DESIGN AND IMPLEMENTATION OF SMART WASTE BIN AND MONITORING SYSTEM USING INTERNET OF THINGS AND MACHINE LEARNING

by

Tun Aung Wai

A Research Report Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Internet of Things

Examination Committee:	Dr. Attaphongse Taparugssanagorn (Chairperson)
	Dr. Chaklam Silpasuwanchai
	Dr. Chantri Polprasert

Nationality:	Myanmar
Previous Degree:	Bachelor of Engineering in Electronics Technology University, Mandalay Myanmar
Scholarship Donor:	AIT SCHOLARSHIPS

Asian Institute of Technology
School of Engineering and Technology
Thailand
December 2023

AUTHOR'S DECLARATION

I, Tun Aung Wai, declare that the research work carried out for this research work was in accordance with the regulations of the Asian Institute of Technology. The work presented in it are my own and has been generated by me as the result of my own original research, and if external sources were used, such sources have been cited. It is original and has not been submitted to any other institution to obtain another degree or qualification. This is a true copy of the research final defense report.

Date: December 1, 2023

Name: Tun Aung Wai

Signature:

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to Dr. Attaphongse Taparugssanagorn, my primary supervisor, for his invaluable guidance and support throughout my research. I am also grateful to the faculty members of the Telecommunications Department and my research committee members, Dr. Chaklam Silpasuwanchai and Dr. Chantri Polprasert, at the Asian Institute of Technology.

I would like to extend my thanks to Ms. Premma Rao and Arunya P. S. S. Dumunnage for their assistance during my academic onboarding and ongoing support. Additionally, I am grateful to the institution for providing me with the AIT scholarship.

Lastly, I would like to express my gratitude to my parents, Mr. San Aung and Mrs. Win Win Khaing, for their unwavering support and belief in me. I would also like to acknowledge my family and friends for their continuous help and encouragement throughout my research journey.

ABSTRACT

This study presents the development of an innovative Smart Waste Management System (SWMS) designed to boost the efficiency of waste handling and mitigate environmental damage. The proposed system utilizes real-time waste segregation to enhance sorting accuracy. It comprises a series of garbage bins equipped with various sensors and actuators, all controlled by a microcontroller that processes sensor information and manages actuator operations. The microcontroller is embedded with a live classification model to ensure precise waste categorization.

Data gathered by the system, which includes types of waste and the status of each bin, is sent to a cloud-based platform for further analysis. This data is made available via an Internet of Things (IoT) application intended for use by waste management teams. The objective of this research is to design and test an SWMS that elevates the precision of waste sorting and to assess its functionality through empirical trials. The report will detail the methodology, system design, and findings from the implementation and testing phases.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

SWMS	= Smart Waste Management System
IoT	= Internet of Things
ESP32 CAM	= ESP32 Camera Module
ML	= Machine Learning
Blynk	= Blynk App (A mobile platform for IoT)
CNN	= Convolutional Neural Network
SVM	= Support Vector Machine
AWS	= Amazon Web Services
LoRa	= Long Range
LCD	= Liquid Crystal Display
MQTT	= Message Queuing Telemetry Transport
API	= Application Programming Interface
PIR	= Passive Infrared Sensor
TCP/IP	= Transmission Control Protocol/Internet Protocol
EEPROM	= Electrically Erasable Programmable Read-Only Memory
AJAX	= Asynchronous JavaScript and XML
LED	= Light Emitting Diode
ESP32CAM	= ESP32 Camera Module
SSL/TLS	= Secure Sockets Layer/Transport Layer Security
ADC	= Analog-to-Digital Converter
GPS	= Global Positioning System
FOMO	= Features of Mobile Objects
SSID	= Service Set Identifier
GPIO	= General-Purpose Input/Output
VCC	= Voltage at the Common Collector
GND	= Ground
HTTP	= Hypertext Transfer Protocol
MSE	= Mean Squared Error
mAh	= Milliampere-hour
YOLOv5	= You Only Look Once, version 5
Line Notify	= Notification Service from Line Messaging Platform

CHAPTER 1
INTRODUCTION

1.1 Background of the Study

The growing challenge of managing waste in rapidly urbanizing regions is a pivotal concern in modern urban development. Conventional waste management approaches are often plagued by inefficiencies like inconsistent collection schedules and a lack of live monitoring, leading to heightened environmental issues (Kumar, Prandi, Rathore, & Lloret, 2017). This situation highlights the imperative for innovative waste management strategies, notably incorporating IoT and advancements in machine learning (ML).

IoT's capacity to interconnect a diverse range of sensors and devices marks a transformative shift in waste management. Equipped with advanced sensors, such as the ESP32 Camera Module (ESP32 CAM) and ultrasonic sensors, smart bins can effectively gauge waste levels, enabling more systematic collection schedules and reducing operational expenses (Maksimovic & Natalizio, 2017). The ESP32 CAM is especially notable for its blend of functionality and energy efficiency, making it a suitable choice for SWMS in restricted settings like student dormitories.

In waste categorization, machine learning algorithms, especially efficient ones like MobileNetV2, are set to revolutionize waste segregation. Integrating these algorithms into microcontrollers, as seen with the ESP32 CAM, signifies a major leap towards autonomous and more effective waste management systems. Platforms such as Edge Impulse simplify the process of training and implementing these ML models on microcontrollers, thereby broadening the scope for smaller-scale projects (Vijayalakshmi & Uthariaraj, 2018).

This study seeks to harness these technological advancements for developing a SWMS. The objective is to improve the precision and purity in waste segregation, contributing to more sustainable waste handling methods. By merging IoT with ML, the system is anticipated to boost the efficiency and effectiveness of waste management, in line with the overarching goal of sustainable urban growth.

1.2 Statement of the Problem

The necessity for sustainable waste handling has intensified with the quick expansion of urban areas. Outdated and unsustainable, traditional waste management techniques are leading to considerable environmental pollution, resource wastage, and potential health hazards. These inefficiencies not only increase waste disposal costs but also negatively impact the living standards in urban settings. There is an urgent requirement for advanced and efficient waste management systems, particularly in larger urban contexts.

A promising response to inefficient waste management is the introduction of intelligent waste monitoring systems and advanced smart bins. Such systems can provide immediate data on waste accumulation, bin status, and their locations, thereby optimizing collection routes and diminishing collection expenses. Smart bins, furnished with sensor and automation technology, can separate waste types, compact waste to save space, and notify authorities when nearing full capacity. However, their high cost remains a significant obstacle to widespread adoption, making them unaffordable for smaller entities.

This research aims to create a prototype of an affordable, efficient, and sustainable smart waste bin. This bin will not only identify and sort different waste materials but also use Line Notify to alert when nearing full capacity. Additionally, the system will assist in locating full bins on a Google map, improving waste collection routes and lowering costs. The results of this study are expected to contribute to the creation of more advanced and cost-effective smart waste management systems, making them accessible to smaller businesses and communities, and promoting sustainable waste management in various environments.

1.3 Objectives

The study's objectives are as follows:

1. To optimize machine learning for microcontrollers: Identifying the most effective methods for training and deploying machine learning models in our smart bin system.
2. To achieve a cost-effective, low-power smart bin design: Creating a smart bin system that is economical in terms of cost and energy while maintaining key functions like waste identification and notification alerts.

3. To implement instant notification capabilities: Enabling the system to send immediate alerts regarding waste levels and battery life for maintenance.
4. To develop a mechanical sorting mechanism: Testing and implementing a sorting mechanism, initially using a servo motor, based on the classification results.

1.4 Scope and Limitation

1.4.1 Scope

The research is centered on developing a smart waste bin capable of identifying and segregating various waste types, using machine learning for accurate waste detection. It also focuses on real-time monitoring of waste levels in bins, employing sensors to initiate notifications when bins are full, thus enhancing waste collection efficiency. Another significant aspect is designing an economically viable and sustainable system, suitable for small-scale environments like businesses or community centers. The system will include user interaction features, particularly through the Thingspeak application, providing updates on bin statuses and other pertinent details. The study encompasses the development and testing of a prototype, with a focus on evaluating its real-world performance to determine its practicality and efficiency.

1.4.2 Limitations

The smart bin's effectiveness is dependent on the performance of the selected hardware components, such as sensors and microcontrollers. The precision of waste classification relies on the chosen ML model and the extent and quality of the training data. Identifying certain waste types may be limited by these factors. The system's functionality might also be hampered by issues with network connectivity, particularly in areas with limited internet access. While the system is tailored for smaller-scale applications, scaling it up for larger municipal usage could pose challenges in cost, complexity, and management. The system's success is partly reliant on user participation and acceptance. Hesitation to adopt new technology or changes in user habits can affect the system's efficiency. Concerns about the long-term maintenance and the durability of the smart bins under various environmental conditions could impact the system's sustainability. Dependence on specific software platforms like Blynk and Google Maps may introduce constraints related to platform stability, updates, and compatibility.

1.5 Organization of the report

The study is structured as follows:

1. Chapter 1 includes an introduction to the study, covering the background of smart waste management systems, objectives, scope, and limitations.
2. Chapter 2 presents a literature review on standard waste management systems, SWMS, related research, comparisons of waste segregation methods, database types, power consumption, and a summary of the chapter.
3. Chapter 3 details the project overview, including hardware and software architecture, waste level detection, battery voltage measurement, security features, data handling and visualization, and an analysis of performance.
4. Chapter 4 discusses the findings of the study.
5. The final chapter concludes the report, summarizing the achieved objectives and suggesting potential future developments.

CHAPTER 2
LITERATURE REVIEW

2.1 Normal Waste Management Systems

Standard waste management practices typically involve the gathering, transportation, and handling of waste. These conventional methods usually include picking up waste from residential and commercial areas and moving it to a central facility for sorting, disposal, or recycling.

The limitations and drawbacks of standard waste management systems are notable:

- **Inaccuracy in Waste Sorting:** Common waste management approaches often depend on labor-intensive and prone to error manual sorting. This results in a lack of precision and cleanliness in sorting, leading to potential adverse environmental and economic outcomes.
- **Labor and Cost Intensiveness:** These systems demand significant efforts in terms of collection, transport, and sorting of waste. This proves to be costly, especially in urban areas with higher labor expenses. Furthermore, the cost of transporting waste can increase substantially if the processing centers are far from the collection points.
- **Lack of Timely Data:** Conventional waste management systems generally do not incorporate real-time data gathering and analysis, making it challenging to enhance efficiency or identify areas for improvement.
- **Environmental Impact:** Traditional methods of waste management can lead to environmental issues such as air and water pollution, emission of greenhouse gases, and harm to natural habitats.
- **Need for Sustainable Solutions:** Given their cost, inefficiency, and environmental harm, there is an increasing demand for more innovative and sustainable approaches in waste management.

Figure 2.1 illustrates a prevalent issue in these systems: bins become overfilled due to delayed collections, leading to waste spillover in public spaces, which can pose environmental and health risks.

Figure 2.1

(Faris, 2018)Full waste bin due to untimely collection

2.2 Smart Waste Management Systems

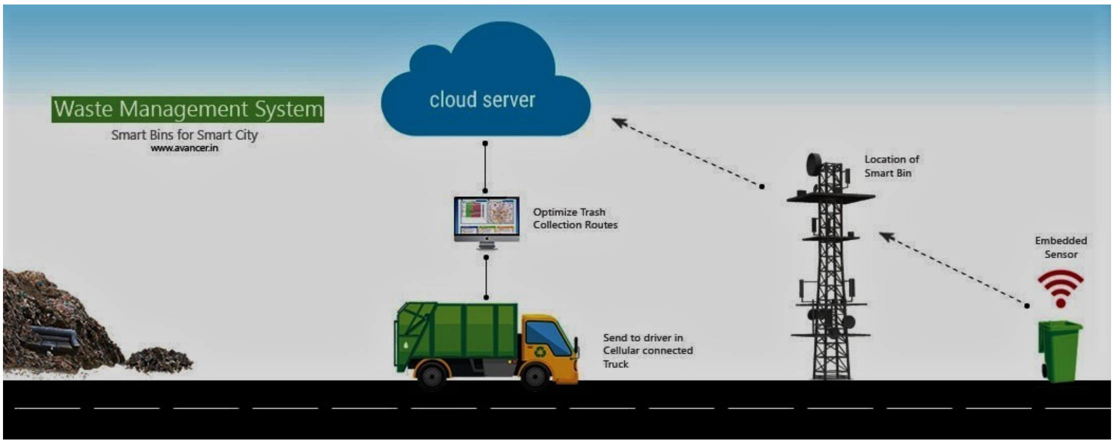
SWMS represent a new era in waste management, utilizing advanced technologies like sensors, real-time data collection and analysis, and machine learning. These innovations significantly enhance the efficiency and effectiveness of waste collection, transportation, and processing. Figure 2.2 showcases one of the key advantages of SWMS, which is the optimization of collection routes. The benefits of SWMS over conventional waste management systems are numerous and include:

- Enhanced accuracy and purity in waste sorting.
- Reduced labor and transportation costs.
- Improved availability of data for decision-making and operational optimization.
- Better environmental performance, contributing to sustainable waste management practices.

These benefits collectively make SWMS a more cost-effective and sustainable solution for managing waste materials.

