Automated Waste Segregation Using IoT and Machine Learning

Anshul Anilkumar Mundakatil
Engineering Department
San Jose State University
San Jose, United States Of America
anshulanilkumar.mundakatil@sjsu.edu

Akshil Anilkumar Mundakatil
Engineering Department
San Jose State University
San Jose, United States Of America
akshilanilkumar.mundakatil@sjsu.edu

***Abstract*—Improper waste management has become a significant global challenge, contributing to environmental degradation, resource wastage, and health hazards. The traditional methods of segregating waste often rely on manual labor, which is time-consuming, error-prone, and inefficient in handling large volumes of waste. To address this pressing issue, this research focuses on designing and developing an IoT-based automated waste segregation system capable of distinguishing between recyclable and non-recyclable materials. The system leverages advanced machine learning techniques and object detection algorithms to identify waste categories in real time and segregates them into appropriate bins. Using a camera for image input, a trained neural network for classification, and an actuator mechanism for sorting, the proposed solution minimizes human intervention and ensures efficient waste management. Preliminary evaluations indicate that the system achieves high accuracy in classifying common recyclables such as plastics, metals, and paper, while effectively identifying non-recyclables. By enhancing recycling rates and reducing human involvement in waste handling, this work aims to contribute to a cleaner environment and support sustainable development initiatives.**

***Index Terms*—IoT, Machine Learning**

I. Introduction

Waste segregation is a critical component of sustainable waste management. Improper disposal of waste not only harms the environment but also impedes recycling efforts, leading to increased landfill use and resource depletion. Recycling, as a core strategy, relies heavily on the accurate separation of materials, which is often labor-intensive and prone to error when performed manually.

The integration of IoT technologies with machine learning offers a promising approach to automate this process. IoT devices enable real-time monitoring and control, while machine learning models can classify waste items with high precision. The convergence of these technologies allows for the development of an efficient waste segregation system that minimizes human intervention, reduces sorting errors, and enhances recycling rates.

This paper proposes an IoT-based waste segregation device capable of distinguishing recyclable items such as plastics, metals, and paper from non-recyclables. The system employs a camera for real-time image capture, a convolutional neural network for material classification, and an actuator-driven sorting mechanism to direct waste into appropriate bins. By automating the segregation process, the device aims to reduce operational costs in recycling plants and encourage broader adoption of sustainable practices.

II. System Overview

This section provides a detailed description of the hardware and software components employed in the automated waste segregation system.

A. Hardware Components

1) Base Shield V2 : Base Shield V2 is an expansion board designed for Arduino-based microcontrollers, providing compatibility with various Grove modules and sensors. It simplifies prototyping by offering a modular and plug-and-play interface, reducing the need for complex wiring. The shield supports analog, digital, and I2C connections, making it ideal for integrating multiple hardware components like sensors and actuators.

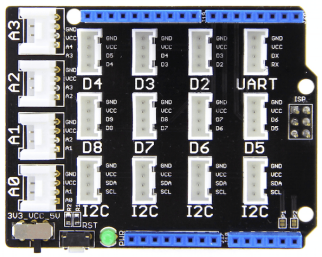


Fig. 1. Base Shield

2) Gear Stepper Motor with Driver : The gear stepper motor is a precise, high-torque motor designed for controlled movement applications. Integrated with a driver circuit facilitates smooth and accurate rotation, even under load. The motor operates in discrete steps, enabling precise angular movements. This characteristic is beneficial for creating a tiltable base mechanism, as it can reliably adjust to predefined angles. Adding a gear assembly enhances torque, making it capable of handling heavier loads without compromising accuracy.

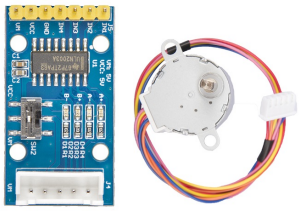


Fig. 2. Gear Stepper Motor with Driver

3) Intel Edison for Arduino : The Intel Edison is a compact development platform featuring robust computing capabilities and wireless connectivity. Designed to support the Arduino ecosystem, it integrates seamlessly with Arduino-compatible shields like the Base Shield V2. Powered by an Intel Atom processor, it offers computational efficiency, making it suitable for IoT and machine learning tasks. Its compatibility with multiple programming environments and libraries makes it an adaptable choice for complex IoT solutions.

