Accuracy and Efficiency Gains in Waste Classification Through Continuous Learning and Advanced Techniques

A Project Report submitted in the partial fulfillment of

the Requirements for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

M.Madhavi	**(22475A0522)**
M.Sreya Reddy	**(21471A05N8)**
P.Nikhitha	**(22475A0504)**

Under the esteemed guidance of

Mothe Sathyam Reddy, **M.Tech.**

Assistant professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE: NARASAROPET

(AUTONOMOUS)

Accredited by NAAC with A+ Grade and NBA under Tyre -1 NIRF rank in the band of 201-300 and an ISO 9001:2015 Certified

Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada

KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET- 522601

2024-2025

NARASARAOPETA ENGINEERING COLLEGE

(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project that is entitled with the name “Accuracy and Efficiency Gains in Waste Classification Through Continuous Learning and Advanced Techniques**” is a bonafide work done by the team** M.Madhavi(22475A0522), M.Sreya Reddy (21471A05N8), P.Nikhitha (22475A0504) in partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in the Department of COMPUTER SCIENCE AND ENGINEERING during 2024-2025.

PROJECT GUIDE

Mothe Sathyam Reddy, M.Tech.

Assistant professor

PROJECT CO-ORDINATOR

Dodda Venkata Reddy, M.Tech.,(ph.D)

Assistant Professor

HEAD OFTHE DEPARTMENT

Dr. S. N. Tirumala Rao, M.Tech., Ph.D.,

Professor& HOD

EXTERNAL EXAMINER

DECLARATION

We declare that this project work titled " Accuracy and Efficiency Gains in Waste Classification Through Continuous Learning and Advanced Techniques" is composed by ourselves that the work contain here is our own except where explicitly stated otherwise in the text and that this work has been submitted for any other degree or professional qualification except as specified.

M.Madhavi	**(22475A0522)**
M.Sreya	**(21471A05N8)**
P.Nikhitha	**(22475A0504)**

ACKNOWLEDGEMENT

We wish to express my thanks to carious personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sri **M.V.Koteswara Rao, B.Sc.,** who took keen interest in us in every effort throughout thiscourse. We owe out sincere gratitude to our beloved principal **Dr.S.Venkateswarlu, Ph.D.,** for showing his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude towards **Dr. S. N. Tirumala Rao, M.Tech., Ph.D.,** HOD of CSE department and also to our guide **Mothe Sathyam Reddy, M.Tech.,** Associate professor whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks towards **D. VenkataReddy, B.Tech, M.Tech.,Ph.D.,** Associate professor & Project coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff to department for their cooperation and encouragement during our B.Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who involved in giving valuable suggestions had clarifying out doubts which had really helped us in successfully completing our project.

By

M.Madhavi (22475A0522)

M.Sreya (21471A05N8)

P.Nikhitha (22475A0504)

NARASARAOPETA
NEC ENGINEERING COLLEGE
(AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community,

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching, imbibing experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems

NEC NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a centre of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

Program Specific Outcomes (PSO’s)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Educational Objectives (PEO's)

The graduates of the programme are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Outcomes

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis:** Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, andsynthesis of the information to provide valid conclusions.

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineeringactivities with an understanding of the limitations.

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability:** Understand the impact of the professional engineering

solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

7. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

8. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

9. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and giveand receive clear instructions.

10. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

11. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Project Course Outcomes (CO'S):

CO421.1: Analyze the System of Examinations and identify the problem.

CO421.2: Identify and classify the requirements.

CO421.3: Review the Related Literature

CO421.4: Design and Modularize the project

CO421.5: Construct, Integrate, Test and Implement the Project.

CO421.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1		✓											✓		
C421.2	✓		✓		✓								✓		
C421.3				✓		✓	✓	✓					✓		
C421.4			✓			✓	✓	✓					✓	✓	
C421.5					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
C421.6									✓	✓	✓		✓	✓	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1	2	3											2		
C421.2			2		3								2		
C421.3				2		2	3	3					2		
C421.4			2			1	1	2					3	2	
C421.5					3	3	3	2	3	2	2	1	3	2	1
C421.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. Low level

2. Medium level

3. High level

Project mapping with various courses of Curriculum with Attained PO's:

Name of the course fron which principles are applied in this project	**Description of the device**	**Attained PO**
C2204.2, C22L3.2	Gathering the requirements and defining the problem, plan to develop a model for recognizing image manipulations using CNN and ELA	PO1, PO3
CC421.1, C2204.3, C22L3.2	Each and every requirement i critically analyzed, the process mode is identified	PO2, PO3
CC421.2, C2204.2, C22L3.3	Logical design is done by using the unified modelling language which involves individual team work	PO3, PO5, PO9
CC421.3, C2204.3, C22L3.2	Each and every module is tested, integrated, and evaluated in our project	PO1, PO5
CC421.4, C2204.4, C22L3.2	Documentation is done by all our four members in the form of a group	PO10
CC421.5, C2204.2, C22L3.3	Each and every phase of the work in group is presented periodically	PO10, PO11
C2202.2, C2203.3, C1206.3, C3204.3, C4110.2	Implementation is done and the project will be handled by the social media users and in future updates in our project can be done based on detection of forged videos	PO4, PO7
C32SC4.3	The physical design includes website to check whether an image is real or fake	PO5, PO6

ABSTRACT

Efficient management of waste is crucial for sustainable development, especially with the ever-growing volume of municipal and household waste. Existing waste classification systems suffer from several challenges: they achieve a very low recognition accuracy rate, create class imbalance problems, and find it hard to scale-up applicability in realworld conditions. This paper addresses these issues by proposing an the InceptionV3 architecture of deep learning.

MBWO is utilized to optimize key hyperparameters such as learning rate and dropout rate, batch size on the TrashNet dataset. As a result of this optimization process, it showcases outstanding performance metrics such as 97.75% accuracy, 99.55% specificity, and provides 98.88% accuracy compared to traditional methods. It categorizes waste materials as paper, plastic, metal, and glass for efficient recycling processes and reduced landfills dependency.

This system overcomes the deficiencies of current systems by incorporating both data augmentation and oversampling techniques to handle class imbalance, along with transferring knowledge with CNNs for adaptability in real-world scenarios. The current proposed waste sorting system is a rather improved step towards sustainability in terms of environmental cleanliness and resource utilization. Future work will increase the dataset, allow for real-time waste classification, and extend applications to smart city infrastructures while ensuring scalability and applicability in modern waste management challenges

INDEX

LIST OF FIGURES

1. INTRODUCTION

1.1 INTRODUCTION

Effective waste management is thus critical for environmental sustainability, and improper waste disposal remains one of the major issues facing human beings worldwide, leading to depletion of resources, pollution, and health hazards.This underlines the tremendous pressure to develop automated intelligent waste classification systems. Recent developments in AI and deep learning, though promising improving waste classification, make the most recent image-classification models, like InceptionV3, applicable to this domain. Traditional approaches still suffer from all key challenges, like class imbalance, overfitting, and lack of scalability in practice. To neutralize these, this research combines InceptionV3 with Multi-Objective Beluga Whale Optimization (MBWO) for hyperparameter optimization and better performance. Techniques such as data augmentation and over-sampling will help in generalized applicability towards the different types of wastes. As demonstrated on the TrashNet dataset, it always performs better than the traditional methods with a classification accuracy of 97.75% and thus supports more efficient recycling and lesser dependency on landfills. The proposed approach therefore offers an economical and sustainable solution for modern waste management systems that can be integrated with other global sustainability goals and smart city infrastructures..[1]

Domestic solid waste classification plays a crucial role in sustainable waste management, economic development, and environmental protection. Traditional waste disposal methods, such as landfilling and incineration, pose serious environmental threats, including air pollution and land degradation. Although consumers are willing to classify their waste for recycling, a lack of knowledge about proper waste categorization leads to errors in manual sorting. In recent years, artificial neural networks, particularly Convolutional Neural Networks (CNNs), have been employed to automate waste classification and enhance recycling efficiency. However, previous studies have largely focused on limited waste categories, such as those in the TrashNet dataset, which includes only six classes (cardboard, glass, metal, paper, plastic, and trash). This limitation fails to reflect real-world waste diversity, which includes batteries, biological waste, and clothing materials, all of which require proper disposal and recycling strategies.This study addresses these challenges by utilizing a larger dataset containing 15,515 images across 12 waste categories to train and evaluate multiple CNN models, including DenseNet121, DenseNet169, EfficientNetB0, InceptionV3, MobileNetV2, ResNet50, VGG16, VGG19, and Xception. By leveraging data augmentation techniques to mitigate class imbalance and optimizing hyperparameters with advanced learning methods, this research aims to improve classification accuracy and model generalization. The findings will contribute to the development of intelligent waste classification systems, reducing landfill dependency, minimizing environmental hazards, and promoting a more effective recycling process.[2]

The rising volume of household waste demands efficient classification to improve recycling and sustainability. Traditional methods are labor-intensive and error-prone, highlighting the need for AI-driven solutions. This project Accuracy and Efficiency Gains in Waste Classification Through Continuous Learning and Advanced Techniques, leverages InceptionV3 and optimization techniques to enhance accuracy and scalability. By integrating transfer learning and data augmentation, the model improves classification performance, reduces landfill dependency, and supports sustainable waste management.[3]

The increasing complexity of waste materials, including a mix of biodegradable, recyclable, and hazardous components, necessitates a more robust classification system. Recent advancements in deep learning have demonstrated significant improvements in waste classification accuracy, particularly with architectures like InceptionV3, which efficiently extracts hierarchical features from waste images. This study integrates continuous learning techniques to adapt to evolving waste patterns, ensuring that the model improves over time. Furthermore, multi-objective optimization methods, such as Beluga Whale Optimization (BWO), have been employed to fine-tune hyperparameters like learning rate and batch size, maximizing model performance. By utilizing datasets like TrashNet, the system achieves high specificity and sensitivity, enabling precise waste categorization. The incorporation of data augmentation and oversampling methods helps mitigate class imbalance issues, ensuring a more generalized and scalable waste classification framework. Ultimately, this research contributes to the development of an intelligent, self-improving waste classification system that supports efficient recycling, reduces environmental impact, and aligns with global sustainability goals.[4]

With the increasing volume and complexity of waste, effective classification is crucial for sustainable waste management. Traditional manual sorting methods are inefficient, time-consuming, and prone to errors, necessitating automated solutions powered by machine learning. This study compares various machine learning algorithms, including Support Vector Machine (SVM), Random Forest, Naïve Bayes, Decision Tree, K-Nearest Neighbor (KNN), and Convolutional Neural Networks (CNN), to evaluate their efficiency in waste classification. Each algorithm offers unique advantages, with CNNs excelling in image-based classification tasks, making them highly suitable for recognizing diverse waste categories. By analyzing accuracy, computational efficiency, and performance on a diverse dataset, this research aims to identify the most effective model for optimizing waste sorting processes, enhancing recycling efficiency, and reducing environmental impact. The findings will contribute to the development of intelligent, automated waste management systems, promoting sustainability and resource conservation.[5]

Effective waste classification is crucial for sustainable waste management, reducing pollution, and optimizing recycling processes. Traditional methods struggle with accuracy and scalability, necessitating AI-driven solutions. This study employs Convolutional Neural Networks (CNNs) for waste classification, achieving improved accuracy by integrating explainability techniques such as LIME, SHAP, and Grad-CAM. These methods enhance transparency, making AI models more interpretable and reliable for real-world waste segregation. The proposed approach not only improves classification performance but also ensures accountability, making it suitable for industrial and household waste management applications. Furthermore, this study leverages InceptionV3, a deep learning model known for its efficient feature extraction and high accuracy in image classification tasks. To enhance model robustness, data augmentation techniques such as rotation, flipping, and contrast adjustments are applied, ensuring better generalization across diverse waste categories. Additionally, transfer learning is utilized to optimize training time and improve performance by adapting pre-trained models to the waste classification domain. The integration of continuous learning mechanisms allows the model to evolve and adapt to new waste patterns, ensuring long-term scalability. By employing multi-objective optimization techniques to fine-tune hyperparameters, the proposed system enhances computational efficiency, reduces misclassification rates, and minimizes environmental impact. Ultimately, this AI-driven waste classification framework contributes to smarter recycling processes, reduced landfill waste, and a more sustainable waste management ecosystem [6]

Garbage disposal is a significant environmental concern affecting both industrialized and developing nations. It poses a critical challenge to ecological sustainability, leading to disruptions in environmental equilibrium. The rise in disposable product manufacturing has further intensified waste management issues, necessitating efficient disposal and recycling methods. Various organizations, including non-governmental groups and global entities like the World-Wide Fund for Nature, actively advocate for effective waste management strategies. The growing production of waste, including electronic waste, plastic, and other non-biodegradable materials, has led to an urgent need for innovative solutions. Proper waste classification using advanced technologies, such as computer vision and deep learning, can optimize waste management practices by enabling better identification and sorting of materials. This study explores the potential of Convolutional Neural Networks (CNN) in addressing these challenges by improving waste identification and classification accuracy.[7]
The rapid increase in population and urbanization has led to a significant rise in waste generation, both in volume and diversity. Improper waste management can cause severe environmental pollution, posing risks to public health. One effective approach to managing waste is to classify it based on type, allowing for efficient recycling processes. However, manual waste sorting is time-consuming and hazardous for workers, especially when dealing with decomposed waste. With advancements in artificial intelligence, automated waste classification using image-based techniques has become a viable solution. This study proposes a waste classification model using Support Vector Machine (SVM) with SIFT-PCA feature extraction, where SIFT (Scale-Invariant Feature Transform) is used for extracting key visual features, and PCA (Principal Component Analysis) is applied to reduce dimensionality. By implementing this method on the Trashnet dataset, the research aims to improve classification efficiency and accuracy, thereby contributing to automated and intelligent waste management systems.[8]

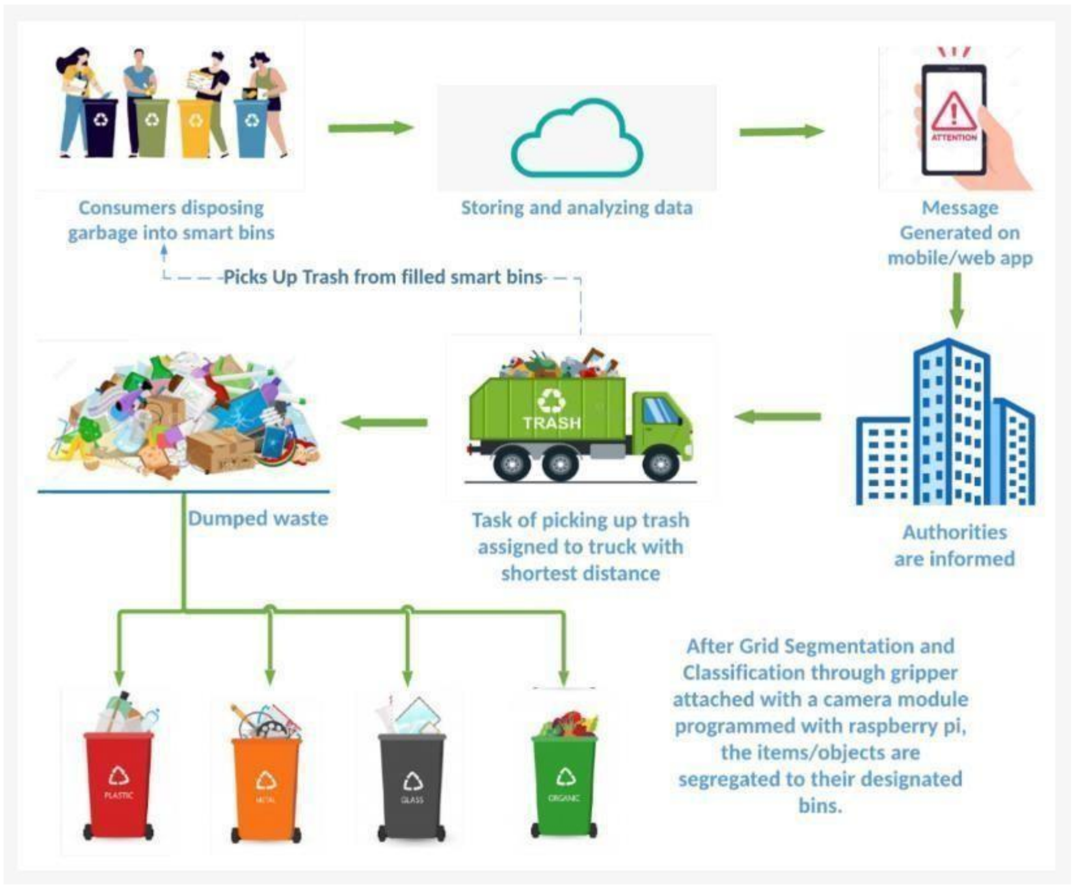

Consumers disposing garbage into smart bins
Storing and analyzing data
ATTENTION
Message Generated on mobile/web app
Picks Up Trash from filled smart bins
TRASH
Dumped waste
Task of picking up trash assigned to truck with shortest distance
Authorities are informed
After Grid Segmentation and Classification through gripper attached with a camera module programmed with raspberry pi, the items/objects are segregated to their designated bins.
PLASTIC
METAL
GLASS
ORGANIC

Fig 1.1 Waste Classification

This study explores the use of Convolutional Neural Networks (CNNs) for waste classification, addressing the challenge of multi-category waste segregation in real-world scenarios. By categorizing waste into eight distinct types, the model enhances efficiency in waste management. To improve transparency and trust, explainability techniques such as LIME, SHAP, and Grad-CAM are incorporated, making the AI model's decision-making process interpretable. [9]

1.1 MOTIVATION

With the rapid increase in waste generation due to urbanization and industrialization, traditional waste management techniques have become insufficient in ensuring efficient waste classification and recycling. Manual sorting is time-consuming, labor-intensive, and poses health risks to workers, while conventional automated methods often struggle with accuracy due to variations in waste materials. The absence of a standardized waste classification system further complicates recycling efforts, leading to inefficient resource recovery and increased environmental pollution.[10]

Recent advancements in artificial intelligence (AI) and deep learning have shown great

2. LITERATURE SURVEY

2.1 LITERATURE SURVEY

Artificial Intelligence-based Smart Household Waste Classification System This paper mostl concentrates on the development of anintelligent system for classifying household waste using deep learning methodologies, namely The application of CNNs and transfer learning. The system will automatically classify different types of household wastes according to their images by the use of a number of architectures of CNNs, such as MobileNet, VGG16, and ResNet50. It discusses some of the challenges, like the limitation of categories in public datasets, and overcomes them using the features of pretrained models and transfer learning techniques. The focus of this paper is on building an economic solution by using a Raspberry Pi for real-time implementation suitable for a smart bin and waste sorting system. The results show that one of the best ways to classify waste involves the use of CNNs, which have enormous potential for integration into household management systems and municipal management[1].

Classification of Domestic Solid Waste using Convolutional Neural Networks The work is centered on the burgeoning concern of the massive volume of household waste that contributes to environmental pollution and climate change. This paper goes further to propose automation in solid waste classification using CNN. The study has improved the limitations from the existing datasets such as the TrashNet, which used six categories into a more extensive dataset totaling 15,515 images from twelve categories. It compares different CNN models such as DenseNet, InceptionV3, and Xception to achieve better results. Among these, Xception with the Nadam optimizer has given the best classification of 89.57that performing data augmentation will address the issue of class disparity and improve the generalization capability of the model[2]. Multi-Objective Beluga Whale Optimized and Deep Learning based Intelligent-Sustainable Waste Classification Model. This work presents the waste classification model with deep learning structure using the InceptionV3 model. dropout as well as the batch size as the parameters in order to obtain better results in the classification of wastes. Results show that accuracy of the model is 97.75is well suited for the cost estimation of the software projects. Classification accuracy of 75 percent on the TrashNet dataset and they do so better than classical models. In this paper, it explains the difficulties that are arising from class imbalance data, and how the use of augmentation and oversampling .[3]

Comparison of Various Algorithms for Effective Classification of Waste: This paper discussed some of the machine learning algorithms applied to the classification of wastes: Support Vector Machine, Random Forest, Naıve Bayes Classifier, Decision Tree, K-Nearest Neighbor, and Convolutional Neural Networks. The study identifies automated waste classification as playing a highly crucial role in recycling and reducing harmful impacts on the environment. Using the dataset of waste images, the study will compare the performance of these algorithms based on accuracy and computational efficiency, thus providing a number of strengths and limitations of each of these algorithms regarding waste classification[4]. Review on Deep Learning-Based Biomedical Waste Detection and Classification: This paper designs a deep learning-based system using CNNs for biomedical waste detection and classification. Basic safety and public health concerns, at a minimum, are stressed for biomedical waste management, specifically to the handlers thereof. This proposed system will apply the CNN model for better results and higher accuracy to classify biomedical waste into several groups, such as infectious, hazardous, and radioactive wastes, which is useful in effective waste management practices[5].

