

WASTE MANAGEMENT DETECTION

Minor project report submitted in partial fulfillment of the requirement for the degree of
Bachelor of Technology

in

Computer Science and Engineering

By

Aanchal Aggarwal (211254)

Khushi Bhati (211276)

Aditya Gupta (211230)

UNDER THE SUPERVISION OF

Mr. PRATEEK THAKRAL



Department of Computer Science & Engineering and Information Technology

**Jaypee University of Information Technology, Waknaghat,
173234, Himachal Pradesh, INDIA**

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 18/05/24

Type of Document (Tick): ☒ B.Tech./B.Sc./BBA/Other

Name: Aanchal Aggarwal, Aditya Gupta, Kushi Bhat Department: CSE Enrolment No 211254, 211230, 211274

Contact No. 7404990746 E-mail. 211254@jitsolan.in

Name of the Supervisor: Mr. Prateek Thakral

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): Waste

Management Detection

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

- Total No. of Pages = 35
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references = 2


(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found Similarity Index at 10 (%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.


(Signature of Guide/Supervisor)

TABLE OF CONTENT

Title	Page No.
Certificate	II
Acknowledgement	III
Abstract	IV
Chapter-1 (Introduction)	1-4
Chapter-2 (Feasibility Study, Requirements Analysis and Design)	5-11
Chapter-3 (Implementation)	12-26
Chapter-4 (Results)	27-29
References	30-31

CERTIFICATE

We hereby certify that the work which is being presented in the project report titled "WASTE MANAGEMENT DETECTION" in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering and submitted to the Department of Computer Science And Engineering, Jaypee University of Information Technology, Wagnaghat is an authentic record of our own work carried out during the period from January 2024 to May 2024 under the supervision of Mr.Prateek Thakral , Department of Computer Science and Engineering, Jaypee University of Information Technology, Wagnaghat.

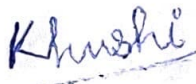
The matter presented in this project report has not been submitted for the award of any other degree of this or any other university.

Aanchal Aggarwal



(211254)

Khushi Bhati




(211276)

Aditya Gupta



(211230)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.



Mr. Prateek Thakral

Assistant Professor(Grade – II)

Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Wagnaghat,

ACKNOWLEDGEMENT

Firstly, we express our heartiest thanks and gratefulness to almighty God for his divine blessing to make it possible to complete the project work successfully.

We are really grateful and wish our profound indebtedness to Supervisor **Mr. Prateek Thakral, Assistant Professor(Grade – II)**, Department of CSE Jaypee University of Information Technology, Wakhnaghat. Deep Knowledge & keen interest of our supervisor in the field of “**Research Area**” motivated us to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

We would like to express our heartiest gratitude to **Mr. Prateek Thakral**, Department of CSE, for his kind help to finish our project.

We would also generously welcome each one of those individuals who have helped us straightforwardly or in a roundabout way in making this project a win. In this unique situation, we might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

Aanchal Aggarwal
(211254)



Khushi Bhati
(211276)



Aditya Gupta
(211230)



ABSTRACT

The short path method, such as Prim's or Kruskal's, can be used in waste management detection to improve the accuracy of trash segregation based on organic and inorganic materials, supporting effective recycling procedures. Our project's main goal is to reduce the amount of rubbish that spreads throughout open spaces and other regions. These algorithms can be used to identify the best routes for waste collection and segregation, reducing operating expenses and trip distances. This method lessens the ecological impact of garbage accumulation, which not only streamlines the waste management process but also promotes environmental conservation. A sustainable and efficient waste management system is the project's goal to reduce negative environmental effects. Effective waste management is essential to the environment's sustainability, and automation can significantly boost output in this field. The purpose of this work is to examine Convolutional neural networks for trash detection and classification (CNNs) and a pre-trained ResNet50V2 model.

A dataset containing a variety of trash types was used to produce training, validation, and test sets. Preprocessing the data included picture augmentation, normalization, and scaling to improve model performance. When trained and evaluated on this dataset, the proprietary CNN model and the ResNet50V2 model both demonstrated excellent potential in accurately categorizing waste categories. With the custom CNN model achieved excellent accuracy on both training and validation sets. Among other performance factors, confusion matrices and categorization reports showed effectiveness in locating garbage.

Various Deep-learning deep learning architectures have been created with the express purpose of enhancing the precision and effectiveness of medical image segmentation. Because of its U-shaped architecture, U-Net can capture context and enable precise localization with less input while still performing well. U-Net++ improves this further by adding nested and dense skip connections for better feature propagation and accuracy. By increasing contrast in small places, the pre-processing technique known as CLAHE (Contrast Limited Adaptive Histogram Equalisation) makes medical images easier to see. With the use of atrous convolutions and atrous DeepLab v3 uses spatial pyramid pooling (ASPP) to acquire multi-scale information. is a great tool for complex image segmentation. This is extended in DeepLab v3+ with the addition of a decoder module for finer segmentation, especially close to boundaries. Combining these methods provides reliable solutions for segmenting medical images.

Chapter 01 : INTRODUCTION

1.1 Introduction

Waste management is a continuous concern of environmental sustainability because of the labor-intensive and complex process of separating waste. Our work addresses this by improving the accuracy of waste classification using deep learning approaches, namely Convolutional Neural Networks (CNNs) and the pre-trained ResNet50V2 model. The processing and analysis of grid-like data, such as photographs, by CNNs has transformed jobs like image identification and categorization. Our goal is to reduce human error and increase recycling efficiency by automating the classification and identification of trash categories from photos using CNNs. For this purpose, the ResNet50V2 model is especially useful. It is a deep network variant that is intended to reduce the vanishing gradient issue by using identity shortcut connections. Pre-trained on large datasets such as ImageNet, ResNet50V2 provides strong transfer learning capabilities that enhance image recognition.

Apart from these models, sophisticated image processing and segmentation methods like U-Net and U-Net++ are utilized to enhance the classification procedure. With its unique U-shaped architecture, U-Net is well suited for accurate waste type identification and performs exceptionally well in biological picture segmentation. With nested and dense skip connections, U-Net++ improves this capacity even further, offering better segmentation results and feature representation. Contrast Limited Adaptive Histogram Equalization (CLAHE) is applied to improve image contrast to improve feature extraction and classification.

Furthermore, advanced segmentation models like DeepLabV3 and DeepLabV3+ contribute to fine-tuning the segmentation process. These models, with their ability to capture multi-scale contextual information through various convolutions, provide high-resolution segmentation results. We aim to reduce contamination in recycling streams and promote environmental sustainability by integrating these sophisticated deep-learning models and image-processing techniques.

1.2 Objective

Creating a garbage classification system based on deep learning is the main goal of this research.

- ❖ Make use of CNNs or convolutional neural networks.
- ❖ Using image data, accurately identify and categorize several trash categories (plastics, metals, paper, and organic materials).
- ❖ Reduce the intensity of labor.
- ❖ Minimizing human mistakes.
- ❖ Boost the effectiveness and efficiency of recycling initiatives.
- ❖ To guarantee model durability and dependability, use a substantial dataset of unusable photographs for training and assessment.
- ❖ To promote widespread adoption and use, design a user-friendly interface that is easy to integrate into existing waste management systems.
- ❖ Increase the precision of waste sorting.
- ❖ Recycled stream pollution is lower.
- ❖ Encourage the efficient use of resources.
- ❖ Decrease the amount of waste dumped in landfills.
- ❖ Raise the rate of recycling.
- ❖ Support sustainable environmental practices.

1.3 Motivation

- ❖ **Environmental Sustainability:** Preserving our world requires cutting back on landfill waste and raising recycling rates.
- ❖ **Efficiency Gains:** By increasing processing speed and accuracy and lessening the workload for human workers, automated garbage sorting can increase efficiency.
- ❖ **Economic Benefits:** Recycling processes can become more profitable and operating costs can be reduced with effective waste management.
- ❖ **Resource Conservation:** Better recycling results from properly sorted garbage help to preserve natural resources.