Figure 2.2

Route optimization in Smart Waste Management Systems



2.3 The Benefits of Smart Waste Management: A Comparison with Traditional Systems

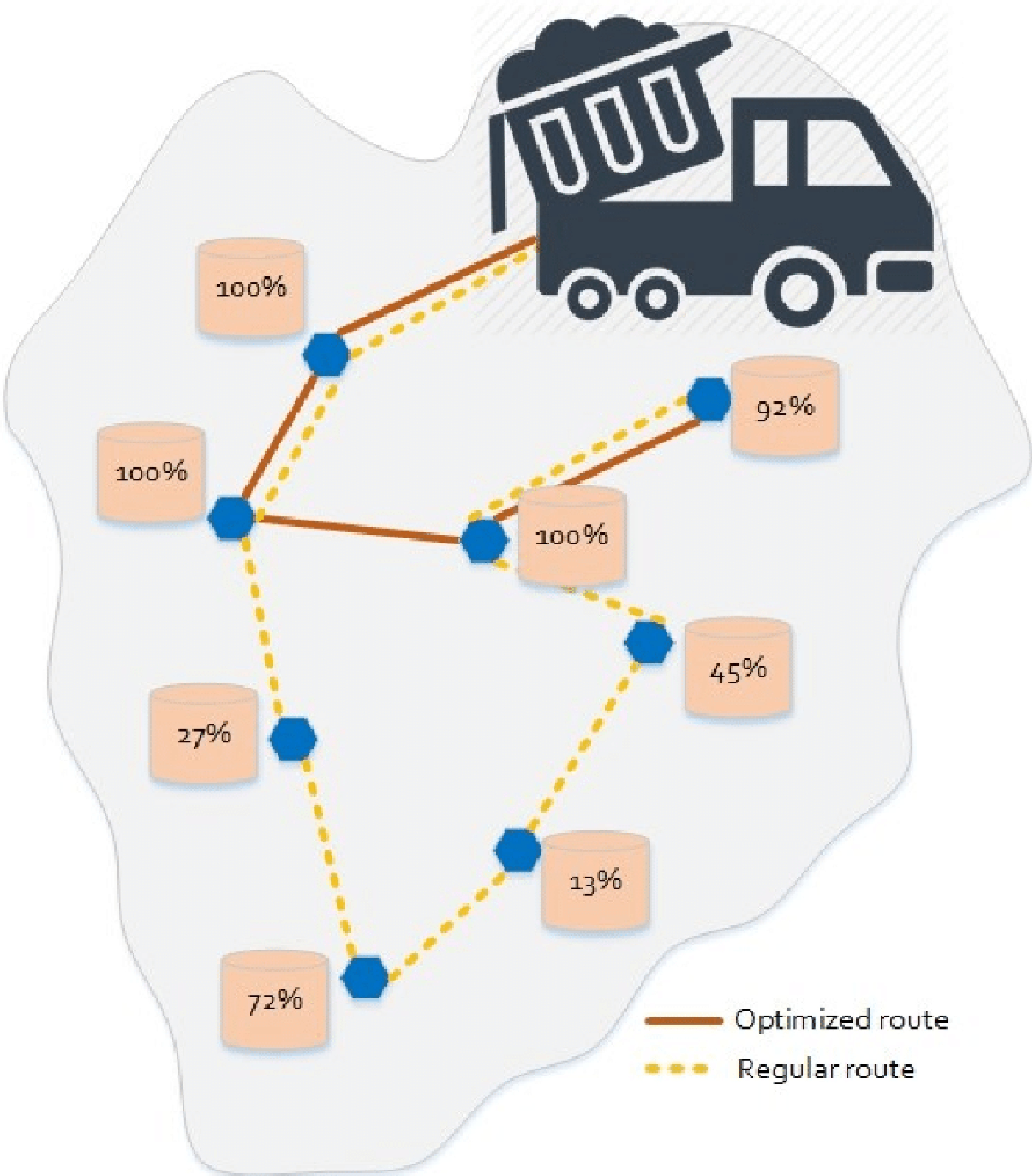
The traditional method of monitoring trash in cans involved manual effort, which was time-consuming and costly. However, with current technological advancements, such manual processes are no longer necessary. Manual monitoring often leads to waste collection trucks visiting dumpsters that are not yet full, wasting time and resources. With rapid urbanization, there is an increasing need for smarter waste management solutions. Implementing IoT-based waste management systems offers several benefits for businesses and local communities:

1. **Reduction in Collection Costs:** Smart garbage bins equipped with real-time sensors provide fill level information, reducing the cost of collection. IoT technology analyses this data to optimize routes for collection trucks, avoiding unnecessary trips to empty containers, thus saving fuel and labor.
2. **Elimination of Missed Pickups:** Unlike traditional methods, smart waste management systems prevent bins from becoming too full. Authorities receive timely alerts when bins approach full capacity, allowing for more efficient scheduling of collection.
3. **Enhanced Data Analysis:** Smart systems enable detailed analysis of waste data. Sensors measure the volume, weight, and type of waste, aiding in optimizing collection routes and schedules, identifying recycling opportunities, and highlighting inefficiencies.

4. **CO2 Emission Reduction:** Optimized collection routes result in a smaller carbon footprint, making waste management more environmentally sustainable.

Figure 2.3

(Aazam et al., 2016)Route optimisation with Optimized Route



2.4 Related Researches on Smart Waste Management

Research in (Sharmin & Al-Amin, 2016) primarily involves two stages: first, the collection and transmission of data from waste bins, followed by determining the optimal route for waste collection. Load sensors, specifically the SEN-10245 model capable of measuring up to 50 kg, are used to gauge the waste weight. Ultrasonic sensors are employed to assess the fill level. The Arduino Uno plays a crucial role in sending sensor data to a central server via a GPRS module. Data mining techniques are applied to analyze when waste bins are nearly full, with data collection occurring between 9.30 a.m. and 4.30 p.m. An ant colony optimization algorithm assists in finding the most efficient path for each collection truck.

A study in (Bharadwaj, Rego, & Chowdhury, 2016) proposes an IoT-based solid waste management system utilizing a range of sensors like DHT22 for temperature, MQ-135 for gas detection, IR sensor, PIR sensor, and a load cell for various measurements, including the presence of harmful gases, waste quantity, and user presence. Data communication is carried out using LoRa and then transmitted to the cloud for monitoring. With each bin equipped with its sensor set, the system manages five different types of waste across four bins, impacting the overall cost.

(Samion et al., 2018) introduces an automated system for sorting recyclable materials. The recycling bin incorporates various sensors: inductive for plastic, capacitive for metal, photoelectric for paper, and proximity sensors for motor positioning. These sensors, linked to an Arduino Uno, are activated when waste enters the bin to identify its material type. After detection, a DC motor moves the waste to the correct compartment, and a pusher then segregates the recyclable material.

The study by (Cerchecci et al., 2018) employs an ultrasonic sensor to monitor the amount of waste in a container, with data sent via LoRa communication. The system's focus is on measuring waste volume, and it includes power management components like a counter and switching regulator.

(Hulyalkar, Deshpande, Makode, & Kajale, 2018) proposes a smart bin system integrating machine learning, image processing, and IoT. The system utilizes a Convolutional Neural Network (CNN), trained on 400-500 images per waste category, to classify waste types. TensorFlow and Keras are used to implement the CNN, which comprises eight layers and undergoes 50 training epochs. The Raspberry Pi microcontroller executes image processing, achieving approximately 84% accuracy in waste classification.

In (Saminathan & Musipatla, 2019), a proposed IoT-based smart bin has three compartments with distinct functionalities: a metal detector and IR sensor in the first, an IR and moisture sensor in the second for differentiating dry and wet waste, and a divided third compartment for separate waste collection. Data transmission is enabled via WiFi to a specific server. The bin features a rotating table with three divisions for dry, wet, and metal waste, rotating based on the type of waste detected.

(Ziouzios & Dasygenis, 2019) describes a high-tech bin designed for recyclable collection, using a Raspberry Pi 3, Xilinx PYNQ-Z1 FPGA board, and a pre-trained ResNet-34 CNN. LoRa network is utilized for data transmission from the sensor node to the gateway, with the system achieving 92.1% detection accuracy and an average processing time of 1.82 seconds.

An intelligent waste classification system is proposed in (Adedeji & Wang, 2019), employing a 50-layer ResNet-50 model with SVM for categorizing waste into different types like glass, metal, paper, and plastic. The system removes the top classification layer of ResNet-50 for feature extraction and uses SVM for classification, achieving 87% accuracy.

(S.Vigneshwaran, 2019) suggests a smart trash can using LoRa technology, addressing the limitations of WiFi, Bluetooth, and cellular networks in terms of noise, interference, and inefficiency. LoRa enables long-range transmission over 10 km, suitable for high-capacity networks. The system includes a cloud platform, LoRa gateway, remote diagnostic system, and various sensors and modules for comprehensive waste monitoring.

The study in (Rashmi, Ameenulla, & Kumar, 2019) aims to provide a cloud-based solution for monitoring solid waste odor in cities, using AWS Kinesis for real-time data processing. Sensors installed at various disposal sites collect data on pollutants, which is then processed and stored in a database, accessible in XLS format. The AQI is used for air quality assessment.

(Atayero, Williams, Badejo, & Popoola, 2019) utilizes ultrasonic sensors with an Arduino Uno microcontroller and LCD display for measuring solid waste volume in bins. ThingSpeak IoT platform is used for data collection and storage, compatible with various programming languages and devices. A Twitter account for each bin is set up for sending notifications via "ThingTweet."

(Pereira, Parulekar, Phaltankar, & Kamble, 2019) proposes a smart bin system focused on waste segregation and recycling. It uses ultrasound sensors (HC-SR04) for bin opening, servo motors for door operation, and capacitive copper plates for wet and dry waste segregation. The system also employs an HC-05 Bluetooth Module for data transmission and Rstudio for data visualization.

The goal of (Chung, Peng, & Yeh, 2020) is to develop an automated garbage can operation system with environmental monitoring, utilizing LoRaWAN communication networks. The system features proximity sensors for waste type identification, embedded motors for sorting, and a C graphical interface for monitoring. Environmental parameters are measured using various sensors, and data is transmitted via TCP/IP to a MariaDB cloud database.

According to (Catarinucci et al., 2020), RFID tags on trash cans measure waste weight, with data sent to a cloud server via a GPRS-enabled RFID reader. The server stores data in a relational database, accessed through APIs and visualized on web application dashboards.

Medehal, Annaluru, Bandyopadhyay, and Chandar (2020) focuses on collecting data on trash levels and collection timestamps using ultrasonic sensors. The ESP8266 controller and Blynk app are used for data visualization. The study introduces a unique approach to sensor data analysis, dividing bins into sections for complete coverage.

In (Savla, Parab, Kekre, Gala, & Narvekar, 2020), the proposed system includes a wifi-enabled microcontroller, various sensors for environmental monitoring, and a web server and mobile app for data access. The backend server uses the SVM algorithm to predict AQI, and a Caffe model for image verification.

(Khan & Naseer, 2020) presents an IoT-based university garbage monitoring system using ultrasound sensors, Arduino Uno R3, and ESP8266. The system's unique feature is a printed campus map dashboard with LED indicators for full bins, facilitating efficient waste collection.

Paper (Rajesh et al., 2021) discusses a system using an ultrasonic sensor for measuring trash levels in bins, with data displayed on an LCD and transmitted via a WiFi module to a central hub. The system activates when bins are full and employs various components for efficient operation and data transmission.

Finally, (Nagesh, Kotari, & Chethan, 2021) describes intelligent bins equipped with ultrasonic sensors and ESP microcontrollers, transmitting data to the cloud via MQTT protocol. The system features a dynamic website and an Android application for municipal workers and residents, offering real-time bin status updates on a city map.

2.5 Recent and Related Work

Table 2.1 provides an overview of the techniques, software, and hardware utilized in several related papers, offering valuable insights into the diverse approaches used for waste management research.

Table 2.1

Related Papers

Paper Cloud Platform	Sensors/Modules Used Communication Method	Machine Learning Model Database, Data Visualization
(Sharmin & Al-Amir, 2016) -	Weigh Sensor, Ultrasonic Sensor, GPRS module -	- -
(Samion et al, 2018) -	Inductive Sensors, Capacitive Sensors, Photoelectric Sensors, Arduino Uno -	- -

(Cerchecci et al, 2018) -	A counter and switching regulators to control the amount of power consumed -	- -
(Hulyalkar, Deshpande, Makode, & Kajale, 2018) -	 -	CNN -
(Saminathan & Musipatla, 2019) -	Moisture Sensor -	-
(Ziouzios & Dasygenis, 2019) -	Pi 3, FPGA board LoRa	Pre-trained ResNet-34 -
(Adedeji & Wang, 2019)	 -	SVM -
(Rashmi, Ameenulla, & Kumar, 2019) AWS kinesis	- -	- -

(Pereira, Parulekar, Phaltankar, & Kamble, 2019) Rstudio	Ultrasonic Sensor, ULN2003A IC, Capacitive Copper Plates, HC-05 Bluetooth Module -	- -
(Chung, Peng, & Yeh, 2020) -	Electrostatic Capacitance Type Proxi- mity Switch TCP/IP	- MariaDB
(Medehal et al, 2020) -	ESP8266 -	- Blynk
(Savla, Parab , Kekre, Gala , & Narvekar, 2020) -	7805 regulator (12V to 5V), Arduino Uno -	- LCD display

Compared to the related papers, our proposed system stands out with a more complete design. Firstly, our system takes into account an important factor which is power consumption, which the other papers did not address. We have also considered the scalability of the system by using Node-RED which enables easy implementation of more bins and nodes to the system. Additionally, in contrast to the other papers which open the bin door based on the detection of someone approaching the bin, our system opens the related bin based on the classified waste category which provides more accuracy. Other related papers proposed different designs such as sorting waste in the bin using wet and dry sensors or computer vision models with robot arms for sorting, but these approaches have drawbacks and can be expensive. Our design is simple, cost-effective, and also educates and reminds humans to distinguish waste into categories. Overall, our proposed system has a more complete design, is easy to implement, and addresses important factors that were not considered in the related papers.

2.5.1 Comparison of Different Methods for Waste Segregation

Various methods for waste segregation offer distinct advantages and limitations. For instance, the wet sensor method is effective in distinguishing between dry and wet waste, which is beneficial in certain waste management systems (Incrocci et al., 2010). However, its capability is limited to identifying only these two waste types, which might not suffice for more comprehensive waste management systems.

On the other hand, capacitive copper plates are advantageous due to their lower computational power requirements. Despite this, their accuracy can be compromised by environmental factors and a limited detection range for different materials (NagarajD, Badami, & Santosh, 2017).

In comparison, Convolutional Neural Network (CNN) models stand out with enhanced accuracy and reliability. Unlike the other two methods, CNNs have the capability to recognize a broader spectrum of materials, making them more versatile for varied waste segregation needs.

Table 2.2

Comparison of waste detection methods

Method	Advantages	Limitations
CNN Models	Improved accuracy and reliability. Ability to distinguish between dry and wet waste.	Requires higher computational power. Limited accuracy due to sensitivity to environmental conditions.
Capacitive Copper Plates	Less sensitive to environmental conditions. Ability to distinguish between dry and wet waste.	Limited to detecting only two types of waste (dry and wet). Limited accuracy due to a limited number of materials.
Wet Sensors	Ability to distinguish between dry and wet waste.	Limited accuracy due to sensitivity to environmental conditions.

2.5.2 Comparison of Different Methods for Waste Segregation

This Table 2.2 provides a brief comparison of the three methods for waste segregation, highlighting their advantages and limitations. The choice of method may depend on the specific requirements and goals of a given waste management system.

Relational Database vs. Time-Series Database

In a relational database, data represents the current state of an entity. These types of databases are good for:

- Content management systems
- Storing transactional data
- Storing data that needs to be updated over time
- Long-term storage
- Throughout 100K+ to 1M+ inserts per second
- Bytes stored per time/value tuple: 2-10 vs 30-100 (rdbms)

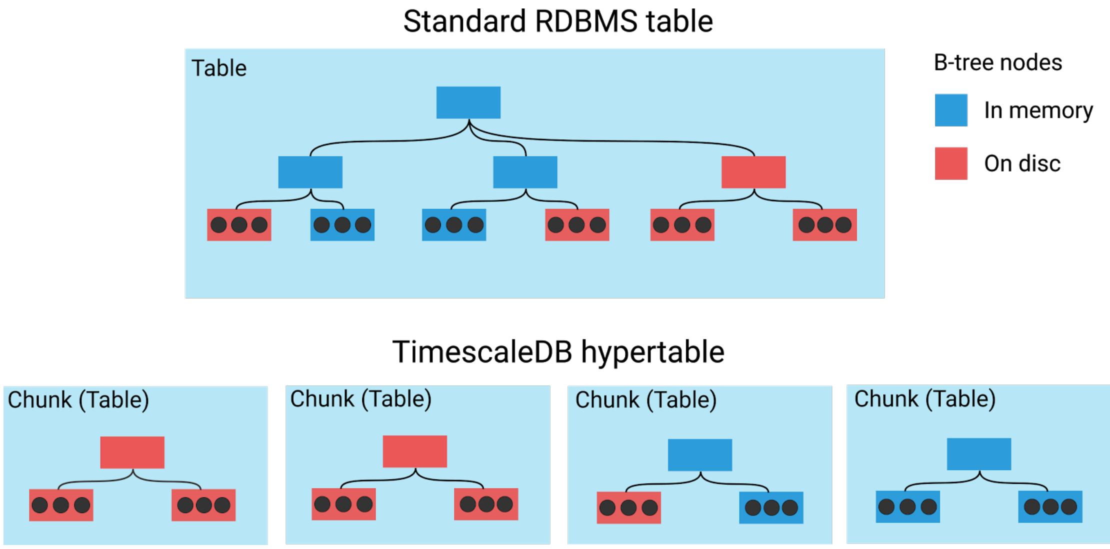
Time-series databases are useful for:

- Monitoring systems over time
- Analytical/reporting data
- Append-only changes (write once, read many)
- Short-lived data sets

In this type of database, the change of state can be visualized over time. In fact, data stored in a time-series database is always related by time.