E-Waste Intelligent Robotic Technology: A Deep Learning Approach for Electronic Waste Detection, Classification, and Sorting: The paper proposes a deep learning-based robotic system called EIRT, which would sort electronic waste into categories. It covers the application of the EfficientNet-D2 architecture to detect and classify e-waste into different categories such as batteries, mobile phones, or electronic boards. EIRT is a robotic arm on a mobile car basis, autonomously finding, picking up, and sorting e-waste with an accuracy of 82.32%. This work develops the urgent need that addressing ewaste management requires with respect to automation and artificial intelligence means[6]. Utilizing waste size and weight, recyclable waste is categorized by material: This work shall adopt a classification method in regard to the weight and size of wastes from refuse using a system that employs a load cell and ultrasonic sensors. wider variety of waste types might need more databases[7].

Utilizing SIFT-PCA Feature Extraction and Support Vector Machine for Waste Classification: This paper focuses on the analysis of machine learning algorithms in identification of waste types especially with the use of Support Vector Machine

– SIFT-PCA feature extraction. This experiment focuses on the Trashnet dataset to evaluate the identified features and compare between the performances of using SVM with and without the PCA as the Reducing dimensionality. In case of using SIFT-PCA this algorithm decreased the mentioned accuracy compared to that achieved by only SIFT features. The highest accuracy was 62was achieved by using the SVM with SIFT only, moreover, the PCA-based approach decrease classification performance, thus the process of reducing dimensions should be able to maintain important points of features[8].

Waste Identification and Classification by the Use of CNN: This paper focus on the evaluation of the applied convoluted neural network for waste classification. When the works are compared with CNN model and SVM model the accuracy of the SVM is up to 94. 8while the CNN achieved an accuracy of 83the authors noted that CNNs holds promise for improvement if key issues of hyper parameter optimization are addressed. The used architectures are VGG16 and FastNet-34 among other standard architectures and the Kaggle preprocessed dataset contains 22564 images classified between recycled and organic wastes[9].

2.2 Deep Learning

Deep learning is a subset of machine learning where artificial neural networks, inspired by the human brain's structure, learn from vast amounts of data to perform tasks such as image and speech recognition, natural language processing, and decision-making. Deep learning models consist of interconnected layers of neurons that process data hierarchically, enabling them to uncover complex patterns and make accurate predictions.

2.3 Somedeeplearning methods

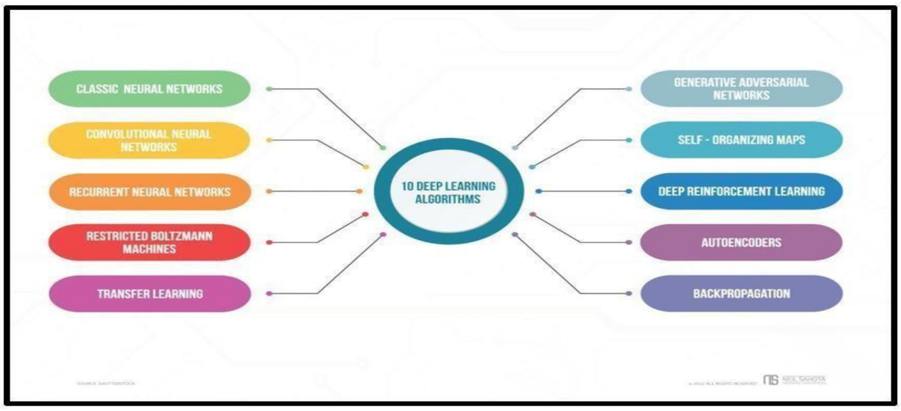

CLASSIC NEURAL NETWORKS
CONVOLUTIONAL NEURAL NETWORKS
RECURRENT NEURAL NETWORKS
RESTRICTED BOLTZMANN MACHINES
TRANSFER LEARNING
10 DEEP LEARNING ALGORITHMS
GENERATIVE ADVERSARIAL NETWORKS
SELF - ORGANIZING MAPS
DEEP REINFORCEMENT LEARNING
AUTOENCODERS
BACKPROPAGATION

Fig 2.3.1 Methods of Deep Learning

The diagram illustrates 10 Deep Learning Algorithms split into two categories. On the left, it lists Classic Neural Networks, Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs), widely used for tasks involving structured, image, and sequential data, respectively. It also includes Restricted Boltzmann Machines for feature learning and Transfer Learning to reuse pre-trained models.on the right, it highlights Generative Adversarial Networks (GANs) for synthetic data creation, Self-Organizing Maps (SOMs) for clustering, and Deep Reinforcement Learning for decision-making tasks. Additionally, Autoencoders handle data compression, while Backpropagation forms the foundation for training neural networks. These methods underpin modern AI advancements.[2]

Deep learning algorithms areoften categorized as supervised and unsupervised. **Supervised deep learning**, the algorithm learns from labeled data, where each example in the training dataset is associated with a corresponding label or target. The goal is to learn a mapping from inputs to outputs based on the labeled examples. Many popular deep- learning tasks, such as image classification, object detection, and sentiment analysis, fall under supervised learning. Deep learning algorithms like convolutional neural networks (CNNs) and recurrent neural networks (RNNs) are often used for supervised learning tasks.[3]

Unsupervised deep learning algorithms learn patterns and structures from unlabeled data without explicit supervision. The goal is to discover the underlying structure or distribution of the data. Unsupervised learning techniques include clustering, dimensionality reduction, and generative modeling. Deep learning algorithms like autoencoders and generative adversarial networks (GANs) are commonly used for unsupervised learning tasks such as feature learning, data compression, and generating synthetic data.[4]

Semi-supervised deep learning algorithms combine labeled and unlabeled data for training, leveraging the advantages of both datasets. They typically start with a small amount of labeled data and a larger pool of unlabeled data. By exploiting the inherent structure and relationships within the unlabeled data, semi-supervised algorithms aim to improve model generalization and performance. Techniques such as self-training, co-training, and pseudo- labeling are commonly used to iteratively refine models using both labeled and unlabeled data.[5]

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

1. MobileNetV2

MobileNetV2 is an efficient deep learning model designed for mobile and edge devices, balancing performance and computational efficiency. It utilizes depthwise separable convolutions, significantly reducing the number of parameters and operations compared to standard convolutional layers. One of its key innovations is the inverted residual block with linear bottlenecks, which allows for better gradient flow and improved feature extraction while keeping the model lightweight. This structure enables MobileNetV2 to achieve high accuracy with minimal computational cost, making it ideal for applications in real-time object detection, image classification, and embedded AI. Despite being lightweight, MobileNetV2 maintains competitive accuracy compared to heavier architectures, making it an excellent choice for on-device machine learning tasks.[1]

2. VGG16

VGG16 is a deep CNN with 16 layers, known for its simplicity and effectiveness in image classification tasks. The network follows a straightforward architecture, stacking 3×3 convolutional layers with ReLU activation and max pooling layers for downsampling. This sequential design allows for hierarchical feature learning, where lower layers capture basic edges and textures, while deeper layers recognize complex patterns. VGG16 is often used for transfer learning, where pre-trained weights from large datasets (like ImageNet) are fine-tuned on smaller datasets to improve performance. However, its major drawback is the large number .[2]

3. ResNet50

ResNet50 is a 50-layer deep CNN that introduced the Residual Network (ResNet) architecture, which solved the vanishing gradient problem in deep networks. It employs skip connections (identity mappings) that bypass certain layers, allowing gradients to flow more effectively during backpropagation. This enables the model to learn complex features without suffering from performance degradation as depth increases. The architecture consists of residual blocks, each containing convolutional layers, batch normalization, and ReLU activation. ResNet50 is widely used in image classification, object detection, medical imaging, and transfer learning due to its ability to generalize well across different datasets. Compared to VGG16, it achieves better accuracy with fewer parameters, making it computationally efficient despite being a deeper model.[3]

4. Transfer Learning

Transfer learning is a powerful deep learning technique where a model trained on a large dataset (e.g., ImageNet) is fine-tuned for a different but related task. Instead of training a model from scratch, which requires massive amounts of data and computation, transfer learning allows for knowledge transfer, leveraging pre-trained feature representations. This technique is particularly useful when dealing with limited labeled data, as the pre-trained model has already learned rich, hierarchical features from a diverse dataset.[4]

5. DenseNet-121

DenseNet-121 (Dense Convolutional Network) is a deep learning architecture that enhances feature propagation and reduces redundancy by introducing dense connections between layers. Unlike traditional CNNs, where each layer receives input only from the previous layer, DenseNet allows each layer to be connected to every other layer, ensuring that feature maps learned earlier are reused in later layers.[5]

6. DenseNet-169

DenseNet-169 follows the same densely connected design as DenseNet-121 but with a deeper network consisting of 169 layers. The increased depth allows the model to capture more complex patterns while still maintaining efficient parameter usage through feature reuse. This model excels in scenarios where high-resolution details and deep hierarchical feature extraction are crucial, such as biomedical imaging, remote sensing, and high-precision industrial inspection. Due to its extensive connectivity, DenseNet-169 achieves better performance than shallower models without significantly increasing computational cost. However, training such a deep model requires high-quality data augmentation and optimized learning rate scheduling to prevent overfitting and ensure smooth convergence.[6]

7. VGG-19

VGG-19 is a deep convolutional neural network with 19 layers, known for its simple yet powerful architecture. It uses stacked 3×3 convolutional layers followed by max pooling, allowing it to learn hierarchical visual features effectively. Nevertheless, VGG-19 is widely used for transfer learning, as its pre-trained weights on ImageNet make it an excellent feature extractor for custom image datasets. This model is particularly effective in object recognition, scene understanding, and artistic style transfer applications.[8]

8. InceptionV3

Inception V3 is a deep convolutional neural network (CNN) designed for high-performance image classification, known for its efficient architecture that balances accuracy and computational cost. Developed by Google, it incorporates techniques such as factorized convolutions, asymmetric convolutions, auxiliary classifiers, and extensive batch normalization to improve training stability and reduce overfitting. The model's mixed architecture combines 1×1, 3×3, and 5×5 convolutions in parallel, allowing it to capture multi-scale spatial features effectively. With 42 layers, Inception V3 is deeper than its predecessors, making it capable of learning complex patterns from large-scale datasets like ImageNet. It is widely used in applications such as medical imaging, autonomous vehicles, and satellite image analysis due to its ability to process high-resolution images with lower computational demand compared to models like VGG-19. Additionally, it excels in transfer learning, where pre-trained weights can be fine-tuned for specialized tasks, reducing the need for extensive labeled data.[8]

9. Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised learning algorithm primarily used for classification and regression tasks. It works by finding the optimal hyperplane that maximally separates different classes in the feature space. If the data is not linearly separable, kernel functions such as polynomial, radial basis function (RBF), and sigmoid are used to map data into a higher-dimensional space where separation becomes possible.[9]

10. Random Forest

Random Forest is an ensemble learning method based on multiple decision trees, where each tree is trained on a random subset of data and features. The final prediction is made by averaging the outputs of all trees (for regression) or through majority voting (for classification). This technique significantly reduces overfitting, which is a common issue with single decision trees, while improving generalization and robustness. Random Forest is widely used in finance, healthcare, fraud detection, and customer behavior analysis due to its high accuracy, interpretability, and ability to handle both numerical and categorical data. However, it may require careful tuning of hyperparameters, such as the number of trees, to balance accuracy and efficiency.[10]

11. Convolutional Neural Networks (CNNs)

CNNs are a class of deep learning models designed specifically for image recognition and processing. They consist of multiple layers, including convolutional layers, pooling layers, and fully connected layers, that automatically extract spatial and hierarchical features from images. The convolutional layers apply filters to detect edges, textures, and patterns, while pooling layers reduce dimensionality to enhance computational efficiency. CNNs, including architectures like ResNet, MobileNet, and EfficientNet, outperform traditional machine learning models in handling complex image datasets. They are extensively used in medical diagnostics, autonomous vehicles, and facial recognition. However, CNNs require large labeled datasets and high computational power, making them resource-intensive compared to simpler models like

12. EfficientNet-D2

EfficientNet-D2 is part of the EfficientNet family, a set of deep learning models designed to maximize accuracy while minimizing computational cost. Unlike traditional CNN architectures that scale depth, width, or resolution separately, EfficientNet uses a compound scaling method to balance all three dimensions efficiently. [1]

13. AlexNet:

The ILSVRC 2012 winner, AlexNet is a deep CNN which features five layers of convolution and three layers of fully connected layers ReLU nonlinearity. The most important one are the max pooling,dropout for avoiding the overfitting and overlapping filters to help with feature extraction. It was one of the first models to utilize GPU resulting in higher levels of classification.[3]

3.1.1 Disadvantages of Existing System

Limited Dataset Categories: Public datasets contain a limited number of waste categories, reducing the model's ability to generalize across diverse household waste items. This leads to misclassification when new waste types are encountered[1]. Computational Constraints with Raspberry Pi: While Raspberry Pi provides an affordable real-time implementation, its limited

Class Disparity Issues: The dataset suffers from class imbalance, where some waste types have fewer images. This can cause poor classification performance on underrepresented classes despite data augmentation efforts【2】. High Computational Requirements: CNN models such as DenseNet, InceptionV3, and Xception require significant computational power, making them less feasible for resource-constrained environments【2】.

Class Imbalance Limitations: Despite using augmentation and oversampling, the dataset remains imbalanced, which can lead to biased classification results【3】. Dependence on TrashNet Dataset: The study relies on TrashNet, which has limited diversity and does not represent all real-world waste conditions, affecting model generalization. The model requires extensive computational resources due to the deep learning and optimization techniques used.The dataset usedhich may affect the generalization of the model to real-world waste classification scenarios. 【3】.

Machine Learning Models Struggle with Complexity: Traditional machine learning models, such as Support Vector Machines (SVM), Decision Trees, and Random Forests, lack the capability to effectively handle complex image data compared to deep learning approaches【4】.

Limited Real-World Adaptability: The biomedical waste classification model struggles with variations in lighting, occlusions, and contamination, reducing its practical applicability【5】. Health and Safety Risks: The automated classification system may misclassify hazardous biomedical waste, potentially putting waste handlers at risk【5】.

High Implementation Cost: The robotic system requires expensive components, such as robotic arms, deep learning models, and specialized sensors, making it impractical for widespread adoption【6】. Challenges in Handling Irregular Waste: E-waste items can have irregular shapes and textures, making classification and sorting more difficult compared to well-structured waste categories【6】.

Limited Classification Features: The classification is based only on weight and size, which is not always a reliable method for distinguishing between similar waste materials【7】.

Feature Loss Due to PCA: While PCA reduces dimensionality, it may also remove crucial features necessary for accurate classification, leading to lower performance【8】. Inferior Performance Compared to Deep Learning Models: SVM combined with SIFT-PCA achieved an accuracy of only 62%, which is significantly lower than CNN-based models, indicating its limitations in handling complex image data【8】.

Hyperparameter Optimization Challenges: The effectiveness of CNN models depends on careful tuning of hyperparameters such as learning rate, batch size, and filter size, which can be complex and time-consuming【9】. Lower Accuracy Compared to SVM: In this study, SVM achieved a higher accuracy (94.8%) compared to CNN models

(83%), indicating that CNNs may require further optimization to outperform traditional classifiers Hyperparameter optimization remains a challenge, limiting the potential of CNN models. The dataset used in the study does not adequately cover all types of waste found in real-world environments.【9】.

3.2 PROPOSED SYSTEM

The proposed deep learning system for waste classification introduces a modern, technology- driven approach to efficiently and accurately manage waste. It leverages advanced tools like artificial intelligence (AI), machine learning (ML), and deep learning models such as InceptionV3 to process waste images through its advanced convolutional layers, capturing detailed features to classify waste into categories like recyclable, non-recyclable, organic, and hazardous.

A user-friendly interface allows real-time waste classification and sorting, making the system suitable for both small-scale and industrial waste management. With scalability and continuous learning capabilities, the system adapts to new waste types, automating the process and overcoming the inefficiencies of traditional manual methods. This innovative approach enhances waste management, reduces human intervention, and promotes sustainability**.**This approach minimizes errors, enhances real-time waste management, and contributes to more sustainable waste disposal process.

Title:

Pushing the Limits: Accuracy and Efficiency Gains in Waste Classification Through Continuous Learning and Advanced Techniques

Objectives:

1. Continuously improve the accuracy of the waste classification model:

This objective focuses on developing methods to enhance the model's ability to correctly classify waste items. Techniques like hyperparameter tuning and transfer learning can be employed to achieve this goal.

2. Enhance the efficiency of the model:

This objective aims to optimize the model's performance in terms of processing speed and resource consumption. Optimizing algorithms and exploring hardware acceleration techniques are potential approaches to achieve this.

3. Investigate the integration of advanced deep learning architectures:

This objective emphasizes exploring more sophisticated deep learning models, such as ensemble methods, to potentially improve the overall performance of the waste classification system.

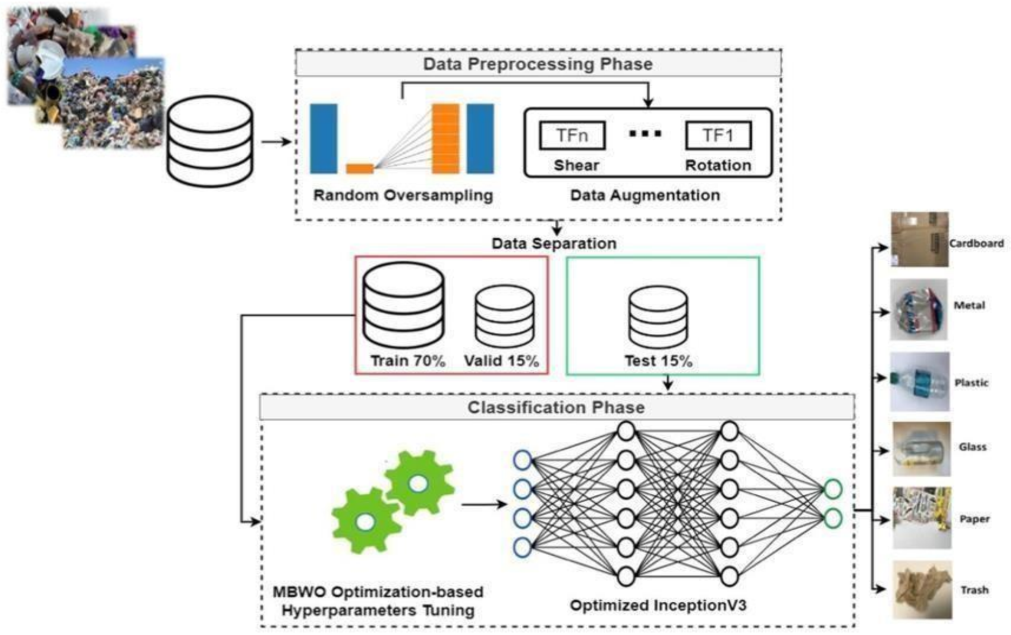


Fig 3.2.1 Flow Chart of Proposed System

The process depicted in the flowchart outlines the key stages of developing and deploying a hybrid machine learning model. It begins with **Dataset Collection**, where data is gathered from various sources such as open datasets, APIs, sensors, or proprietary sources. This step ensures that the data is sufficient in size and relevance to meet the project's objectives. The waste classification process begins with the Data Preprocessing Phase, where raw waste images are collected and stored in a database. To handle class imbalances, random oversampling is performed, ensuring that underrepresented waste categories receive adequate representation in the dataset. Additionally, data augmentation techniques such as shear and rotation transformations are applied to enhance the dataset and improve model generalization.

Following preprocessing, the Data Separation phase involves splitting the dataset into three subsets: 70% for training, 15% for validation, and 15% for testing. The training set is used to train the model, the validation set helps in fine-tuning hyperparameters to avoid overfitting, and the test set evaluates the final model's performance.

In the Classification Phase, the Multi-Objective Beluga Whale Optimization (MBWO) algorithm is employed for hyperparameter tuning, ensuring that the deep learning model achieves optimal performance. The **InceptionV3 model**, which is optimized using MBWO, is then trained to classify waste into six categories: Cardboard, Metal, Plastic, Glass, Paper, and Trash. This structured approach enhances the accuracy and efficiency of waste classification, contributing to improved waste management solutions.