By using deep learning, Artificial Intelligence (AI) can be used to address practical environmental issues.

1.4 Language Used

GOOGLE COLLABORATORY , PYTHON , PYTHON 3 , TENSORFLOW , PYTORCH , NUMPY , PANDAS , STREAMLIT , PICKLE , etc.

For this research, Python was utilized, and the TensorFlow and Keras frameworks were used to design and train the neural networks.

Results for Python 3.6+ TensorFlow 2. x Keras OpenCV NumPy Scikit-learn.

Both a customized CNN model and an improved ResNet50V2 model were effectively produced by the project. In waste classification tasks, both models showed good accuracy; however, the ResNet50V2 model performed better. These findings suggest that it is feasible to include these AI models in automated waste sorting systems, which might completely transform the way waste is managed. In the future, additional datasets, improved models, and practical implementation will be the key areas for improvement.

1.5 Technical Requirements (Hardware)

GOOGLE COLLABORATORY , STREAMLIT, PICKLE , AMD Ryzen 3 3250U with Radeon Graphics 2.60 GHz , 8.00 GB (5.94 GB usable) RAM , 64-bit operating system , x64-based processor .

1.6 Deliverables/Outcomes

Both a customized CNN model and an improved ResNet50V2 model were effectively produced by the project. In waste classification tasks, both models showed good accuracy; however, the ResNet50V2 model performed better. These findings suggest that AI models could be incorporated into automated waste sorting systems, which would completely transform the waste management process. Additional datasets, improved models, and practical implementation will be the main areas of future improvement.

Chapter 02:Feasibility Study, Requirements Analysis and Design

2.1 Feasibility Study

S.No.	Title/Author	Year	Objective	Work done	Research Gap
[1]	Deep learning-based waste detection in natural and urban environments by Sylvia Majchrowska, Agnieszka Mikołajczyk, Maria Farlin, Zuzanna Klawikowska, Marta A. Plantykov, Arkadiusz Kwasigroch.	2022	The paper advocates for the pivotal role of artificial intelligence, specifically deep learning, in advancing waste management practices	It highlights a benchmark model achieving up to 75% accuracy in classifying household litter across diverse environments. The research underscores DL's potential for automating waste sorting in households and industrial settings.	Future directions aim to enhance DL models for improved performance, particularly in recognizing smaller litter items and optimizing classification accuracy and speed.
[2]	A Deep Learning-Based Intelligent Garbage Detection System Using an Unmanned Aerial Vehicle by Vishal Verma, Deepali Gupta, Shefali Gupta, Mudita Gupta	2022	This paper advocates for adopting deep convolutional neural networks (CNNs) in solid waste detection, showcasing superior accuracy with CNN1	94% accuracy over CNN2. The study highlights the efficacy of the Adam optimizer compared to RMSprop for optimizing CNN performance in waste management tasks	Emphasizing cost and time savings for municipal corporations, this research underscores the transformative potential of CNN technology in enhancing waste management efficiency and environmental monitoring practices.

[3]	A real-time rural domestic garbage detection algorithm with an improved YOLOv5s network model by Xiangkui Jiang, Rui Ding	2022	This paper advocates for leveraging deep learning techniques to enhance garbage classification, emphasizing the need for improved detection algorithms and network models.	The proposed YOLOv5s-CSS model demonstrates superior speed and accuracy in real-time domestic garbage detection, showcasing the effectiveness of the approach	Despite acknowledged limitations with small or unbalanced datasets and evolving garbage types, the study highlights future directions toward embedding .
[4]	Garbage Classification and Detection for Urban Management by Kishan PS, Jitendra Jaiswal	2021	This paper proposed leveraging CNN for garbage classification to enable robotic intervention in waste management, achieving an accuracy	Achieving an accuracy of 85.52%. It also explored Faster SVM, but CNN outperformed with 89.7% accuracy on their dataset. The future scope includes integrating IoT for garbage detection and environmental monitoring advancement.	challenges in regions with limited waste infrastructure, emphasizing the significance of innovative solutions like the proposed IoT-based system for promoting environmentally friendly operations.

[5]	<p>A Garbage Detection and Classification Method Based on Visual Scene Understanding the Home Environment by Yuezhong Wu, Xuehao Shen, Qiang Liu</p> <hr/>	2021	<p>This paper advocates for a novel approach to intelligent garbage detection and classification on edge devices through visual scene understanding. Unlike traditional methods that rely on perceptual detection, assuming items as garbage by default.</p>	<p>this method leverages knowledge graphs and visual algorithms for intelligent decision-making in scene analysis.</p>	<p>The proposed system highlights future research directions, emphasizing the need for deeper exploration into item attributes and expanding modalities like voice interaction to enhance edge device intelligence and usability</p>
[6]	<p>IoT-Based Waste Management System Through Cloud Computing and WSN.” by M. Humera Khanam, V. Yamini, C. Sucharitha, Samia Anjum, and Y. C. Thejaswini</p>	2020	<p>This paper advocates for advanced technologies in urban solid waste management to address environmental concerns. It introduces a three-layered IoT architecture for real-time waste monitoring, aiming to boost efficiency and sustainability</p>	<p>It also highlights challenges in regions with limited waste infrastructure, emphasizing the significance of innovative solutions</p>	<p>The proposed an IoT-based system for promoting environmentally friendly operations.</p>
[7]	<p>solid waste monitoring system for smart and connected communities” by Aderemi A. Atayero, Rotimi Williams</p>	2019	<p>The paper highlights the importance of timely waste information for urban management.</p>	<p>Atayero et al. (2019) developed an IoT-enabled waste monitoring system, employing ultrasonic sensors and cloud computing for real-time data transmission</p>	<p>Previous studies emphasize the necessity of monitoring for efficient waste collection and recycling.</p>

[8]	E-waste management using hybrid optimization-enabled deep learning in IoT-cloud platform” by Puppala Ramya a,Ramya V, Babu Rao M	2019	This paper examines various e-waste management approaches, including using deep learning models like TensorFlow and YOLOv3 algorithms to minimize operational costs and enhance waste detection	Techniques by Hussain et al. and Sami et al. Transfer learning, as proposed by Baker et al.,	shows promise in mitigating overfitting but may come with high costs.
[9]	solid waste monitoring system for smart and connected communities” by Aderemi A. Atayero,Rotimi Williams	2019	The paper highlights the importance of timely waste information for urban management. Atayero et al. (2019) developed an IoT-enabled waste monitoring system, employing ultrasonic sensors and cloud computing for real-time data transmission.	Previous studies emphasize the necessity of monitoring for efficient waste collection and recycling. Challenges include high costs and limited access to information.	Atayero et al.'s approach offers practical implementation and scalability for addressing urban waste management.
[10]	E-waste management using hybrid optimization-enabled deep learning in IoT-cloud platform” by Puppala Ramya a,Ramya V, Babu Rao M	2019	This paper examines various e-waste management approaches, including using deep learning models like TensorFlow and YOLOv3 algorithms	To minimize operational costs and enhance waste detection. Techniques by Hussain et al. and Sami et al	Transfer learning, as proposed by Baker et al., shows promise in mitigating overfitting but may come with high costs.

2.1.1 Problem Definition

The limitations and inefficiencies of conventional waste sorting techniques are the main issues this initiative attempts to solve. Manual sorting methods are time-consuming, prone to mistakes, and frequently inconsistent, which results in poor recycling results and environmental damage. Furthermore, these difficulties are made worse by the volume and complexity of garbage that is growing, calling for creative ways to improve waste management procedures.