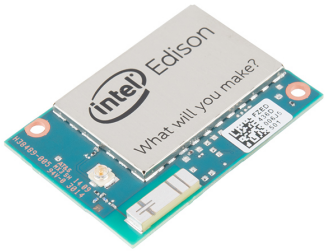


Fig. 3. Intel Edison

4) Grove - LCD RGB Backlight : The LCD Backlight is a simple display module used to show messages and provide visual feedback. It has a screen to display text and a backlight that can change colors. In this project, it is used to show whether an item is ”Recyclable” or ”Non-Recyclable” based on the classification. The backlight ensures the message is easy to see, making it a helpful addition to the waste segregation system.

Fig. 4. LCD RGB Backlight

B. Software Components

1) Google Teachable Machine: Google Teachable Machine is a user-friendly platform that enables the creation and deployment of machine learning models. The platform is designed to work with input data from cameras, microphones, or other sensors. For this project, the image recognition module is utilized to classify waste into recyclable and non-recyclable categories. The Teachable Machine uses a pre-trained MobileNet model, a lightweight convolutional neural network designed for mobile and edge devices.

2) MobileNet: MobileNet is a family of efficient convolutional neural networks optimized for mobile and embedded vision applications. It uses depthwise separable convolutions, which break down the process into two steps: first, filtering each channel of the image separately, and then combining the results. This design reduces the computation needed while still maintaining accuracy, making it ideal for real-time image classification on devices with limited resources.

3) TensorFlow Lite: TensorFlow Lite is a lightweight framework designed to deploy machine learning models on edge devices with limited resources. It optimizes models trained in TensorFlow by reducing their size and computation requirements through techniques such as quantization and pruning. TensorFlow Lite enables low-latency inference directly on devices, allowing for real-time predictions without requiring a constant connection to the cloud, making it well-suited for IoT applications like waste segregation. [3]

III. System Architecture

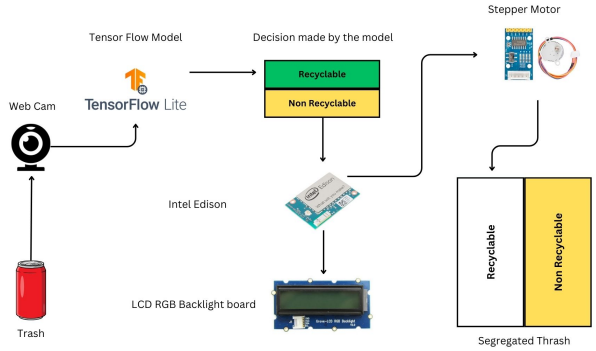


Fig. 5. System Architecture

As shown in Figure 5. the system architecture starts with a webcam which captures images of waste items, which are processed by a TensorFlow Lite model running on a separate computing device. The model analyzes the image and determines whether the waste is recyclable or non-recyclable. This decision is sent to an Intel Edison microcontroller, which controls a stepper motor to direct the waste into the appropriate bin. An LCD RGB backlight board displays the classification result in real-time for user feedback.

IV. Implementation

1) Apparatus setup The first step in implementing the waste segregation system involves setting up the apparatus. This includes creating a setup where a trained MobileNet model classifies items into recyclable and non-recyclable categories. Objects used for training include: Can, paper, aluminum foil, plastic bottle, plastic cup, cardboard, Straw, battery, sauce packet, leaves, mask, Styrofoam, plastic bag/package bags, spoon/fork.

The apparatus as discussed earlier comprises a camera for image capture, Intel Edison for processing, a stepper motor for bin movement, and an LCD for displaying results.

2) Model Training and Deployment The classification model was trained using Google Teachable Machine. The objects were classified into two categories: recyclable and non-recyclable. Real-time data collection was done using a webcam, capturing varied inputs for each category directly through the platform. The trained model was exported as a TensorFlow Lite file for deployment.

The model is loaded in the Python environment using the TensorFlow Lite interpreter. The Python code then orchestrates the entire system, capturing frames from the webcam, preprocessing the data, running inference using the TensorFlow Lite model, and controlling the motor. The main.py file executes in a continuous loop, processing frames in real-time and ensuring the sorting process is carried out automatically. We set the following commands for output handling, clockwise command for recyclable items and anticlockwise command for non-recyclable items.

3) Command Transmission and System Feedback The model's classification output was transmitted as signals to the Intel Edison chip, which controlled the stepper motor responsible for waste segregation. Based on the output, commands such as "clockwise" or "anticlockwise" were sent to rotate the lid of the bin, ensuring precise deposition into the appropriate compartment.

Simultaneously, an LCD backlight provided real-time feedback by displaying the classification result ("Recyclable" or "Non-Recyclable"), enhancing user understanding and transparency in system operation.