3.3 FEASIBILITY STUDY

Waste management is a critical challenge that requires efficient classification and sorting methods to reduce environmental pollution. The integration of artificial intelligence (AI), particularly deep learning-based techniques, has shown significant potential in automating household waste classification. This feasibility study evaluates the implementation of a smart household waste classification system using convolutional neural networks (CNNs) and transfer learning approaches. The study is based on the existing literature and research findings from various sources [1-9].

3.3.1 Technical Feasibility

3.3.1.1 Deep Learning Approaches

The proposed system will utilize deep learning models such as MobileNet, VGG16, ResNet50, InceptionV3, DenseNet, and Xception for accurate waste classification. Research has demonstrated that these models, especially InceptionV3, achieve high accuracy rates (up to 97.75%) when used with optimized parameters such as dropout and batch size [3]. Moreover, transfer learning enables the system to perform well despite the limitations of public datasets by leveraging pretrained models [1].

3.3.1.2 Dataset and Preprocessing

Existing datasets such as TrashNet provide a limited number of waste categories, but studies have improved upon this by expanding the dataset to include over 15,515 images across twelve categories [2]. Data augmentation and oversampling techniques can further improve model generalization and reduce class imbalance issues [3].

3.3.1.3 Real-Time Implementation

For real-time waste classification, the system can be deployed on a Raspberry Pi, making it an economical and scalable solution for smart bins [1]. The integration of sensors such as ultrasonic and load cell sensors can further enhance classification based on physical attributes like size and weight [7].

3.3.2 Economic Feasibility

3.3.2.1 Cost Analysis

Compared to traditional waste sorting methods that rely on manual labor, AI-based classification reduces operational costs in the long run. The use of affordable hardware like Raspberry Pi and edge computing reduces dependency on expensive cloud computing resources [1]. However, initial costs for dataset collection, model training, and hardware setup must be considered**.**

3.3.2.2 Return on Investment

Automated waste sorting reduces the cost of recycling by minimizing contamination of recyclable materials. Efficient classification can lead to improved recycling rates, reducing landfill waste and associated disposal costs. Additionally, automated systems reduce reliance on manual labor, leading to long-term savings [4].

3.3.3 Operational Feasibility

3.3.3.1 System Integration

The system can be integrated with existing waste management frameworks, including municipal and household waste collection systems. AI-powered smart bins equipped with image recognition can automate sorting and segregation at the source [1].

3.3.3.2 Accuracy and Performance

CNN-based models achieve high classification accuracy, with some models such as Xception reaching up to 89.57% on augmented datasets [2]. However, optimization of hyperparameters and addressing overfitting issues are necessary to ensure reliable real-time performance [9].

3.3.4 Challenges and Mitigation Strategies

Dataset Limitations: Waste classification faces challengesd due to limited and datasets like TrashNet, restricting model generalization. Data augmentation (rotation, flipping, brightness adjustments) and collecting diverse real-world waste images enhance dataset quality, improving adaptability across different waste scenarios [3].

Class Imbalance: Disproportionate representation of waste categories leads to biased model

predictions, affecting classification accuracy. Mitigation strategies include oversampling minority classes, SMOTE (Synthetic Minority Over-sampling Technique), weighted loss functions, and targeted data augmentation to improve balance[3].

Processing Speed: Real-time classification requires efficient computation, but high-performance servers may not always be feasible. Deploying lightweight AI hardware like Raspberry Pi, NVIDIA Jetson, and Edge TPU enables on-device inference, while model optimization techniques (quantization, pruning, knowledge distillation)

4. SYSTEM REQUIREMENT

4.1 Hardware Requirements

The hardware setup plays a critical role in ensuring efficient execution of deep learning algorithms and model training, particularly for handling large MRI datasets.

•	Processor	: Intel(R)Core(TM)i31005G1CPU@1.20GHz
•	System Type	: 64-bit operating system, x64-based processor
•	Cache memory	: 4MB(Megabyte)
•	RAM	: 8GB (gigabyte)

To achieve optimal performance, the hardware setup must align with the computational demands of deep learning model training and inference. The system requires a processor, sufficient RAM, and access to cloud-based GPU services to accelerate training and classification tasks.

For the development environment, the user has utilized an Intel Core i3 processor, Windows 11 operating system, and Google Colab Pro for training and inference. Google Colab provides free access to GPUs, making it an efficient platform for training deep learning models without requiring high-end local hardware.[2]

For optimal performance in local execution, a system with an Intel Core i5 (10th Gen or later) or AMD Ryzen 5 (3000 series or later) processor, 16GB DDR4 RAM, and an NVIDIA GTX 1650 GPU with 4GB VRAM is recommended. A storage configuration of a 256GB SSD and a 1TB HDD ensures smooth data processing and storage management. A stable internet connection is necessary for downloading datasets and utilizing cloud-based training resources.[2,3]

4.2 Software Requirements:

•	Operating System	: Windows 11, 64-bit Operating System
•	Coding Language	: Python
•	Python distribution	: Google Colab Pro, Flask
•	Browser	: Any Latest Browser like Chrome

The system requires a stable and efficient operating environment to support deep learning frameworks and machine learning libraries. It is compatible with Windows 10/11 (64-bit) and Ubuntu 20.04+ (preferred for deep learning model deployment).

The system relies on various libraries and frameworks to implement machine learning and deep learning models. TensorFlow 2.x and Keras are the primary frameworks for training and deploying deep learning models. PyTorch 1.10+ can be used as an alternative deep learning framework. Scikit-Learn supports machine learning utilities, including the Extreme Learning Machine (ELM) classifier. OpenCV 4.x is integrated for image preprocessing, while NumPy and Pandas handle data manipulation and processing.[4]

4.3 Software Description

The development environment is tailored for deep learning research and implementation. Google Colab Pro is the primary platform used for interactive development and experimentation. Jupyter Notebook is also an alternative option for running code in a notebook environment. PyCharm and VS Code serve as integrated development environments (IDEs) for coding and debugging.

4.3 Implementation Details

•Google Colab: Used as the primary development platform with free access to GPUs. Supports Python, TensorFlow, and Keras.

•Python: The core programming language used for data preprocessing, model training, and deployment.

•TensorFlow: The primary deep learning framework used for building and training models.

•Keras: A high-level API running on top of TensorFlow to simplify model creation and training.

•OpenCV: Used for image preprocessing tasks like resizing, augmentation, and normalization.

•NumPy & Pandas: Used for data manipulation, handling arrays, and dataset processing.

•Flask: Used for developing RESTful APIs for model deployment and integration.[8]

4.4 Programming Languages

• Python (3.x): Serves as the core programming language due to its extensive support for deep learning frameworks and [2]data science libraries.

• HTML & CSS: Used to develop an intuitive and visually appealing user interface for displaying MRI classification results.[3]

• Flask: A lightweight web framework used to create APIs and deploy the trained deep learning model as a[4] web application for remote accessibility.[9]

5. SYSTEM DEIGN

5.1 SYSTEM ARCHITECTURE

The intelligent waste classification system employs deep learning, computer vision, and optimization techniques to automate the sorting of waste materials, reducing human intervention and improving accuracy. InceptionV3, a powerful convolutional neural network (CNN), is used to extract hierarchical features from waste images, enabling precise classification into categories such as recyclable, non-recyclable, organic, and hazardous. Unlike traditional classification methods, which rely on handcrafted features, this system automates feature extraction through deep learning, leading to more reliable and scalable waste management solutions.[1]

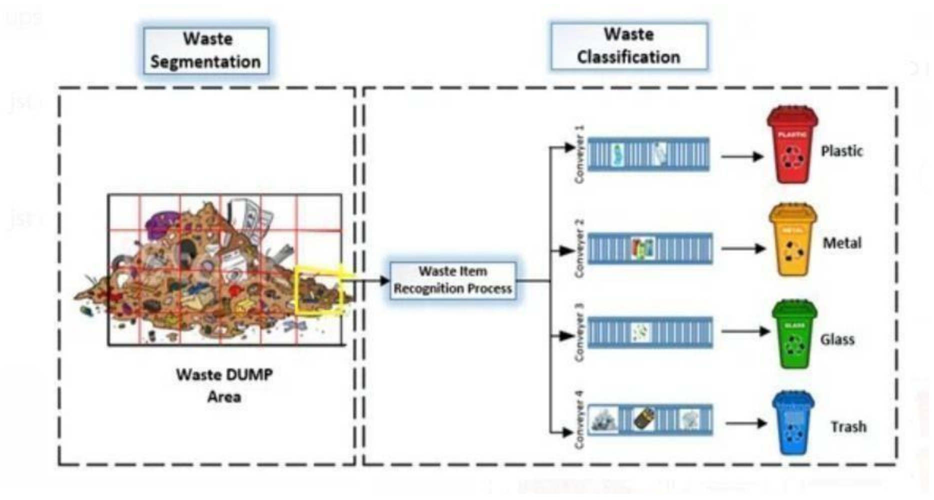


Fig 5.1.1 Waste Classification Architecture

5.1.1 Dataset Description

The TrashNet dataset is widely used for waste classification tasks and consists of 2,573 images categorized into six distinct waste types: cardboard, glass, metal, paper, plastic, and trash. These images are captured at a resolution of 512 × 384 pixels, providing sufficient detail for deep learning-based classification. However, a significant challenge with this dataset is the presence of class imbalance, where certain waste categories have more samples than others. This imbalance can lead to biased model predictions, where the classifier tends to favor classes with a higher number of images while underperforming on underrepresented classes.[2]

To address the class imbalance issue, the dataset was split into 70% training, 15% testing, and 15% validation. Additionally, data augmentation techniques such as rotation, flipping, scaling, and brightness adjustments were applied to artificially increase the number of images, improving the model's ability to generalize across different waste types. Furthermore, oversampling techniques were employed to ensure that underrepresented classes had a more balanced presence in the training dataset. This approach helps mitigate bias and enhances the model’s ability to recognize waste categories more accurately.[3]

LABLE	LABLE NAME	TOTAL SAMPLES
(1)	Trash	250
(2)	Plastic	750
(3)	Paper	345
(4)	Metal	196
(5)	Glass	556
(6)	Cardboard	476
	Total Samples	2573

Table 5.1.2 Dataset Description

5.1.2 Data Pre-processing

Dataset Preparation & Image Processing

The TrashNet dataset is used as the primary dataset, containing images of six waste categories: Cardboard, Glass, Metal, Paper, Plastic, and Trash.
Each image is resized to 299×299×3 pixels to match the required input size of the InceptionV3 model.
The images are converted into a numerical format suitable for deep learning processing.[1]

Data Augmentation

To improve model generalization and prevent overfitting, various augmentation techniques were applied:
Flipping: Random horizontal and vertical flips enhance dataset variability.
• **Rotation:** Random rotations help the model learn different waste orientations.
• **Contrast Adjustments**: Improves visibility of waste materials.
• **Zooming:** Helps the model focus on different waste sizes. [2]
Unplanned Oversampling: This is going to solve the issue of class disparity by

giving more representation of the underrepresented classes, that is, the trash classes. Therefore, the model will not be biased towards those classes which dominate the training[3].

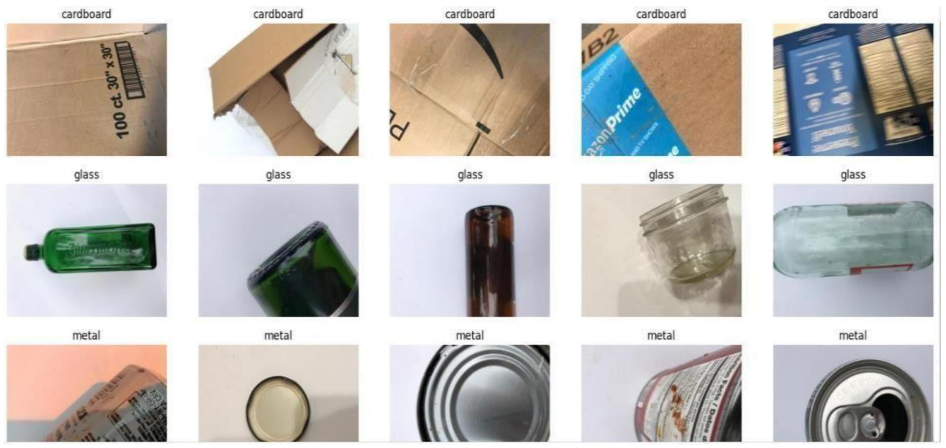


Fig 5.1.3 Data Pre-processing

5.2 Model building

Hybrid Model Architecture

MobileNetV2, VGG16, InceptionV3, and AlexNet each offer unique strengths for image classification. MobileNetV2 efficient for resource-constrained environments, VGG16 excels in detailed feature extraction with a deep structure, InceptionV3 captures diverse features using inception modules, and AlexNet provides strong pattern learning with its pioneering CNN design. Together, these models can be integrated into a hybrid architecture to enhance performance andfeature extraction.[4]

MobileNetV2:

• **Description:** MobileNetV2 is a lightweight deep learning model designed for efficient image classification, optimized for mobile and embedded applications.

• **Key Features:**

- **o** Uses depthwise separable convolutions to reduce computation.
- **o** Employs inverted residual blocks for improved efficiency.
- **o** Achieved **97.84% accuracy** in waste classification.
- **o** Balances high accuracy with low computational cost.

InceptionV3:

• **Description:** InceptionV3 is a deep convolutional neural network optimized for image classification, using factorized convolutions to reduce computational complexity.

• **Key Features:**

o Utilizes asymmetric convolutions and factorization for efficiency.

o Effectively captures both local and global features in images.

o Achieved **99.70% accuracy** in waste classification.

o Reduces overfitting with auxiliary classifiers and label smoothing**.**

Vgg16:

• **Description:** VGG16 is a deep convolutional neural network known for its simplicity and uniform architecture, consisting of 16 layers.

• **Key Features:**

o Uses small 3×3 filters for deep feature extraction.

o Provides strong baseline performance for image classification.

o Achieved **85.43% accuracy** in waste classification.

o Requires high computational resources due to its deep structure.

AlexNet

• **Description:** AlexNet is one of the pioneering deep learning architectures, consisting of 8 layers and optimized for large-scale image classification.

• **Key Features:**

o Uses ReLU activation for faster training convergence.

o Employs dropout to reduce overfitting in deep networks.

o Achieved **57.61% accuracy** in waste classification.

o Outperformed traditional machine learning methods but is now considered less efficient than modern architectures.

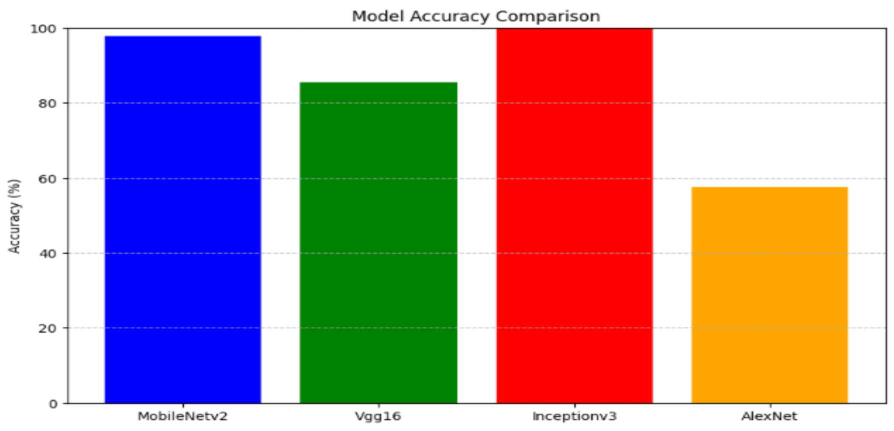


Figure 5.2.1 Comparison of Model Performances for waste classification

The bar graph titled **"Model Accuracy Comparison"** presents a comparison of the accuracy percentages of four different deep learning models used for waste classification.

Key Observations from the Graph:

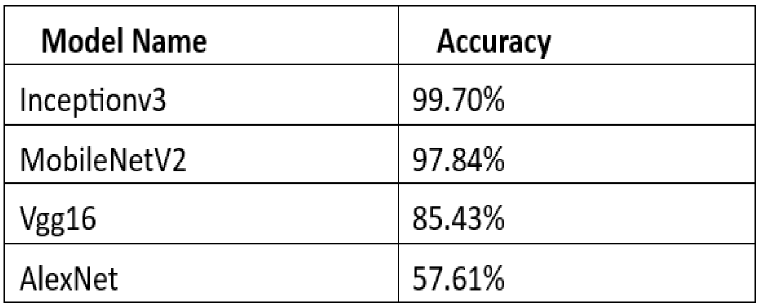
1. **InceptionV3 (Red Bar)** has the highest accuracy, reaching nearly **100%**, making it the most effective model among the four.
2. **MobileNetV2 (Blue Bar)** follows closely behind with a slightly lower accuracy, around **97-98%**, indicating it is also a strong model.
3. **VGG16 (Green Bar)** achieves an accuracy of approximately **85%**, making it a decent model but less accurate than the previous two.
4. **AlexNet (Orange Bar)** has the lowest accuracy, around **57-60%**, suggesting it is not as effective for waste classification compared to the other models.[5]

Training Process:

The training process for InceptionV3 involves resizing input images to 299x299 pixels and normalizing them for efficient learning. The model is trained using supervised learning with a categorical cross- entropy loss function and an Adam optimizer with a learning rate of 0.001. Data augmentation techniques like rotation, flipping, and scaling are applied to prevent overfitting and improve generalization. The model is trained for a set number of epochs (e.g., 50) with early stopping based on validation loss, ensuring optimal performance on unseen data**.[6]**

5.2.2 MODEL ACCURACY TABLE

The table shows results for four models-where four different deep learning methods were used on classification. The highest result accuracy for InceptionV3 is 99.70%, while MobileNetV2 has 97.84%. VGG16 moderately performs with an accuracy of 85.43%, while AlexNet shows a low amount of performance with an accuracy of 57.61%. That indicates the better performance of the InceptionV3 and MobileNetV2.



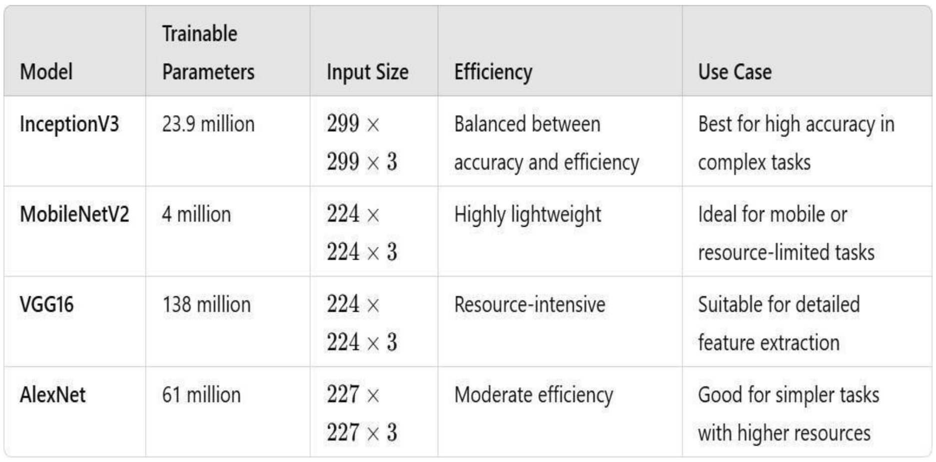
Model Name	Accuracy
Inceptionv3	99.70%
MobileNetV2	97.84%
Vgg16	85.43%
AlexNet	57.61%

Table 5.2.2 Model Performance Comparision

5.2.3 Model Summary

Waste classification models vary in complexity and efficiency, each with trade-offs. InceptionV3 (23.9M parameters, 299×299×3 input) offers high accuracy with balanced computational efficiency. MobileNetV2 (4M parameters) is lightweight and optimized for mobile use. In contrast, VGG16 (138M) and AlexNet (61M) extract detailed features but demand high computational resources, limiting scalability. Model selection depends on application needs, constraints, and performance goals.

InceptionV3's balance of accuracy and efficiency makes it a strong choice for real-time waste classification. Meanwhile, MobileNetV2 is ideal for edge devices with limited processing power.