2.1.2 Problem Analysis

- ❖ Labor Intensity: Manual waste sorting costs a lot of money since it takes a lot of labor and time.
- ❖ Ineffective Use of Resources: Waste that is not adequately sorted contaminates recycling streams, which lowers the effectiveness of recycling procedures.
- ❖ Environmental Impact: Poor waste management poses serious environmental risks by causing pollution, habitat destruction, and resource depletion.

2.1.3 Solution

Modern deep learning techniques, in particular Convolutional Neural Networks (CNNs), are used in the proposed method to automate waste classification in order to address the challenges that have been identified. A CNN model is trained on a variety of trash picture datasets to enable the system to precisely recognize and classify various waste categories, such as plastics, metals, paper, and organic components.

The solution's process entails the following crucial steps:

Gathering and preparing data: A large dataset of rubbish photos is gathered and annotated with the relevant waste categories. To improve model generalization, data preprocessing methods like resizing, normalization, and augmentation are used.

Model Development: TensorFlow and Keras libraries are used to develop and implement a customized CNN architecture. To enhance performance, a pre-trained ResNet50V2 model is adjusted.

Education and Modern deep learning techniques, in particular Convolutional Neural Networks (CNNs), are used in the proposed method to automate waste classification in order to address the challenges that have been identified. A CNN model is trained on a variety of trash picture datasets to enable the system to precisely recognize and classify various waste categories, such as plastics, metals, paper, and organic components.

2.2 Requirements

2.2.1 Functional Requirements

- ❖ **Image Classification:** Waste photos should be able to be categorized by the system into pre-established groups including paper, plastics, metals, and organic materials.
- ❖ **User Interface:** By providing photos for categorization and accessing the classification results via an easy-to-use interface, users need to be able to interact with the system.
- ❖ **Model Training:** Using labeled garbage picture datasets, the system ought to enable CNN models to be trained.
- ❖ **Model Evaluation:** Using validation datasets, the system should analyze the performance of taught models to determine their correctness and dependability after training.
- ❖ **Model Deployment:** To enable real-time waste classification, deploy the trained model into a production environment.
- ❖ **Feedback Mechanism:** Include a feedback mechanism that enables users to comment on classification outcomes, thereby increasing the accuracy of the model.
- ❖ **Ability to manage huge garbage image databases efficiently,** including the ability to organize them.

2.2.2 Non-Functional Requirements

- ❖ **Accuracy:** To guarantee dependable sorting results, the system must have high accuracy in trash classification.
- ❖ **Scalability:** The system must be scalable to manage substantial waste image quantities and future user and data expansion.
- ❖ **Performance:** To give users their results on time, the categorization process needs to be effective and low-latency.

- ❖ **Protection:** Make sure that privacy and data protection safeguards are in place to safeguard sensitive information, including user credentials and trash image data.
- ❖ **Usability:** The user interface should be easy to use and straightforward so that new users may get started with little to no training.
- ❖ **Reliability:** To minimize downtime and guarantee continuous availability, the system must function dependably under a variety of circumstances.

To enable future improvements, the system should have a modular architecture, comprehensive documentation, and be easy to maintain.

2.3 E-R Diagram / Data-Flow Diagram (DFD)

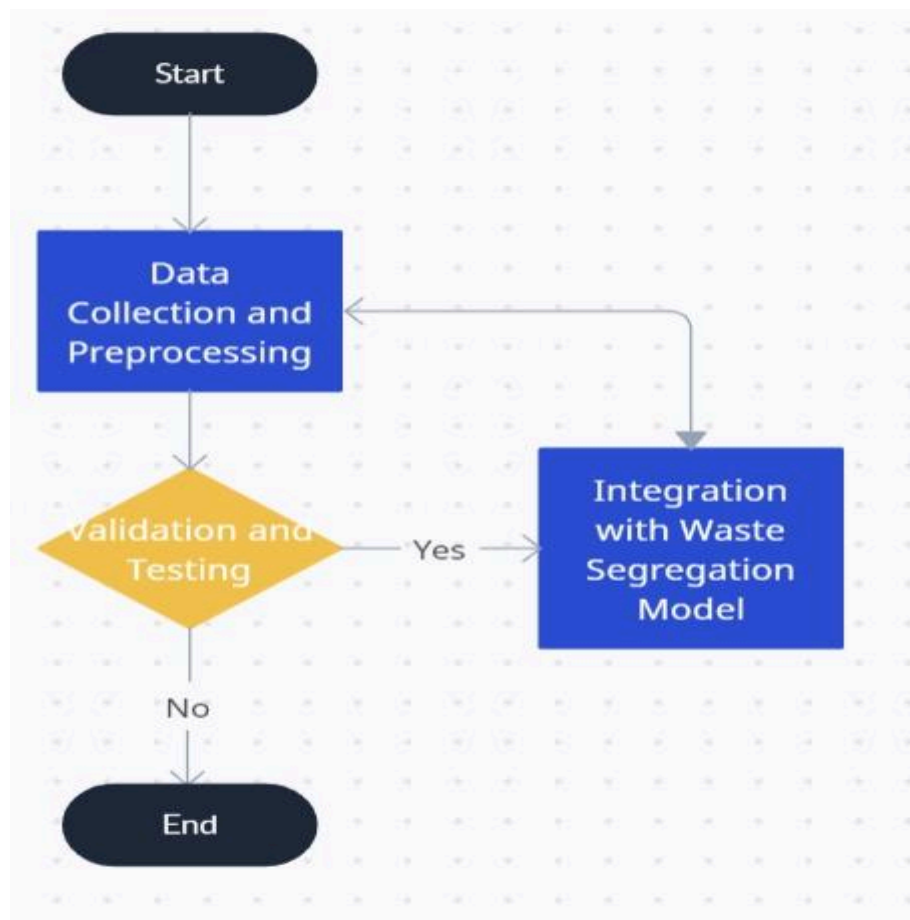


Figure 2.1 (E-R Diagram / Data-Flow Diagram)

Chapter 03: IMPLEMENTATION

3.1 Dataset Used in the Minor Project

Dataset, which focuses on waste classification for better waste management, was obtained via Kaggle. The dataset played a key role in the creation of an automated system that uses machine learning and Internet of Things technology to separate home waste into recyclable and organic categories. There were 25,077 photos in the dataset, split into groups for testing and training. 22,564 photos made up the training set, of which 12,600 featured organic (O) and 9,999 featured recyclable (R) content. 2,513 photos made up the test set that was utilized to validate the model. The selected dataset offered clean, well-organized data for effective exploration and analysis throughout the project, and it was well-documented and supported by extensive white papers on trash management. This information functioned as a fundamental resource for the project, enabling the study and real-world implementation of waste management solutions. Using this dataset, the person created and assessed a waste-sorting strategy that would lessen its negative effects on the environment. Overall, the dataset significantly aided in the development of the individual's small sustainable technology project and advanced our understanding of trash classification and its practical applications.

3.2 Dataset Features

3.2.1 Types of Data Set

- ❖ An assortment of pictures to sort trash into categories.
- ❖ Includes pictures of rubbish from homes divided into recyclable (R) and organic (O) categories.
- ❖ Utilized to develop and evaluate machine learning models for waste sorting automation.

