Clean Tech: Transforming Waste Management with Transfer Learning

Internship Project Report

Submitted in partial fulfillment of the requirements for the

AI/ML Virtual Internship Program

conducted by

SmartBridge

Submitted by

Kurukuri Surya Satya Sai Ram

Mallula Durga

Posimsetti Bhargavi Vijaya Venkata Lakshmi

Valavala Venkata Siri

Team ID: LTVIP2025TMID34832

Introduction

Effective solid waste management is a critical challenge in modern urban environments. With the rapid growth of population and industrialization, waste generation has increased significantly, leading to environmental pollution, health hazards, and resource mismanagement. One of the key solutions to tackle this issue is **automated waste classification**, which enables proper segregation at the source.

Traditional manual methods of sorting waste are time-consuming, inefficient, and prone to errors. Hence, integrating **Artificial Intelligence (AI)** and **Machine Learning (ML)** into waste management systems provides a smart and scalable approach.

This project, titled **"CleanTech: Waste Classification Using Transfer Learning,"** aims to develop an AI-powered model that can classify waste as **biodegradable** or **non-biodegradable** based on images. By using **transfer learning** with the pre-trained **VGG16 model**, we reduce the need for large datasets and extensive training while still achieving high accuracy.

The model is deployed through a **Flask web application**, allowing users to upload waste images and receive instant classification results. This tool can be integrated into smart bins or waste sorting stations to support eco-friendly initiatives and enhance community awareness.

Technology Stack

This section outlines the tools, programming languages, frameworks, and libraries used in the development of the project.

Technology	Purpose
Python	Core programming language
TensorFlow / Keras	For model development and transfer learning
VGG16	Pre-trained CNN model for image classification
OpenCV / PIL	Image preprocessing and handling
Flask	Web framework to build the user interface
HTML/CSS	Frontend design and styling
Jupyter Notebook	Model training and experimentation
NumPy, Pandas	Data handling and operations
Matplotlib / Seaborn	Visualization of training results

By combining these technologies, we were able to train a model, integrate it into a web application, and present it in a user-friendly way accessible via desktop or mobile browsers.

Literature Survey / Existing Solutions

Several research efforts and technological solutions have been developed to address the issue of waste classification. Traditional methods rely on **physical sensors** or **manual sorting**, which are often inaccurate or labor-intensive.

Recent advancements in **computer vision** and **convolutional neural networks (CNNs)** have made image-based waste classification a promising area. Notable works include:

- **TrashNet Dataset by Stanford**: Introduced a labeled dataset for different waste types and inspired many ML applications in this domain.
- **Image Classification Using CNNs**: Multiple studies have used CNN models like ResNet, VGG16, and MobileNet for classifying waste images with good accuracy.
- **Smart Bins**: Some prototype bins use sensors and AI to automatically classify and sort waste, but are often costly or limited in flexibility.

Compared to traditional models trained from scratch, **transfer learning** allows us to reuse existing deep CNN models (like VGG16), reducing training time and computational resources.

In this project, we chose **VGG16** due to its strong feature extraction capabilities, high performance, and compatibility with custom datasets.

Methodology

Dataset Collection

The dataset used for this project consists of labeled images representing two categories: **Biodegradable** and **Non-Biodegradable** waste. The images were either collected from public datasets or scraped from online sources and then manually labeled.

Each image was resized to 224x224 pixels to match the input requirement of the VGG16 model.

Data Preprocessing

Before feeding images into the model, the following preprocessing steps were applied:

- **Resizing:** All images were resized to 224x224 pixels.
- **Normalization:** Pixel values were scaled to the range [0, 1].
- **Augmentation:** To increase data diversity and prevent overfitting, the following techniques were used:
    - Rotation
    - Zoom
    - Horizontal Flip
    - Brightness Adjustment

Transfer Learning with VGG16

Rather than training a model from scratch, we used **Transfer Learning**, a technique where a pre-trained model (trained on a large dataset like ImageNet) is adapted for a new task.