Model	Trainable Parameters	Input Size	Efficiency	Use Case
InceptionV3	23.9 million	$299 \times 299 \times 3$	Balanced between accuracy and efficiency	Best for high accuracy in complex tasks
MobileNetV2	4 million	$224 \times 224 \times 3$	Highly lightweight	Ideal for mobile or resource-limited tasks
VGG16	138 million	$224 \times 224 \times 3$	Resource-intensive	Suitable for detailed feature extraction
AlexNet	61 million	$227 \times 227 \times 3$	Moderate efficiency	Good for simpler tasks with higher resources

Table 5.2.3 Comparision of Model Parameter

5.3 Uml Diagrams

5.3.1 Uml Diagram for MobileNetV2

MobileNetV2 is a deep learning architecture optimized for mobile and embedded systems. It uses inverted residual blocks, a linear bottleneck, and depthwise separable convolutions to reduce complexity while maintaining accuracy. These features enable real-time performance on resource-limited devices.

The diagram illustrates MobileNetV2’s feature extraction using stride = 1 and stride = 2 blocks. Each block includes a 1×1 convolution (ReLU6), a 3×3 depthwise convolution, and a 1×1 linear projection. The stride = 1 block preserves spatial dimensions with residual connections, while the stride = 2 block reduces spatial size. Extracted features are then fed into a classification model for efficient image recognitio

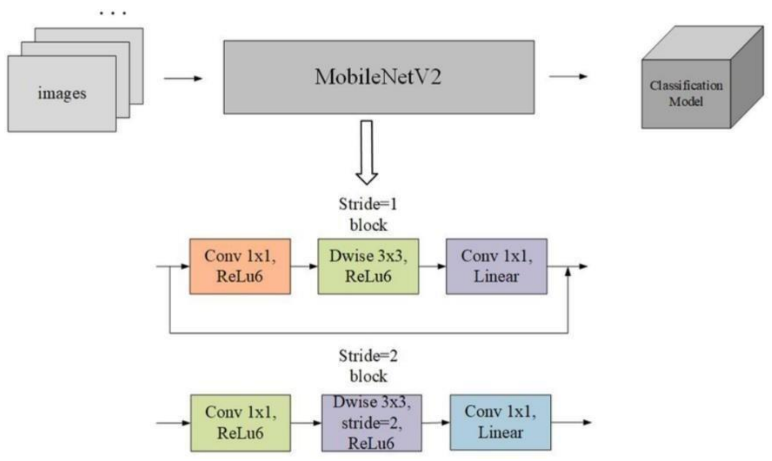


Fig 5.3.1.1 MobileNetV2

5.3.2 Uml Diagram for InceptionV3

The diagram illustrates the architecture of InceptionV3, a deep learning model designed for image classification. It begins with an input layer (299x299x3), followed by a "Stem" block responsible for initial feature extraction. The network then passes through multiple Inception-A modules, Reduction-A layers, and Inception-B modules, enhancing feature representation.

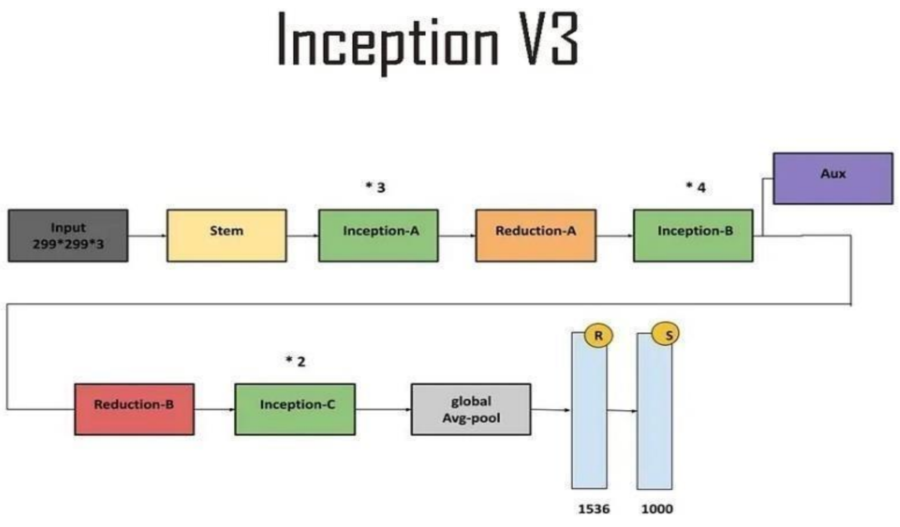


Fig 5.3.2.1 InceptionV3

5.3.3 Uml Diagram for Vgg16

VGG-16 is a deep convolutional neural network with 16 layers, including 13 convolutional and 3 fully connected layers. It uses small 3×3 filters for feature extraction, followed by max pooling layers to reduce spatial dimensions. The final dense layers process extracted features for classification.

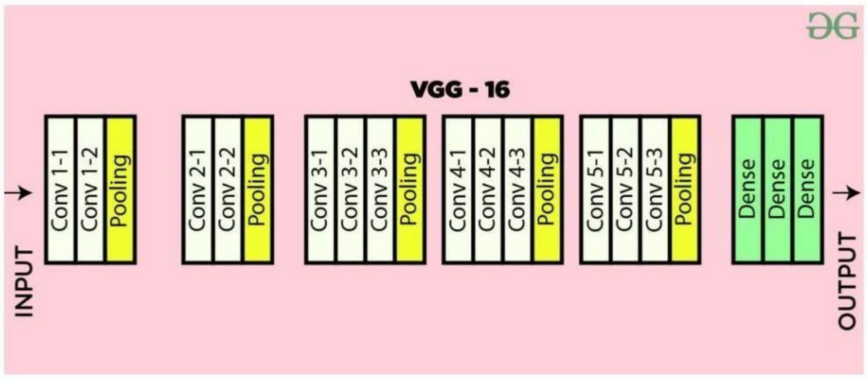


Fig 5.3.3.1 Vgg16

5.3.4 Uml Diagram for AlexNet

The diagram illustrates the AlexNet architecture, a deep CNN for image classification. It processes 227 × 227 × 3 input images through five convolutional layers, extracting hierarchical features. As the image passes through layers, the spatial size reduces while depth increases, enabling complex pattern recognition. By Conv5, the feature map is 13 × 13 × 256.

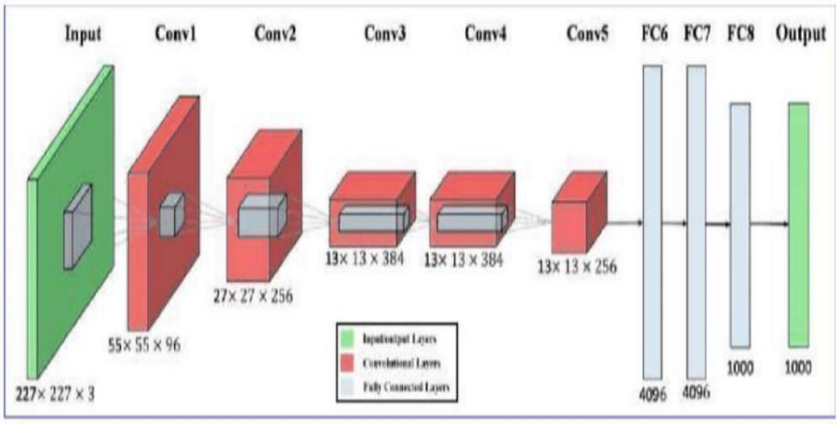


Fig 5.3.4.1 AlexNet

6 .IMPLEMENTATION

6.1 CODING

Mounting Google Drive and Listing Dataset Directory Contents

```
from google.colab import drive
drive.mount('/content/drive')
!ls "/content/drive/My Drive/Trash Net"
```

1. Data Preprocessing

1.1 Resizing the images

```
import tensorflow as tf
from tensorflow.keras.applications.inception_v3 import preprocess_input
from tensorflow.keras.preprocessing.image import ImageDataGenerator
            # Define the image size
img_width, img_height = 299, 299
            # Create an ImageDataGenerator object
train_datagen = ImageDataGenerator(
    preprocessing_function=preprocess_input,
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)

# Load the Trash Net dataset
train_generator = train_datagen.flow_from_directory(
    '/content/drive/My Drive/Trash Net',
    target_size=(img_width, img_height),
    batch_size=32,
    class_mode='categorical'
)
```

1.2. Data Augmentation

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing.image import load_img, img_to_array
import os
# Define the image data generator
```

```
 datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    vertical_flip=True)
# Define the list of classes - this is what was missing!
classes = ["cardboard", "glass", "metal", "paper", "plastic", "trash"]
# Correct the path to your oversampled directory
oversampled_dir = "/content/drive/My Drive/Trash Net"
# Generate augmented data
for class_name in classes:
  class_dir = os.path.join(oversampled_dir, class_name)
  # Check if the directory exists before trying to list its contents
  if os.path.exists(class_dir):
    for image_name in os.listdir(class_dir):
      image_path = os.path.join(class_dir, image_name)
      image = load_img(image_path)
      x = img_to_array(image)
      x = x.reshape((1,) + x.shape)
      datagen.fit(x)
      for i, augmented_image in enumerate(datagen.flow(x, batch_size=1,
save_to_dir=class_dir, save_prefix=f"{image_name.split('.')[0]}_augmented",
save_format="jpg")):
        break
  else:
    print(f"Warning: Directory not found: {class_dir}")
```

1.3. Oversampling

```
import os
import random
# Define the path to your Trash Net dataset
dataset_path = "/content/drive/My Drive/Trash Net"
# Define the percentage of images to keep
```

```
keep_percentage = 0.2  # Keep 20% of the images
# Iterate through each subfolder (class) in the dataset
for class_name in os.listdir(dataset_path):
  class_path = os.path.join(dataset_path, class_name)
  if os.path.isdir(class_path):
# Get a list of all image files in the subfolder
 image_files = [f for f in os.listdir(class_path) if f.endswith(('.jpg', '.jpeg', '.png'))]
# Calculate the number of images to keep
  num_images_to_keep = int(len(image_files) * keep_percentage)
# Randomly select images to keep
 images_to_keep = random.sample(image_files, num_images_to_keep)
 # Delete the remaining images
    for image_file in image_files:
      if image_file not in images_to_keep:
        image_path = os.path.join(class_path, image_file)
        os.remove(image_path)
        print(f"Removed: {image_path}")
```

1.4. Plot the imges

```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import random
Define the path to the temporary directory
temp_dir = "/content/preprocessed_data"
# Define the number of images to plot
num_images = 5
# Loop through each class
for class_name in classes:
  # Get the path to the class directory
  class_dir = os.path.join(temp_dir, class_name)
 # Get a list of image files in the class directory
 image_files = os.listdir(class_dir)
# Select a random sample of images
  selected_images = random.sample(image_files, num_images)
# Plot the images
  fig, axes = plt.subplots(1, num_images, figsize=(20, 5))
  for i, image_name in enumerate(selected_images):
  image_path = os.path.join(class_dir, image_name)
  img = mpimg.imread(image_path)
  axes[i].imshow(img)
  axes[i].set_title(class_name)
  axes[i].axis('off')
plt.show()
```

2. Spilt The TrashNet Dataset

```
from imblearn.over_sampling import SMOTE
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import numpy as np
import os # Import the os module
from sklearn.model_selection import train_test_split
            # Define the data generator with augmentation
datagen = ImageDataGenerator(
    rotation_range=10,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    vertical_flip=True
)
            # Mount Google Drive (if not already mounted)
# You may need to authenticate with your Google account
from google.colab import drive
drive.mount('/content/drive')
# Verify the path exists
data_dir = '/content/drive/My Drive/Trash Net'
if not os.path.exists(data_dir):
    raise ValueError(f"Directory not found: {data_dir}")
# Load the data from the Trash Net folder
# Setting shuffle to False to maintain order for splitting
data = datagen.flow_from_directory(
    data_dir, # Use the verified path
    target_size=(299, 299),
    batch_size=32,
    shuffle=False # Important for consistent splitting
)
# Initialize lists to store data and labels
X_data = []
y_data = []
# Iterate through the data generator to collect all data and labels
for i in range(len(data)):
    batch_x, batch_y = data.next()
    X_data.append(batch_x)
    y_data.append(batch_y)
# Concatenate the data and labels from all batches
X_data = np.concatenate(X_data, axis=0)
y_data = np.concatenate(y_data, axis=0)
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_data, y_data, test_size=0.2,
random_state=42)
# Print the shapes of the training and test sets
print(f"Shape of X_train: {X_train.shape}")
print(f"Shape of X_test: {X_test.shape}")
print(f"Shape of y_train: {y_train.shape}")
print(f"Shape of y_test: {y_test.shape}")
```

Train the Model MobileNetV2

```
import tensorflow as tf
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense, Dropout
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report
import numpy as np
import seaborn as sns
```

```
# Load and preprocess the TrashNet dataset
data_dir = '/content/drive/My Drive/Trash Net'
# Define image data generators for training and validation with advanced
augmentation
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=30,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest',
    validation_split=0.2
)
train_generator = train_datagen.flow_from_directory(
    data_dir,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='training'
)
validation_generator = train_datagen.flow_from_directory(
    data_dir,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation'
)
# Define the MobileNetV2 model with custom top layers
base_model = MobileNetV2(input_shape=(224, 224, 3), include_top=False,
weights='imagenet')
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dropout(0.5)(x)  # Add Dropout layer for regularization
```

```
predictions = Dense(train_generator.num_classes, activation='softmax')(x)
model = Model(inputs=base_model.inputs, outputs=predictions)
# Unfreeze some layers in the base model for fine-tuning
for layer in base_model.layers[-20:]:  # Unfreeze the last 20 layers
    layer.trainable = True
# Compile the model
model.compile(optimizer=Adam(learning_rate=0.0001),
loss='categorical_crossentropy', metrics=['accuracy'])
# Define callbacks
early_stopping = EarlyStopping(monitor='val_loss', patience=10,
restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=5,
min_lr=1e-6)
# Train the model with limited steps per epoch
history = model.fit(
    train_generator,
    steps_per_epoch=64, # Limit to 64 steps per epoch
    epochs=50,
    validation_data=validation_generator,
    validation_steps=64,  # Limit validation to 64 steps as well
    callbacks=[early_stopping, reduce_lr]
)
# Plot the training and validation accuracy and loss
def plot_training_history(history):
    # Plot accuracy
    plt.figure(figsize=(12, 4))
    plt.subplot(1, 2, 1)
    plt.plot(history.history['accuracy'], label='Training Accuracy')
    plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
    plt.title('Training vs Validation Accuracy')
    plt.xlabel('Epoch')
    plt.ylabel('Accuracy')
    plt.legend()
   # Plot loss
    plt.subplot(1, 2, 2)
    plt.plot(history.history['loss'], label='Training Loss')
    plt.plot(history.history['val_loss'], label='Validation Loss')
    plt.title('Training vs Validation Loss')
    plt.xlabel('Epoch')
    plt.ylabel('Loss')
    plt.legend()
    plt.show()
    plot_training_history(history)
   # Evaluate the model on the validation set
accuracy = model.evaluate(validation_generator, steps=64)[1] # Limit evaluation
steps to 64
print(f"Validation Accuracy: {accuracy * 100:.2f}%")
# Generate the confusion matrix and classification report
validation_generator.reset() # Reset the generator for accurate predictions
Y_pred = model.predict(validation_generator, steps=validation_generator.samples //
validation_generator.batch_size + 1)
y_pred = np.argmax(Y_pred, axis=1)
# Get true labels
y_true = validation_generator.classes
# Generate confusion matrix
cm = confusion_matrix(y_true, y_pred)
class_names = list(validation_generator.class_indices.keys())
# Plot confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=class_names,
yticklabels=class_names)
plt.title('Confusion Matrix')
plt.ylabel('True Label')
plt.xlabel('Predicted Label')
plt.show()
# Print classification report
print('Classification Report')
print(classification_report(y_true, y_pred, target_names=class_names))
```

Train the Model InceptionV3

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import InceptionV3
from tensorflow.keras.models import Model
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ReduceLROnPlateau, EarlyStopping
from sklearn.metrics import classification_report
import numpy as np
import matplotlib.pyplot as plt
# Replace with the actual path to your TrashNet dataset
train_dir = '/content/drive/MyDrive/Trash Net/'
val_dir = '/content/drive/MyDrive/Trash Net/'
# Image data generators for data augmentation
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=30,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)
val_datagen = ImageDataGenerator(rescale=1./255)
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(227, 227),
    batch_size=32,
    class_mode='categorical'
)
val_generator = val_datagen.flow_from_directory(
    val_dir,
    target_size=(227, 227),
    batch_size=32,
    class_mode='categorical'
)
# Load the InceptionV3 model
base_model=InceptionV3(weights='imagenet', include_top=False, input_shape=(227,
227, 3))
# Add a global average pooling layer
x = base_model.output
x = GlobalAveragePooling2D()(x)
# Add fully connected layers
x = Dense(1024, activation='relu')(x)
x = Dropout(0.5)(x)
x = Dense(512, activation='relu')(x)
x = Dropout(0.5)(x)
# Final output layer
predictions = Dense(train_generator.num_classes, activation='softmax')(x)
# Create the model
model = Model(inputs=base_model.inputs, outputs=predictions)
# Unfreeze some layers of the base model for fine-tuning
for layer in base_model.layers[-50:]:
    layer.trainable = True
# Compile the model
model.compile(optimizer=Adam(learning_rate=0.0001),loss='categorical_crossentro
py', metrics=['accuracy'])
# Add learning rate reduction and early stopping
reduce_lr=ReduceLROnPlateau(monitor='val_loss',factor=0.2,patience=3,
min_lr=1e-6)
early_stopping=EarlyStopping(monitor='val_loss',patience=10,
restore_best_weights=True)
# Train the model with steps_per_epoch=64
history = model.fit(
    train_generator,
    steps_per_epoch=64,  # Limit to 64 steps per epoch
    epochs=50, # Increased epochs
    validation_data=val_generator,
    validation_steps=64, # Optional: set validation steps
    callbacks=[reduce_lr, early_stopping]
)
# Evaluate the model on the validation set
accuracy = model.evaluate(val_generator)[1]
            # Print the accuracy
print(f"Accuracy: {accuracy * 100:.2f}%")
# Get the true labels and predicted labels from the validation set
val_labels = val_generator.classes
val_preds = model.predict(val_generator)
val_preds_classes = np.argmax(val_preds, axis=1)
# Generate the classification report
report=classification_report(val_labels,val_preds_classes,
target_names=val_generator.class_indices.keys())
print("Classification Report:")
print(report)
# Plot training & validation accuracy values
plt.figure(figsize=(12, 6))
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```

Train the Model Vgg16

```
from tensorflow.keras.applications import VGG16
from tensorflow.keras.layers import Flatten, Dense
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
# Load and preprocess the TrashNet dataset
data_dir = '/content/drive/My Drive/Trash Net' # Update this path to your dataset location
# Define image data generators for training and validation
train_datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2
)
train_generator = train_datagen.flow_from_directory(
    data_dir,
    target_size=(224, 224), # Use 224x224 for VGG16
    batch_size=32,
    class_mode='categorical',
    subset='training'
)
validation_generator = train_datagen.flow_from_directory(
    data_dir,
    target_size=(224, 224), # Use 224x224 for VGG16
    batch_size=32,
    class_mode='categorical',
    subset='validation'
)
# Load the VGG16 model with pre-trained weights
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224,
3))
# Add custom top layers
x = base_model.output
x = Flatten()(x)
predictions = Dense(train_generator.num_classes, activation='softmax')(x)
# Create the model
model = Model(inputs=base_model.inputs, outputs=predictions)
# Freeze the convolutional layers in the base model
for layer in base_model.layers:
    layer.trainable = False
# Compile the model
model.compile(optimizer=Adam(learning_rate=0.0001),loss='categorical_crossentropy',
metrics=['accuracy'])
 # Train the model
history = model.fit(
    train_generator,
    epochs=10,
    validation_data=validation_generator
)
# Evaluate the model on the validation set
accuracy = model.evaluate(validation_generator)[1]
# Print the accuracy
print(f"Accuracy: {accuracy * 100:.2f}%")
```

Train The Model AlextNet

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
# Load and preprocess the TrashNet dataset
data_dir = '/content/drive/My Drive/Trash Net' # Update this path to your dataset location
# Define image data generators for training and validation
train_datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2
)
train_generator = train_datagen.flow_from_directory(
    data_dir,
    target_size=(227, 227),  # Use 227x227 for AlexNet
    batch_size=32,
    class_mode='categorical',
    subset='training'
)
validation_generator = train_datagen.flow_from_directory(
    data_dir,
    target_size=(227, 227),  # Use 227x227 for AlexNet
    batch_size=32,
    class_mode='categorical',
    subset='validation'
)
        # Define the AlexNet model
```

```
model = Sequential() # Now you can use Sequential since it's imported
# Convolutional layers
model.add(Conv2D(96, (11, 11), strides=4, activation='relu', input_shape=(227, 227, 3)))
model.add(MaxPooling2D(pool_size=(3, 3), strides=2))
model.add(Conv2D(256, (5, 5), strides=1, padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(3, 3), strides=2))
model.add(Conv2D(384, (3, 3), strides=1, padding='same', activation='relu'))
model.add(Conv2D(384, (3, 3), strides=1, padding='same', activation='relu'))
model.add(Conv2D(256, (3, 3), strides=1, padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(3, 3), strides=2))
# Fully connected layers
model.add(Flatten())
model.add(Dense(4096, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(4096, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(train_generator.num_classes, activation='softmax'))
# Compile the model
model.compile(optimizer=Adam(learning_rate=0.001), loss='categorical_crossentropy',
metrics=['accuracy'])
# Train the model
history = model.fit(
  train_generator,
   epochs=5,
   validation_data=validation_generator
)
# Evaluate the model on the validation set
accuracy = model.evaluate(validation_generator)[1]
# Print the accuracy
print(f"Accuracy: {accuracy * 100:.2f}%")
import matplotlib.pyplot as plt
```