4) Process Overview The system operates as follows: a waste item is placed in front of the camera, where an image is captured and processed through the pre-trained machine learning model. The model classifies the item as either "Recyclable" or "Non-Recyclable," [Fig 6.] and the classification output is transmitted to the Intel Edison chip. Based on the output, the Edison chip [Fig 7.] sends commands to the stepper motor [Fig 8.] to rotate the lid in the appropriate direction, ensuring correct waste segregation. Simultaneously, an LCD backlight displays the classification result for user confirmation [Fig 9.]. This integrated process combines machine learning, hardware control, and user feedback to achieve efficient and automated waste management.

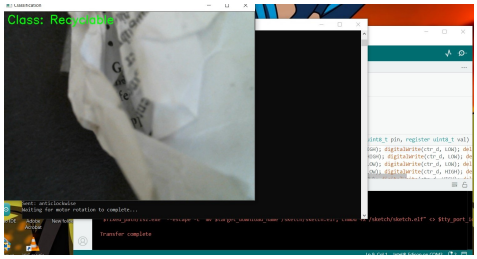


Fig. 6. Model classification result - Recyclable

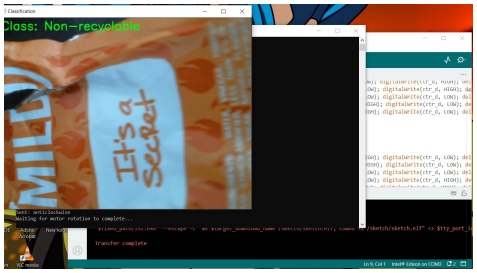


Fig. 7. Model classification result - Non Recyclable

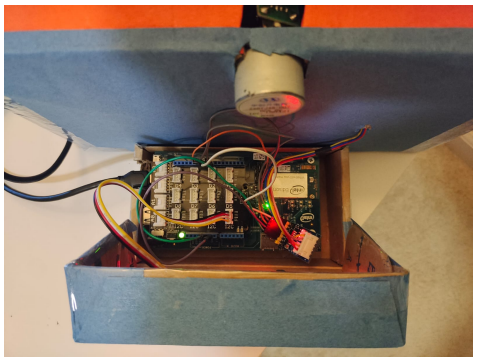


Fig. 8. Intel Edison Chip

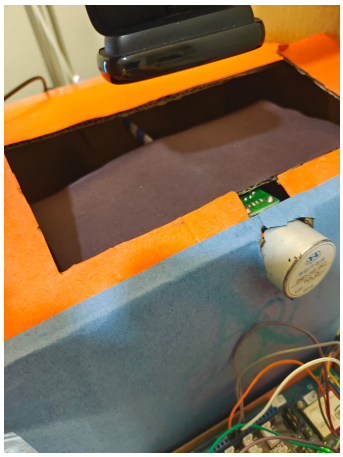


Fig. 9. Stepper Motor

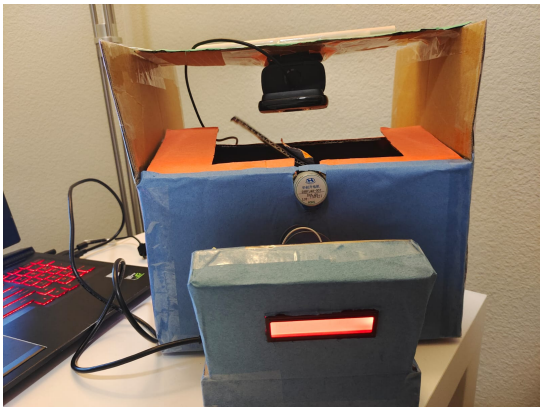


Fig. 10. Apparatus sorting a sauce packet as non-recyclable.

V. Conclusion and Future Work

The developed waste segregation system successfully integrates machine learning and hardware components to automate the classification of recyclable and non-recyclable materials. Utilizing a pre-trained model from Google Teachable Machine, Intel Edison for command processing, and a stepper motor for bin segregation, the system demonstrates efficient and accurate waste management. The inclusion of an LCD display further enhances user interaction by providing clear feedback on classification results.

Future iterations could expand the system's capabilities by incorporating multi-class classification to identify specific material types, enabling finer waste sorting. The integration of IoT devices, such as weight sensors, temperature monitors, and moisture detectors, could provide additional parameters to refine classification accuracy and enable context-aware waste processing. Expanding the dataset with diverse real-world samples would enhance the model's robustness and adaptability. Furthermore, IoT connectivity could support remote monitoring, real-time analytics, and scalable waste management across various environments.

References

[1] Agarwal C, Yewale B, Jagadish C. Automatic waste segregation and management. International Journal of Engineering Research and Technology (IJERT). 2020 Jun;9.