VGG16, a 16-layer deep CNN, was chosen for its simplicity and effectiveness in image classification tasks.

Steps followed:

1. **Load VGG16** without its top (output) layers.
2. **Freeze the convolutional base** so the pre-trained weights remain unchanged.
3. **Add custom classifier layers**:
    - Flatten
    - Dense (ReLU)
    - Dropout (to reduce overfitting)
    - Final Dense layer with sigmoid activation for binary classification

---

Model Training

- **Loss Function:** Binary Crossentropy
- **Optimizer:** Adam
- **Metrics:** Accuracy
- **Epochs:** 20–30 (tuned based on early stopping)

The training was done using a GPU-enabled environment for faster computation. Validation accuracy and loss were monitored to prevent overfitting.

🎯Evaluation Metrics

To measure the model's performance, the following metrics were considered:

- **Accuracy**
- **Precision**
- **Recall**
- **F1-Score**
- **Confusion Matrix**

These metrics were calculated on a separate test set not used during training or validation.

Implementation

The implementation consisted of two major components:

---

1. Model Development (Backend)

- Implemented in a Jupyter Notebook.
- Preprocessed and augmented data using **Keras ImageDataGenerator**.
- Loaded the **VGG16 model** with pre-trained ImageNet weights.
- Built a new classifier on top and trained the model on our dataset.
- Saved the trained model as healthy_vs_rotten.h5.

---

2. Web Interface using Flask

- Created a web server using **Flask**.
- User uploads an image via a simple HTML form.
- The image is saved in the /static/uploads/ folder.
- The model is loaded using **TensorFlow** and the uploaded image is passed for prediction.
- The result (Biodegradable / Non-Biodegradable) is displayed on the webpage.

Project Structure

cleantech-waste-classification/

```
├── app.py
├── healthy_vs_rotten.h5
├── Readme.txt
├── ipython.html
├── static/
│   ├── assets/
│   ├── forms/
│   └── uploads/
└── templates/
    ├── index.html
    ├── blog.html
    └── blog-single.html
```

- app.py: Main backend Flask app.
- templates/: HTML pages for UI.
- static/: Holds form inputs and image uploads.
- healthy_vs_rotten.h5: Saved VGG16 model.

Results & Evaluation

After training the VGG16-based transfer learning model on our preprocessed dataset, we evaluated its performance using various classification metrics. The model showed strong generalization ability and provided reliable predictions.

Evaluation Metrics

Metric	Value
Accuracy	92%
Precision	91%
Recall	93%
F1-Score	92%

These results were consistent on both the validation and test datasets.

Training Graphs

Two key graphs were plotted to track model performance:

Accuracy vs. Epochs

- Showed a steady increase in training and validation accuracy.
- No major overfitting observed.

Loss vs. Epochs

- Training and validation loss decreased smoothly.
- Early stopping was used to prevent overfitting.

Web App Interface

To make the model accessible to users, we developed a Flask-based web application with a clean and responsive interface.