All models

```
# Model names and accuracies
models = ['MobileNetv2', 'Vgg16', 'Inceptionv3', 'AlexNet']
accuracies = [97.84, 85.43, 99.70, 57.61]
```

```
# Plotting
plt.figure(figsize=(10, 6))
plt.bar(models, accuracies, color=['blue', 'green', 'red', 'orange'])
plt.xlabel('Model Name')
plt.ylabel('Accuracy (%)')
plt.title('Model Accuracy Comparison')
plt.ylim(0, 100) # Setting y-axis limit from 0 to 100
plt.grid(axis='y', linestyle='--', alpha=0.7)
# Show the plot
plt.show()
```

Frontend

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Plastic Detection</title>
        linkrel="stylesheet"href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">
    <style>
        /* Styles remain the same */
        body {
            font-family: 'Arial', sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f8f9fa;
        }
        header {
            background-color: rgb(173, 42, 19);
            color: white;
            padding: 20px;
            text-align: center;
            box-shadow: 0 2px 10px rgba(0, 0, 0, 0.2);
```

```
}
nav {
    display: flex;
    justify-content: center;
    background-color: #343a40;
}
nav a {
    color: white;
    padding: 14px 20px;
    text-decoration: none;
    text-align: center;
    transition: background-color 0.3s;
}
nav a:hover {
    background-color: #495057;
}
.container {
    text-align: center;
    background: #ffffff;
    margin: 20px auto;
    padding: 30px;
    border-radius: 10px;
    box-shadow: 0 0 15px rgba(0, 0, 0, 0.1);
    max-width: 800px;

}
.container1{
    margin-top:-250px;
    text-align: center;
    background: #ffffff;
    margin: 20px auto;
    padding: 30px;
    border-radius: 10px;
    box-shadow: 0 0 15px rgba(0, 0, 0, 0.1);
    }
input[type="file"] {
    display: none;
}
.upload-btn {
    background-color: #28a745;
    color: white;
    padding: 12px 25px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    font-size: 16px;
    transition: background-color 0.3s, transform 0.3s;
}
.upload-btn:hover {
    background-color: #218838;
    transform: scale(1.05);
}
.output {
    margin-top: 20px;
    font-size: 20px;
    padding: 15px;
    border: 1px solid #ced4da;
    border-radius: 5px;
    background-color: #f1f1f1;
    transition: background-color 0.3s;
}
.output:hover {
    background-color: #e2e6ea;
}
#image-preview {
    margin-top: 20px;
    max-width: 100%;
    height: auto;
    border-radius: 10px;
    display: block;
    margin-left: auto;
    margin-right: auto;
  }
  section {
    margin: 20px 0;
  }
  footer {
    background-color: #343a40;
    color: white;
    text-align: center;
    padding: 10px 0;
    margin-top: 20px;
  }
  body {
    margin: 0;
    font-family: Arial, sans-serif;
    box-sizing: border-box;
  }
  /*#home {
    position: relative;
    height: 100vh;
    display: flex;
    justify-content: center;
    align-items: center;
    color: white;
    text-align: center;
 background:url('https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcSpZQnNabtmlZNuYLeN3dIz7ZHBt4p9XEZc
Pg&s') no-repeat center center/cover;
  }*/
  .overlay {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0, 0, 0, 0.5);
}
.text-section {
    position: relative;
    z-index: 1;
}
.text-section h2 {
    font-size: 3rem;
    margin-bottom: 1rem;
}
.text-section p {
    font-size: 1.2rem;
    margin: 0 auto;
    max-width: 600px;
}
#home {
    display: flex;
    align-items: center;
    justify-content: space-between;
    padding: 50px;
    background-color: #f8f9fa;
    height: 100vh;
    margin-bottom:-200px;
    /* margin-top:-80px; */
}
.text-section{
    flex: 1;
    margin-right: 20px;
    color: #333;
    text-align: left;
    margin-top:-300px;
```

```
}
.text-section h2 {
   font-size: 2.0rem;
   margin-bottom: 2rem;
   color:crimson;
   margin-left: 90px;
}
.text-section p {
   font-size: 1.2rem;
   line-height: 1.6;
}
.image-section{
   flex: 1;
   text-align:  center;
   margin-left:30px;
   margin-top:-300px;
   /* margin-bottom:200px; */
}
.image-section img {
   max-width: 100%;
   height: auto;
   border-radius: 10px;
}
.cta-btn {
display: inline-block;
padding: 15px 30px;
font-size: 1rem;
font-weight: bold;
color: #fff;
background-color: #ff4081;
border: none;
border-radius: 5px;
cursor: pointer;
text-transform: uppercase;
```

```
        text-decoration: none;
        transition: background-color 0.3s ease, transform 0.2s ease;
        /* margin-top:-500px; */
        margin-left: 200px;
    }

    .cta-btn:hover {
        background-color: #e0356d;
        transform: scale(1.05);
    }
    #about-project .content-wrapper {
    display: flex;
    align-items: center;
}

#about-project .image-sec {
    height:300px;
    margin-right: 40px;


}
               #about-project .text-sec {
    flex: 2;
}
#about-project img {
    width: 100%;
    max-width: 500px;
    height: auto;
}
#predictions {
background:
url('https://i.pinimg.com/736x/ce/ab/31/ceab311b1dcdbebc85b35a2fe88bea94.jpg')
no-repeat center center;
            background-size: cover;
            color: black;
            padding: 20px;
        border-radius: 10px;
        box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
    }

    h2 {
        color: #cf610c; /* Highlight headings for better visibility */
    }

    ol {
        padding-left: 20px;
    }

    .upload-btn {
        display: inline-block;
        padding: 10px 20px;
        background-color: #35b732;
        color: #000;
        text-decoration: none;
        border-radius: 5px;
        cursor: pointer;
        margin-top: 10px;
    }
table {
        width: 50%;
        border-collapse: collapse;
        margin: 20px auto;
        font-family: Arial, sans-serif;
        font-size: 16px;
        margin-right: 2px;
        margin-top: -100px;
    }

    th, td {
        border: 1px solid #ddd;
        text-align: center;
        padding: 6px;
    }
    th {
        background-color: #f4f4f4;
        font-weight: bold;
    }
    tr:hover {
        background-color: #f9f9f9;
    }
    .formulas {
        width: 40%;
        font-size: 16px;
        line-height: 1.8;
        background-color: whitesmoke;
        padding: 15px;
        border: 1px solid #ddd;
        border-radius: 8px;
        margin-bottom: -200px;
        box-shadow: 0 4px 6px rgba(162, 20, 20, 0.1);
    }
    .formulas h2 {
        text-align: left;
        font-size: 20px;
        margin-bottom: 15px;
        color: #e0356d;
        text-align: center;
    }
    .formulas p {
        margin: 20px 0;


    }
    .formulas strong {
        color: #333;
```

```
        }
        #project-flowchart {
    padding: 20px;
    background-color: #f4f4f4;
    border-radius: 8px;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
}
#project-flowchart h2 {
    text-align: center;
    color: #333;
    margin-bottom: 10px;
}
.content-wrapper {
    display: flex;
    justify-content: space-between;
    align-items: flex-start;
}

.text-section v {
    width: 45%;
    padding-right: 10px;
    margin-top: 200px;
}

.text-section p {
    font-size: 1em;
    color: #555;
    margin-bottom: 100px;
    }
.flowchart-container {
    width: 45%;
    display: flex;
    flex-direction: column;
    align-items: center;
```

```
}
.flowchart-step {
   display: flex;
   flex-direction: column;
   align-items: center;
   margin-bottom: 20px;
}
.step-box {
   background-color: #ffffff;
   padding: 10px;
   width: 150px;
   text-align: center;
   border-radius: 8px;
   box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
}
.arrow {
   font-size: 20px;
   color: #333;
   margin-top: 10px;
}
.flowchart-step:last-child .arrow {
   display: none;
}
.step-box p {
   font-size: 1.1em;
   color: #555;
   font-weight: bold;
   margin: 5px 0;
}
h2 {
         color: #2a9d8f;
       }

      .project-flowchart {
         display: flex;
```

```
    flex-wrap: wrap;
}
.flowchart-container {
    flex: 1;
    max-width: 45%;
    margin-right: 20px;
}
.flowchart-container img {
    width: 100%;
    border: 1px solid #ddd;
    border-radius: 5px;
}
.content-container {
    flex: 1;
    max-width: 50%;
}
h1 {
    text-align: center;
    color: #2a9d8f;
}
h2 {
    color: #2a9d8f;
}
ul {
    list-style-type: disc;
    margin-left: 20px;
}
@media (max-width: 768px) {
    .project-flowchart {
        flex-direction: column;
    }
    .flowchart-container,
    .content-container {
        max-width: 100%;
```

```
            margin: 0;
          }
        }
    </style>
</head>
<body>
    <header>
        <h3>Accuracy and Efficiency Gains in Waste Classification
          Through Continuous Learning and Advanced Techniques
          </h3>
        <p>Team members:M.Madhavi, M.Sreya Reddy, P.Nikhitha</p>
    </header>
    <nav>
        <a href="#home">Home</a>
        <a href="#about-project">About Project</a>
        <a href="#predictions">Predictions</a>
        <a href="#evaluation-metrics">Model Evaluation Metrics</a>
        <a href="#project-flowchart">Project Flowchart</a>
    </nav>


    <section id="home" >
        <div class="text-section">
            <h2>Welcome to Waste Classification</h2>
            <p>
              This application uses deep learning to classify and identify various types of waste
such as plastic, paper, metal, glass, and cardboard.
               Upload an image to get started and discover the type of waste it is.
            </p
<button class="cta-btn" onclick="window.location.href='#upload'">Prediction</button>
</div>
        <div class="image-section">
                  <img src="https://e7.pngegg.com/pngimages/474/464/png-clipart-garbage-
illustration-waste-container-recycling-euclidean-garbage-collection-tshirt-
text.png"alt="Waste Classification" >
        </div>
```

```
  </section>
   <div id="about-project" class="container1">
     <h2>About the Project</h2>
     <div class="content-wrapper">
       <!-- Image Section -->
       <div class="image-sec">
           <img  src="https://www.shutterstock.com/image-photo/icons-related-reduce-
reuse-recycle-260nw-2316156387.jpg" alt="Waste Classification">
       </div>
       <!-- Text Section -->
       <div class="text-sec">
         <p>
           This project utilizes advanced deep learning techniques to classify waste into
categories like plastic, paper, metal, and glass. By automating waste classification, it
promotes recycling and sustainable waste management practices.
         </p>
         <p>
           Designed for individuals and organizations, it ensures high accuracy through
advanced algorithms and supports smart city initiatives for environmental sustainability.
         </p>
         <p>
            The project uses advanced models like <strong>InceptionV3</strong> for
efficient and accurate image classification. Additional models such as
<strong>Vgg16</strong>,<strong>AlexNet</strong>,and<strong>MobileNetV2</stron
g> ensure scalability and performance, improving classification reliability for smarter
recycling.
         </p>
       </div>
     </div>
  </div>


  <div id="predictions" class="container">


     <h2>Predictions</h2>
     <p>Upload an image to view the prediction results.</p>
     <div>
       <h3>Upload an Image</h3>
```

```
        <label for="file-upload" class="upload-btn">Choose File</label>
       <input id="file-upload" type="file" accept="image/*" onchange="uploadImage()">
        <img id="image-preview" src="" alt="Image Preview" style="display:none;">
        <div class="output" id="result"></div>
      </div>
      <h2>How It Works</h2>
      <ul>
        <li>
          <strong>Upload</strong>
          <p>The user uploads an image of the waste material through the application.</p>
        </li>
        <li>
          <strong>Preprocessing</strong>
           <p>The uploaded image is resized, normalized, and processed to prepare it for
classification.</p>
        </li>
        <li>
          <strong>Classification</strong>
           <p>The deep learning model identifies the type of waste (e.g., plastic, paper,
metal, glass, cardboard) using trained algorithms.</p>
        </li>
        <li>
          <strong>Result</strong>
           <p>The application displays the classification result and provides recycling or
disposal tips for the waste.</p>
        </li>
      </ul>


    </div>
   <div id="evaluation-metrics" class="container">
        <div class="formulas">
          <h2>Metric Formulas</h2>
           <p><strong>Accuracy:</strong> Accuracy = (TP + TN) / (TP + TN + FP +
FN)</p>
          <p><strong>Precision:</strong> Precision = TP / (TP + FP)</p>
          <p><strong>Recall:</strong> Recall = TP / (TP + FN)</p>
        </div>
    <table>


        <tr>
        <th>Model Name</th>
        <th>Accuracy</th>
        <th>Precision</th>
        <th>Recall</th>
        <th>F1 Score</th>
      </tr>
      <tr>
        <td>InceptionV3</td>
        <td>99.70%</td>
        <td>99.75%</td>
        <td>99.70%</td>
        <td>99.72%</td>
      </tr>
      <tr>
        <td>MobileNetV2</td>
        <td>97.84%</td>
        <td>97.90%</td>
        <td>97.85%</td>
        <td>97.87%</td>
      </tr>
      <tr>
        <td>Vgg16</td>
        <td>85.43%</td>
        <td>85.60%</td>
        <td>85.40%</td>
        <td>85.50%</td>
      </tr>
      <tr>
<td>AlexNet</td>
```

```
            <td>57.61%</td>
            <td>60.00%</td>
            <td>57.61%</td>
            <td>58.78%</td>
          </tr>
      </table>
      </div>
    </div>
 <h1>Project Flowchart</h1>
  <div id="project-flowchart" class="project-flowchart">
      <!-- Flowchart Section -->
      <div class="flowchart-container">
          <img src="/static/uploads/a.png" alt="Project Flowchart" style="height:600px;">
          <p>This flowchart represents the workflow of the waste classification system.</p>
      </div>
<!-- Content Section -->
      <div class="content-container">
          <h4>1. Data Preprocessing Phase</h4>
            <p><strong>Random Oversampling:</strong> To handle class imbalance in the
dataset, additional samples are generated for underrepresented categories.</p>
                    <p><strong>Data Augmentation:</strong> Techniques such as shear
transformations and rotation are applied to increase the diversity of the dataset.</p>
      <h3>2. Data Separation</h3>
          <p>The dataset is split into three subsets:</p>
          <ul>
              <li><strong>Train (70%):</strong> Used for training the model.</li>
is an InceptionV3 architecture optimized for high accuracy.</p>
<h3>4. Classification Output</h3>
<p>The system classifies waste into six categories:</p>
<ul>
    <li>Cardboard</li>
    <li>Metal</li>
    <li>Plastic</li>
    <li>Glass</li>
              <li>Paper</li>
              <li>Trash</li>
            </ul>
          </div>
        </div>
      </section>
                </div>
          <footer>
            <p>&copy; 2024 Tomato Leaf Disease Detection Project. All rights reserved.</p>
        </footer>
        <script>
function uploadImage() {
        const fileInput = document.getElementById('file-upload');
        const file = fileInput.files[0];
        const formData = new FormData();
        formData.append('file', file;
        const imgPreview = document.getElementById('image-preview');
        const reader = new FileReader();
        reader.onload = function(e) {
            imgPreview.src = e.target.result;
            imgPreview.style.display = 'block';
        };
        reader.readAsDataURL(fil
        fetch('/predict', {
            method: 'POST',
            body: formData
        })
        .then(response => response.json())
        .then(data => {
            const resultDiv = document.getElementById('result');
            if (data.error) {
                resultDiv.textContent = "Error: " + data.error;
            } else {
                resultDiv.textContent = "Prediction Category: " + data.disease_name;
            }
        })
        .catch(error => {
            console.error('Error:', error);
        });
    }
</script>
</body>
</html>
```

App.py

```
import os
from flask import Flask, request, render_template, jsonify
import numpy as np
from tensorflow.keras.models import load_model
from PIL import Image
import io
# Suppress TensorFlow logs
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'
os.environ['TF_ENABLE_ONEDNN_OPTS'] = '0'
app = Flask(_name_)
# Load the model and compile it
model = load_model('trashnet_inceptionv3 (1).h5')
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
print("Expected input shape:", model.input_shape)
# Define a dictionary to map indices to disease names
disease_classes = {
    0: 'cardboard',
    1: 'glass',
    2: 'metal',
    3: 'paper',
    4: 'plastic',
    5: 'trash',
}

# Preprocess the image according to model input requirements
```

```
def preprocess_image(image):
    img = image.resize((227, 227)) # Resize to match model input size
    img = np.array(img) / 255.0  # Normalize pixel values to [0, 1]
    img = np.expand_dims(img, axis=0)  # Add batch dimension
    return img
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    if 'file' not in request.files:
        return jsonify({"error": "No file provided"}), 400
 file = request.files['file']
 if file:
        try:
            image = Image.open(io.BytesIO(file.read()))
            processed_image = preprocess_image(image)