[2] Saha HN, Auddy S, Pal S, Kumar S, Pandey S, Singh R, Singh AK, Banerjee S, Ghosh D, Saha S. Waste management using internet of things (iot). In2017 8th annual industrial automation and electromechanical engineering conference (IEMECON) 2017 Aug 16 (pp. 359-363). IEEE.

[3] Carney M, Webster B, Alvarado I, Phillips K, Howell N, Griffith J, Jongejan J, Pitaru A, Chen A. Teachable machine: Approachable Web-based tool for exploring machine learning classification. InExtended abstracts of the 2020 CHI conference on human factors in computing systems 2020 Apr 25 (pp. 1-8).

[4] A. Gopi, J. A. Jacob, R. M. Puthumana, R. A. K, K. S and B. Manohar, "IoT based smart waste management system," 2021 8th International Conference on Smart Computing and Communications (ICSCC), Kochi, Kerala, India, 2021, pp. 298-302, doi: 10.1109/ICSCC51209.2021.9528293. keywords: Cloud computing;Waste materials;Costs;Microcontrollers;Urban areas;Acoustics;Sensor systems;IoT;Smart Waste Management;Blynk App,

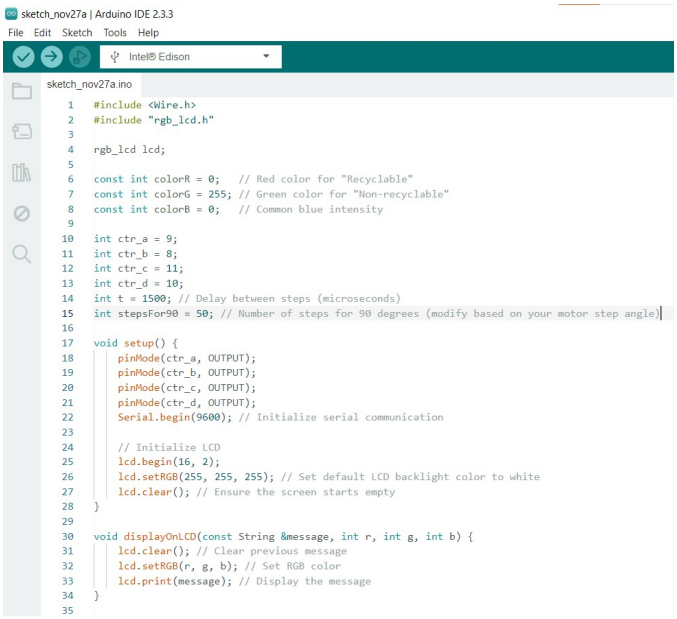
VI. Appendix

Python Code

```python
import serial
import time
import numpy as np
import tensorflow as tf
import cv2  # OpenCV for webcam and display

# Initialize serial communication
ser = serial.Serial('COM3', 9600)  # Update 'COM3' to your Arduino's port
time.sleep(2)  # Wait for the connection to establish

# Load TFLite model
interpreter = tf.lite.Interpreter(model_path="model/model.tflite")  # Replace with your model's path
interpreter.allocate_tensors()

# Get model input and output details
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()

def preprocess_frame(frame):
    """
    Preprocess the input frame to match the model's expected input size and type.
    Adjusted for UINT8 input.
    """
    input_shape = input_details[0]['shape']  # Model's expected input shape
    frame_resized = cv2.resize(frame, (input_shape[1], input_shape[2]))  # Resize to model's input size

    if input_details[0]['dtype'] == np.uint8:  # For UINT8 models
        input_data = np.array(frame_resized, dtype=np.uint8)  # Ensure UINT8 type
    else:  # For FLOAT32 models
        input_data = np.array(frame_resized, dtype=np.float32) / 255.0  # Normalize to 0-1 range

    input_data = np.expand_dims(input_data, axis=0)  # Add batch dimension
    return input_data

def classify_frame(frame):
    """
    Classify the frame using the TFLite model.
    Returns the predicted class index.
    """
    input_data = preprocess_frame(frame)
    interpreter.set_tensor(input_details[0]['index'], input_data)
    interpreter.invoke()
    output_data = interpreter.get_tensor(output_details[0]['index'])
    return np.argmax(output_data)  # Return the class index

def send_command(command):
    """
    Send the command to the Arduino via serial.
    """
    ser.write((command + '\n').encode())
    print(f"Sent: {command}")
```

```python

# Initialize webcam
cap = cv2.VideoCapture(1)  # Change '1' to the desired webcam index

if not cap.isOpened():
    print("Error: Could not open webcam.")
    exit()

try:
    while True:
        # Capture a frame from the webcam
        ret, frame = cap.read()
        if not ret:
            print("Error: Could not read frame from webcam.")
            break

        # Perform classification
        prediction = classify_frame(frame)

        # Determine action based on prediction
        if prediction == 1:  # Recyclable
            label = "Recyclable"
            send_command("clockwise")
        elif prediction == 0:  # Non-recyclable
            label = "Non-recyclable"
            send_command("anticlockwise")
        else:  # None
            label = "None (No action)"

        # Display the label on the frame
        cv2.putText(frame, f"Class: {label}", (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)

        # Show the frame
        cv2.imshow("Classification", frame)

        # Delay to allow motor rotation to complete
        if prediction == 0 or prediction == 1:  # If action was taken
            print("Waiting for motor rotation to complete...")
            time.sleep(2)  # Adjust delay as needed for motor rotation

        # Exit the loop if 'q' is pressed
        if cv2.waitKey(1) & 0xFF == ord('q'):
            print("Exiting program.")
            break

except KeyboardInterrupt:
    print("Program interrupted.")

finally:
    # Release resources
    cap.release()
    cv2.destroyAllWindows()
    ser.close()
    print("Webcam and serial connection closed.")
```