Figure 2.4

(Timescale, 2019)Relational Database vs Time-Series Database



2.6 Power Consumption

The authors of the paper (Akintade, Yesufu, & Kehinde, 2019) on power consumption models for microcontroller-enabled low-cost IoT monitoring nodes note that the ESP8266-12E module's high current consumption limits the practicality of ESP8266-enabled IoT monitoring nodes powered by small batteries. To address this issue, the authors suggest methods to reduce power consumption, including deep sleep mode, reduced transmit power, and optimized code (Chéour, Khriji, abid, & Kanoun, 2020). They explain that deep sleep mode is suitable for time-based monitoring when there is a long time interval between sensor readings or when it is not necessary to continuously transmit data or monitor a physical parameter. Light sleep mode, on the other hand, is used in event-based monitoring, where the physical parameter is continuously monitored, and a threshold is set such that data obtained from the physical phenomenon is transmitted only if it exceeds the set threshold. These measures can significantly extend the battery life of an ESP module, enhancing the overall performance and reliability of IoT monitoring nodes.

2.7 Conclusion

The literature review covers the current state of waste management systems, including traditional methods and the benefits of implementing SWMS. It also discusses related research on smart waste management, including a comparison of different systems, sensors, and machine learning techniques. The review also explores the benefits of instance segmentation for waste classification and compares different methods for waste segregation. Finally, the review discusses the differences between relational databases and time-series databases, and highlights the issue of power consumption in microcontroller-enabled IoT monitoring nodes.

Based on the related research, our proposed system is a smart waste management system that uses classification to guide waste disposal and improve the accuracy of waste sorting. It consists of a network of trash bins equipped with sensors and actuators, and a microcontroller that processes the data from the sensors and controls the actuators. The microcontroller also runs a real-time classification model to identify the type of waste being disposed and only allows the disposal of waste in the correct bin. Data from the system is transmitted to a cloud platform for storage and analysis, and is also made available to waste management staff through an IOT application. Our proposed system is designed to be low-cost, effective, and sustainable, and can improve the accuracy and purity of waste sorting.

CHAPTER 3
METHODOLOGY

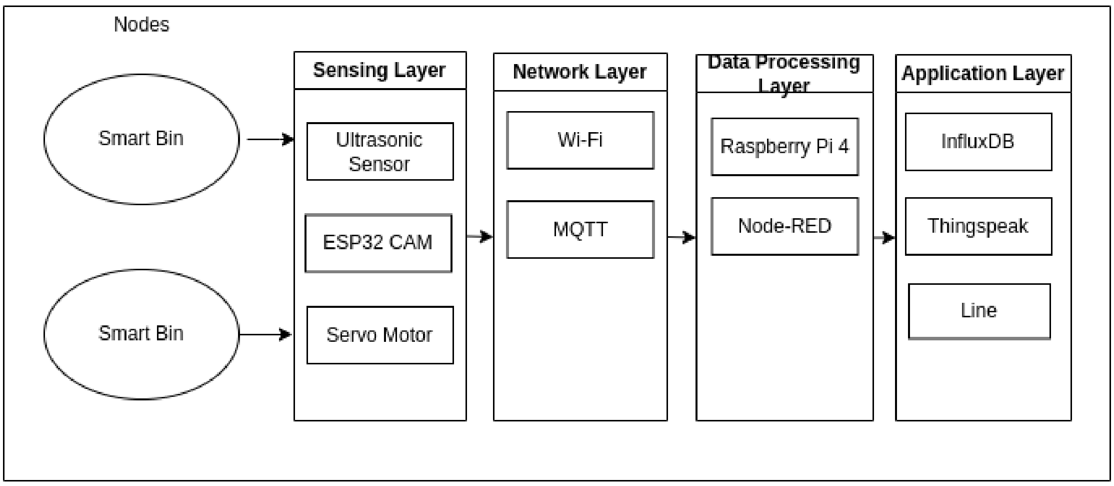
3.1 Overview of Smart Bin System Functionality

The functionality of our intelligent bin system is designed with user ease in mind. The process starts with the user pressing a button, which activates the ESP32CAM from its energy-saving deep sleep state, causing the servo motor to unlatch the bin. The ultrasonic sensor then assesses the amount of waste inside the bin, a crucial measurement in this system. This information, along with the current battery status, is communicated by the ESP32CAM to the Node-RED server. After sending this data, the ESP32CAM reverts to sleep mode, a key feature for maintaining the system's energy efficiency.

Server-side, Node-RED analyzes the incoming data against specific parameters. If the level of waste inside the bin surpasses 70 %, the battery charge falls under 20 %, or the bin hasn't been emptied in more than two weeks, notifications are dispatched through Line Notify. Concurrently, the ESP32CAM sorts the waste based on its type, directing the servo motor accordingly. This combination of technological elements not only streamlines waste management but also enables continuous monitoring and automatic updates, making the system particularly advantageous for settings with multiple bins, such as in student dormitories.

Figure 3.1

System Block Diagram



3.2 Hardware Architecture and Components Selection

The smart bin system is engineered with a selection of components, each meticulously chosen for their unique roles in enhancing the system's functionality and design efficiency. At the core of the system is the ESP32 cam OV2640, a module renowned for its dual functionality. This module not only classifies waste in real-time but also facilitates data transmission, making it a pivotal component in our system. Its compact size and integrated camera are perfectly suited for efficient waste sorting, while its ability to seamlessly connect with the Node-RED server ensures smooth data communication and system management.

For the mechanical operations of the bin, particularly the opening and sorting of waste, the SG90 servo motor is employed. This motor was selected for its precision in rotational movement, which is crucial for controlled and accurate operation. The ultrasonic sensor, another key component, plays a vital role in measuring the waste level inside the bin. This non-intrusive sensor was chosen for its reliability and accuracy in distance measurement, utilizing high-frequency sound waves to determine the amount of waste present.

The system's power management is addressed through the inclusion of a 500mAh battery. This particular capacity was chosen to strike a balance between compactness and efficiency, catering to the system's energy needs while maintaining a lightweight and portable design. Additionally, the integration of a push button acts as a user interface and an external interrupt for the ESP32, enabling it to conserve energy by waking from deep sleep mode as needed.

Rounding off the hardware architecture is the Raspberry Pi4, which serves as the host for the Node-RED server and the ML API. This setup not only underpins the system's computational capabilities but also ensures its adaptability and scalability.

Figure 3.2 presents the hardware block diagram of our smart bin system. This diagram visually encapsulates the interconnectivity and functionality of each component within the system, offering an overview of how they collectively contribute to the system's operation and efficiency.

Figure 3.2

Hardware Block Diagram

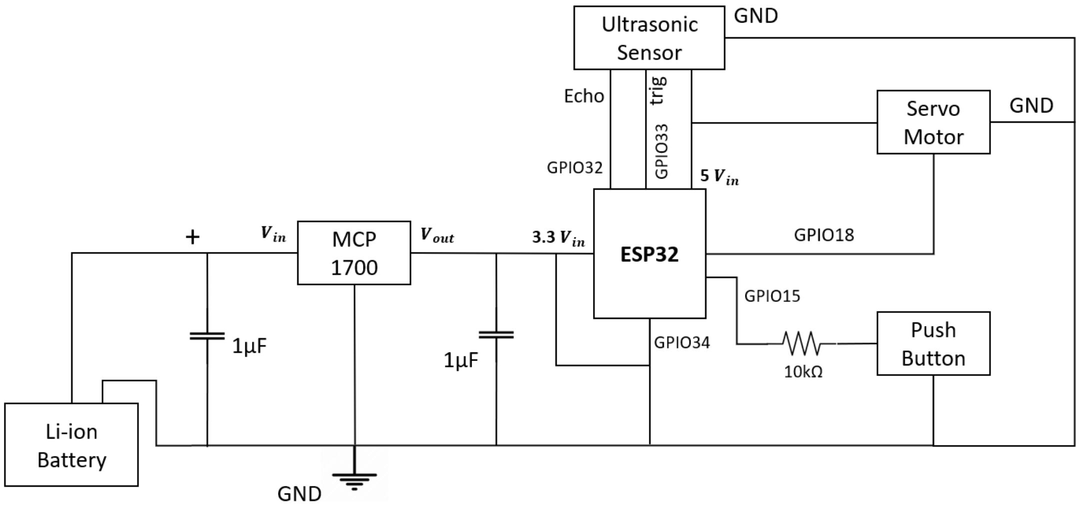


Table 3.1

Specifications of Components Used

No	Component	Technical Details
1	500 mAh Battery	Discharge cut-off voltag: 2.45V 3.0V, Operating temperature: -20°C +60 °C, Standard Charging: 3.0 hours, Battery type: Lithium polymer
2	Ultrasonic Sensor (HC-SR04)	Operating voltage: DC 5V, Operating current: $<$15mA, Detection distance: 2cm - 400cm, Measurement angle: 15 degrees, Trigger signal: 10us TTL pulse
3	SH90 Servo Motor	Operating voltage: DC 4.8V - 6.0V, Torque: 2.2 kg-cm at 4.8V, 2.5 kg-cm at 6.0V, Speed: 0.16 sec/60 degrees at 4.8V, 0.14 sec/60 degrees at 6.0V, Operating temperature: -20C to 60C, Dimensions: 22.8mm x 12.2mm x 28.5mm
4	Raspberry Pi 4 Model B	Processor: Broadcom BCM2711 quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz, RAM: 4GB, Connectivity: Dual-band (2.4GHz and 5.0GHz) wireless LAN, Bluetooth 5.0, Gigabit Ethernet, Ports: 2 $\times$ USB 3.0, 2 $\times$ USB 2.0, 2 $\times$ micro-HDMI, 3.5mm audio jack, 40-pin General-Purpose Input/Output (GPIO) header, 2-lane MIPI CSI camera port, 2-lane MIPI DSI display port, microSD card slot
5	ESP32 CAM OV2640	Power Supply Core: 1.3V DC ± 5%, Analog: 2.5 3.0V DC, I/O: 1.7V to 3.3V, S/N ratio: 40dB, Output Format: YUV/RGB/Raw RGB Data

3.3 Implementation of Software and System Framework

Our smart bin system employs Node-RED as its core server, handling the data communication from the ESP32. Chosen for its intuitive interface and efficient data processing, Node-RED stands as a prime server solution for our application. Its capability to function effectively on low-cost hardware like Raspberry Pi or a basic laptop enhances the system's cost-efficiency. Furthermore, Node-RED's flexible and modular design offers scalability, an essential attribute for potentially expanding our system to manage additional bins in the future. The incorporation of Node-RED in our setup is depicted in the Software Architecture Block Diagram, shown in Figure 3.3.

3.4 Software Framework

The heart of our software structure is the Node-RED server, hosted on a Raspberry Pi. Data transmission from the ESP32 to the InfluxDB through Node-RED involves a series of steps utilizing MQTT:

1. **Setting up MQTT Broker:** We install the Mosquitto MQTT broker on the Raspberry Pi to enable a communication channel between the ESP32 and Node-RED.

2. **Linking ESP32:** The ESP32 connects to the MQTT broker through a client library and is set up to send sensor data to an MQTT topic.

3. **Configuring Node-RED:** Within Node-RED, an MQTT input node subscribes to the ESP32's topic, allowing the server to collect sensor data.

4. **Handling Data:** Node-RED processes this data using its in-built nodes for analysis, transformation, and initiating actions based on the received information.

5. **Storing Data in InfluxDB:** The processed data is stored in InfluxDB for future analysis and visualization, necessitating the configuration of InfluxDB nodes with relevant database details.

The sensor data resides in InfluxDB until the bin reaches full capacity or other specified conditions are met, triggering alerts. The system also transmits data concerning power and battery voltage to InfluxDB. Figure 3.4 presents a flowchart that outlines the operational procedure of our system, supplementing the details provided in the Software Architecture Block Diagram.

Figure 3.3

Software Architecture Block Diagram

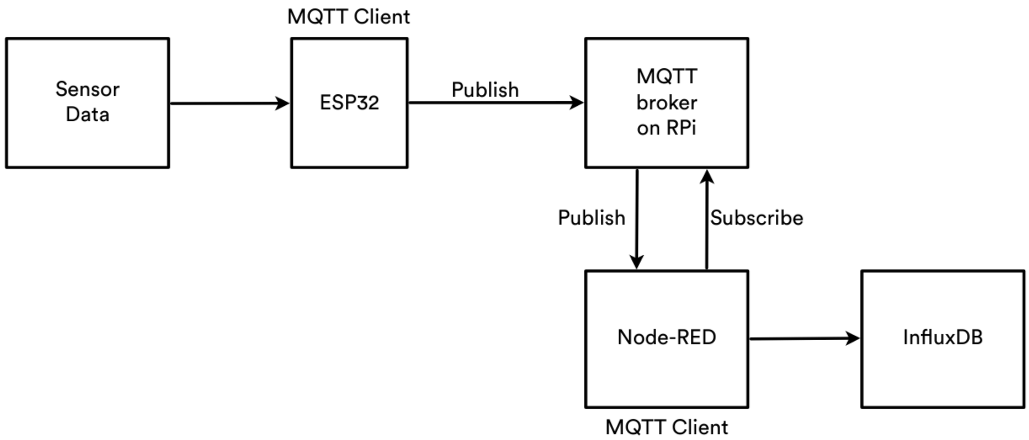
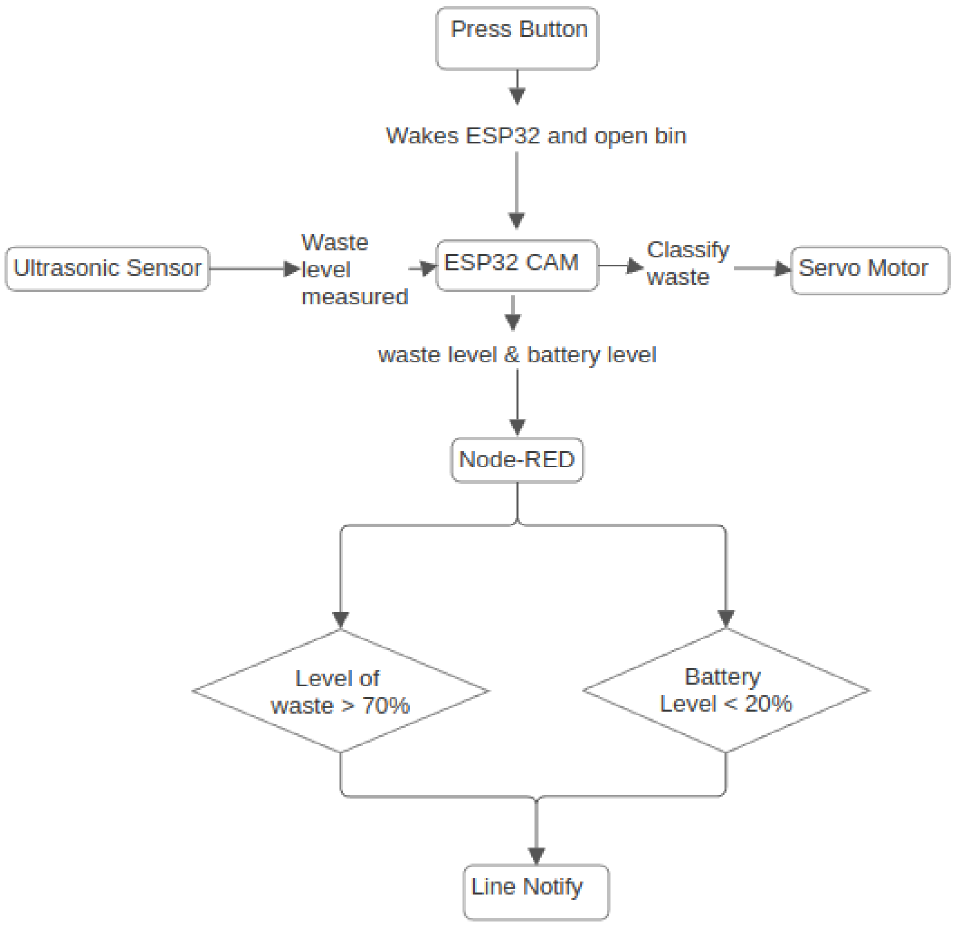