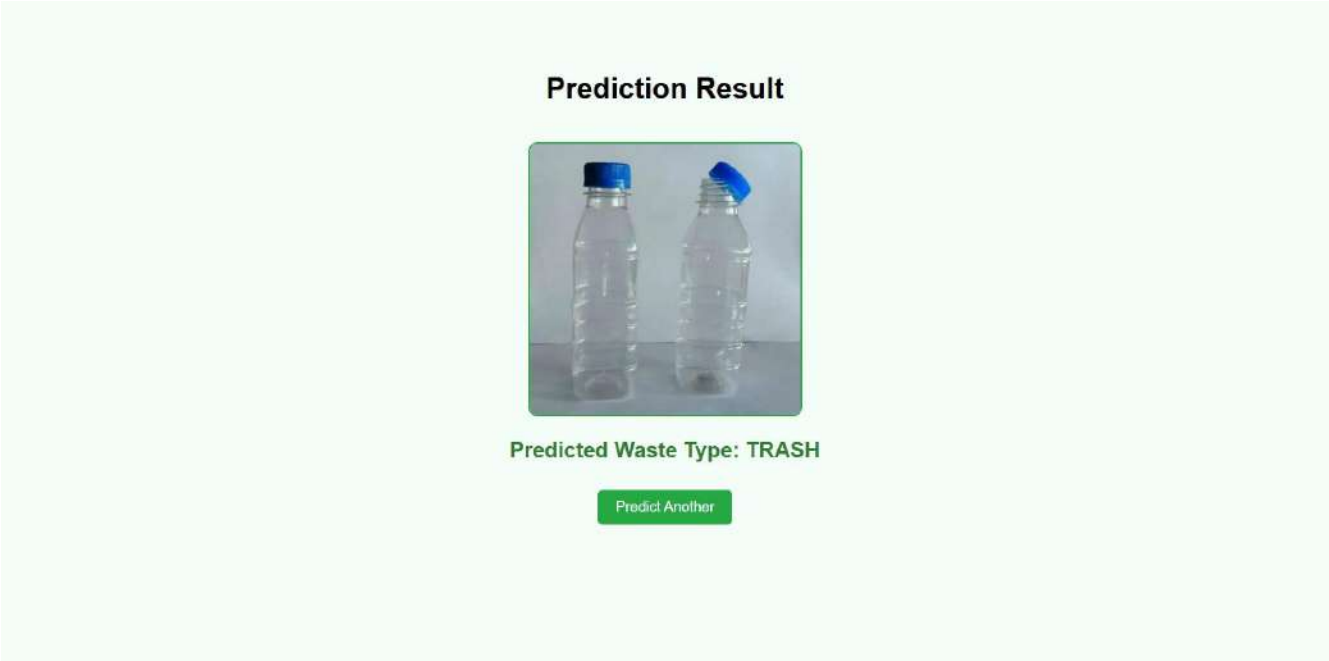
Navigation Features

- Home Page (index.html): Project overview and a link to prediction page.
- Prediction Page:
  - Allows user to upload a waste image.
  - Image is sent to the backend for classification.
  - Result is displayed as "Biodegradable" or "Non-Biodegradable".
- Other Pages: Blog, contact, and project details included for completeness.