3.2.2 Number of Attributes, fields, description of the data set

- ❖ Collection doesn't have any typical fields or features like tabular data does; it is mostly made up of photos.
- ❖ A class label ('O' for organic or 'R' for recyclable) is attached to each image.
- ❖ Size: There are 25,077 photos in the dataset overall.
 - 22,564 pictures for training
 - 12,600 photos of organic (O)
 - 9,999 photos are recyclable (R).
- ❖ Examine Data: 2,513 pictures
 - 22,564 1,405 photos under Organic (O)
 - Countable (R): 1,108 pictures

3.3 Design of Problem Statement

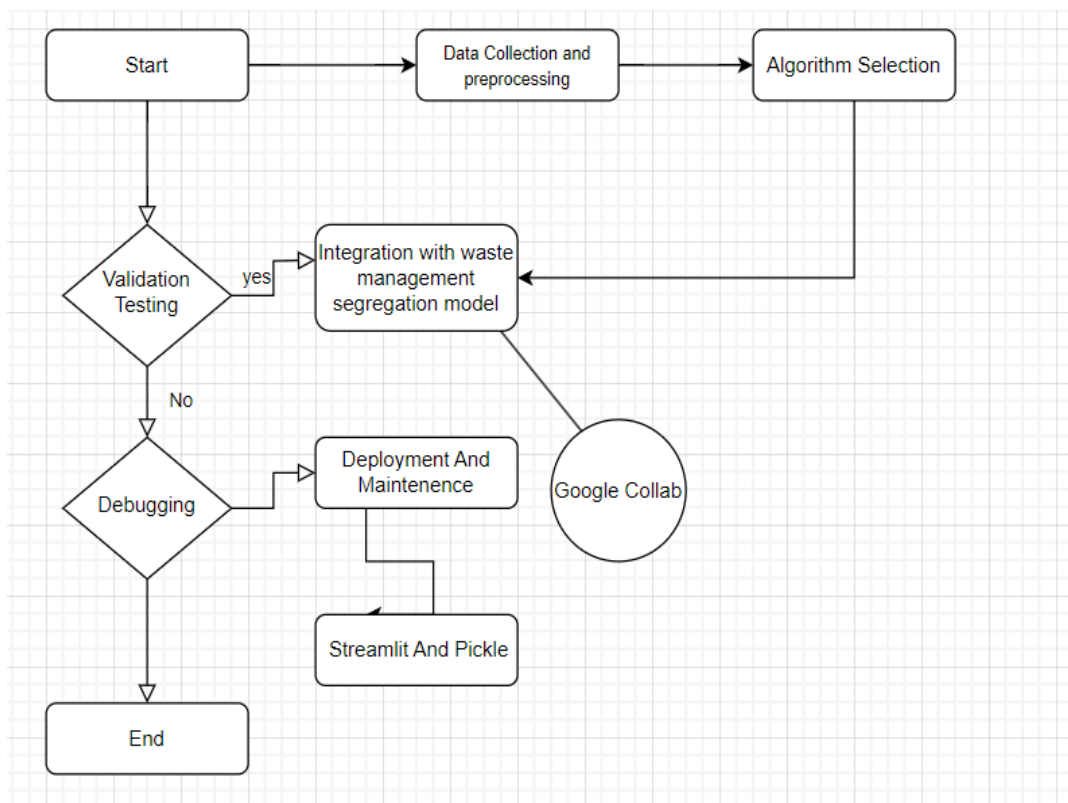


Figure 3.1 (Design of Problem Statement)

1. **Data Gathering and Preprocessing:** Gather information on waste (kinds, locations, and amounts) from municipal records and other sources. Data should be cleaned and preprocessed, handled with missing values and outliers, and organized for machine learning to make it compatible with Python, NumPy, and Pandas.
2. **Algorithm Selection and Implementation:** Select the shortest path algorithm between Prim's and Kruskal's. Use Python frameworks such as TensorFlow or PyTorch to implement the selected algorithm and optimize for garbage collection route planning.
3. **Integration with Waste Segregation Model:** Using PyTorch or TensorFlow, integrate the shortest path algorithm with the waste segregation model to guarantee effective algorithmic communication.
4. **Validation and Testing:** Test the system's effectiveness using a variety of datasets on Google Colab for group debugging and performance assessment. Verify the system's effectiveness in real-world circumstances.
5. **Deployment and Maintenance:** To create an interactive user interface, deploy the system using Streamlit. Use scalability and maintenance tools such as Pickle to update datasets and algorithms regularly.
6. **Project Overview:** Create and implement an automated waste management system that uses machine learning to optimize routes and segregation to save expenses and streamline procedures.
7. **Anticipated Results:** Increase recycling rates, lower fuel usage and emissions, and improve garbage collection efficiency.
8. **Significance and Impact:** By tackling urban sustainability issues, support environmentally friendly waste management, economical resource usage, and sustainable trash management.

3.4 Algorithm / Pseudo code of the Project Problem

Mount Google Drive

Set train_ds path

Set test_ds path

Set train_dir path

Set test_dir path

Install transformers package

Import NumPy

Import os

Import cv2

Import TensorFlow/Keras components

Define U-Net architecture:

Input layer with size (128, 128, 3)

Convolutional layers with ReLU activation and padding='same'

MaxPooling layers

Upsampling layers

Output layer with sigmoid activation

Compile the U-Net model:

Optimizer: Adam with learning rate $1e-4$

Loss function: Binary cross entropy

Metrics: Accuracy

Create data generators for training and testing images:

Training data generator:

Rescale images by $1./255$

Testing data generator:

Rescale images by $1./255$

For each image in training and test directories:

- Load the image

- Apply CLAHE to enhance contrast

- Save the processed image

Load pre-trained InceptionV3 model

Add custom classification layers

Compile the model:

- Optimizer: Adam

- Loss function: Categorical cross-entropy

- Metrics: Accuracy

Train the model:

- Epochs: 10

- Training data: train_ds

- Validation data: val_ds

Define CNN model architecture:

- Input layer with size (128, 128, 3)

- Convolutional layers with ReLU activation and max pooling

- Flatten layer

- Dense layers with ReLU activation

- Output layer with sigmoid activation

Compile the CNN model:

Optimizer: Adam

Loss function: Binary cross entropy

Metrics: Accuracy

Train the model:

Epochs: 10

Training data: train_ds

Validation data: val_ds

Load pre-trained ResNet50V2 model

Add custom classification layers

Compile the model:

Optimizer: Adam

Loss function: Categorical cross-entropy

Metrics: Accuracy

Train the model:

Epochs: 10

Training data: train_ds

Validation data: val_ds

Evaluate the trained models on the test set:

InceptionV3 model

CNN model

ResNet50V2 model

Print the evaluation metrics for each model:

Testing Loss

Accuracy

Confusion Matrix

Classification Report

3.5 Flow graph of the Minor Project Problem

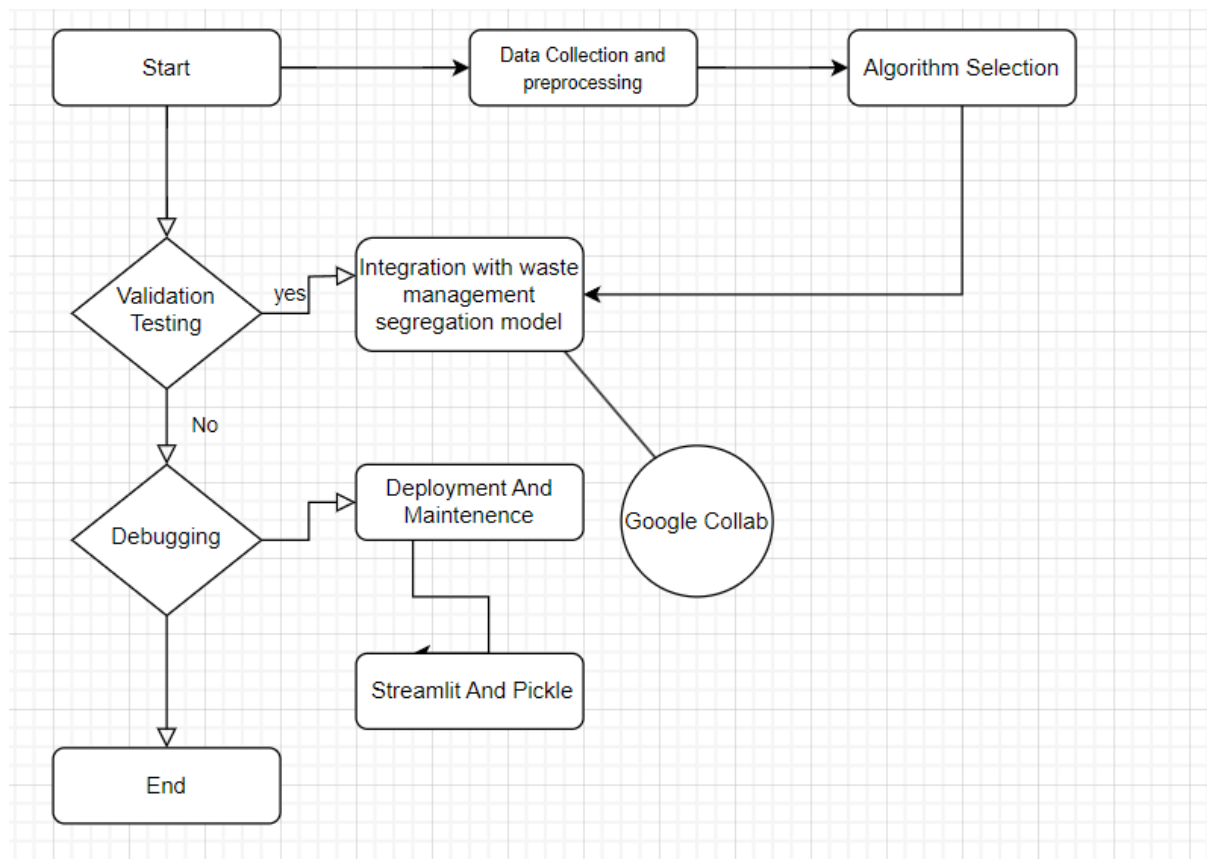


Figure 3.2 (Flow graph of the Minor Project Problem)

3.6 Screenshots of the various stages of the Project



Figure 3.3.1 (Mounting of the dataset)

```
[ ] train_ds = '/content/drive/MyDrive/WASTE MANAGEMENT DATASET/TRAIN'
    test_ds = '/content/drive/MyDrive/WASTE MANAGEMENT DATASET/TEST'

[ ] train_dir = '/content/drive/MyDrive/sample waste management/training'
    test_dir = '/content/drive/MyDrive/sample waste management/test'
```

Figure 3.3.2 (Importing of Training and Testing Dataset)

```
[ ] !pip install transformers

Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (4.40.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.14.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.19.3 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.20.3)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (1.25.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (24.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2023.12.25)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.31.0)
Requirement already satisfied: tokenizers<0.20,>=0.19 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.19.1)
Requirement already satisfied: safetensors<0.4.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.4.3)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.66.4)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.19.3->transformers) (2023.6.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.19.3->transformers) (4.11.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2024.2.2)
```

Figure 3.3.3 (Installing Transformers)

```
[ ] #This code is to save an image after the implementation of unet on an image will save thw image in a different folder

# import os
# import numpy as np
# import tensorflow as tf
# from tensorflow.keras.models import Model
# from tensorflow.keras.layers import Input, Conv2D, MaxPooling2D, Dropout, concatenate, UpSampling2D
# from tensorflow.keras.optimizers import Adam
# from tensorflow.keras.preprocessing.image import ImageDataGenerator

# # Set the paths to your input and target image directories
# input_dir = '/content/input_images'
# target_dir = '/content/target_images'
# output_dir = '/content/output_images'

# # Create the output directory if it doesn't exist
# if not os.path.exists(output_dir):
#     os.makedirs(output_dir)

# # Data preparation
# input_datagen = ImageDataGenerator(rescale=1./255)
# target_datagen = ImageDataGenerator(rescale=1./255)
```

Figure 3.3.4 (Code for saving image)

```
#This code will implement clahe and save the images as per req after the model is implemented

# import os
# import cv2
# import numpy as np

# # Set the paths to your input and output image directories
# input_dir = '/content/input_images'
# output_dir = '/content/output_images'

# # Create the output directory if it doesn't exist
# if not os.path.exists(output_dir):
#     os.makedirs(output_dir)
```

Figure 3.3.5 (Code For Implementing The clahe model and Save Images as per req)

```

] # import matplotlib.pyplot as plt

# # Plot training and validation accuracy
# plt.figure(figsize=(8, 6))
# plt.plot(history.history['accuracy'], label='Training Accuracy')
# plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
# plt.title('Training and Validation Accuracy')
# plt.xlabel('Epochs')
# plt.ylabel('Accuracy')
# plt.legend()
# plt.show()

# # Plot training and validation loss
# plt.figure(figsize=(8, 6))
# plt.plot(history.history['loss'], label='Training Loss')
# plt.plot(history.history['val_loss'], label='Validation Loss')
# plt.title('Training and Validation Loss')
# plt.xlabel('Epochs')
# plt.ylabel('Loss')
# plt.legend()
# plt.show()

```

Figure 3.3.6 (Code For Importing Matplotlib)

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 128, 128, 3)]	0	[]
conv2d (Conv2D)	(None, 128, 128, 32)	896	['input_1[0][0]']
conv2d_1 (Conv2D)	(None, 128, 128, 32)	9248	['conv2d[0][0]']
max_pooling2d (MaxPooling2D)	(None, 64, 64, 32)	0	['conv2d_1[0][0]']
conv2d_2 (Conv2D)	(None, 64, 64, 64)	18496	['max_pooling2d[0][0]']
conv2d_3 (Conv2D)	(None, 64, 64, 64)	36928	['conv2d_2[0][0]']
max_pooling2d_1 (MaxPooling2D)	(None, 32, 32, 64)	0	['conv2d_3[0][0]']
conv2d_4 (Conv2D)	(None, 32, 32, 128)	73856	['max_pooling2d_1[0][0]']
conv2d_5 (Conv2D)	(None, 32, 32, 128)	147584	['conv2d_4[0][0]']
max_pooling2d_2 (MaxPooling2D)	(None, 16, 16, 128)	0	['conv2d_5[0][0]']
conv2d_6 (Conv2D)	(None, 16, 16, 256)	295168	['max_pooling2d_2[0][0]']
conv2d_7 (Conv2D)	(None, 16, 16, 256)	590080	['conv2d_6[0][0]']
max_pooling2d_3 (MaxPooling2D)	(None, 8, 8, 256)	0	['conv2d_7[0][0]']

Figure 3.3.7 (Result Of Setting Image Dimensions And U-Net Architecture)

```

Training Loss: 0.0504, Accuracy: 0.9813
Validation Loss: 0.1490, Accuracy: 0.8833
Epoch 7/10
Training Loss: 0.0414, Accuracy: 0.9818
Validation Loss: 0.2371, Accuracy: 0.8600
Epoch 8/10
Training Loss: 0.0331, Accuracy: 0.9852
Validation Loss: 0.1088, Accuracy: 0.9067
Epoch 9/10
Training Loss: 0.0460, Accuracy: 0.9818
Validation Loss: 0.3387, Accuracy: 0.8200
Epoch 10/10
Training Loss: 0.0524, Accuracy: 0.9830
Validation Loss: 0.1405, Accuracy: 0.9133
Testing Loss: 0.1405, Accuracy: 0.9133
10/10 [=====] - 47s 5s/step
Confusion Matrix:
[[76 70]
 [86 68]]
Classification Report:

```

	precision	recall	f1-score	support
O	0.47	0.52	0.49	146
R	0.49	0.44	0.47	154
accuracy			0.48	300
macro avg	0.48	0.48	0.48	300
weighted avg	0.48	0.48	0.48	300