Figure 3.4

System Flow Chart



3.5 Monitoring Battery Voltage

The ESP32's analog input is capable of measuring voltage from 0V to 33V, corresponding to an Analog-to-Digital Converter (ADC) range of 0 to 4095, as shown in Equation 3.1. The formula below calculates the battery's voltage level:

$$Battery\ Voltage\ Level = \frac{ADC\ Reading \times 3.3V}{Maximum\ ADC\ Reading\ (4095)} \tag{3.1}$$

When the battery voltage falls below 3.1V, the ESP32 communicates a "0" message to a MQTT topic, signaling low battery on the Thingspeak's dashboard. Conversely, a reading above 3.1V results in sending a "1" message, indicating sufficient battery voltage.

3.6 Network Security Enhancements

The Wireless Network Name (SSID) is utilized alongside WPA2 and PSK configurations for network data authentication and encryption. To further protect device communication, the implementation of SSL/TLS encryption is recommended, ensuring the security of network traffic against unauthorized access and alterations.

3.7 Optimizing Power Usage and System Performance

Deep Sleep Mode Functionality: The ESP32's deep sleep mode plays a pivotal role in conserving energy. This mode is activated when the system is not in use (i.e., when the user is not pressing the push button), substantially reducing power usage. This strategy not only prolongs the battery lifespan but also supports the system's eco-friendly objectives. The push button's dual functionality of waking the ESP32 and triggering the servo motor showcases the system's effective, user-friendly design.

3.8 Notification and Communication Mechanism

Incorporation of Line Notify: Our system's integration with Line Notify provides a streamlined avenue for immediate updates and surveillance. This functionality enables users or administrators to get prompt alerts regarding the bin's conditions, like the waste level or battery status. The choice of Line Notify is attributed to its broad accessibility and compatibility with Node-RED, enhancing overall user interaction and the system's operational effectiveness.

3.9 Data Management and Display

In our setup, we utilize InfluxDB, a time-series database, for storing sensor-derived data from waste bins, covering aspects like waste amount and battery voltage. We have established a live monitoring setup using the Thingspeak application, granting stakeholders the opportunity to observe and interpret data instantaneously. This facility aids in making well-informed decisions. Thingspeak is an IoT ecosystem that facilitates the creation and control of interconnected devices. Its intuitive app builder, based on a drag-and-drop mechanism, allows for crafting personalized mobile applications to operate these devices. Moreover, Thingspeak offers a vast collection of widgets for application interface design. It is compatible with various hardware platforms, including ESP32 and Raspberry Pi, which enhances its adaptability for diverse IoT initiatives.

3.10 Machine Learning Implementation for Waste Sorting

In the domain of waste sorting, our system leverages an optimized MobileNet model, a choice driven by its optimal balance of operational efficiency and precision. MobileNet's design is streamlined, making it suitable for real-time functions on devices with limited processing capacity, such as the ESP32 cam. We refined this model on a specialized dataset to boost its proficiency in identifying different waste types, thereby increasing the dependability and customization of the classification process. For our smart bin system, we selected MobileNetV2, inspired by its architecture's efficiency, particularly in environments with constrained computational resources. MobileNetV2's compact and effective design is ideal for edge computing devices like the ESP32 cam OV2640 used in our setup.

3.10.1 Insights into MobileNetV2's Design

MobileNetV2 stands out with its unique architecture that includes inverted residuals and linear bottlenecks. This design allows MobileNetV2 to retain high accuracy levels while being substantially compact. A critical feature of its structure is the implementation of depthwise separable convolutions, drastically cutting down the model's computational demands. This reduction makes the model quicker and more efficient, especially for operations on devices like the ESP32 cam. Considering the ESP32's limited processing power and memory, MobileNetV2 emerges as the perfect fit due to its effective architectural design.

By tailoring the MobileNetV2 model to a dataset specifically designed for different waste materials, we have notably enhanced its accuracy in distinguishing various types of waste under diverse conditions. This precision is crucial for effective sorting and management within our smart bin system, directly contributing to the system's operational effectiveness and environmental friendliness.

Figure 3.5 depicts MobileNetV2's architecture, emphasizing features like inverted residuals and depthwise separable convolutions. This diagram aids in understanding why MobileNetV2's structure is so well-suited for our smart bin system's requirements, striking a balance between computational efficiency and the need for precise waste classification.

Figure 3.5

MobileNetV2 Architecture

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

3.11 Employment of the TrashNet Dataset

For the training of our model, we utilized the TrashNet dataset, a renowned repository of images of various waste materials. Comprising categories like glass, paper, metal, plastic, cardboard, and general trash, the TrashNet dataset offers a comprehensive range of materials for foundational model training. This dataset has played a significant role in propelling forward the field of waste classification research (Source: K. Yang et al., 2016, "Where Does Your Trash Go: A Journey Through the Waste Stream").

3.12 Integration of the TrashBox Dataset

In an effort to broaden and strengthen our training dataset, we incorporated the TrashBox dataset. The creation of TrashBox was driven by the necessity for more diverse and representative data in waste classification, surpassing the scope of existing datasets like TrashNet and TACO. TrashBox includes an extensive array of waste items captured under varied conditions, thereby enriching our model training regimen (Source: M. S. Islam et al., 2020, "TrashBox: A Dataset for Automatic Trash Classification").

3.13 Implementation of FOMO for Object Detection

To achieve an effective object detection mechanism suitable for low-resource devices, we adopted the Features of Mobile Objects (FOMO) model. Tailored for environments with limited resources, FOMO focuses on key features of mobile objects to facilitate real-time object detection. This method is particularly suited for IoT applications, like our smart bin system, which require prompt and accurate identification of waste items.

3.14 Leveraging Edge Impulse for Model Development

Our project leveraged Edge Impulse, a prominent platform for machine learning in edge computing environments. Edge Impulse offers a comprehensive suite for the entire cycle of machine learning model development, from creation and training to deployment, particularly for devices with constrained resources. The platform was pivotal in the efficient development and fine-tuning of our machine learning models, including MobileNetV2 and FOMO.

3.14.1 Contribution to Model Development and Refinement

Utilizing Edge Impulse enabled us to streamline the development and training of our models. The platform's extensive features facilitated smooth data import, model training, validation, and deployment processes. A notable benefit of Edge Impulse is its optimization for edge device capabilities, ensuring that our models are both lightweight and effective for real-time waste classification tasks.

3.14.2 Benefits of Deployment on Edge Devices

Edge Impulse's deployment capabilities were instrumental in transferring our trained models to the ESP32 cam OV2640. The platform's support for converting models to formats compatible with edge devices was vital, allowing for effective performance and reduced memory demands. This aspect was especially critical due to the ESP32's constrained computational capacity.

3.14.3 Enhancing System Performance and Efficiency

Edge Impulse significantly contributed to the success of our project in terms of energy efficiency, cost-effectiveness, and precise waste classification. The platform's emphasis on machine learning solutions tailored for edge computing directly enhanced the overall functionality and performance of our smart bin system.

3.15 Communication and Notification Mechanism

Integration with Line Notify: Incorporating Line Notify into our system established a robust channel for immediate alerts and ongoing monitoring. This integration enables users and administrators to promptly receive updates regarding the bin's condition, including waste level and battery status. The selection of Line Notify was based on its broad accessibility and its straightforward integration with Node-RED, which significantly improved the user experience and operational effectiveness of our system.

CHAPTER 4
EXPERIMENT

4.1 Experimental Methodology

Our experimental approach focused on determining the most viable and resource-efficient technique for incorporating a machine learning-based waste classification framework into our intelligent waste bin system. Key objectives included straightforward integration, minimal energy usage, and prompt system response. The smart bin's practical tests were carried out in the dormitory area of the Asian Institute of Technology's student campus.

4.1.1 Dataset Preparation

In the course of preparing our dataset for these trials, we leveraged two primary sources. Initially, we adopted the TrashNet dataset, which we augmented with an additional 100 waste images captured via an iPhone camera. Moreover, we integrated the TrashBox dataset, known for its extensive and detailed coverage in waste categorization. Combining these resources resulted in a total of 2,138 items from the TrashNet and custom datasets, and an impressive 14,176 items from the TrashBox and custom datasets.

The employment of these datasets played a pivotal role in the evolution of a robust machine learning model, as illustrated in Figure 4.1. By blending the varied waste images from TrashNet with real-world campus imagery and merging them with the extensive TrashBox dataset, we created a rich and diverse training landscape. This diversity was essential in elevating the precision and adaptability of our waste sorting system, equipping it to manage waste segregation with both efficiency and efficacy.

Figure 4.1

ML Architecture

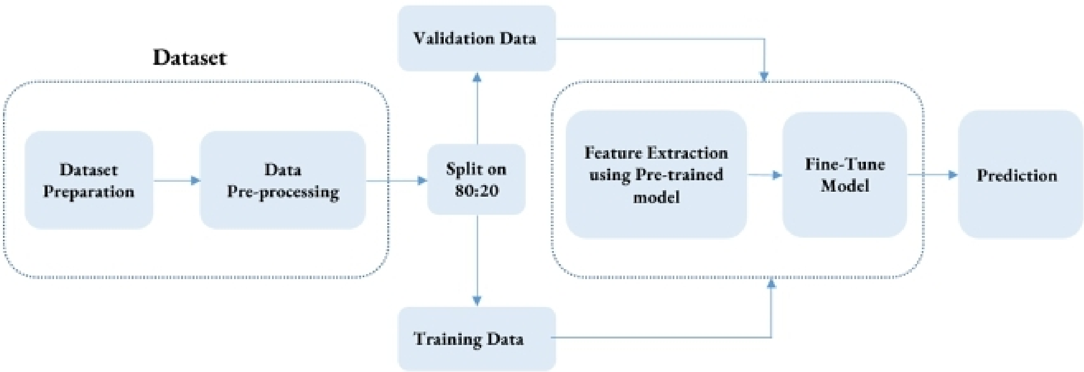


Table 4.1

Number of Images in each class of trashnet dataset for training and validation

Class	Training Data	Validation Data
Cardboard	239	82
Glass	405	102
Metal	36	92
Paper	477	123
Plastic	398	98

Table 4.2

Number of Images in each class of trashbox dataset for training and validation

Class	Training Data	Validation Data
Cardboard	1501	379
Glass	1934	366
Metal	1650	415
Paper	1731	437
Plastic	1784	425
Medical	1265	300
E-waste	1934	482

4.1.2 Enhancing the Model Using the TrashNet Dataset

In our quest to improve our model's effectiveness, we focused our training process on the TrashNet dataset, enriched with a collection of custom images. Initial outcomes were modest, but the extensive and varied composition of the TrashNet dataset offered substantial potential for model development. This dataset provided a wide array of waste material images, allowing our model to refine its pattern recognition capabilities. The diverse and comprehensive nature of the TrashNet dataset played a crucial role in enhancing the model's ability to accurately identify features, thereby increasing its precision in classifying various waste types and its adaptability to different waste materials.

Figure 4.2

Training and Validation Accuracy and Loss Over Epochs for MobileNetV2 Model on TrashNet Dataset

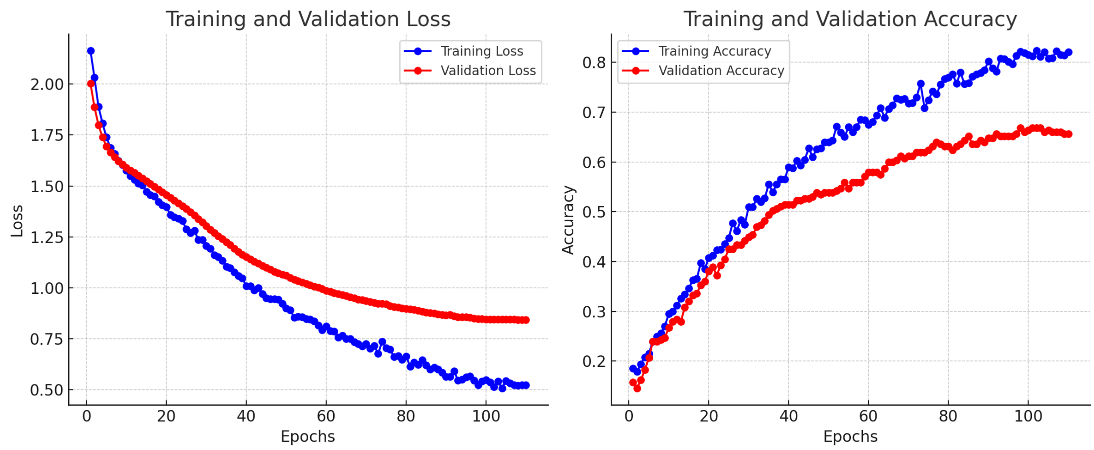
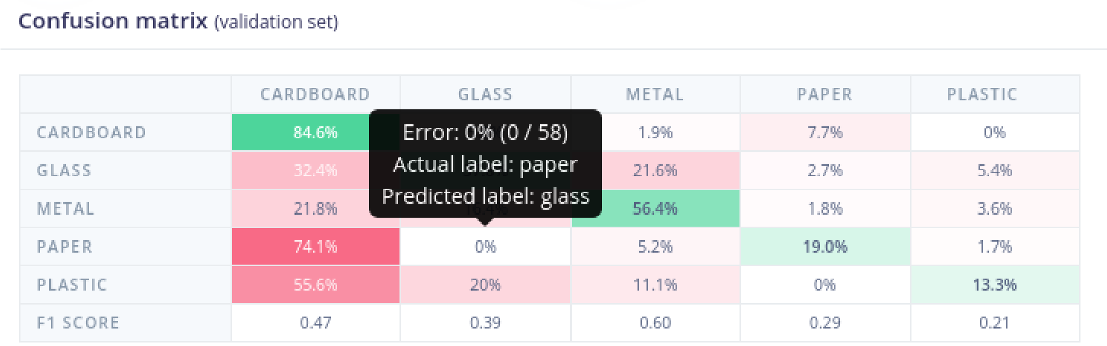


Figure 4.2 illustrates the evolution of training and validation accuracy and loss throughout the model's learning journey, which included 100 epochs of initial training followed by a fine-tuning phase. The graph indicates a steady decline in both training and validation loss, evidencing the model's progressive learning. Moreover, training accuracy consistently outperformed validation accuracy, signifying a strong learning capacity while also highlighting areas where model generalization could be further improved.