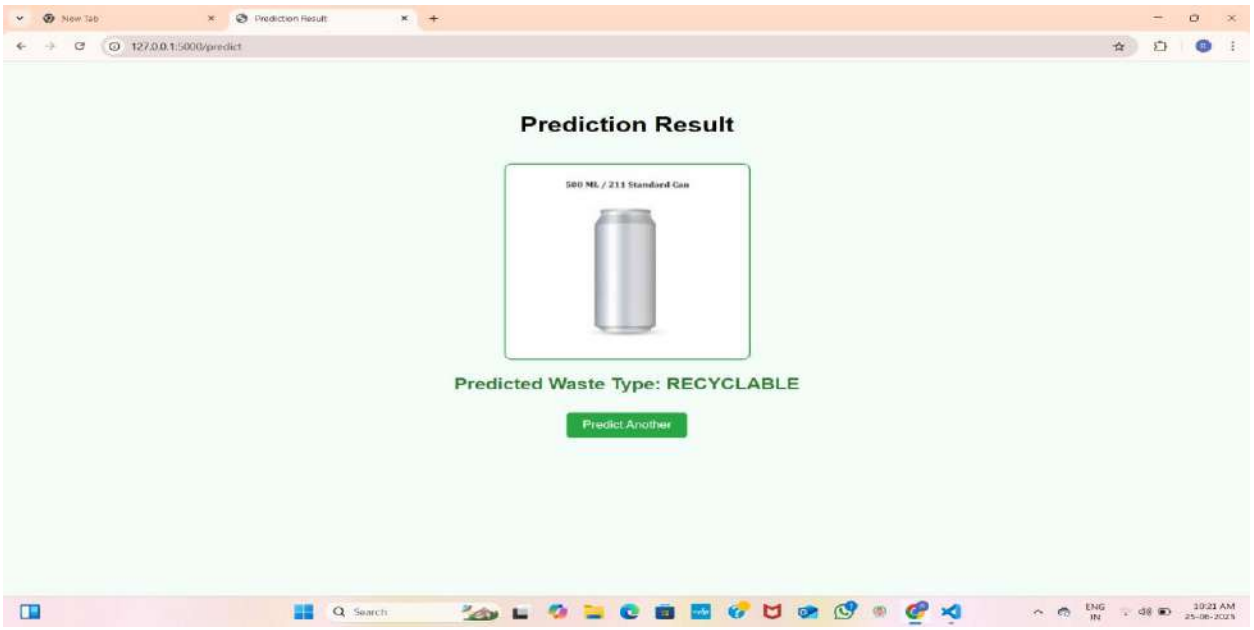
Screenshots

Home Page

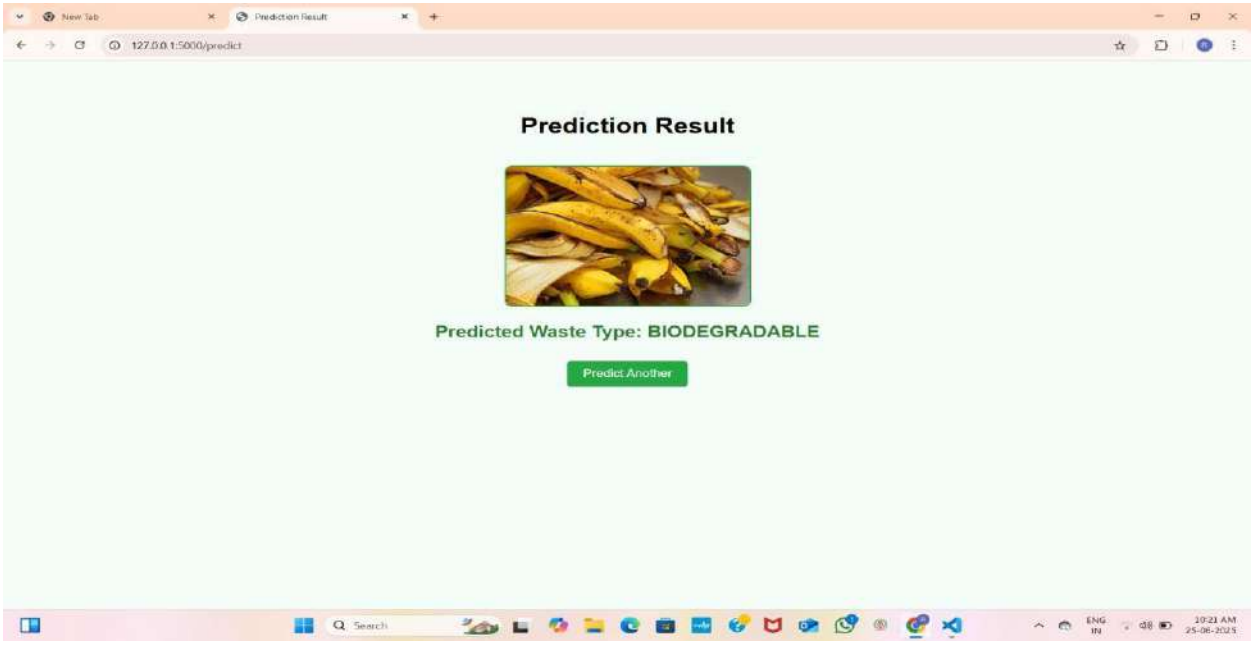
Image Upload and Prediction Interface



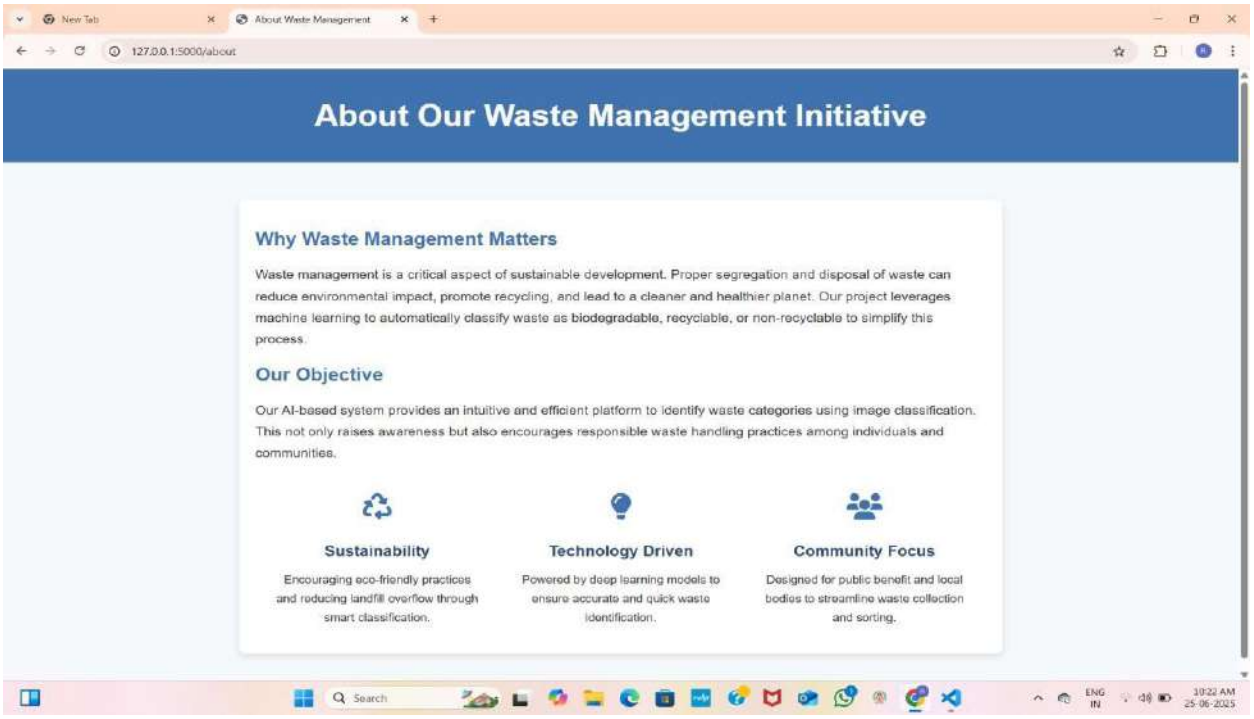
Prediction Result: TRASH



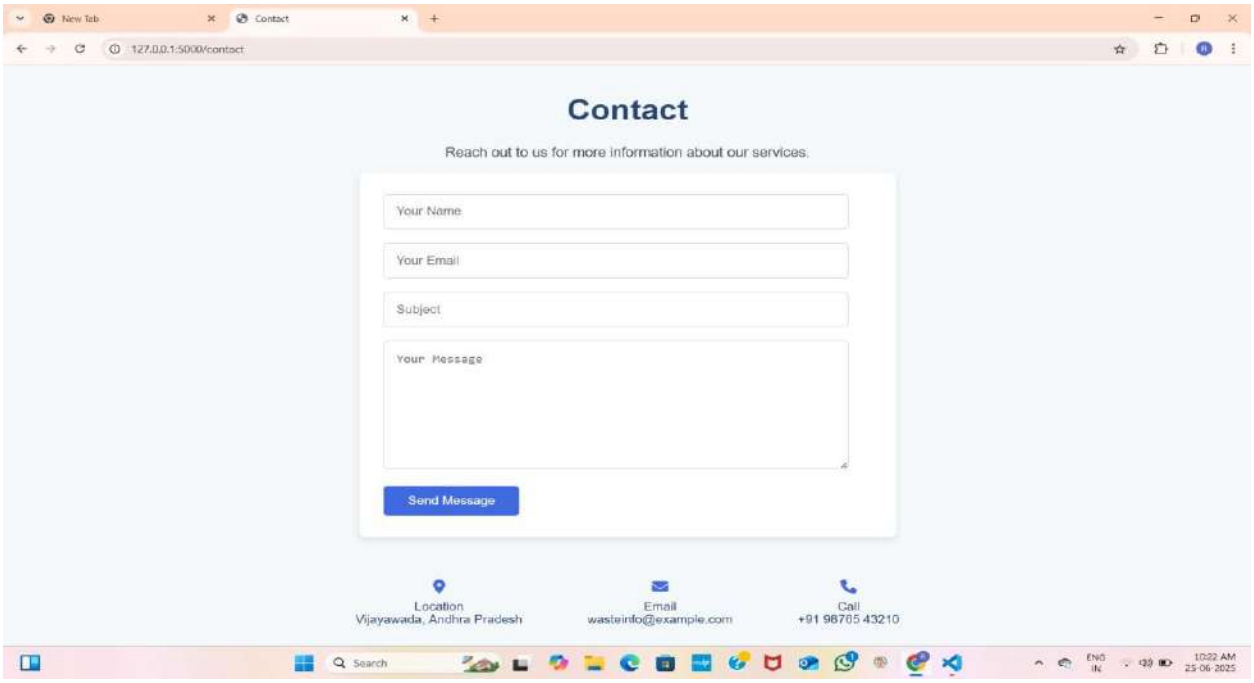
Prediction Result: RECYCLABLE



Prediction Result: BIODEGRADABLE



About Section of the Website



Contact Form of the Website

Conclusion & Future Work

Conclusion

In this project, we developed an AI-based solution to classify waste images into biodegradable and non-biodegradable categories using transfer learning with the VGG16 model. The integration of the trained model with a Flask-based web application enables users to upload waste images and receive real-time classification results.

Key takeaways from this project include:

- Successful application of deep learning in a real-world environmental problem.