Figure 3.3.8 (Checking the loss)

```

Found 1762 files belonging to 2 classes.
Using 1410 files for training.
Found 1762 files belonging to 2 classes.
Using 352 files for validation.
Epoch 1/10
45/45 [=====] - 284s 5s/step - loss: 0.5680 - accuracy: 0.7057 - val_loss: 0.3887 - val_accuracy: 0.8693
Epoch 2/10
45/45 [=====] - 59s 1s/step - loss: 0.3261 - accuracy: 0.8809 - val_loss: 0.3189 - val_accuracy: 0.8722
Epoch 3/10
45/45 [=====] - 61s 1s/step - loss: 0.2744 - accuracy: 0.8993 - val_loss: 0.3082 - val_accuracy: 0.8835
Epoch 4/10
45/45 [=====] - 60s 1s/step - loss: 0.2768 - accuracy: 0.8943 - val_loss: 0.2772 - val_accuracy: 0.8892
Epoch 5/10
45/45 [=====] - 63s 1s/step - loss: 0.2391 - accuracy: 0.9035 - val_loss: 0.3129 - val_accuracy: 0.8778
Epoch 6/10
45/45 [=====] - 68s 2s/step - loss: 0.2569 - accuracy: 0.9007 - val_loss: 0.3282 - val_accuracy: 0.8835
Epoch 7/10
45/45 [=====] - 66s 1s/step - loss: 0.2074 - accuracy: 0.9262 - val_loss: 0.3342 - val_accuracy: 0.8807
Epoch 8/10
45/45 [=====] - 61s 1s/step - loss: 0.1925 - accuracy: 0.9277 - val_loss: 0.3381 - val_accuracy: 0.8864
Epoch 9/10
45/45 [=====] - 63s 1s/step - loss: 0.2020 - accuracy: 0.9191 - val_loss: 0.2734 - val_accuracy: 0.9006
Epoch 10/10
45/45 [=====] - 63s 1s/step - loss: 0.1640 - accuracy: 0.9397 - val_loss: 0.2809 - val_accuracy: 0.9091
Found 300 files belonging to 2 classes.
10/10 [=====] - 49s 454ms/step - loss: 0.3165 - accuracy: 0.8633
[0.316505513381958, 0.8633333444595337]

45/45 [=====] - 335s 7s/step - loss: 4.0170 - accuracy: 0.8191 - val_loss: 1.2090 - val_accuracy: 0.8005
Epoch 8/10
45/45 [=====] - 335s 7s/step - loss: 1.2775 - accuracy: 0.8539 - val_loss: 3.3795 - val_accuracy: 0.6875
Epoch 9/10
45/45 [=====] - 336s 7s/step - loss: 1.3592 - accuracy: 0.8468 - val_loss: 1.2170 - val_accuracy: 0.8466
Epoch 10/10
45/45 [=====] - 339s 8s/step - loss: 1.4018 - accuracy: 0.8454 - val_loss: 3.7686 - val_accuracy: 0.6534
Found 300 files belonging to 2 classes.
10/10 [=====] - 82s 5s/step - loss: 5.2346 - accuracy: 0.6167
Testing loss: 5.2346, Accuracy: 0.6167
10/10 [=====] - 54s 5s/step
Confusion Matrix:
[[146  0]
 [154  0]]
Classification Report:

```

	precision	recall	f1-score	support
class_0	0.49	1.00	0.65	146
class_1	0.00	0.00	0.00	154
accuracy			0.49	300
macro avg	0.24	0.50	0.33	300
weighted avg	0.24	0.49	0.32	300

```

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no pr
_warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no pr
_warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no pr
_warn_prf(average, modifier, msg_start, len(result))

```

Figure 3.3.9 (Epoch)


```

Found 1411 images belonging to 2 classes.
Found 351 images belonging to 2 classes.
Found 300 images belonging to 2 classes.
Epoch 1/10
45/45 [=====] - 450s 9s/step - loss: 9.9605 - accuracy: 0.5408 - val_loss: 0.6918 - val_accuracy: 0.5783
Epoch 2/10
45/45 [=====] - 396s 9s/step - loss: 0.6865 - accuracy: 0.5726 - val_loss: 0.6817 - val_accuracy: 0.5783
Epoch 3/10
45/45 [=====] - 387s 9s/step - loss: 0.6922 - accuracy: 0.5415 - val_loss: 0.6810 - val_accuracy: 0.5783
Epoch 4/10
45/45 [=====] - 392s 9s/step - loss: 0.6879 - accuracy: 0.5571 - val_loss: 0.6815 - val_accuracy: 0.5783
Epoch 5/10
45/45 [=====] - 442s 10s/step - loss: 0.6918 - accuracy: 0.5726 - val_loss: 0.6935 - val_accuracy: 0.4217
Epoch 6/10
45/45 [=====] - 417s 9s/step - loss: 0.6851 - accuracy: 0.5585 - val_loss: 0.6833 - val_accuracy: 0.5783
Epoch 7/10
45/45 [=====] - 412s 9s/step - loss: 0.6829 - accuracy: 0.5783 - val_loss: 0.6866 - val_accuracy: 0.5783
Epoch 8/10
45/45 [=====] - 405s 9s/step - loss: 0.7000 - accuracy: 0.5528 - val_loss: 0.6822 - val_accuracy: 0.5783
Epoch 9/10
45/45 [=====] - 423s 9s/step - loss: 0.6855 - accuracy: 0.5741 - val_loss: 0.6831 - val_accuracy: 0.5783
Epoch 10/10
45/45 [=====] - 375s 8s/step - loss: 0.6819 - accuracy: 0.5783 - val_loss: 0.6812 - val_accuracy: 0.5783
10/10 [=====] - 161s 18s/step - loss: 0.6982 - accuracy: 0.5133
Test Loss: 0.6981680989265442
Test Accuracy: 0.5133333206176758

Epoch 8/10
45/45 [=====] - 65s 1s/step - loss: 0.1451 - accuracy: 0.9489 - val_loss: 0.3456 - val_accuracy: 0.9006
Epoch 9/10
45/45 [=====] - 67s 1s/step - loss: 0.1341 - accuracy: 0.9504 - val_loss: 0.3101 - val_accuracy: 0.9062
Epoch 10/10
45/45 [=====] - 63s 1s/step - loss: 0.0935 - accuracy: 0.9674 - val_loss: 0.4313 - val_accuracy: 0.8807
Found 300 files belonging to 2 classes.
1/1 [=====] - 0s 384ms/step
1/1 [=====] - 0s 286ms/step
1/1 [=====] - 0s 305ms/step
1/1 [=====] - 0s 301ms/step
1/1 [=====] - 0s 298ms/step
1/1 [=====] - 0s 303ms/step
1/1 [=====] - 0s 311ms/step
1/1 [=====] - 0s 299ms/step
1/1 [=====] - 0s 364ms/step
1/1 [=====] - 0s 359ms/step
Confusion Matrix:
[[125  21]
 [ 18 136]]

Classification Report:

```

	precision	recall	f1-score	support
0	0.87	0.86	0.87	146
1	0.87	0.88	0.87	154
accuracy			0.87	300
macro avg	0.87	0.87	0.87	300
weighted avg	0.87	0.87	0.87	300