Figure 4.3

Confusion Matrix for Classification of Waste Types on the TrashNet Dataset (Validation Set)



	CARDBOARD	GLASS	METAL	PAPER	PLASTIC
CARDBOARD	84.6%	[illegible]	1.9%	7.7%	0%
GLASS	32.4%	[illegible]	21.6%	2.7%	5.4%
METAL	21.8%	[illegible]	56.4%	1.8%	3.6%
PAPER	74.1%	0%	5.2%	19.0%	1.7%
PLASTIC	55.6%	20%	11.1%	0%	13.3%
F1 SCORE	0.47	0.39	0.60	0.29	0.21

As further elucidated in Figure 4.3, the confusion matrix from the validation set provides a granular look at the model's classification acumen post-training with the TrashNet dataset. The matrix showcases high classification rates for certain categories, like 'Cardboard', yet reveals some confusion between classes such as 'Glass' and 'Paper', indicating potential avenues for refinement in future training iterations.

The utilization of the MobileNetV2 architecture, coupled with TensorFlow's computational framework, laid the groundwork for an optimized training protocol. The incorporation of pretrained weights was a strategic move to expedite the feature learning stage. The subsequent fine-tuning phase, which rendered 65 percent of the base model's layers trainable, was aimed at enhancing the model's sensitivity to the subtleties of the dataset's features.

Our training approach was further augmented by a series of data augmentation techniques, including random image flipping, resizing, cropping, and brightness variation, to ensure a robust and varied training experience. This approach, combined with the use of the Adam optimizer and categorical cross-entropy loss function, was calibrated to foster an optimal training environment, adjusting learning rates judiciously during both the initial training and fine-tuning phases.

Despite the overall satisfactory performance as depicted in Figure 4.3, the varying F1 scores across classes call attention to the need for a more balanced precision-recall equilibrium. The disparities in F1 scores suggest potential enhancements in model training, which could include a more refined data augmentation strategy, incorporation of additional training data, or a methodical reassessment of the model's structural design.

The insights drawn from the confusion matrix are invaluable, serving not just as a quantitative measure of the model's current efficacy but also as a qualitative compass for subsequent model development phases.

4.1.3 Enhanced Training Using the TrashBox Dataset

After the initial training phase with the TrashNet dataset and custom images yielded less-than-ideal results, we opted to shift our training focus to the TrashBox dataset. This decision was driven by the substantially larger and more varied collection of images in TrashBox, offering a richer array of waste material images. This enhanced dataset breadth is expected to significantly improve the model's learning efficacy by exposing it to a wider spectrum of features and variations. The goal is to achieve a more versatile model with heightened accuracy in waste classification, ultimately leading to more consistent performance in real-world waste sorting tasks.

Figure 4.4

Progression of Training and Validation Metrics for the MobileNetV2 Model on the TrashBox and Custom Dataset

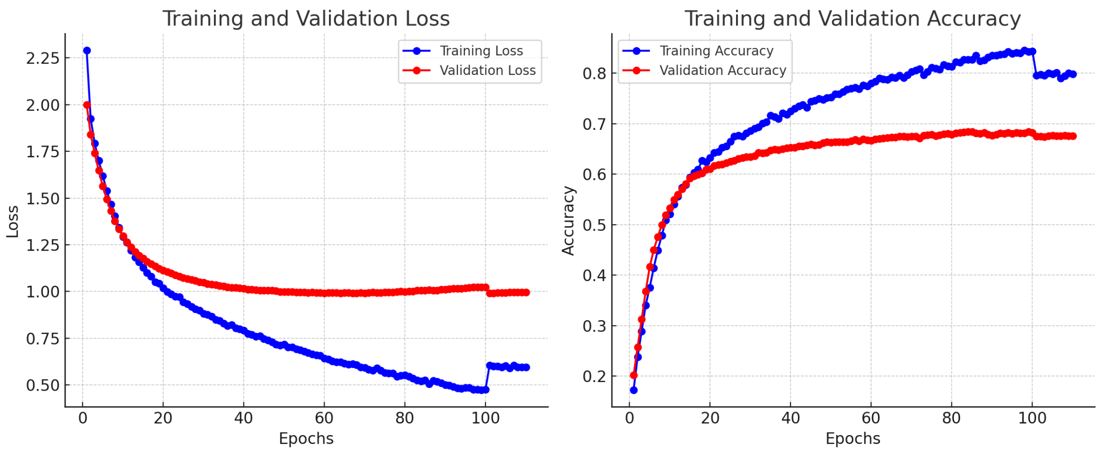


Figure 4.4 presents the training progress, showcasing the accuracy and loss during training and validation phases of our model, which combines the TrashBox dataset and custom images. We utilized TensorFlow and the MobileNetV2 model, applying transfer learning techniques with pretrained weights. The training process encompassed an initial phase of 100 epochs, followed by fine-tuning over an additional 10 epochs, adjusting about 65

Data augmentation methods, such as random flipping, resizing, cropping, and brightness alterations, were employed to enhance the model's ability to generalize. These techniques were applied to the training data, creating a more comprehensive training approach. We compiled the model using the Adam optimizer and categorical cross-entropy loss function, with a learning rate of 0.0001 initially, reduced to 0.000045 during the fine-tuning stage.

As illustrated in Figure 4.4, both training and validation loss (depicted by blue and red lines, respectively) display a decreasing trend, signifying effective model learning. Additionally, the training and validation accuracy (also in blue and red) show an upward trajectory, with training accuracy consistently higher than validation accuracy, which is typical in machine learning models.

These findings highlight the success of integrating transfer learning with data augmentation in training specialized datasets like TrashBox, further enriched with custom imagery. This combination has led to improved model performance, as evidenced by the gradual reduction in loss and increase in accuracy throughout the training epochs.

Figure 4.5

Detailed Classification Performance on the Validation Set with the TrashBox and Custom Dataset

Confusion matrix (validation set)

	CARDBOARD	E-WASTE	GLASS	MEDICAL	METAL	PAPER	PLASTIC
CARDBOARD	69.3%	6%	3.7%	3%	2%	8.7%	7.3%
E-WASTE	5.8%	73.8%	2.9%	5.2%	6.8%	4.2%	1.3%
GLASS	2.8%	7.3%	67.4%	3.5%	7.3%	3.5%	8.2%
MEDICAL	3.5%	8.6%	4.7%	61.5%	6.6%	3.5%	11.7%
METAL	2.4%	5.1%	4.2%	2.7%	69.8%	7.3%	8.5%
PAPER	8.0%	5.4%	2.5%	3.2%	7.3%	65.3%	8.3%
PLASTIC	2.4%	3.5%	7.5%	7.5%	9.4%	6.4%	63.4%
F1 SCORE	0.71	0.73	0.69	0.63	0.67	0.65	0.63

Figure 4.5 displays the confusion matrix for our validation set, providing an in-depth analysis of the model's classification effectiveness across different classes within the TrashBox and custom datasets. The matrix's diagonal elements represent the correct predictions for each class, with notable accuracy in 'E-Waste' (73.8%) and 'Glass' (67.4%), illustrating the model's reliable identification abilities.

The matrix also reveals instances of misclassification, indicating which classes the model often confuses. For example, the model frequently mistook 'Paper' for 'Cardboard' (8.7%), suggesting challenges in distinguishing between visually similar categories. The F1 scores for each class are also provided, with 'E-Waste' and 'Cardboard' demonstrating the highest scores of 0.73 and 0.71, respectively.

These outcomes affirm the model's competency in distinguishing various waste types with considerable accuracy, showing promise for real-world waste sorting applications. The confusion matrix not only underscores the model's strengths in recognizing specific waste categories but also identifies potential areas for further model enhancement or additional training data.

4.1.4 Enhanced Model Training on the TrashBox Dataset

Recognizing the need to bolster the performance of our model, we directed our training efforts toward the TrashBox dataset, which was initially complemented by a series of custom images. Although the initial results were not as promising, the comprehensive and varied nature of the TrashBox dataset presented an opportunity for the model to enhance its ability to discern intricate patterns in the waste material images. The depth and breadth of the TrashBox dataset were instrumental in advancing the model's feature recognition prowess, thereby improving classification accuracy and the ability to generalize across diverse waste types.

Figure 4.6

Training and Validation Accuracy and Loss Over Epochs for MobileNetV2 Model on TrashBox Dataset

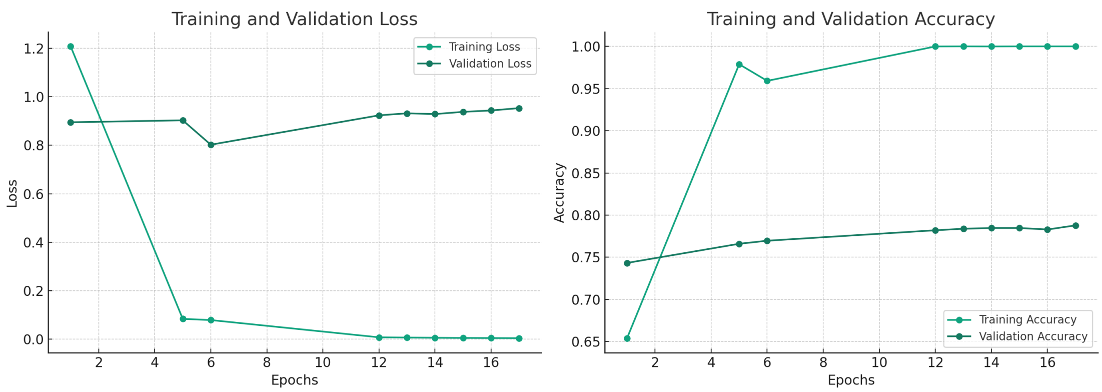
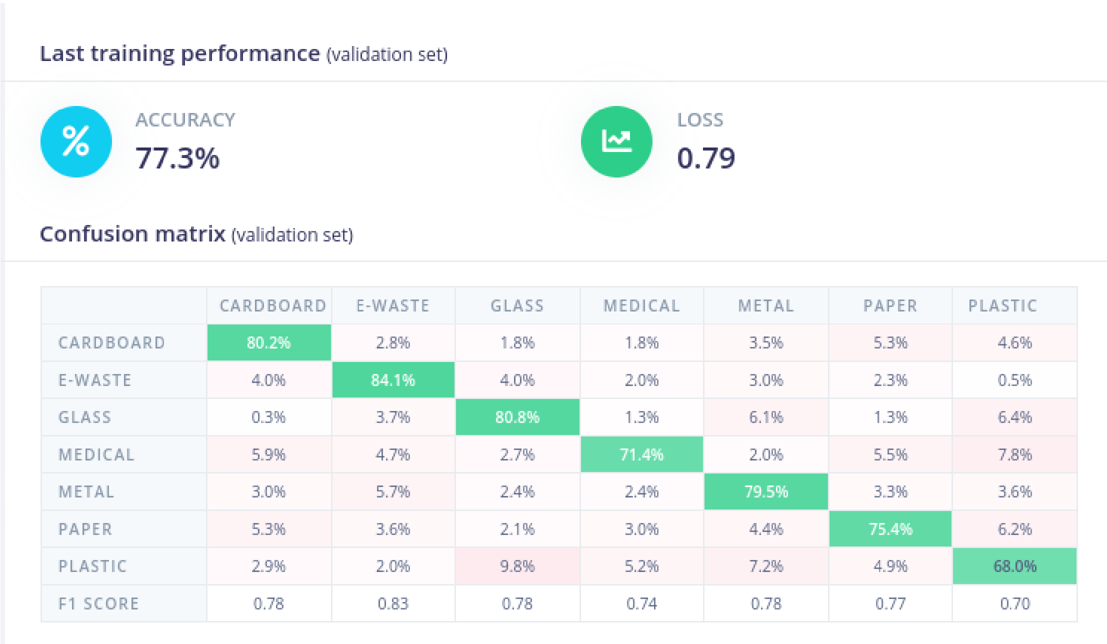


Figure 4.2 illustrates the evolution of training and validation accuracy and loss throughout the model's learning journey, which included 100 epochs of initial training followed

by a fine-tuning phase. The graph indicates a steady decline in both training and validation loss, evidencing the model's progressive learning. Moreover, training accuracy consistently outperformed validation accuracy, signifying a strong learning capacity while also highlighting areas where model generalization could be further improved.

Figure 4.7

Confusion Matrix for Classification of Waste Types on the TrashNet Dataset (Validation Set)



Last training performance (validation set)

ACCURACY 77.3%

LOSS 0.79

Confusion matrix (validation set)

	CARDBOARD	E-WASTE	GLASS	MEDICAL	METAL	PAPER	PLASTIC
CARDBOARD	80.2%	2.8%	1.8%	1.8%	3.5%	5.3%	4.6%
E-WASTE	4.0%	84.1%	4.0%	2.0%	3.0%	2.3%	0.5%
GLASS	0.3%	3.7%	80.8%	1.3%	6.1%	1.3%	6.4%
MEDICAL	5.9%	4.7%	2.7%	71.4%	2.0%	5.5%	7.8%
METAL	3.0%	5.7%	2.4%	2.4%	79.5%	3.3%	3.6%
PAPER	5.3%	3.6%	2.1%	3.0%	4.4%	75.4%	6.2%
PLASTIC	2.9%	2.0%	9.8%	5.2%	7.2%	4.9%	68.0%
F1 SCORE	0.78	0.83	0.78	0.74	0.78	0.77	0.70

As further elucidated in Figure 4.3, the confusion matrix from the validation set provides a granular look at the model's classification acumen post-training with the TrashNet dataset. The matrix showcases high classification rates for certain categories, like 'Cardboard', yet reveals some confusion between classes such as 'Glass' and 'Paper', indicating potential avenues for refinement in future training iterations.

The utilization of the MobileNetV2 architecture, coupled with TensorFlow's computational framework, laid the groundwork for an optimized training protocol. The incorporation of pretrained weights was a strategic move to expedite the feature learning stage. The subsequent fine-tuning phase, which rendered 65 percent of the base model's layers trainable, was aimed at enhancing the model's sensitivity to the subtleties of the dataset's features.

Our training approach was further augmented by a series of data augmentation techniques, including random image flipping, resizing, cropping, and brightness variation, to ensure a robust and varied training experience. This approach, combined with the use of the Adam optimizer and categorical cross-entropy loss function, was calibrated to foster an optimal training environment, adjusting learning rates judiciously during both the initial training and fine-tuning phases.