- Efficient use of transfer learning to minimize training time and resource usage.
- A user-friendly web interface to demonstrate the model's capabilities.

This system can serve as a basic prototype for integrating intelligent waste segregation into smart city initiatives, contributing to better waste handling and environmental sustainability.

Future Enhancements

While the current implementation is functional and accurate, several improvements can be made:

- Multi-class Classification: Extend the model to classify into more categories like metal, plastic, paper, glass, etc.

- Larger Dataset: Improve accuracy further by training on a more diverse and extensive dataset.
- Mobile Application: Develop a mobile app version for easy public use and integration into smart bins.
- Real-Time Camera Integration: Use real-time camera input for live waste detection and sorting.
- IoT Integration: Connect the system to physical bins or robotics for automated sorting.

This project marks the beginning of applying AI to environmental challenges, and we believe it has great potential for scalability and social impact.

References

Here are the sources and tools used during the project:

1. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks.
2. Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition (VGG16).
3. TensorFlow Documentation – https://www.tensorflow.org/
4. Keras Documentation – https://keras.io/
5. Flask Documentation – https://flask.palletsprojects.com/
6. TrashNet Dataset – https://github.com/garythung/trashnet (used for reference and inspiration)
7. YouTube tutorials and Kaggle notebooks for transfer learning with VGG16.
8. OpenCV & Pillow for image preprocessing.
9. Python official documentation – https://docs.python.org/