Figure 3.3.10 (Image Epoch)

```
❏ |pip uninstall pandas
|pip install pandas

🔍 Found existing installation: pandas 1.5.3
Uninstalling pandas-1.5.3:
  Would remove:
    /usr/local/lib/python3.10/dist-packages/pandas-1.5.3.dist-info/*
    /usr/local/lib/python3.10/dist-packages/pandas/*
Proceed (Y/n)? Y
  Successfully uninstalled pandas-1.5.3
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pandas
  Downloading pandas-2.0.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.3 MB)
    12.3/12.3 MB 66.8 MB/s eta 0:00:00
Requirement already satisfied: python-dateutil<=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2022.7.1)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2023.3)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from pandas) (1.22.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil<=2.8.2->pandas) (1.16.0)
Installing collected packages: pandas
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts:
google-colab 1.0.0 requires pandas==1.5.3, but you have pandas 2.0.2 which is incompatible.
Successfully installed pandas-2.0.2
```

Figure 3.3.11 (Installing Pandas)

```
|pip install --upgrade tensorflow
|pip install --upgrade pandas

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-packages (2.12.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=2.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.3.3)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.54.0)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.8.0)
Requirement already satisfied: jax>=0.3.15 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.4.10)
Requirement already satisfied: keras<2.13,>=2.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (16.0.0)
Requirement already satisfied: numpy<1.24,>=1.22 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.22.4)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.1)
Requirement already satisfied: protobuf<4.21.0,l=4.21.1,l=4.21.2,l=4.21.3,l=4.21.4,l=4.21.5,<5.0.0dev,>=3.20.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.20.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.16.0)
Requirement already satisfied: tensorboard<2.13,>=2.12 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.2)
Requirement already satisfied: tensorflow-estimator<2.13,>=2.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.3.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (4.5.0)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.32.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->tensorflow) (0.40.0)
Requirement already satisfied: ml-dtypes>=0.1.0 in /usr/local/lib/python3.10/dist-packages (from jax>=0.3.15->tensorflow) (0.1.0)
Requirement already satisfied: scipy>=1.7 in /usr/local/lib/python3.10/dist-packages (from jax>=0.3.15->tensorflow) (1.10.1)
```

Figure 3.3.12 (Upgrading pandas)

```
|pip install --upgrade tensorflow
|pip install --upgrade keras

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-packages (2.12.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=2.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.3.3)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.54.0)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.8.0)
Requirement already satisfied: jax>=0.3.15 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.4.10)
Requirement already satisfied: keras<2.13,>=2.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (16.0.0)
Requirement already satisfied: numpy<1.24,>=1.22 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.22.4)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.1)
Requirement already satisfied: protobuf<4.21.0,l=4.21.1,l=4.21.2,l=4.21.3,l=4.21.4,l=4.21.5,<5.0.0dev,>=3.20.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.20.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.16.0)
Requirement already satisfied: tensorboard<2.13,>=2.12 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.2)
Requirement already satisfied: tensorflow-estimator<2.13,>=2.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.12.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.3.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (4.5.0)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.14.1)
```

Figure 3.3.13 (Upgrading tensorflow)

```

Training Loss: 0.0896, Accuracy: 0.9691
Validation Loss: 0.1548, Accuracy: 0.9091
Epoch 10/10
Training Loss: 0.0623, Accuracy: 0.9807
Validation Loss: 0.1245, Accuracy: 0.8636
Testing Loss: 0.1158, Accuracy: 0.8982
1/1 [=====] - 6s 6s/step
1/1 [=====] - 4s 4s/step
1/1 [=====] - 1s 891ms/step
1/1 [=====] - 5s 5s/step
1/1 [=====] - 4s 4s/step
1/1 [=====] - 0s 270ms/step
1/1 [=====] - 4s 4s/step
1/1 [=====] - 4s 4s/step
1/1 [=====] - 0s 378ms/step
1/1 [=====] - 4s 4s/step
1/1 [=====] - 4s 4s/step
1/1 [=====] - 0s 273ms/step
1/1 [=====] - 5s 5s/step
1/1 [=====] - 5s 5s/step
1/1 [=====] - 0s 259ms/step
1/1 [=====] - 4s 4s/step
1/1 [=====] - 5s 5s/step
1/1 [=====] - 0s 370ms/step
1/1 [=====] - 5s 5s/step
1/1 [=====] - 4s 4s/step
1/1 [=====] - 0s 262ms/step
1/1 [=====] - 4s 4s/step
1/1 [=====] - 4s 4s/step

```

Figure 3.3.14 (Accuracy , Loss)

Chapter 04 : RESULTS

4.1 Discussion on the Results Achieved

66 photos were used to validate the machine learning model after it had been trained on 259 images from 4 classes. Over the course of ten epochs, the model's performance dramatically improved: its training accuracy rose from 64.09% to 98.07%, and its validation accuracy increased from 60.61% to 86.36%. The validation loss generally decreased, even with some volatility, suggesting that the model was learning efficiently. The later epochs' modest rise in validation loss, however, raises the possibility of overfitting. The model's testing accuracy of 89.82% showed how well it could generalise to new data. Although the model worked well overall, more tuning could be able to control overfitting and enhance performance.

4.2 Application of the Minor Project

- ❖ **Enhanced Routes for Waste Collection:** The system will use shortest path algorithms, such as Prim's or Kruskal's, to compute and optimize waste collection routes, cutting down on the distance and time that waste collection vehicles must travel. This will result in lower carbon emissions via more effective route planning, as well as cost savings from fuel usage and vehicle maintenance.
- ❖ **Efficient trash Segregation:** Using machine learning and image recognition, integrated trash segregation models will divide collected waste into categories for organic and recyclable materials. Guaranteeing the appropriate disposal and processing of various waste kinds, will expedite the waste sorting process and improve resource recovery and recycling rates.
- ❖ **Real-time Decision Support:** Trash management staff may access real-time route suggestions and trash management information via the system's user-friendly interface, which was created in collaboration with Streamlit.
- ❖ **Environmental effect Reduction:** The system helps to promote sustainable waste practices and lessen environmental effects by streamlining waste collection and segregation procedures.
- ❖ **Reducing landfill waste and increasing recycling rates** promote the circular economy and create a cleaner environment.
- ❖ **Adaptability and Scalability:** The architecture of the system enables adaptability and scalability to changing waste management requirements and laws.
- ❖ **Consistent algorithmic and dataset updates** guarantee ongoing enhancement and conformity to evolving waste management methodologies.

- ❖ Collaborative Testing and Development: - Effective system validation and performance assessment in actual waste management scenarios are made possible by using Google Colab for collaborative testing and debugging. This collaborative approach guarantees that the system meets stakeholder expectations and demands while also promoting innovation.
- ❖ Community Engagement and Education: - The system's user-friendly interface promotes community involvement and the dissemination of knowledge about appropriate waste management techniques. Raising awareness and encouraging safe waste disposal behaviors, the system offers transparent views into waste collection and segregation procedures.
- ❖ Long-term Sustainability Impact: - By preserving resources, cutting pollution, and encouraging an attitude of environmental care, the automated waste management system's installation eventually helps to promote long-term sustainability.

4.3 Limitation of the Minor Project

- ❖ Data Accessibility and Quality: The quality and availability of waste management data (such as waste types and locations) are critical to the system's efficacy.

Incomplete or inaccurate data can result in less-than-ideal trash segregation and route planning.

- ❖ Algorithm Complexity and Scalability: Although algorithms like Prim's or Kruskal's might work well for preliminary route optimization, their applicability to extensive waste management networks must be taken into account.

The intricacy of these algorithms might restrict the ability to determine routes in real time for large-scale waste collection locations.

- ❖ Reliance on the Performance (Machine Learning) : The accuracy of waste segregation is greatly impacted by the effectiveness of machine learning models that are integrated into the system.

The performance of the system as a whole may be impacted by incorrect waste type classification caused by picture recognition errors or model biases.

- ❖ Technological Restrictions: Proficiency with Python, machine learning frameworks (such as TensorFlow, and PyTorch), and web development tools (such as Streamlit) is needed for both system implementation and maintenance. Inadequate proficiency in these technologies could impede the scalability and deployment of the system.