Sketch Code




```ino
#include <Wire.h>
#include "rgb_lcd.h"

rgb_lcd lcd;

const int colorR = 0;   // Red color for "Recyclable"
const int colorG = 255; // Green color for "Non-recyclable"
const int colorB = 0;   // Common blue intensity

int ctr_a = 9;
int ctr_b = 8;
int ctr_c = 11;
int ctr_d = 10;
int t = 1500; // Delay between steps (microseconds)
int stepsFor90 = 50; // Number of steps for 90 degrees (modify based on your motor step angle)

void setup() {
    pinMode(ctr_a, OUTPUT);
    pinMode(ctr_b, OUTPUT);
    pinMode(ctr_c, OUTPUT);
    pinMode(ctr_d, OUTPUT);
    Serial.begin(9600); // Initialize serial communication

    // Initialize LCD
    lcd.begin(16, 2);
    lcd.setRGB(255, 255, 255); // Set default LCD backlight color to white
    lcd.clear(); // Ensure the screen starts empty
}

void displayOnLCD(const String &message, int r, int g, int b) {
    lcd.clear(); // Clear previous message
    lcd.setRGB(r, g, b); // Set RGB color
    lcd.print(message); // Display the message
}

void rotateClockwise(int steps) {
    for (int i = 0; i < steps; i++) {
        digitalWrite(ctr_a, LOW); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
        digitalWrite(ctr_a, LOW); digitalWrite(ctr_b, LOW); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, LOW); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, LOW); digitalWrite(ctr_c, LOW); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, LOW); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, LOW); digitalWrite(ctr_d, LOW); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, LOW); delayMicroseconds(t);
        digitalWrite(ctr_a, LOW); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, LOW); delayMicroseconds(t);
    }
}

void rotateCounterClockwise(int steps) {
    for (int i = 0; i < steps; i++) {
        digitalWrite(ctr_a, LOW); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, LOW); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, LOW); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, LOW); digitalWrite(ctr_d, LOW); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, LOW); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, LOW); digitalWrite(ctr_c, LOW); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
        digitalWrite(ctr_a, HIGH); digitalWrite(ctr_b, LOW); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
        digitalWrite(ctr_a, LOW); digitalWrite(ctr_b, LOW); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
        digitalWrite(ctr_a, LOW); digitalWrite(ctr_b, HIGH); digitalWrite(ctr_c, HIGH); digitalWrite(ctr_d, HIGH); delayMicroseconds(t);
    }
}

void loop() {
    if (Serial.available() > 0) {
        String command = Serial.readStringUntil('\n'); // Read input from serial
        command.trim(); // Remove extra spaces or newlines

        if (command == "clockwise") {
            displayOnLCD("Recyclable", colorR, colorG, colorB); // Display "Recyclable"
            rotateClockwise(stepsFor90); // Rotate 90° clockwise
            delay(1000); // Wait for 1 second
            rotateCounterClockwise(stepsFor90); // Return to original position
        } else if (command == "anticlockwise") {
            displayOnLCD("Non-recyclable", colorG, colorR, colorB); // Display "Non-recyclable"
            rotateCounterClockwise(stepsFor90); // Rotate 90° counterclockwise
            delay(1000); // Wait for 1 second
            rotateClockwise(stepsFor90); // Return to original position
        } else {
            lcd.clear(); // Clear the LCD for "None"
        }
    }
}
```