Despite the overall satisfactory performance as depicted in Figure 4.3, the varying F1 scores across classes call attention to the need for a more balanced precision-recall equilibrium. The disparities in F1 scores suggest potential enhancements in model training, which could include a more refined data augmentation strategy, incorporation of additional training data, or a methodical reassessment of the model's structural design.

The insights drawn from the confusion matrix are invaluable, serving not just as a quantitative measure of the model's current efficacy but also as a qualitative compass for subsequent model development phases.

4.1.5 Training with FOMO for Object Detection

Our initial approach using a classification model yielded promising results for images taken from a camera. However, the model's real-time classification performance was suboptimal. To address this, we transitioned to an object detection model, utilizing the TrashNet dataset supplemented with custom images from the camera, resulting in 2141 annotated items. These images were meticulously labeled to train our model with the FOMO (Few-shot Object Detection with Model distillation) algorithm, optimizing for both accuracy and speed.

The training process was executed using TensorFlow and a MobileNetV2 architecture, chosen for its balance between efficiency and accuracy. The model was trained with the following key parameters:

- Epochs: `100`
- Batch size: `128`
- Learning rate: `0.001`, with a learning rate finder to adjust if necessary
- Object weight in loss function: `100`, indicating a high importance given to the objects over the background
- Input shape: `MODEL_INPUT_SHAPE`, respecting the requirement for square input due to MobileNetV2 constraints

- Pretrained weights were used to leverage transfer learning, enhancing the initial learning curve

The model was compiled with the Adam optimizer and a custom weighted cross-entropy loss function to accommodate the class imbalance in our dataset. The training was further augmented by a set of callbacks including centroid scoring for validation data and percentage-trained logging for monitoring progress.

Figure 4.8

Training and Validation Loss, and Precision over Epochs.

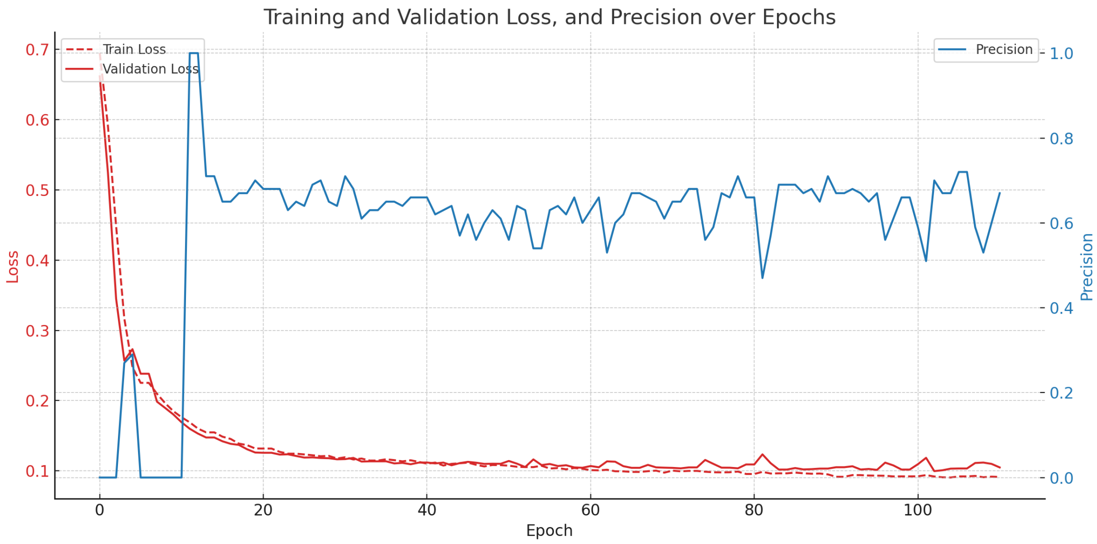


Figure 4.8 displays the evolution of training loss, validation loss, and precision over the epochs. The graph demonstrates the model's ability to consistently improve over time, reflected by the decrease in loss and increase in precision.

Figure 4.9 presents the confusion matrix from the validation set, providing insights into the model's classification performance across different categories. The matrix reveals a substantial true positive rate for various classes, with some inevitable misclassifications indicative of areas for future model improvement.

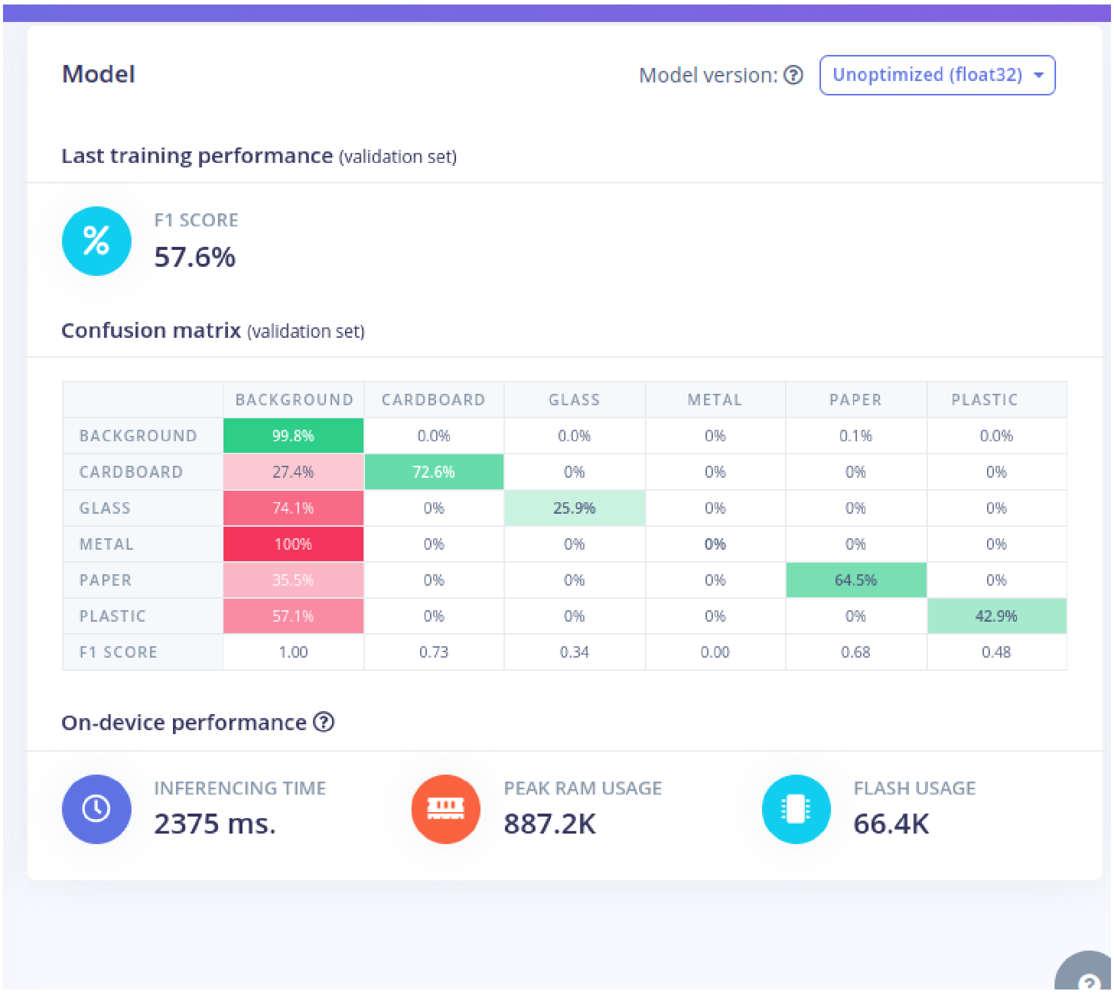
.

4.1.6 Annotation of Training Data

To ensure the object detection model recognizes and localizes objects accurately, we annotated 2141 items with bounding boxes. Captured from various angles and lighting conditions, the images reflect real-world scenarios. Annotations on the TrashNet dataset, supplemented with custom images, encompass a diverse range of waste materials. Ex-

Figure 4.9

Confusion Matrix for Object Detection Model on the Validation Set.



	BACKGROUND	CARDBOARD	GLASS	METAL	PAPER	PLASTIC
BACKGROUND	99.8%	0.0%	0.0%	0%	0.1%	0.0%
CARDBOARD	27.4%	72.6%	0%	0%	0%	0%
GLASS	74.1%	0%	25.9%	0%	0%	0%
METAL	100%	0%	0%	0%	0%	0%
PAPER	35.5%	0%	0%	0%	64.5%	0%
PLASTIC	57.1%	0%	0%	0%	0%	42.9%
F1 SCORE	1.00	0.73	0.34	0.00	0.68	0.48

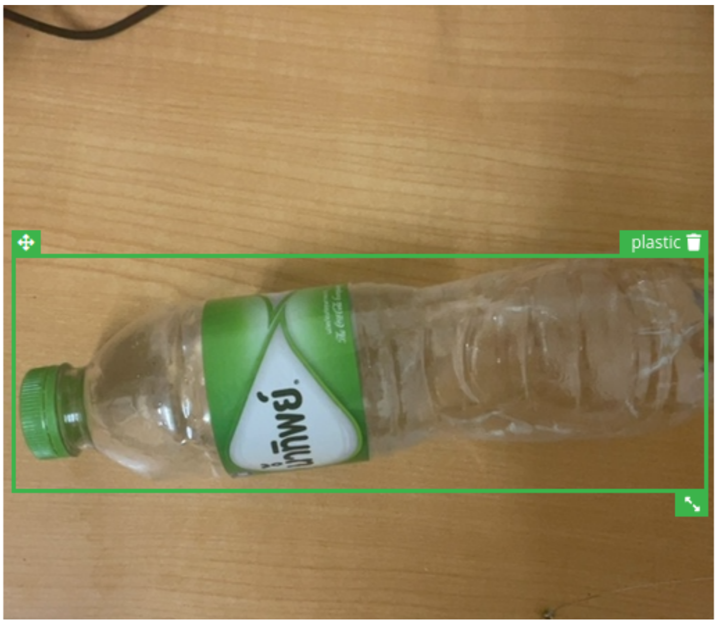
amples of annotated images used for training the object detection model are shown in Figure **??**, depicting bounding box annotations for recyclable materials—Glass (a) and Paper (b). These annotations are crucial for enhancing accurate object detection in waste sorting.

[b]0.3

[b]0.3

Figure 4.10

Examples of annotated images for object detection.



4.2 Training with yolov5

The model was trained using the Trashnet dataset combined with a custom dataset consisting of 135 images. The training process utilized the You Only Look Once, version 5 (YOLOv5) architecture. As shown in Figure 4.11, the accuracy achieved was 59.95%, with a precision score of 57.2%. The size of the trained model was approximately 3MB, indicating a lightweight model suitable for real-time applications.

Figure 4.11

Feature explorer showing the correct and incorrect object detections.

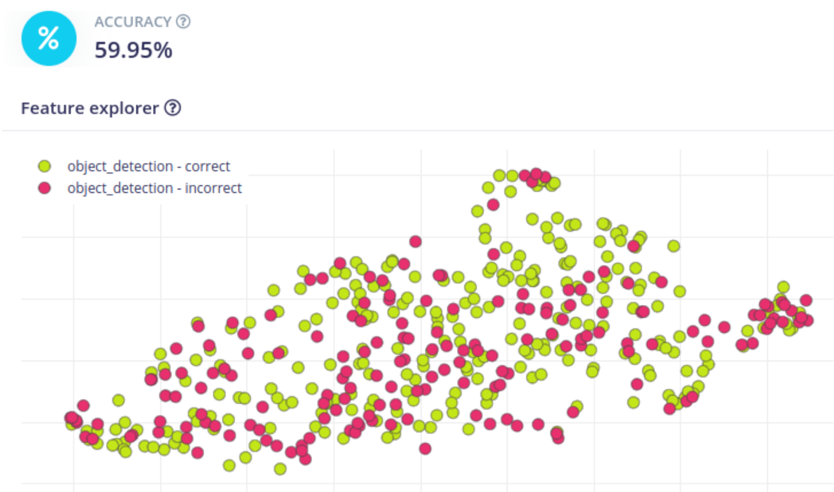


Table 4.3

Comparison of Different Training Methods

Metric	Training with TrashBox	Enhanced Training on TrashNet	Training with FOMO
Dataset Used	TrashBox	TrashNet	TrashNet + Custom
Model Architecture	MobileNetV2	MobileNetV2	MobileNetV2
Epochs	110	100	100
Data Augmentation	Yes	Yes	Yes
Optimizer	Adam	Adam	Adam
Loss Function	Categorical Cross-Entropy	Categorical Cross-Entropy	Weighted Cross-Entropy
Learning Rate	0.0001 to 0.000045	0.05	0.001
Pretrained Weights	Yes	Yes	Yes
Special Features	Fine-tuning	Pretrained weights, Fine-tuning	Learning rate finder, Callbacks
Accuracy	73.8% (E-Waste)	77.3%	59.95%
Size	1MB	3MB	987KB

In conclusion, our transition to an object detection framework and subsequent training on enhanced datasets have culminated in models that excel in both static image classification and real-time detection tasks. Table 4.3 provides a comprehensive comparison of the different training methods we employed, highlighting the unique features and performance metrics of each approach. Our methodology, particularly the use of the FOMO algorithm, has proven effective in training object detection models on specialized datasets augmented with custom imagery, as evidenced by the accuracy and model sizes detailed in the table. The comparative analysis underscores the strengths of each training method and guides the direction for future model optimizations.

4.3 Integration of Servo Motor Control with ESP32-Based Classification System

In our development process, we achieved a significant milestone by successfully integrating a servo motor control mechanism with the ESP32 module, which had been trained to discern and categorize objects into five classifications: glass, paper, plastic, organic, and e-waste. The classified signal from the ESP32 triggered specific rotational movements of the servo motor, with each class corresponding to a distinct angular segment, thereby directing the object towards its designated disposal segment. Figure 4.12 displays a subset of the classification results obtained during our testing phase. This system's ingenuity lies in its simplicity and potential for enhancing the efficiency of waste sorting in intelligent recycling systems.