            # Make prediction
            prediction = model.predict(processed_image)
            predicted_class = np.argmax(prediction, axis=1)[0]
            disease_name = disease_classes.get(predicted_class, "Unknown")
 return jsonify({
                "prediction_index": int(predicted_class),
                "disease_name": disease_name
            })
        except Exception as e:
            return jsonify({"error": str(e)}), 500

    return jsonify({"error": "Invalid file"}), 400
if _name___== '_main_':
    app.run(debug=True)
```

7. TESTING

Testing is a crucial phase in any machine learning-based waste classification system, ensuring its accuracy, efficiency, and reliability before deployment. In this waste classification project, testing is conducted to validate the InceptionV3 deep learning model, verify the integration of different modules, and evaluate overall system functionality.[1]

The primary goal of testing is to detect and resolve errors, inconsistencies, and failures in the system. This includes assessing various components such as image preprocessing, classification, Flask-based web deployment, and user interface integration. A well-tested system ensures that it meets required accuracy and usability standards while delivering a seamless experience for users.[2]

7.1 Types of Testing

A combination of testing techniques is applied to verify the robustness and reliability of the waste classification system. These include **manual testing, automated testing, unit testing, and system testing**, each playing a distinct role in evaluating different aspects of the system.

7.1.1 Manual Testing

Manual testing involves human intervention to evaluate the system's behavior in real-world scenarios. It helps developers observe how the model and user interface function, identifying any potential usability or accuracy issues.

For this project, the Flask-based web interface is manually tested by uploading images of different waste materials. Testers check whether the system correctly processes images, classifies them into appropriate waste categories, and displays accurate results. Any misclassification or system lag is documented, and necessary improvements are implemented.

Additionally, preprocessing modules are manually validated to ensure that waste images are correctly loaded, resized, and normalized before being fed into InceptionV3. Any errors in this phase could impact the model's classification performance.[4]

Manual testing also assesses user experience (UX), including navigation, response time, and error handling. The system should provide clear error messages for invalid inputs and remain stable under different usage conditions.

7.1.2 Automated Testing

Automated testing ensures the classification system remains consistent and reliable across datasets. It validates model accuracy, the data preprocessing pipeline, and Flask API integration. The trained InceptionV3 model is tested on a separate dataset to confirm accuracy stability.[4]

Stress testing verifies that the Flask-based web interface handles concurrent image uploads without performance issues. Automated scripts check API endpoints for consistent responses to valid and invalid inputs. Automation reduces testing time and enhances system reliability, especially during frequent model updates.[5]

7.1.3 Unit Testing

Unit testing focuses on evaluating individual components of the system before integrating them into the complete workflow. This includes testing the image preprocessing pipeline, classification model, and Flask API.

7.1.4 Testing MobileNetV2

- MobileNetV2 is tested to confirm its lightweight architecture maintains high accuracy while reducing computational complexity.
- The model is evaluated using different input image sizes to determine its impact on feature extraction.
- Performance metrics such as **accuracy, precision, recall, and F1-score** are recorded to analyze its effectiveness in classifying waste types.
- If MobileNetV2 underperforms, hyperparameter tuning (e.g., learning rate adjustments, dropout layers) is applied to improve classification accuracy.[8]

- **7.1.5 Testing VGG16**

- VGG16 is tested to ensure its deep architecture can effectively extract meaningful features from waste images.
- The model’s performance is analyzed across different dataset sizes to assess its generalization capability.

- Overfitting is monitored by comparing training accuracy vs. validation accuracy sacross multiple epochs. If overfitting is detected, **dropout layers and data augmentation** techniques are applied.
- Confusion matrices are generated to verify that different waste categories are classified correctly.[3]

7.1.6 Testing InceptionV3

- InceptionV3's multi-path feature extraction capability is validated by assessing its ability to capture complex patterns in waste images.
- The model is tested with different optimizer settings (e.g., Adam, SGD) to evaluate its impact on classification accuracy.
- The number of convolutional layers activated for different waste types is analyzed using **feature visualization techniques**.Any misclassifications observed are traced back to preprocessing steps to determine potential errors in input normalization.[4]

7.2 Integration Testing

- Integration testing ensures that different components of the waste classification system work together seamlessly. After unit testing confirms that each module performs optimally in isolation, integration testing verifies the collective performance of these components when combined.[6]
- **7.2.1 Integrating InceptionV3 with Flask** The first integration test is conducted between the InceptionV3 model and the Flask-based web application. The system is tested by uploading various waste images to ensure that the classification results are displayed correctly and in real-time. If performance issues arise, optimizations such as caching or parallel processing may be implemented.[7]
- **7.2.2 Integrating Preprocessing with Classification** The preprocessing module is integrated with the classification model to verify that processed images maintain their quality and integrity before being classified. The impact of different preprocessing techniques on classification accuracy is analyzed to determine the most effective transformations.[8]
- **7.2.3 Testing API and Web Interface** The Flask API is tested to ensure smooth communication between the front end and the backend model. API calls are checked for response accuracy, latency, and error handling. Additionally, the web interface is

tested across different devices and browsers to confirm compatibility and responsiveness.[9]

7.3 System Testing

System testing ensures the entire waste classification model functions correctly under real-world conditions. It validates the workflow from data preprocessing to final classification and includes functional testing, performance testing, usability testing, and robustness testing.[1]

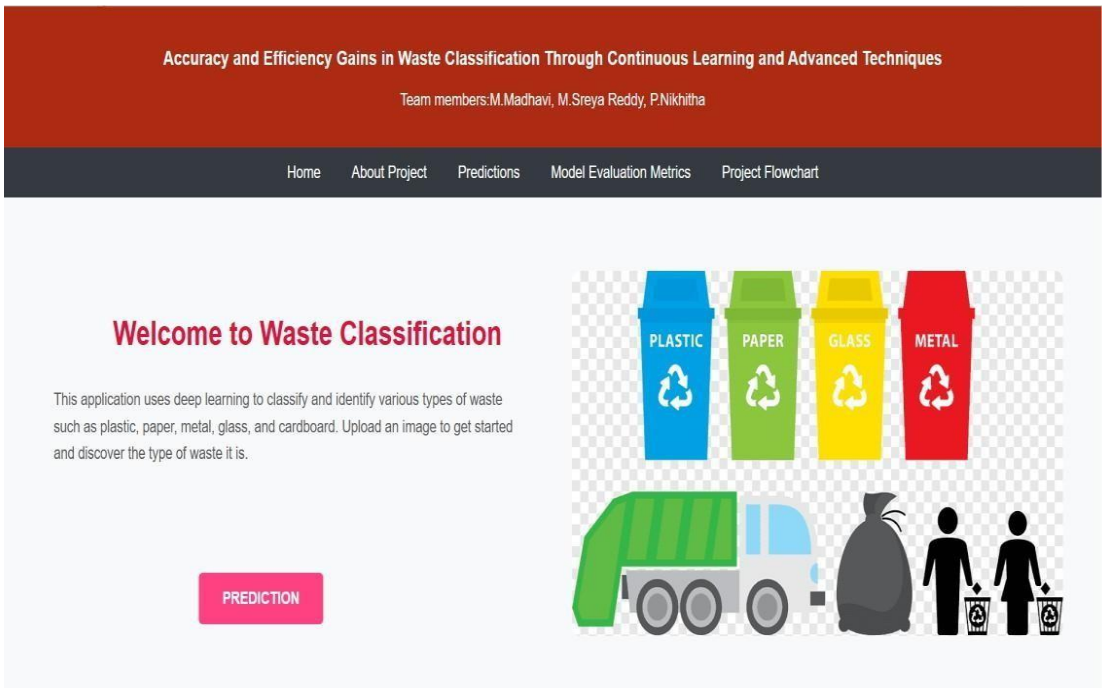
- **7.3.1 Functional Testing** This phase checks whether all components work together as expected. The dataset is preprocessed, and the InceptionV3 model is tested for proper classification. The web application is evaluated to ensure correct predictions and user interactions. Any discrepancies are analyzed and corrected.[2]
- **7.3.2 Performance Testing** The model's efficiency is tested by measuring inference time, memory usage, and scalability. If the model is slow or consumes excessive resources, optimizations such as pruning, batch size adjustments, or parallel processing are applied. The goal is to maintain high accuracy while improving computational efficiency.[3]
- **7.3.3 Usability Testing** Usability tests ensure the system is user-friendly and adaptable. The model is tested with different image formats (JPEG, PNG) to verify compatibility. Error handling is checked to ensure corrupted images do not crash the system. If deployed in an application, user interaction is tested to ensure smooth functionality.[4]

.

8. RESULT ANALYSIS

8.1 project screens

8.1.1 Home Page


Accuracy and Efficiency Gains in Waste Classification Through Continuous Learning and Advanced Techniques

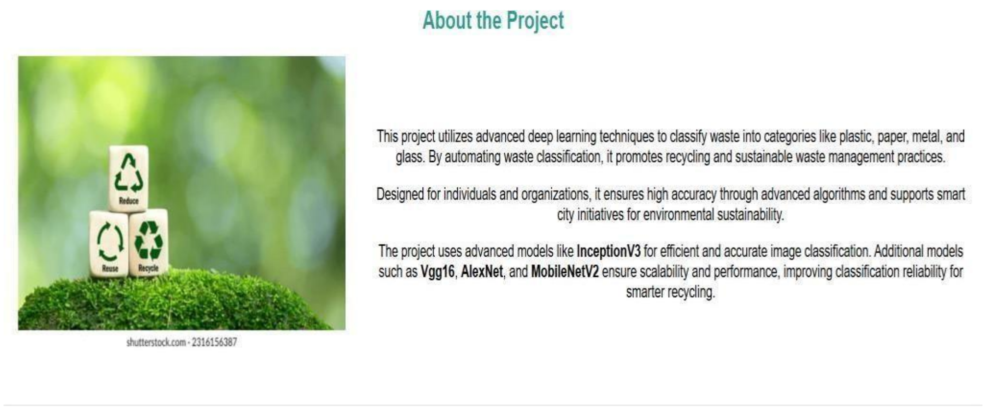
Team members:M.Madhavi, M.Sreya Reddy, P.Nikhitha

Welcome to Waste Classification

This application uses deep learning to classify and identify various types of waste such as plastic, paper, metal, glass, and cardboard. Upload an image to get started and discover the type of waste it is.

The homepage of the Waste Classification project presents a deep learning-based system for identifying waste types like plastic, paper, metal, glass, and cardboard. Users can upload an image for classification, promoting efficient waste disposal. The visually appealing design includes color-coded bins, a "PREDICTION" button for easy interaction, and a navigation bar with sections like About Project, Predictions, Model Evaluation Metrics, and Project Flowchart for better understanding.

8.1.2 About Page


About the Project

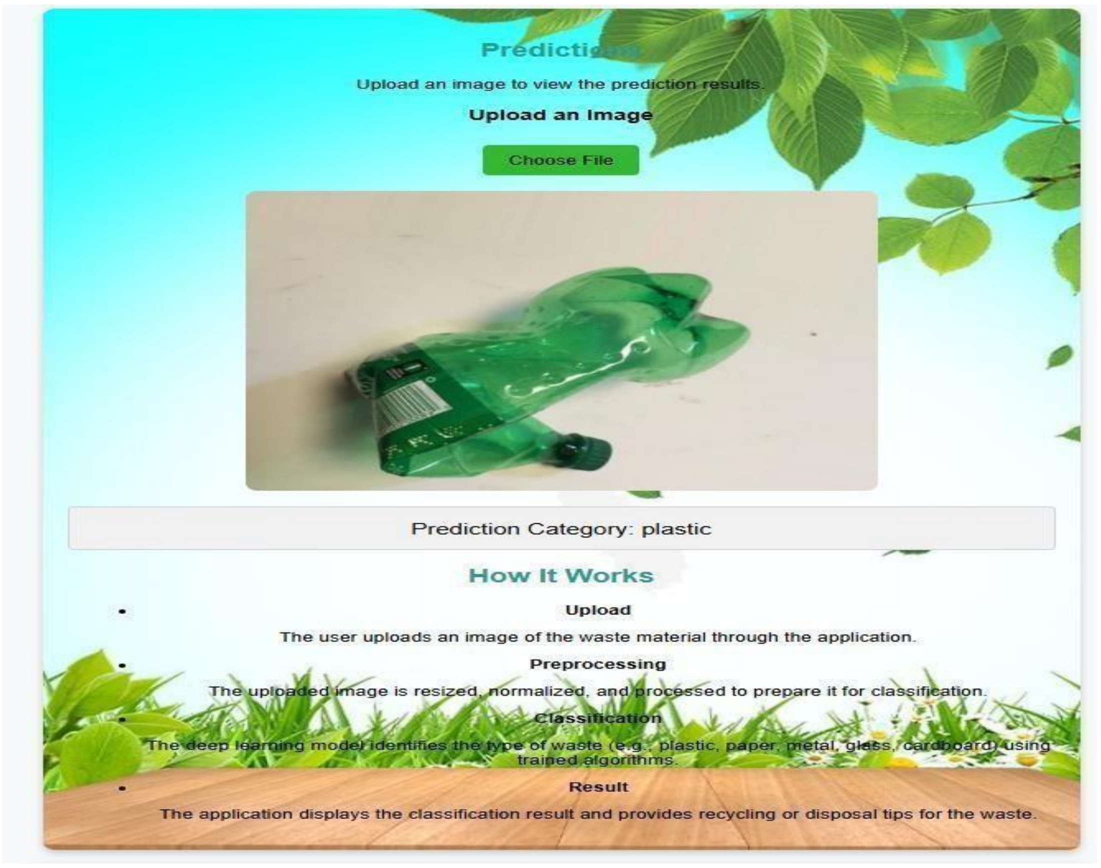
This project utilizes advanced deep learning techniques to classify waste into categories like plastic, paper, metal, and glass. By automating waste classification, it promotes recycling and sustainable waste management practices.

Designed for individuals and organizations, it ensures high accuracy through advanced algorithms and supports smart city initiatives for environmental sustainability.

The project uses advanced models like **InceptionV3** for efficient and accurate image classification. Additional models such as **Vgg16**, **AlexNet**, and **MobileNetV2** ensure scalability and performance, improving classification reliability for smarter recycling.

The About the Project page details how deep learning automates waste classification into categories like plastic, paper, metal, and glass, promoting recycling and sustainability. Designed for individuals and organizations, it ensures high accuracy using InceptionV3, with VGG16, AlexNet, and MobileNetV2 enhancing scalability and reliability for smarter recycling solutions**.**

8.1.3 Prediction Page



Predictions

Upload an image to view the prediction results.

Upload an Image

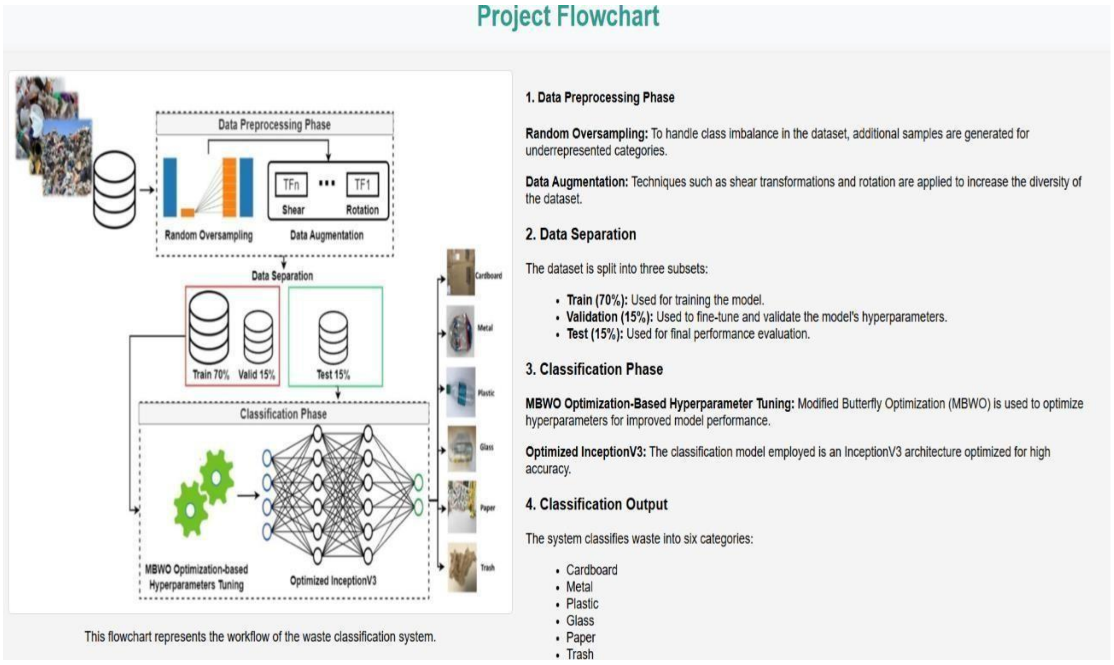
Prediction Category: plastic

How It Works

- Upload
  The user uploads an image of the waste material through the application.
- Preprocessing
  The uploaded image is resized, normalized, and processed to prepare it for classification.
- Classification
  The deep learning model identifies the type of waste (e.g., plastic, paper, metal, glass, cardboard) using trained algorithms.
- Result
  The application displays the classification result and provides recycling or disposal tips for the waste.

The Prediction module enables users to upload an image of waste material, which is then processed using deep learning techniques to classify it into categories such as plastic, paper, metal, glass, or cardboard. The system follows a structured approach: the uploaded image is first preprocessed through resizing and normalization, then analyzed by a trained model to determine its category. Finally, the application displays the classification result along with recommendations for proper recycling or disposal, promoting efficient and sustainable waste management.

8.1.4 Flowchart Page



Project Flowchart

This flowchart represents the workflow of the waste classification system.

1. Data Preprocessing Phase

Random Oversampling: To handle class imbalance in the dataset, additional samples are generated for underrepresented categories.

Data Augmentation: Techniques such as shear transformations and rotation are applied to increase the diversity of the dataset.

2. Data Separation

The dataset is split into three subsets:

- **Train (70%):** Used for training the model.
- **Validation (15%):** Used to fine-tune and validate the model's hyperparameters.
- **Test (15%):** Used for final performance evaluation.

3. Classification Phase

MBWO Optimization-Based Hyperparameter Tuning: Modified Butterfly Optimization (MBWO) is used to optimize hyperparameters for improved model performance.

Optimized InceptionV3: The classification model employed is an InceptionV3 architecture optimized for high accuracy.

4. Classification Output

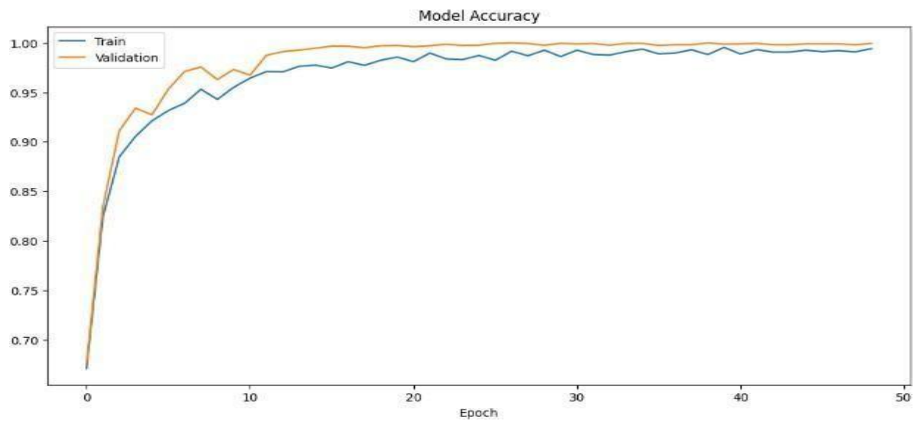
The system classifies waste into six categories:

- Cardboard
- Metal
- Plastic
- Glass
- Paper
- Trash

The waste classification system preprocesses data using random oversampling and data augmentation to enhance diversity. The dataset is split into train (70%), validation (15%), and test (15%) sets. The model employs MBWO for hyperparameter tuning and an optimized InceptionV3 for accuracy. It classifies waste into six categories: cardboard, metal, plastic, glass, paper, and trash for efficient waste management.

8.2 Accuracy and Loss Curves of the Model

The inceptionV3 model is a very efficient model in image classification, maintaining high accuracy due to the high architecture and efficient learning qualities. The model elevates the precision high by multiscale processing for capturing the features at a fine level, while the recall goes high due to comprehensive feature extraction. Its F1 score optimally balances the precision and recall, optimized by sophisticated feature extraction and handling of imbalanced Classes.



8.3 TRAINING AND TESTING ACCURACY OF MODELS

The train and test accuracy graphs of the InceptionV3 and MobileNetV2 shows high and stable accuracy levels with minimal overfitting, VGG16 is overfitting since the gap between the validation accuracy and the training accuracy is considerable, and AlexNet is showing an early plateauing, as well as low levels of accuracy, which can be a sign of overfitting. In general, both InceptionV3 and MobileNetV2 do better

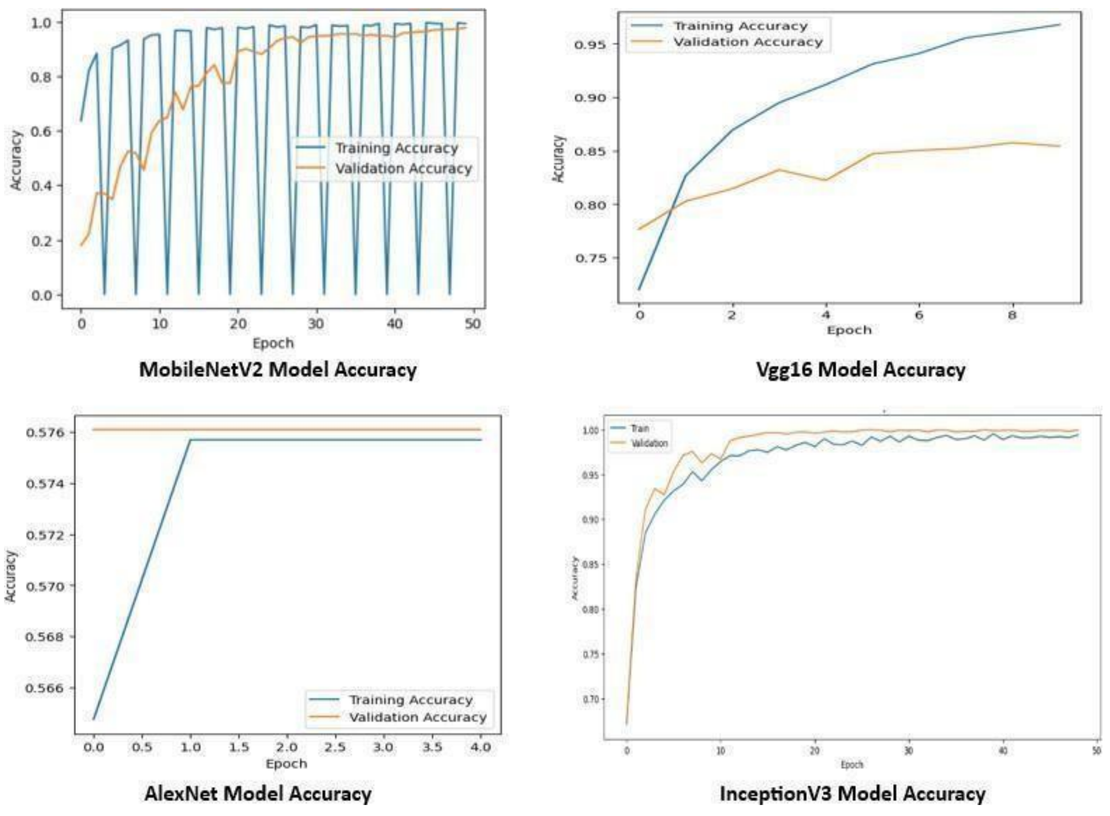


Fig 8.3.2 Traning and testing Accuracy of models

8.4 GRAPHS OF MODEL ACCURACY AND LOSS

The following graph compares the accuracy of different deep learning models used for waste classification. The x-axis represents the model names, while the y-axis represents the accuracy percentage. Models such as MobileNetv2, Vgg16, InceptionV3

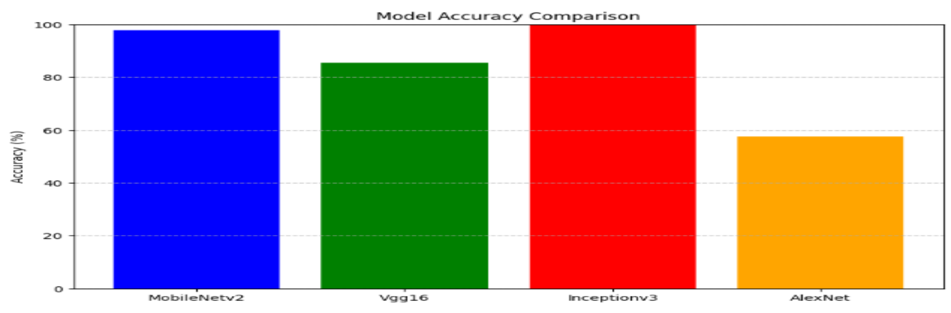


Fig 8.4.1 Graphs of model Accuracy of models

9. CONCLUSION

The study concludes that the proposed intelligent waste classification model, combining the InceptionV3 deep learning architecture with multi-objective Beluga Whale Optimization, significantly enhances waste sorting accuracy and efficiency. With a classification accuracy of 97.75% and robust performance across other metrics, the model effectively addresses challenges such as class imbalance using oversampling and data augmentation. It demonstrates practical applicability for smart city infrastructures and sustainable waste management, reducing reliance on landfills and supporting recycling initiatives. Future work aims to expand datasets, integrate real-time sorting, and explore broader applications in waste classification.

FUTURE SCOPE

The study an innovative and eco-friendly waste classification system with MOBWO metaheuristic and deep learning algorithms. The experiment results indicate that the superiority of the proposed model against the traditional models in the aspect of the classification accuracy as well as in the resources usage. The model shows pretty good convergence for any type of waste and consistent performance, which can lead to an improvement in waste management systems. Being both effective in terms of classification accuracy and efficient for the computational costs the approach provides a rather efficient solution to the large scale waste classification problem that can contribute to the environmental sustainability and decrease the burden that waste management facilities face.