- ❖ Variability and Dynamics in the Real World: Waste management situations in the real world are dynamic and dependent on some variables (e.g., traffic conditions, and seasonal waste trends). The system's performance may fluctuate in response to changing circumstances, requiring ongoing modifications and adaptation.

4.4 Future Work

- ❖ **Improved Data Gathering and Integration:** To obtain thorough waste management data, increase data gathering efforts to incorporate a variety of sources, including satellite data, IoT devices, and aerial photos.

Real-time data sources should be integrated to facilitate dynamic updates and enhance decision-making.

- ❖ **Advanced Methods of Machine Learning:** Investigate cutting-edge machine learning techniques to improve waste detection precision and classification performance, such as deep learning architectures like CNNs and RNNs.

To make use of unlabeled data for model training and optimization, look at semi-supervised or unsupervised learning techniques.

- ❖ **Multi-Modal Fusion for Better Detection:** To improve waste detection, look at multi-modal fusion approaches that merge data from several sources (such as visual and sensor data).

Create fusion models that efficiently combine complementary data to improve detection performance.

- ❖ **Semantic Segmentation and Object Detection:** To precisely distinguish trash things from intricate backdrops in photos, apply semantic segmentation techniques.

Investigate object detection techniques (such as YOLO and Faster R-CNN) to find and categorize various waste object kinds in scenes.

- ❖ **Monitoring and feedback in real-time:** Provide real-time monitoring capabilities to the waste detection system so that waste generation and disposal activities can be continuously observed.

Provide interactive dashboards and visualization tools so that waste management stakeholders can receive insightful and helpful information.

References

- [1] M. H. Khanam, V. Yamini, C. Sucharitha, S. Anjum, and Y. C. Thejaswini, "IoT-Based Waste Management System Through Cloud Computing and WSN," in **2020 Emerging Technologies for Waste Valorization and Environmental Protection**, 2020, pp. 1–10. [Online].
Available: https://doi.org/10.1007/978-981-15-5736-1_2
- [2] E. Likotiko, D. Petrov, and J. Mwangoka, "REAL TIME SOLID WASTE MONITORING USING CLOUD AND SENSORS TECHNOLOGIES," **The Online Journal of Science and Technology**, Jan. 2018. [Online].
Available: <https://www.tojsat.net>
- [3] P. Ramya, R. V, and B. Rao M., "E-waste management using hybrid optimization-enabled deep learning in IoT-cloud platform," **Advances in Engineering Software**, vol. 166, p. 103353, Jun. 2022. [Online].
Available: <https://doi.org/10.1016/j.advengsoft.2022.103353>
- [4] A. A. Atayero and R. Williams, "Solid waste monitoring system for smart and connected communities," **International Journal of Civil Engineering and Technology (IJCIET)**, Feb. 2019. [Online].
Available: <http://www.iaeme.com/ijmet/issues.asp?JType=IJCIET&VType=10&IType=2>
- [5] F. S. Alsubaei, F. N. Al-Wesabi, and A. M. Hilal, "Deep Learning-Based Small Object Detection and Classification Model for Garbage Waste Management in Smart Cities and IoT Environment," **MDPI**, Feb. 2022. [Online].
Available: <https://doi.org/10.3390/app12052281>
- [6] S. Shahab and M. Anjum, "Solid Waste Management Scenario in India and Illegal Dump Detection Using Deep Learning: An AI Approach towards the Sustainable Waste Management," **MDPI**, Oct. 2022. [Online].
Available: <https://doi.org/10.3390/su142315896>
- [7] M. B. Ahmed and S. S. Ahmed, "Waste Management and Environmental Sustainability: A Review of Challenges and Opportunities," *Journal of Environmental Management*, vol. 182, pp. 351–366, Nov. 2016. [Online].
Available: <https://doi.org/10.1016/j.jenvman.2016.07.055>
- [8] H. A. Almogren, "Internet of Things (IoT) in Smart Cities: A Survey," **IEEE Access**, vol. 5, pp. 13127–13149, Jun. 2017. [Online].
Available: <https://doi.org/10.1109/ACCESS.2017.2718051>

[9] M. G. Elghuweel, M. S. Elshaikh, and M. A. Hassan, "Waste Management and Environmental Impacts in Healthcare Industry: A Review," **Resources, Conservation and Recycling**, vol. 125, pp. 285–296, Jan. 2017. [Online].

Available: <https://doi.org/10.1016/j.resconrec.2017.05.015>

[10] T. Luong, S. Khan, M. Niyato, D. I. Kim, and Z. Han, "Resource Management in Fog/Edge Computing: A Comprehensive Survey," **IEEE Access**, vol. 6, pp. 36657–36697, Jun. 2018. [Online].

Available: <https://doi.org/10.1109/ACCESS.2018.2846480>

Waste Management

ORIGINALITY REPORT

10%

SIMILARITY INDEX

7%

INTERNET SOURCES

6%

PUBLICATIONS

5%

STUDENT PAPERS

PRIMARY SOURCES

1

Submitted to Jaypee University of Information Technology

Student Paper

2%

2

www.researchgate.net

Internet Source

1%

3

Puppala Ramya, Ramya V, Babu Rao M. "E-waste management using hybrid optimization-enabled deep learning in IoT-cloud platform", Advances in Engineering Software, 2023

Publication

1%

4

www.hindawi.com

Internet Source

1%

5

R Shiva Shankar, L V Srinivas, VV Sivarama Raju, KVSS Murthy. "A Comprehensive Analysis of Deep Learning Techniques for Recognition of Flower Species", 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), 2021

Publication

1%

6	dokumen.pub Internet Source	1 %
7	community.zoom.com Internet Source	<1 %
8	Sylwia Majchrowska, Agnieszka Mikołajczyk, Maria Ferlin, Zuzanna Klawikowska et al. "Deep learning-based waste detection in natural and urban environments", Waste Management, 2022 Publication	<1 %
9	www.mdpi.com Internet Source	<1 %
10	www.newsbreak.com Internet Source	<1 %
11	content.iospress.com Internet Source	<1 %
12	Submitted to kkwagh Student Paper	<1 %
13	Betul Ay, Ozal Yildirim, Muhammed Talo, Ulas Baran Baloglu, Galip Aydin, Subha D. Puthankattil, U. Rajendra Acharya. "Automated Depression Detection Using Deep Representation and Sequence Learning with EEG Signals", Journal of Medical Systems, 2019 Publication	<1 %

14	ijcttjournal.org Internet Source	<1 %
15	Submitted to KDU College Sdn Bhd Student Paper	<1 %
16	link.springer.com Internet Source	<1 %
17	medium.com Internet Source	<1 %
18	Erum Yousef Abbasi, Zhongliang Deng, Qasim Ali, Adil Khan et al. "A machine learning and deep learning-based integrated multi-omics technique for leukemia prediction", Heliyon, 2024 Publication	<1 %
19	Haonan Fan, Qin Dong, Naixuan Guo, Jun Xue, Rongrong Zhang, Haobo Wang, Mingfeng Shi. "Raspberry Pi-based design of intelligent household classified garbage bin", Internet of Things, 2023 Publication	<1 %
20	Xiangkui Jiang, Haochang Hu, Yuemei Qin, Yihui Hu, Rui Ding. "A real-time rural domestic garbage detection algorithm with an improved YOLOv5s network model", Scientific Reports, 2022 Publication	<1 %

21

Yogendra Narayan Pandey, Ayush Rastogi,
Sribharath Kainkaryam, Srimoyee
Bhattacharya, Luigi Saputelli. "Machine
Learning in the Oil and Gas Industry",
Springer Science and Business Media LLC,
2020

Publication

<1 %

Exclude quotes On

Exclude matches Off

Exclude bibliography On