Figure 4.12

Snapshot of classification results.

```
    No objects found
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    e-waste (0.667969) [ x: 32, y: 64, width: 8, height: 8 ]
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    No objects found
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    No objects found
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    e-waste (0.531250) [ x: 24, y: 64, width: 8, height: 8 ]
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    No objects found
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    No objects found
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    e-waste (0.500000) [ x: 32, y: 64, width: 8, height: 8 ]
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    e-waste (0.753906) [ x: 32, y: 64, width: 8, height: 8 ]
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    No objects found
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    e-waste (0.542969) [ x: 32, y: 64, width: 8, height: 8 ]
Predictions (DSP: 17 ms., Classification: 546 ms., Anomaly: 0 ms.):
    No objects found
```

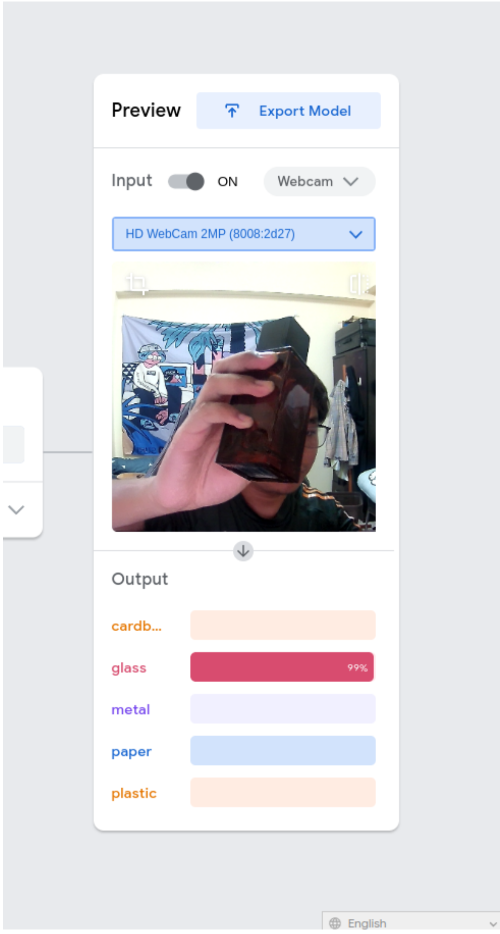
4.4 Alternative Training with Google's Teachable Machine

In addition to the methods employed through Edge Impulse, we explored Google's Teachable Machine as an alternative avenue for model training. This intuitive web-based tool allowed for the importation of the TrashNet dataset and the integration of additional photographic data captured via a webcam. The interface facilitated real-time training and provided immediate feedback on the model's performance.

The Teachable Machine's user-friendly environment is ideal for rapid prototyping and experimentation. With its streamlined process, we successfully trained a model capable of classifying various waste types. This tool offers a no-code solution for machine learning, making it accessible to individuals without a deep technical background in AI.

Figure 4.13

Real-time model training feedback on Google's Teachable Machine.



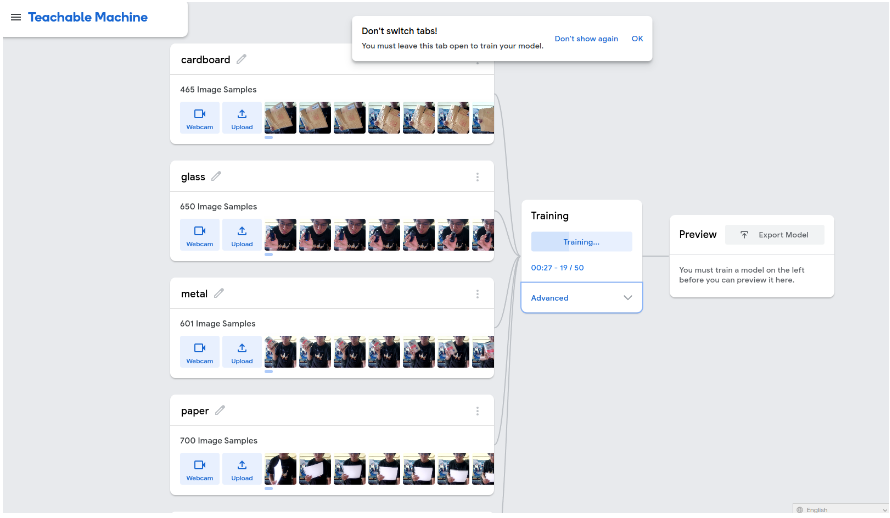
Figures 4.13 and 4.14 showcase the Teachable Machine in action. Figure 4.13 displays the real-time feedback during the model training, while Figure 4.14 gives an overview of the diverse image samples utilized for the training process.

4.4.1 Utilization of the Model from Teachable Machine

The model trained using Teachable Machine can be exported in various formats, including TensorFlow.js, which enables deployment on web applications, and TensorFlow Lite, suitable for mobile and edge devices. The versatility of the export options ensures that the model can be integrated into a wide range of applications, from interactive web interfaces to IoT devices.

Figure 4.14

Overview of image samples used for training on Google's Teachable Machine



4.4.2 Challenges with Node-RED and API Integration

We attempted to utilize the trained model with Node-RED, a flow-based development tool for visual programming, intending to implement Hypertext Transfer Protocol (HTTP) requests for real-time classification. However, challenges were encountered in setting up a reliable HTTP server to interface with the Teachable Machine model. The process required a deeper understanding of server-side programming and HTTP protocol management.

To overcome this hurdle, it would be necessary to establish a dedicated HTTP server capable of handling API requests or to utilize cloud services that provide serverless computing options. These solutions would enable the efficient use of the Teachable Machine model in a networked environment, allowing for real-time data processing and response.

4.5 Attempted Development of a Machine Learning API

Our endeavor to develop a machine learning API using Node-RED was met with significant challenges. The complexity of deploying a performant API capable of real-time image classification was compounded by TensorFlow version mismatches and the hardware constraints of the Raspberry Pi, notably the absence of GPU support.

This experience has underscored the necessity of a thorough understanding of the deployment environment, including both software compatibility and hardware capabilities.

The intricacies of integrating a high-level workflow automation tool like Node-RED with the nuanced requirements of a TensorFlow-based model illuminated the complexities of bringing a machine learning system from development to production.

The time constraints of the research project did not allow for a resolution to these challenges. However, the insights gained from these attempts are invaluable. They highlight the importance of allowing for sufficient time to troubleshoot and adapt to unexpected technical hurdles in machine learning projects.

4.6 Data Gathering

To gather data from the bins, an ESP32 microcontroller is installed on the top of each bin. The Voltage at the Common Collector (VCC) and Ground (GND) pins of the ESP32 are connected to the 5V and ground pins of the microcontroller, respectively. The ultrasonic sensor's trigPin is connected to GPIO 12, and the echoPin is connected to GPIO 14. The ultrasonic sensor measures the level of the waste in centimeters, and the bin's height is 37cm. Therefore, if the sensor data is 37cm, the bin is empty, and if the sensor value is low, it is about full. The ultrasonic sensor has a minimum sensing range of 2cm, but a minimum of 3 cm is set to avoid false readings. When the sensor value reaches 3cm, a notification is sent, indicating that the bin is full. For the battery level, a variable resistor is used to simulate the battery's voltage level, making it easier to test. The complete circuit has been built, and only the ESP32 is powered on. However, the battery level takes a long time to drain completely, so a potentiometer is used to send different voltage values to the ESP32 as a battery level indicator. Figures 4.15 and Figure 4.16 show the results in serial monitor.

Figure 4.15

Ultrasonic sensor's data in serial monitor

```
Low Battery. Need to Charge.    1.79V
Distance (cm): 10.68
Distance (inch): 4.20
1.79V
Low Battery. Need to Charge.    1.79V
Distance (cm): 10.34
Distance (inch): 4.07
1.80V
Low Battery. Need to Charge.    1.80V
Distance (cm): 10.34
Distance (inch): 4.07
1.79V
Low Battery. Need to Charge.    1.79V
```

Figure 4.16

Battery level in serial monitor

```
Low Battery. Need to Charge.    3.00V
Distance (cm): 11.37
Distance (inch): 4.48
3.30V
Battery has Full voltage.       3.30V
Distance (cm): 11.71
Distance (inch): 4.61
3.30V
Battery has Full voltage.       3.30V
Distance (cm): 11.37
Distance (inch): 4.48
3.30V
Battery has Full voltage.       3.30V
Distance (cm): 11.37
Distance (inch): 4.48
3.30V
Battery has Full voltage.       3.30V
```

4.7 Data Transmission and visualization

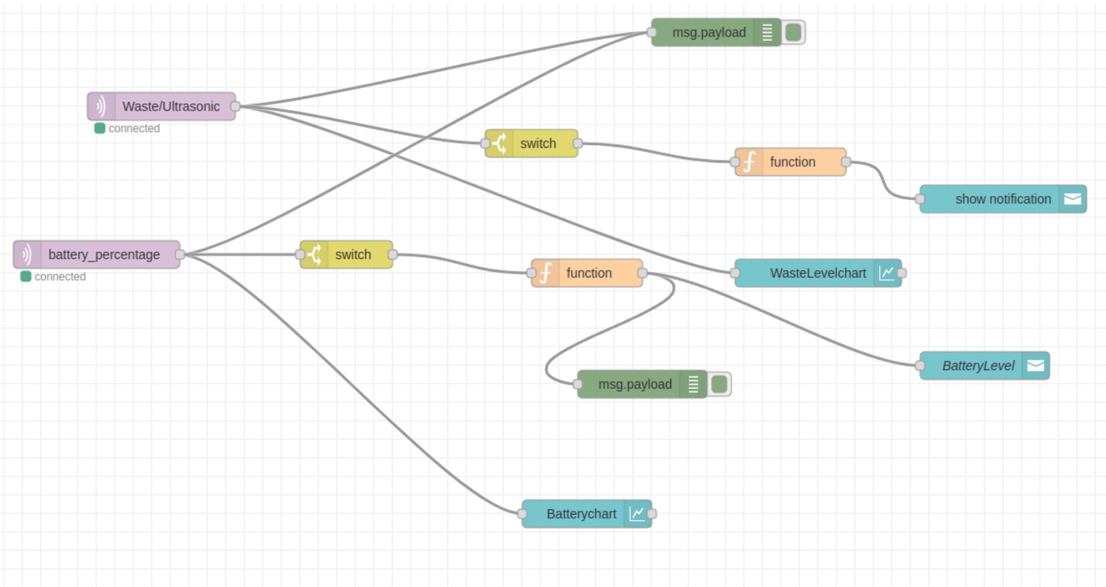
In my experiment, Two methods of data transmission from ESP32 to ThingSpeak and Node-RED were compared. However, due to difficulties in creating an InfluxDB database, the data could only be visualized on the Node-RED dashboard. Despite this, both methods had their advantages and disadvantages. The first method using ThingSpeak had the advantage of built-in graphs and charts for easy data visualization. It also provided APIs for integration with other services. However, it had a potential delay problem in getting real-time data, and firmware changes were required when adding new nodes to the system. In contrast, the second method using MQTT to transmit data to Node-RED had the benefit of being highly scalable and easy to add new nodes to the system using the topic/subscribe method. It also had flow-based programming options for creating custom logic and automating tasks.

4.7.1 Implementation with Node-Red

Figure 4.17 provides an overview of how Node-Red flows from one step to another. It starts with input nodes, which can receive data from sensors or other devices, and then passes that data to processing nodes for analysis or manipulation. Next, the data is sent to output nodes, which can send the data to other devices or systems for further processing or action.

Figure 4.17

Node-RedFlow

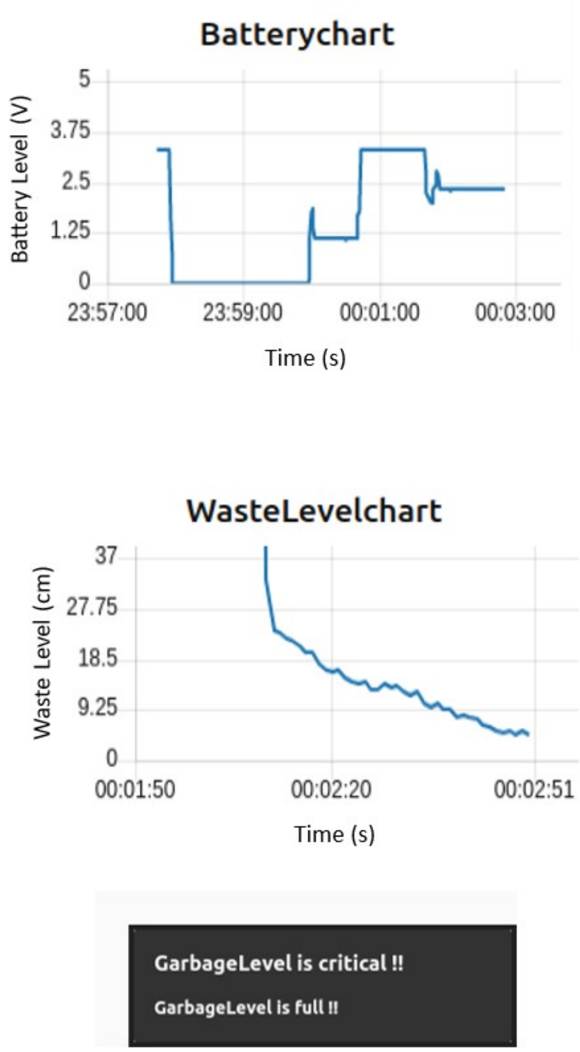


The ESP32 is sending real-time data of battery and garbage level to MQTT in Node-

RED every second. The data from the waste level MQTT node is directed to a switch node, where the value is set as greater than or equal to 3. If the value is greater than or equal to 3, the switch node sends the payload to the function node, which then forwards it to the notification node. The notification is displayed as shown in Figure 4.18.

Figure 4.18

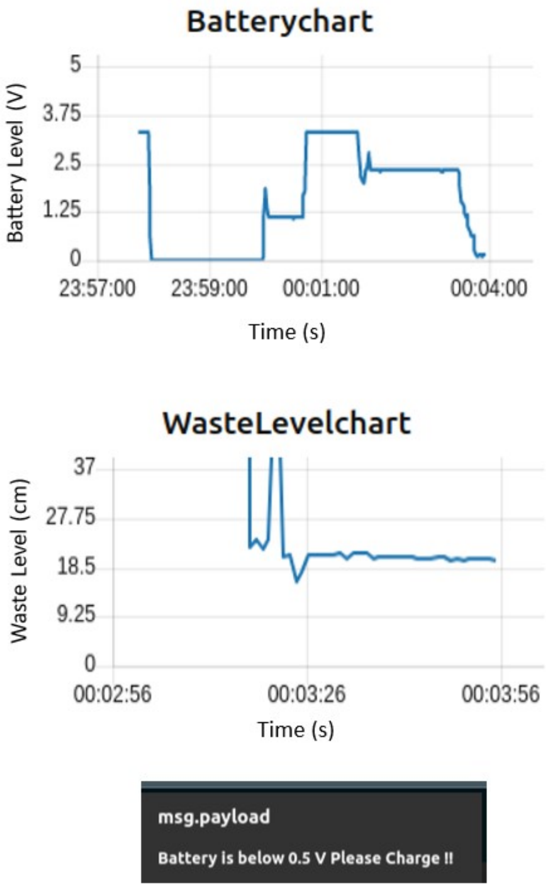
Node-Red Display showing Garbage is full



The same process is implemented for the battery level, but in the switch node, the value is set to send the payload when it is less than or equal to 0.5V. Figure 4.18 displays the notification that is being referred to.

Figure 4.19

Battery Notification



The Node-RED dashboard visually represents real-time ESP32 data, as shown in Figure 4.19. This graphical display plots waste and battery levels against time, offering a clear overview of the system's status. To send notifications via email, the team is currently encountering some challenges but is working towards resolving them.

The data is also being stored in InfluxDB for future use. The time-series data can be downloaded from InfluxDB and used for further analysis. For example, regression models can be developed to predict when the battery will need to be charged based on historical data.