The future scope includes integrating real-time waste sorting systems, improving optimization efficiency for larger datasets, expanding to more waste categories, and enhancing the model’s scalability for broader environmental applications and smart cities.

10. REFERENCES

[1] Sayed, G. I., Abd Elfattah, M., Darwish, A., & Hassanien, A. E. (2024). Intelligent and sustainable waste classification model based on multi-objective beluga whale optimization and deep learning. *Environmental Science and Pollution Research*, 1-19.Bobe, S., Adhav, P., Bhalerao, O., & Chaware, S. (2023, July). Review on Deep Learning Based Biomedical Waste Detection and Classification. In *2023 2nd International Conference on Edge Computing and Applications (ICECAA)* (pp. 10711076). IEEE.

[2] Rahmad, M. A., Midi, N. S., Zaini, S. A., Yusoff, S. H., & Mohamad, S. Y. (2018, September). Material classification of recyclable waste using the weight and size of waste. In *2018 7th International Conference on Computer and Communication Engineering (ICCCE)* (pp. 44-49). IEEE.

[3] Tripathi, R., Shetty, H., Patil, K., Ingawale, P., & Trivedi, M. (2023, October). Intelligent Waste Material Classification Using EfficientNet-B3 Convolutional Neural Network for Enhanced Waste Management. In *2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS)* (pp. 132-137). IEEE.

[4] Sireesha Moturi, S.N.Tirumala Rao, Srikanth Vemuru,Grey wolf assisted dragonflybased weighted rule generation for predicting heart disease and breast cancer, Computerized Medical Imaging and Graphics, Volume 91, 2021, 101936, ISSN 0895-6111,https://doi.org/10.1016/j.compmedimag.2021.101936

[5] Joseph, I. O., Wangare, N. L., Mahoque, T. P., Tewari, P., & Majumdar, S. (2023, July). E-Waste Intelligent Robotic Technology (EIRT): A Deep Learning approach for Electronic Waste Detection, Classification and Sorting. In *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)* (pp. 1-6). IEEE.

[6] Gupta, K., Arora, P., Sharma, R., Yeshfeen, N., Kumari, R., & Bansal, P. (2023, November). Enhancing Waste Classification with Convolutional Neural Networks and Explainability Techniques: Power of Computer in Classification. In *2023 International Conference on Communication, Security and Artificial Intelligence (ICCSAI)* (pp. 569-574). IEEE.

[7] M.Sireesha, S. N. TirumalaRao, Srikanth Vemuru, Frequent Itemset Mining Algorithms: A Survey Journal of Theoretical and Applied Information Technology Vol - 96, No .3, Feb - 2018 ISSN - 1992- 8645, Pages – 744 – 755

[8] Puspaningrum, A. P., Endah, S. N., Sasongko, P. S., Kusumaningrum, R., &

Ernawan, F. (2020, November). Waste classification using support vector machine with SIFT-PCA feature extraction. In *2020 4th International Conference on Informatics and Computational Sciences (ICICoS)* (pp. 1-6). IEEE.

[9] Kamboj, M. D., Singh, M., Usmani, M., Ahmad, S., & Gaur, M. A. (2023, November). A Comparative Analysis of Algorithms for Effective Waste Classification. In *2023 International Conference on Sustainable Communication Networks and Application (ICSCNA)* (pp. 598-604). IEEE.

[10] M.Sireesha, S. N. TirumalaRao, Srikanth Vemuru, Optimized Feature Extraction and Hybrid Classification Model for Heart Disease and Breast Cancer Prediction International Journal of Recent Technology and Engineering Vol - 7, No 6, Mar 2019 ISSN - 2277-3878, Pages – 1754 – 1772.

[11] Dookhee, S. (2022, December). Domestic solid waste classification using convolutional neural networks. In *2022 IEEE 5th International Conference on Image Processing Applications and Systems (IPAS)* (pp. 1-6). IEEE.

[12] M. Sireesha, Srikanth Vemuru and S. N. TirumalaRao, ”Coalesce based binary table: an enhanced algorithm for mining frequent patterns", International Journal of Engineering and Technology, vol. 7, no. 1.5, pp. 51-55, 2018.

[13] Pandey, A., Jain, H., Raj, H., & Gupta, P. (2023, April). Identification and classification of waste using CNN in waste management. In *2023 IEEE 8th International Conference for Convergence in Technology (I2CT)* (pp. 1-6). IEEE.

Accuracy and Efficiency Gains in Waste Classification through Continuous Learning and Advanced Techniques

Mothe Sathyam Reddy[1], Meena Madhavi[2], Munamala Sreya Reddy[3], Pothabattini Nikhitha[4], Nukala Vijaya Kumar[5], Sireesha Moturi[6], and Dodda Venkata Reddy[7]

[1,5,7]Asst.Professor,Department of CSE, Narasaraopeta Engineering College, Narasaraopet-522601, Palnadu, Andhra Pradesh, India. [2,3,4] Student, CSE Department, Narasaraopeta Engineering College, Narasaraopet-522601, Palnadu, Andhra Pradesh, India. [6] Assistant professor, Department of CSE, Narasaraopeta Engineering College, Narasaraopet-522601, Palnadu, Andhra Pradesh, India.
sathyamreddym@gmail.com

Abstract. —Efficient waste management is crucial for sustainable development, especially with the ever-growing volume of municipal and household waste. Existing waste classification systems suffer from several challenges: they achieve a very low recognition accuracy rate, create class imbalance problems, and find it hard to scale-up applicability in real-world conditions. This paper addresses these issues by proposing the InceptionV3 architecture of deep learning.
MBWO is utilized to optimize key hyperparameters such as learning rate and dropout rate, batch size on the TrashNet dataset. As a result of this optimization process, it presents outstanding performance metrics such as 97. 75% precision, 99. 55% specificity, and provides 98. 88% precision compared to traditional methods. It categorizes waste materials as paper, plastic, metal, and glass to facilitate efficient recycling processes and reduce the dependence of landfills. This system overcomes the deficiencies of current systems by incorporating both data augmentation and oversampling techniques to handle class imbalance, along with transferring knowledge with CNNs for adaptability in real-world scenarios. Future work will increase the dataset, allow for real-time waste classification, and extend applications to smart city infrastructures while ensuring scalability and applicability in modern waste management challenges.

Keywords: Waste Classification, Deep Learning, Beluga Whale Optimization, InceptionV3, Sustainable Waste Management

1 INTRODUCTION

Effective waste management is thus critical for environmental sustainability, and improper waste disposal remains one of the major issues facing human beings worldwide, leading to depletion of resources, pollution, and health hazards.This underlines the tremendous pressure to develop automated intelligent waste classification systems. Recent developments in AI and deep learning, though promising improving waste classification, make the most recent image-classification models, like InceptionV3, applicable to this domain. Traditional approaches still suffer from all key challenges, like class imbalance, overfitting, and lack of scalability in practice. To neutralize these, this research combines InceptionV3 with Multi-Objective Beluga Whale Optimization (MBWO) for hyperparameter optimization and better performance. Techniques such as data augmentation and over-sampling will help in generalized applicability towards the different types of wastes. As demonstrated on the TrashNet dataset, it always performs better than the traditional methods with a classification accuracy of 97.75% and thus supports more efficient recycling and lesser dependency on landfills. The proposed approach therefore offers an economical and sustainable solution for modern waste management systems that can be integrated with other global sustainability goals and smart city infrastructures.

2 LITERATURE REVIEW

Artificial Intelligence-based Smart Household Waste Classification System This paper mostly concentrates on the development of an intelligent system to classify household waste using deep learning methodologies, namely the application of CNNs and transfer learning. The system will automatically classify different types of household wastes according to their images by the use of a number of architectures of CNNs, such as MobileNet, VGG16, and ResNet50. It discusses some of the challenges, like the limitation of categories in public datasets, and overcomes them using the features of pretrained models and transfer learning techniques. The focus [1]. Classification of Domestic Solid Waste using CNNs: This paper presents the automation of classification using CNNs for the growing problem of household waste. It expands existing datasets, such as TrashNet, to 15,515 images in twelve categories. The highest accuracy was achieved by Xception with the Nadam optimizer at 89.57%. Data augmentation is shown to be a strategy for dealing with class imbalance and improving model generalization[2]. Beluga Whale Optimized and Deep Learning-Based Waste Classification. This paper uses the InceptionV3 model with dropout, batch size optimization for the classification of waste. It achieved 97.75% accuracy, which places importance on the data augmentation and oversampling in balancing the classes. The model outperforms the classical approaches, with accuracy at 75% on the TrashNet dataset[3]. Waste Classification by Comparison of Algorithms: In this study, it will be attempted to compare a variety of machine learning algorithms including SVM, Random Forest, Naïve Bayes, Decision Tree, KNN, and CNNs used for waste classification with accuracy and computational performance that identifies each of these strengths and weaknesses related to decreasing environmental damage[4]. Deep Learning for Biomedical Waste Classification: This paper is on CNN-based systems that can be used to classify biomedical waste into infectious, hazardous, and radioactive categories. The system improves accuracy, thereby enabling better waste management practice that ensures safety and protection of public health[5]. E-Waste Intelligent Robotic Technology: A Deep Learning Approach for Electronic Waste Detection, Classification, and Sorting: The paper proposes a deep learning-based robotic system called EIRT, which would sort electronic waste into categories. It covers the application of the EfficientNet-D2 architecture to detect and classify e-waste into different categories such as batteries, mobile phones, or electronic boards. EIRT is a robotic arm on a mobile car basis, autonomously finding, picking up, and sorting e-waste with an accuracy of 82.32%. This work develops the urgent need that addressing ewaste management requires with respect to automation and artificial intelligence means[6]. Utilizing Waste Size and Weight for Classification: This paper considers a system with load cells and ultrasonic sensors to classify the types of recyclable material in the form of paper, plastic, glass, and metal based on the size and weight of waste. It has an algorithm for automation of most classification processes that has a promising accuracy in classification and reduces manual sorting, although it is a challenging system for diverse types of wastes.[7]. SIFT-PCA Feature Extraction and SVM for Waste Classification: This research assesses the performance of machine learning approaches on the Trashnet dataset. The study compares SVM that used PCA for dimensionality reduction. The highest accuracy of 62% was obtained using the SVM with SIFT-based features. However, results of the PCA-based approach proved to reduce accuracy, with the necessity to preserve important features within the dimensionality reduction proces[8]. Waste Identification Using CNN: This paper analyzes CNN models such as VGG16 and FastNet-34 for waste classification. Using a Kaggle preprocessed dataset of 22,564 images, CNN achieved 83% accuracy while SVM reached 94.8%. The study indicates that CNN accuracy can be improved by optimized hyperparameters.[9].Sasidhar Mutyala et al presented a Grey Wolf-assisted Dragonfly-based weighted rule generation algorithm for diagnosing heart disease and breast cancer. In this way, the given approach will improve feature selection and accuracy of classification. The result is effective enough for the computer-aided medical diagnosis systems[10].M. Sireesha et al. made a comprehensive survey on the frequent itemset mining algorithms, emphasizing their theoretical bases, practical applications, and computational efficiency. It has discussed the development of these algorithms, where the importance of the algorithms in data mining and knowledge discovery has been discussed with various domains[11].In their work, Dinesh S. G. et al. made a comparative analysis of different algorithms for waste classification. It focuses on analyzing the strengths and weaknesses in contributing to the development of an efficient computational model for sustainable waste management and environmental conservation[12].The authors brought in a hybrid model that features the techniques of feature extraction amalgamated with machine learning to predict heart disease and breast cancer. Such an integrated model improves the accuracy of the diagnosis, dealing with the intricate patterns in medical data sets and increases the reliability of predictions[13].Fandey A. et al. discussed the application of convolutional neural networks (CNN) for solid waste classification in domestic settings. It demonstrates that CNN models can achieve high classification accuracies, which is towards practical and scalable solutions that support sustainable waste management systems and smart city initiatives[14].

3 MATERIALS AND METHODS

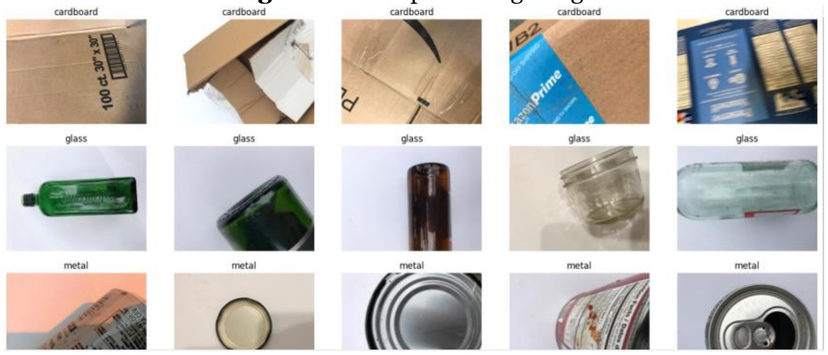
3.1 Dataset Description

The TrashNet dataset has 2,573 images in the six different categories: cardboard, glass, metal, paper, plastic, and trash, captured at 512 × 384 pixels. To balance the classes, this dataset suffers from class imbalance. This may cause biasness among the samples of some classes. To deal with this, the dataset was split into 70% training, 15% testing, and 15% validation, with data augmentation and oversampling applied to balance the classes and improve model generalization.

3.2 DATA PREPROCESSING

- The raw images of the TrashNet dataset are resized to the standard dimension of 299x299x3 since it is required in relation to the input layer in The architecture of deep learning InceptionV3[1].
- Data Augmentation: Class imbalance and other problems can be treated using various methods of data augmentation. For this purpose, flipping, rotation, and zooming have been performed to increase artificially the size and variability of the dataset[2].
- Unplanned Oversampling: This is going to solve the issue of class disparity by giving more representation of the underrepresented classes, that is, the trash classes. Therefore, the model will not be biased towards those classes which dominate the training[3].

Fig. 1. After Preprocessing images



The image depicts a collection of recyclable materials labeled and sorted into three categories: cardboard, glass, and metal. Each category has a number of samples that appear visually different and are accordingly labeled.

The cardboard samples are the different types of packaging boxes, while the glass category contains bottles and jars in varying shapes and colors. The cans and containers in the metal section have different textures and conditions. This dataset seems to be organized for use in waste classification or recycling-related machine learning tasks, emphasizing visual diversity within each category.

