentropy
```

```
from tensorflow.keras.optimizers import Adam

from tensorflow.keras.preprocessing.image import ImageDataGenerator


#Define the paths to our directories

train_dir = r'C:\Nathans Projects\EE297_Final\data\train'

val_dir = r'C:\Nathans Projects\EE297_Final\data\val'


train_datagen =
ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)


val_datagen = ImageDataGenerator(rescale=1./255)


train_generator=train_datagen.flow_from_directory(train_dir,target_size=(256,
256),batch_size=32,class_mode='categorical')


val_generator=val_datagen.flow_from_directory(val_dir,target_size=(256,
256),batch_size=32,class_mode='categorical')




model = Sequential()

model.add(Conv2D(16, (3,3), 1, activation='relu', input_shape=(256,256,3)))

model.add(MaxPooling2D())

model.add(Conv2D(32, (3,3), 1, activation='relu'))

model.add(MaxPooling2D())

model.add(Conv2D(64, (3,3), 1, activation='relu'))

model.add(MaxPooling2D())

model.add(Flatten())

model.add(Dense(256, activation='relu'))

model.add(Dense(3, activation='softmax'))
```

```
model.compile(optimizer=Adam(), loss=CategoricalCrossentropy(), metrics=['accuracy'])

logdir='logs'

tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=logdir)

hist=model.fit(train_generator,epochs=20,validation_data=val_generator,
callbacks=[tensorboard_callback])

fig = plt.figure()

plt.plot(hist.history['loss'], color='teal', label='loss')

plt.plot(hist.history['val_loss'], color='orange', label='val_loss')

fig.suptitle('Loss', fontsize=20)

plt.legend(loc="upper left")

plt.show()


fig = plt.figure()

plt.plot(hist.history['accuracy'], color='teal', label='accuracy')

plt.plot(hist.history['val_accuracy'], color='orange', label='val_accuracy')

fig.suptitle('Accuracy', fontsize=20)

plt.legend(loc="upper left")

plt.show()

print(train_generator.class_indices)

print(val_generator.class_indices)


from tensorflow.keras.models import load_model

model.save(os.path.join(r'C:\Nathans Projects\EE297_Final','brown_clear_green_glass.h5'))


import cv2

import time

import serial

from PIL import Image

import numpy as np

from keras.models import load_model

# Connect to the Arduino Uno
```

```
ser = serial.Serial('COM5', 9600)
# Load the model
model_path = r'C:\Nathans Projects\EE297_Final\brown_clear_green_glass.h5'
model = load_model(model_path)
# Define the labels for each class
labels = ['brown', 'green', 'clear']
while True:
  # if ser.read()==b'p':

      # Capture an image from the webcam
      cap = cv2.VideoCapture(0)
      ret, frame = cap.read()
      cap.release()
      # Resize the image
      img = Image.fromarray(frame)
      img = img.resize((256, 256))
      # Convert the image to a numpy array
      x = np.array(img)
      # Expand the dimensions of the array to match the model's input shape
      x = np.expand_dims(x, axis=0)
      # Make the prediction
      prediction = model.predict(x)
      # Get the predicted class label
      predicted_class = labels[np.argmax(prediction)]
      # Send a signal to the Arduino based on the prediction
      if predicted_class == 'brown':
         # Brown bottle detected, activate the actuator and flash the brown LED
         ser.write(b'1')
         time.sleep(1.5)
         ser.write(b'4') # stop the actuator
         print("Brown bottle detected with probability", prediction)
```

```
        elif predicted_class == 'green':
            # Green bottle detected, activate the actuator and flash the green LED
            ser.write(b'2')
            time.sleep(2.5)
            ser.write(b'4') # stop the actuator
            print("Green bottle detected with probability", prediction)
        else:
            # Clear bottle detected#
                ser.write(b'3')
                print("Clear bottle detected with probability", prediction)
```

Arduino

```
#include <Servo.h>
int clearLed = 10;
int brownLed = 12; // pin number of the brown LED
int greenLed = 11; // pin number of the green LED
// pin number of the button
Servo myservo;

void setup() {
  pinMode(brownLed, OUTPUT);
  pinMode(greenLed, OUTPUT);
  myservo.attach(3);
  Serial.begin(9600);
}

void loop() {
    // Get the signal from the computer and activate the actuator
    int signal = Serial.read();
    if (signal == '1') {
      // Brown bottle detected, flash the brown LED
```

```
    myservo.write(180);
    digitalWrite(brownLed, HIGH);
    delay(1000);
    digitalWrite(brownLed, LOW);
    myservo.write(0);
  } else if (signal == '2') {
    // Green bottle detected, flash the green LED
    digitalWrite(greenLed, HIGH);
    myservo.write(135);
    delay(2000);
    myservo.write(0);
    digitalWrite(greenLed, LOW);
  } else if(signal == '3'){
    // Clear bottle detected, activate the actuator
    myservo.write(45);
    digitalWrite(clearLed, HIGH);
    delay(1000);
    digitalWrite(clearLed, LOW);
    myservo.write(0);
  }

}
```