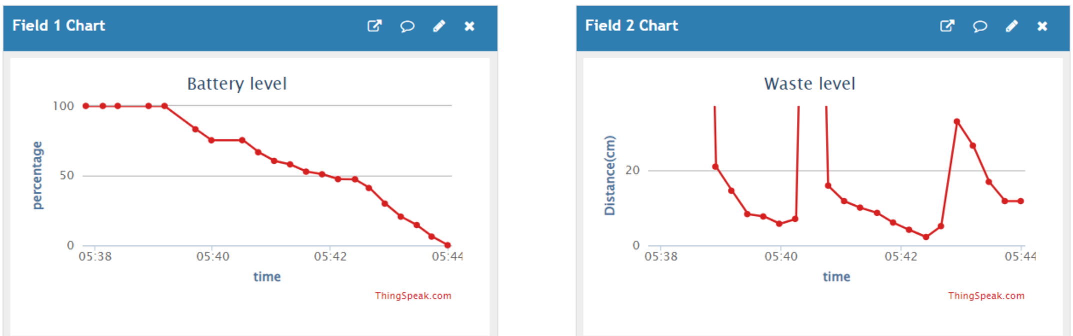
In addition, the data can be visualized on other tools, such as Grafana, to gain more insights and make informed decisions.

4.7.2 Implementation with Thingspeak

The implementation was also done using ThingSpeak. Firstly, a ThingSpeak account was created with an email and password. Then, a channel was created with two fields, one for waste level and the other for battery level (as a %). In the graph, the battery level is observed to be decreasing over time. As for the garbage level, the graph shows a spike when the bin is empty and decreases as waste is added to the bin. When it reaches it is lowest value, it means the bin is full. Once the waste is collected, it will again reach its spike. Although the data was correctly sent in time series, there was a delay of approximately 10 seconds before it appeared on the Thingspeak dashboard, as shown in Figure 4.20, which is a significant disadvantage.

Figure 4.20

Thingspeak Dashboard



4.8 Estimating Battery Life of ESP32 CAM System

Data Collection

To accurately estimate the battery life of the ESP32 CAM system, we gathered data on its power consumption during active and sleep states. The parameters measured included:

- **Active Current Draw:** The current consumed when the ESP32 CAM is fully operational, which includes the sensor reading and actuation periods.
- **Sleep Current Draw:** The current consumed when the ESP32 CAM is in deep sleep mode, significantly lower than the active state.
- **Active Duration:** The length of time the ESP32 CAM remains in an active state during a waste cycle.
- **Sleep Duration:** The length of time the ESP32 CAM stays in sleep mode between active cycles.
- **Waste Cycles Per Day:** The frequency of activation for the ESP32 CAM to manage waste-related operations each day.

Energy Consumption Calculations

Using the collected data, we applied the following equations to calculate energy consumption and estimate the battery life as

$$\text{Energy Consumed Per Cycle (mAh)} = \text{Active Current (mA)} \times \text{Active Time (h)}. \tag{4.1}$$

$$\begin{aligned}\text{Daily Energy Consumption (mAh)} = {} & \text{Energy Consumed Per Cycle (mAh)} \times \\ & \text{Number of Cycles Per Day}.\end{aligned} \tag{4.2}$$

$$\text{Battery Life (days)} = \frac{\text{Battery Capacity (mAh)}}{\text{Daily Consumption (mAh)}}. \tag{4.3}$$

For our experiments, we used a 500mAh battery to provide a consistent baseline for our estimations.

Synthetic Data Generation

To supplement our data and enhance the model's predictive accuracy, we generated synthetic data that closely mimics real-world device behavior. We utilized:

- **Poisson Distribution:** To model the average number of daily activation events.
- **Uniform Distribution:** To represent the range of possible values for active current, sleep current, and active duration.

A RandomForestRegressor was employed due to its efficacy in handling non-linear data and providing reliable predictions.

Sleep Time Calculation

Assuming a 24-hour day, we calculated the sleep time as follows:

$$\text{Sleep Time (seconds)} = 86,400 - (\text{Number of Wake Cycles} \times \text{Active Time}). \tag{4.4}$$

With a set number of wake cycles, we determined the minimum and maximum possible sleep times to better understand the system's operational efficiency.

Model Training and Evaluation

We trained the RandomForestRegressor on the collected and synthetic data, excluding the actual daily energy consumption, to predict the latter. The dataset was divided into training and test subsets, and the model's performance was evaluated using Mean Squared Error (MSE) as the metric.

Results

The model's predictions on daily energy consumption were compared against the actual measurements to validate its accuracy. The sleep time calculations also provided insights into the system's power management efficiency, essential for optimizing battery life.

Insights and Implications The experiment provided critical insights into the power management of ESP32 CAM systems and underscored the importance of efficient energy usage patterns for extending battery life in remote or portable applications.

4.8.1 Integration of Line Notify for Real-Time Alerts

A key component of our smart bin system's effectiveness is its ability to communicate in real time. To achieve this, we integrated Line Notify (Notification Service from Line Messaging Platform) into the system architecture. Line Notify is a service from the Line messaging platform, which enjoys widespread popularity for instant messaging.

Implementation Details

The integration process involved the following steps:

- Setting up Line Notify with our Node-RED server to handle outgoing notifications.
- Configuring the Node-RED flows to trigger alerts based on specific events detected by the ESP32 CAM modules.
- Ensuring secure and reliable communication between the Node-RED server and the Line messaging platform.

Advantages of Line Notify

The choice of Line Notify was informed by several benefits it offers, including:

- **Wide Accessibility:** With its large user base, Line ensures that our alert system is reachable to many users, facilitating broader adoption.
- **Instant Communication:** The immediacy of notifications via Line allows for timely actions and decisions in response to the alerts sent by our smart bin system.
- **Ease of Integration:** The simplicity of integrating Line Notify with our existing Node-RED setup allowed for a smooth implementation process without the need for extensive changes to our infrastructure.

System Alerts

Alerts were configured to notify when:

- A bin reaches its capacity threshold, signaling the need for waste collection.
- Unusual activity or system errors are detected, requiring immediate attention.

The successful deployment of Line Notify has greatly enhanced the responsiveness of our smart bin system, enabling prompt waste management and system maintenance.

4.9 Chapter Summary

We presented a comprehensive summary of our successful efforts to develop an efficient design for our system, focusing on the integration and implementation of this design. Central to our system's performance is the reliance on the accuracy and compactness of the classification model, particularly its ability to fit within the constraints of the ESP32 CAM module.

A key finding of our research is the direct correlation between the model's accuracy and the system's overall performance. By collecting a more extensive and diverse dataset of waste images, we were able to refine the classification model, enhancing its accuracy and reducing detection delays. This improvement did not necessitate additional costs; rather, it was achieved through better data collection and model optimization.

Additionally, we discuss the importance of efficient memory management in the ESP32 module. By optimizing the usage of available memory, we were able to support a more sophisticated machine learning model without requiring hardware upgrades. This approach underscores the potential of IoT technologies and machine learning not only in addressing global problems but also in enhancing system performance through software and algorithmic improvements rather than hardware expansions.

This chapter underscores the significant role of advanced machine learning techniques and strategic memory management in elevating the performance of IoT-based systems, demonstrating that impactful advancements can be achieved without incurring additional hardware costs.

CHAPTER 5
Conclusion

Our smart bin design emerges as an efficient, low-cost, and low-power solution, optimized for ease of integration into existing systems. Utilizing Node-RED and LINE Notify, our design offers scalable deployment and immediate notification capabilities, as detailed in Table 5.1. The deployment of classification models via Edge Impulse illustrates our innovative approach. This initial design sets the stage for future improvements, with a focus on enhancing classification accuracy, optimizing memory management, and increasing detection speed, contributing to the sustainable evolution of waste management systems. While the average price of smart bins with similar functions in the market is significantly higher, our smart bin design achieves a more cost-effective solution at 645 THB without compromising on functionality or performance

5.1 Components and Cost

Table 5.1

Component costs of the smart bin.

Component	Price (THB)
500mAh Battery	100.00
ESP32-CAM ov2640	250.00
MCP1700 Voltage Regulator	50.00
1μF Capacitor (each)	10.00
Ultrasonic Sensor	60.00
Servo Motor	150.00
10kΩ Resistor	5.00
Push Button	20.00
Total	645.00

REFERENCES

Aazam, M., St-Hilaire, M., Lung, C.-H., & Lambadaris, I. (2016, 10). Cloud-based smart waste management for smart cities. In (p. 188-193). doi: 10.1109/CAMAD.2016.7790356

Adedeji, O., & Wang, Z. (2019). Intelligent waste classification system using deep learning convolutional neural network. *Procedia Manufacturing*, *35*, 607-612. Retrieved from `https://www.sciencedirect.com/science/article/pii/S2351978919307231` (The 2nd International Conference on Sustainable Materials Processing and Manufacturing, SMPM 2019, 8-10 March 2019, Sun City, South Africa) doi: https://doi.org/10.1016/j.promfg.2019.05.086

Akintade, O. O., Yesufu, T. K., & Kehinde, L. O. (2019). Development of power consumption models for esp8266-enabled low-cost iot monitoring nodes. *Advances in Internet of Things*.

Atayero, P. A., Williams, R., Badejo, J., & Popoola, S. (2019, 02). Cloud based iot-enabled solid waste monitoring system for smart and connected communities. *International Journal of Civil Engineering and Technology*, *10*, 2308-2315.

Bharadwaj, A. S., Rego, R., & Chowdhury, A. (2016). Iot based solid waste management system: A conceptual approach with an architectural solution as a smart city application. In *2016 ieee annual india conference (indicon)* (p. 1-6). doi: 10.1109/INDICON.2016.7839147

Catarinucci, L., Colella, R., Consalvo, S. I., Patrono, L., Rollo, C., & Sergi, I. (2020). Iot-aware waste management system based on cloud services and ultra-low-power rfid sensor-tags. *IEEE Sensors Journal*, *20*(24), 14873-14881. doi: 10.1109/JSEN.2020.3010675

Cerchecci, M., Luti, F., Mecocci, A., Parrino, S., Peruzzi, G., & Pozzebon, A. (2018, 04). A low power iot sensor node architecture for waste management within smart cities context. *Sensors*, *18*, 1282. doi: 10.3390/s18041282

Chung, C.-Y., Peng, I.-T., & Yeh, J.-C. (2020). Environmental monitoring and smart garbage sorting system based on lora wireless transmission technology. In *2020 ieee 2nd eurasia conference on biomedical engineering, healthcare and sustainability (ecbios)* (p. 43-46). doi: 10.1109/ECBIOS50299.2020.9203665

Chéour, R., Khriji, S., abid, M., & Kanoun, O. (2020). Microcontrollers for iot: Optimizations, computing paradigms, and future directions. In *2020 ieee 6th world* *forum on internet of things (wf-iot)* (p. 1-7). doi: 10.1109/WF-IoT48130.2020.9221219

Faris, S. (2018). *Wireless controllable robotic arm.* [Online image]. Retrieved from https://fabacademy.org/2018/labs/fablabkochi/students/salman-faris/finalproject.html (Accessed on April 5, 2023.)

Hulyalkar, S., Deshpande, R., Makode, K., & Kajale, S. (2018). Implementation of smartbin using convolutional neural networks..

Incrocci, L., Marzialetti, P., Incrocci, G., Balendonck, J., Spagnol, S., & Pardossi, A. (2010). Application of wet sensor for management of reclaimed wastewater irrigation in container-grown ornamentals (prunus laurocerasus l.)..

Khan, M. N., & Naseer, F. (2020). Iot based university garbage monitoring system for healthy environment for students. In *2020 ieee 14th international conference on semantic computing (icsc)* (p. 354-358). doi: 10.1109/ICSC.2020.00071

Kumar, A., Prandi, R., Rathore, N., & Lloret, J. (2017). A survey of the state-of-the-art in smart waste management. *The IoT Solutions for Waste Management*, *23*, 1-10. doi: 10.1016/j.iot.2017.01.004

Maksimovic, M., & Natalizio, E. (2017). Smart waste management solutions: Examples and open issues. In *2017 international conference on engineering, technology and innovation (ice/itmc)* (p. 1047-1053). doi: 10.1109/ICE.2017.8279952

Medehal, A., Annaluru, A., Bandyopadhyay, S., & Chandar, T. (2020). Automated smart garbage monitoring system with optimal route generation for collection. In *2020 ieee international smart cities conference (isc2)* (p. 1-7). doi: 10.1109/ISC251055.2020.9239002

NagarajD, C., Badami, S. I., & Santosh, H. (2017). Automated waste segregator using renesas microcontroller..

Nagesh, U. B., Kotari, M., & Chethan, S. C. (2021). Integration of mqtt protocol with map apis for smart garbage management. In *2021 ieee international conference on distributed computing, vlsi, electrical circuits and robotics (discover)* (p. 145-150). doi: 10.1109/DISCOVER52564.2021.9663654

Pereira, W., Parulekar, S., Phaltankar, S., & Kamble, V. (2019). Smart bin (waste segregation and optimisation). In *2019 amity international conference on artificial intelligence (aicai)* (p. 274-279). doi: 10.1109/AICAI.2019.8701350

Rajesh, K., Rohini, B., Agalya, R., Janani, S., Rajendran, S., & Ramkumar, A. (2021).

Intelligent garbage monitoring system using iot. In *2021 second international conference on electronics and sustainable communication systems (icesc)* (p. 717-722). doi: 10.1109/ICESC51422.2021.9532937

Rashmi, G., Ameenulla, M., & Kumar, S. S. (2019). Cloud based architecture for solid waste garbage monitoring and processing. In *2019 4th international conference on computational systems and information technology for sustainable solution (csitss)* (p. 1-4). doi: 10.1109/CSITSS47250.2019.9031040

Saminathan, T., & Musipatla, A. (2019). Iot based automated waste segregator for efficient recycling..

Samion, N., Kaamin, M., Adan, N., Nadiah, A., Kadir, A., Tun, U., … Pahat, B. (2018, 06). The effectiveness of segregation recyclable materials by automated motorized bin. *Journal of Advanced Manufacturing Technology*, *12*.

Savla, D. V., Parab, A. N., Kekre, K. Y., Gala, J. P., & Narvekar, M. (2020). Iot and ml based smart system for efficient garbage monitoring: Real time aqi monitoring and fire detection for dump yards and garbage management system. In *2020 third international conference on smart systems and inventive technology (icssit)* (p. 315-321). doi: 10.1109/ICSSIT48917.2020.9214202

Sharmin, S., & Al-Amin, S. T. (2016, 11). A cloud-based dynamic waste management system for smart cities. In (p. 1-4). doi: 10.1145/3001913.3006629

S.Vigneshwaran, M. M. M. V., Karthikeyan N. (2019, march). A smart dustbin using lora technology. *Conference World*.

Timescale. (2019, March 28). *Time-series data: Why and how to use a relational database (instead of nosql).* Blog post. Retrieved from `https://www.timescale.com/blog/time-series-data-why-and-how-to-use-a-relational-database-instead-of-nosql-d0cd6975e87c/` (Accessed on April 5, 2023.)

Vijayalakshmi, S., & Uthariaraj, R. (2018). Iot-based smart waste management system for smart cities. *Journal of Advanced Research in Dynamical and Control Systems*, *10*, 699-705.

Ziouzios, D., & Dasygenis, M. (2019). A smart recycling bin for waste classification. In *2019 panhellenic conference on electronics telecommunications (pacet)* (p. 1-4). doi: 10.1109/PACET48583.2019.8956270