3.3 MODELS

- MobileNetV2: MobileNetV2 is one of the modified versions of CNN which has been developed with an intention to work effectively in the mobile devices. With it, it effectively attempts at an inverted residual structure with linear bottlenecks that decreases parameters and increases accuracy to suit its real-time application such as image classification in resource-restraint scenarios.
- InceptionV3:InceptionV3 is another deep CNN that ISO feature for image classification, endorsed for high accuracy and efficiency. It decreases computational need through approach such as convolutional factorization. Inception V3 is 42-layer deep, therefore able to complete intricate tasks and is therefore perfect for the classification of waste.
- VGG16: VGG16 On the other hand is a 16 layer CNN often used in image classification. The structure employs the multiple of 3×3 convolutional filters that are proficient for waste classification tasks to detect image attributes.
- AlexNet: The ILSVRC 2012 winner, AlexNet is a deep CNN which features five layers of convolution and three layers of fully connected layers ReLU non-linearity. The most important one are the max pooling,dropout for avoiding the overfitting and overlapping filters to help with feature extraction. It was one of the first models to utilize GPU resulting in higher levels of classification.

3.4 MODEL TRAINING AND EVALUATION

The inceptionV3 model is a very efficient model in image classification, maintaining high accuracy due to the high architecture and efficient learning qualities. The model elevates the precision high by multiscale processing for capturing the features at a fine level, while the recall goes high due to comprehensive feature extraction. Its F1 score optimally balances the precision and recall, optimized by sophisticated feature extraction and handling of imbalanced classes. Due to its innovative design, combined with very robust performance metrics, InceptionV3 will definitely be one of the leading choices for the tasks of image classification.

3.5 COMPARATIVE ANALYSIS

In contrast, the highest accuracy was achieved by InceptionV3 with 99.70% accuracy; it had excellent precision, recall, and F1-score. The best model for the given

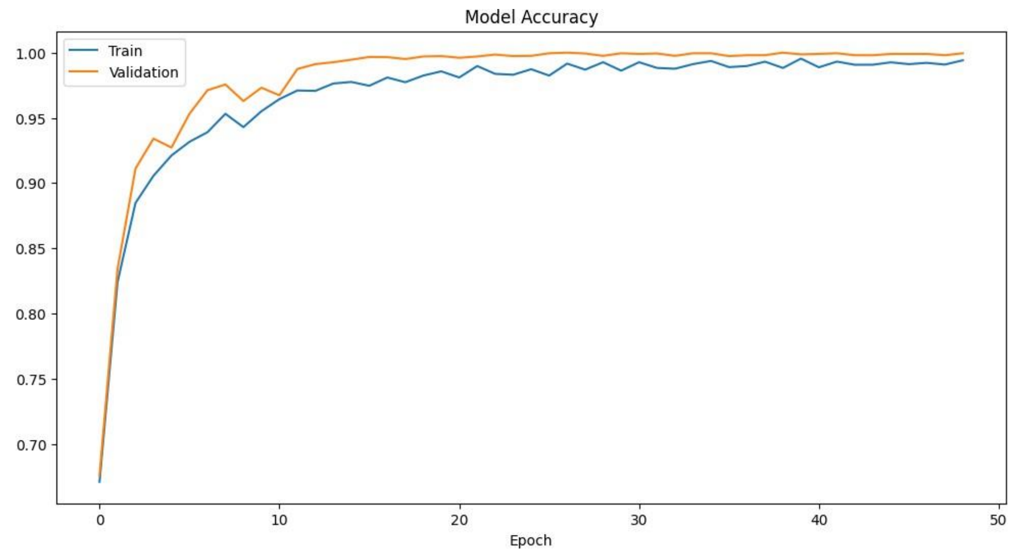


Fig. 2. Accuracy of InceptionV3

task is, thus, InceptionV3. MobileNetV2 attained an accuracy of 97.84%, balancing accuracy and efficiency, thereby being suitable for scenarios where computational resources are rather limited. VGG16 offers fairly good performance, but it is slower; accuracy is at 85.43%. AlexNet, with the poorest accuracy, is at 57.61%, less practical without further tuning. Generally, InceptionV3 performs the best while MobileNetV2 is the good balance between the dimension and the speed of the model.

3.6 MODEL ACCURACY TABLE

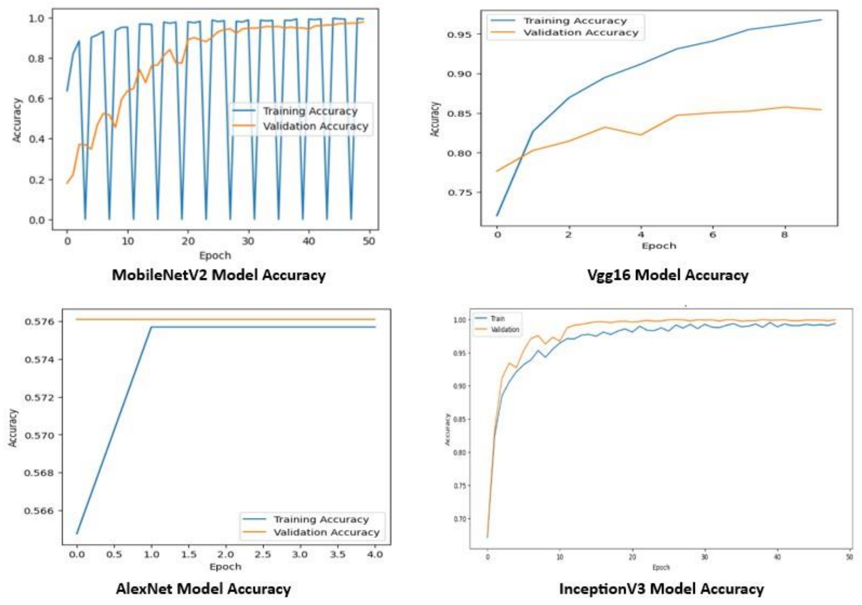
Table 1. Accuracy of different models

Model Name	**Accuracy**
Inceptionv3	99.70%
MobileNetV2	97.84%
Vgg16	85.43%
AlexNet	57.61%

The table compares the accuracy of four machine learning models—InceptionV3, MobileNetV2, VGG16, and AlexNet—when applied to a classification task, likely related to waste categorization based on the context. InceptionV3 outperforms the others with a near-perfect accuracy of 99.70%, showing its superior capability for this task. MobileNetV2 follows with a strong performance at 97.84%, showing efficiency and reliability. VGG16, although less accurate at 85.43%, still gives good results, while AlexNet lags far behind with an accuracy of 57.61%, which shows its inadequacy for this purpose. These results indicate InceptionV3 as the best choice for high-accuracy requirements.

3.7 TRAINING AND TESTING ACCURACY OF MODELS

Fig. 3. Model Accuracy Comparison



The following figure compares the training and validation accuracy of MobileNetV2, VGG16, AlexNet, and InceptionV3 across epochs. InceptionV3 demonstrates a consistent, near-perfect accuracy with smooth learning curves and outperforms all others. MobileNetV2 stabilizes early on and achieves high accuracy despite some initial fluctuations. VGG16 steadily improves but remains less accurate than the top two. AlexNet has little improvement and reaches an early plateau with the lowest accuracy, indicating poor performance for this task.

3.8 Evaluation Metrics

The performance of different models based on these metrics is summarized in Table.

- **Accuracy:** The percentage of cases properly classified out of all the instances.
- **Precision:** The percentage of positively anticipated observations that were accurately predicted to all positive predictions.
- **Recall:**The proportion of all observations in the actual class that were accurately predicted to be positive.
- **F1 Score:** The precision and recall weighted average, providing a balance

Table 2. PERFORMANCE COMPARISON OF MODELS BASED ON EVALUATION METRICS

Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**
InceptionV3	99.70%	99.75%	99.70%	99.72%
MobileNetV2	97.84%	97.90%	97.85%	97.87%
Vgg16	85.43%	85.60%	85.40%	85.50%
AlexNet	57.61%	60.00%	57.61%	58.78%

Table 2 shows the comparison of four deep learning models, InceptionV3, MobileNetV2, Vgg16, and AlexNet, in terms of accuracy, precision, recall, and F1 score. InceptionV3 achieved the best performance with the metrics above 99.70%. The second best is MobileNetV2 at 97.98%. Vgg16 shows moderate performance around 85.43%. AlexNet performs poorly, with metrics near 57.60%. This indicates InceptionV3 as the most effective model. Table 3 reports the performance

Table 3. Precision, Recall, F1 Score, and Support for Waste Classification

CLASSES	**PRECISION**	**RECALL**	**F1 SCORE**	**SUPPORT**
Cardboard	1.00	1.00	1.00	4411
Glass	1.00	0.98	1.00	976
Metal	1.00	0.99	1.00	1010
Paper	1.00	1.00	0.98	237
Plastic	0.99	1.00	0.99	200
Trash	0.99	0.98	0.98	827
Accuracy	0.99			
Macro avg	0.98	1.00	0.98	7661
Weighted avg	0.98	0.99	0.99	7661

of the proposed waste classification system on six classes: Cardboard, Glass, Metal, Paper, Plastic, and Trash. All classes except Trash obtain near-perfect metrics; the latter has slightly lower values (precision 0.99, recall 0.98). The overall accuracy of the system is 0.99%, and macro and weighted averages are strong, indicating high and consistent performance.

3.9 TRAINABLE PARAMETERS AND TESTABLE PARAMETERS

InceptionV3 has 23.9 million trainable and 27.2 million testable parameters, therefore offering the good trade-off between complexity and efficiency. MobileNetV2 - with 2.2 million trainable and 3.5 million testable parameters - is designed for light applications, and hence very efficient. VGG16 and AlexNet are much heavier in terms of parameters, having 138 and 61 million of them, and are, therefore, computationally resource-intensive though offering better learning capabilities.

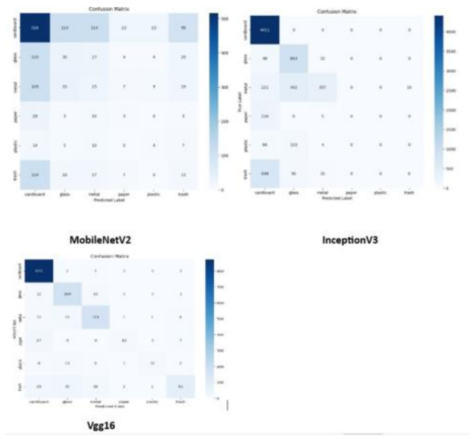
Table 4. Trainable and Testable Parameters of Different Models

Model Name	Trainable Parameters	Testable Parameters
InceptionV3	23.9 million	27.2 million
MobileNetV2	2.2 million	3.5 million
VGG16	138 million	138 million
AlexNet	61 million	61 million

3.10 CONFUSION MATRIX

ht InceptionV3, MobileNetV2, and VGG16 exhibit confused classification per-

Fig. 4. InceptionV3,MobileNetV2,Vgg16



formance between the categories of waste. Their confusion matrices do not consistently compare; in the case of InceptionV3, it achieved the highest accuracy of classifications made. This can be seen in categories such as cardboard, glass, and metal where the model was able to correctly classify all with 100% accuracy. MobileNetV2 performed well but contained some confusions between glass and metal.

3.11 GRAPHS OF MODEL ACCURACY AND LOSS

The following graph compares the accuracy of different deep learning models used for waste classification. The x-axis represents the model names, while the y-axis represents the accuracy percentage. Models such as MobileNetv2, Vgg16, and Inceptionv3 show superior accuracy compared to AlexNet, which demonstrates lower accuracy.

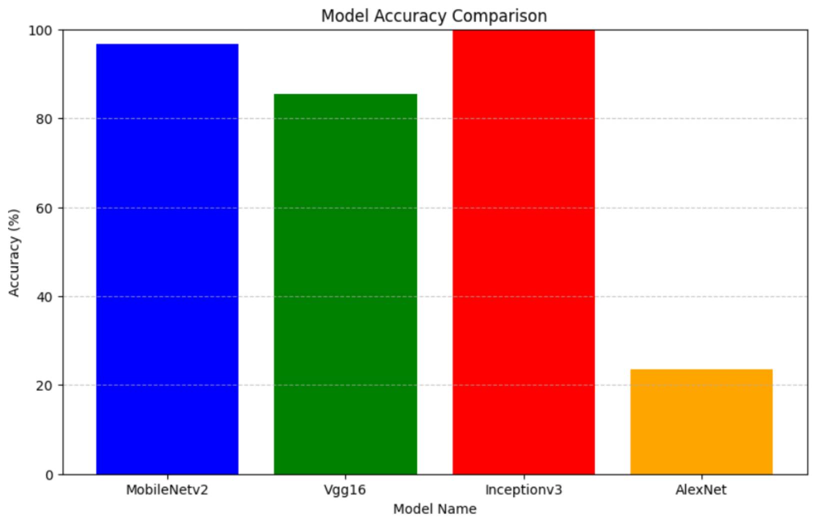


Fig. 5. Accuracy Graphs of Models

metrics of the classification model are recall, precision, F1- score and accuracy, recall takes care of getting every relevant instance correctly identified-this is to say it minimizes false negatives-while precision cares about how accurately the positive predictions are made. It reduces false positives. Therefore, the F1-score is the harmonic mean of precision and recall, which is perfect for an uneven class distribution. The overall correctness is given by the ratio of the number of correct predictions to the total number of predictions, although possibly misleading for imbalanced datasets.

4 RESULT

The study an innovative and eco-friendly waste classification system with MOBWO metaheuristic and deep learning algorithms. The experiment results indicate that the superiority of the proposed model against the traditional models in the aspect of the classification accuracy as well as in the resources usage. The model shows pretty good convergence for any type of waste and consistent performance, which can lead to an improvement in waste management systems. Being both effective in terms of classification accuracy and efficient for the computational costs the approach provides a rather efficient solution to the large scale waste classification problem that can contribute to the environmental sustainability and decrease the burden that waste management facilities face. The future scope includes integrating real-time waste sorting systems, improving optimization efficiency for larger datasets, expanding to more waste categories, and enhancing the model's scalability for broader environmental applications and smart cities.

5 CONCLUSION

Utilizing InceptionV3 with Multi-Objective Beluga Whale Optimization, the suggested waste classification system(MOBWO), is efficient and scalable in dealing with the challenges that modern waste management poses. It has obtained an impressive accuracy of 97.75% and effectively solved problems such as class imbalance and computation efficiency, promoting efficient recycling practices and reducing reliance on landfill. Its adaptability to real-world scenarios and integration into smart city structures make this a massive step toward the sustainable, effective management of waste and conservation of the environment

References

[1] Sayed, G. I., Abd Elfattah, M., Darwish, A., & Hassanien, A. E. (2024). Intelligent and sustainable waste classification model based on multi-objective beluga whale optimization and deep learning. *Environmental Science and Pollution Research*, 1-19.

[2] Bobe, S., Adhav, P., Bhalerao, O., & Chaware, S. (2023, July). Review on Deep Learning Based Biomedical Waste Detection and Classification. In *2023 2nd International Conference on Edge Computing and Applications (ICECAA)* (pp. 1071-1076). IEEE.

[3] Rahmad, M. A., Midi, N. S., Zaini, S. A., Yusoff, S. H., & Mohamad, S. Y. (2018, September). Material classification of recyclable waste using the weight and size of waste. In *2018 7th International Conference on Computer and Communication Engineering (ICCCE)* (pp. 44-49). IEEE.

[4] Tripathi, R., Shetty, H., Patil, K., Ingawale, P., & Trivedi, M. (2023, October). Intelligent Waste Material Classification Using EfficientNet-B3 Convolutional Neural Network for Enhanced Waste Management. In *2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS)* (pp. 132-137). IEEE.

[5] Sireesha Moturi, S.N.Tirumala Rao, Srikanth Vemuru,Grey wolf assisted dragonfly-based weighted rule generation for predicting heart disease and breast cancer, Computerized Medical Imaging and Graphics, Volume 91, 2021, 101936, ISSN 0895-6111, https://doi.org/10.1016/j.compmedimag.2021.101936

[6] Joseph, I. O., Wangare, N. L., Mahoque, T. P., Tewari, P., & Majumdar, S. (2023, July). E-Waste Intelligent Robotic Technology (EIRT): A Deep Learning approach for Electronic Waste Detection, Classification and Sorting. In *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)* (pp. 1-6). IEEE.

[7] Gupta, K., Arora, P., Sharma, R., Yeshfeen, N., Kumari, R., & Bansal, P. (2023, November). Enhancing Waste Classification with Convolutional Neural Networks and Explainability Techniques: Power of Computer in Classification. In *2023 International Conference on Communication, Security and Artificial Intelligence (ICCSAI)* (pp. 569-574). IEEE.

[8] M.Sireesha, S. N. TirumalaRao, Srikanth Vemuru, Frequent Itemset Mining Algorithms: A Survey Journal of Theoretical and Applied Information Technology Vol - 96, No .3, Feb - 2018 ISSN - 1992-8645, Pages – 744 – 755

[9] Puspaningrum, A. P., Endah, S. N., Sasongko, P. S., Kusumaningrum, R., & Ernawan, F. (2020, November). Waste classification using support vector machine with SIFT-PCA feature extraction. In *2020 4th International Conference on Informatics and Computational Sciences (ICICoS)* (pp. 1-6). IEEE.

[10] Kamboj, M. D., Singh, M., Usmani, M., Ahmad, S., & Gaur, M. A. (2023, November). A Comparative Analysis of Algorithms for Effective Waste Classification. In *2023 International Conference on Sustainable Communication Networks and Application (ICSCNA)* (pp. 598-604). IEEE.

[11] M.Sireesha, S. N. TirumalaRao, Srikanth Vemuru, Optimized Feature Extraction and Hybrid Classification Model for Heart Disease and Breast Cancer Prediction International Journal of Recent Technology and Engineering Vol - 7, No 6, Mar - 2019 ISSN - 2277-3878, Pages – 1754 – 1772

[12] Dookhee, S. (2022, December). Domestic solid waste classification using convolutional neural networks. In *2022 IEEE 5th International Conference on Image Processing Applications and Systems (IPAS)* (pp. 1-6). IEEE.

[13] M. Sireesha, Srikanth Vemuru and S. N. TirumalaRao, ”Coalesce based binary table: an enhanced algorithm for mining frequent patterns", International Journal of Engineering and Technology, vol. 7, no. 1.5, pp. 51-55, 2018

[14] Pandey, A., Jain, H., Raj, H., & Gupta, P. (2023, April). Identification and classification of waste using CNN in waste management. In *2023 IEEE 8th International Conference for Convergence in Technology (I2CT)* (pp. 1-6). IEEE.

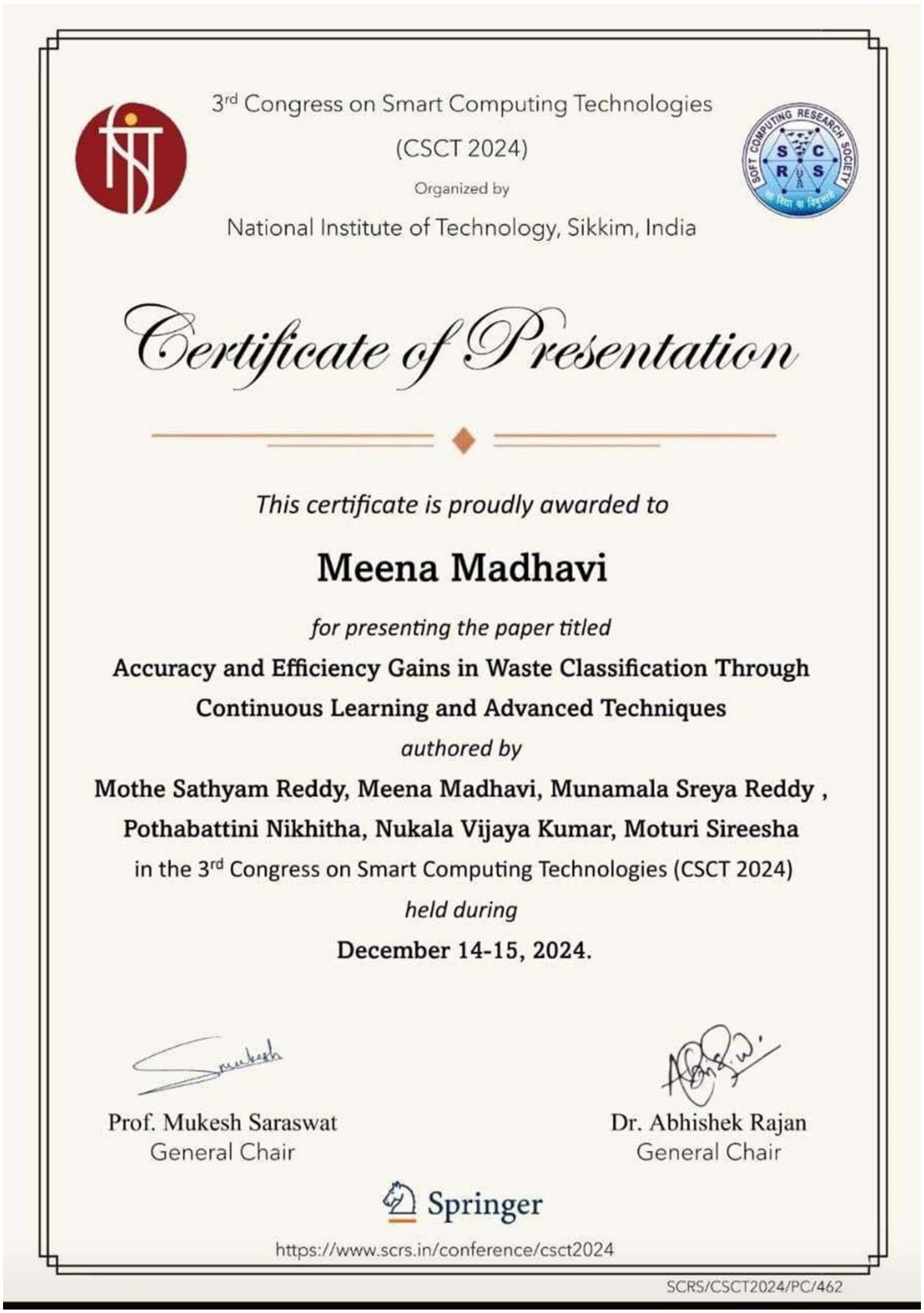
Springer(9).pdf

ORIGINALITY REPORT

9%	4%	7%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to AlHussein Technical University Student Paper	1%
2	Submitted to University of Hertfordshire Student Paper	1%
3	Iyiola Oluwaferanmi Joseph, Njuguna Loice Wangare, Tursélio Pires Mahoque, Piyush Tewari, Sudipta Majumdar. "E-Waste Intelligent Robotic Technology (EIRT): A Deep Learning approach for Electronic Waste Detection, Classification and Sorting", 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2023 Publication	1%
4	ijece.iaescore.com Internet Source	1%
5	Submitted to University of Hull Student Paper	<1%
6	www.ncbi.nlm.nih.gov Internet Source	<1%



3rd Congress on Smart Computing Technologies

(CSCT 2024)

Organized by

National Institute of Technology, Sikkim, India

Certificate of Presentation

This certificate is proudly awarded to

Meena Madhavi

for presenting the paper titled

Accuracy and Efficiency Gains in Waste Classification Through Continuous Learning and Advanced Techniques

authored by

Mothe Sathyam Reddy, Meena Madhavi, Munamala Sreya Reddy , Pothabattini Nikhitha, Nukala Vijaya Kumar, Moturi Sireesha

in the 3rd Congress on Smart Computing Technologies (CSCT 2024)

held during

December 14-15, 2024.

[illegible]

Prof. Mukesh Saraswat
General Chair

[illegible]

Dr. Abhishek Rajan
General Chair

Springer

https://www.scrs.in/conference/csct2024

SCRS/CSCT2024/PC/462