frontier tech leaders

# A new generation of tech specialists

# Waste Classification System

## Group-14

Khin Yadanar Hlaing
Myo Myat Htun

# Outline

Concept note and implementation plan:

- Background
- Objectives
- SDG Relation

Data

- Data Collection
- Exploratory Data Analysis (EDA) and Feature Engineering

Model Selection and Training

- Model Evaluation and Hyperparameter Tuning
- Model Refinement and Testing

Results
Deployment
Future Work

# Background

- **The Problem**: Improper waste management drives pollution, greenhouse gas emissions, and resource depletion.
- **Current State**: Manual waste sorting is labor-intensive and inefficient. Automated AI solutions are emerging but often lack hierarchical depth.
- **Literature Gap:**
    - Existing studies (e.g., Khetarpal 2024, Islam 2025) use CNNs for basic categorization but miss hierarchical sorting (e.g., Recyclable vs. Non-Recyclable to Material Type)
    - Most current models handle limited class scopes or fail to generalize across diverse conditions.

# Objectives

- **Primary Goal**: Develop an automated deep learning system to reduce manual labor and increase recycling efficiency.

- **Innovation**: Implement a multi-level classification architecture that simultaneously identifies:
    a. Recyclability (Binary: Recycle vs. Non-Recycle).
    b. Material Type (Multi-class: Glass, Metal, Paper, Plastic, Organic).

- **Technical Aim**: Utilize Transfer Learning (VGG16) to achieve high accuracy with a scalable, real-time deployment capability

# SDG Relation

- **Alignment**: This project directly supports Sustainable Development Goals (SDGs) related to:
  - **Responsible Consumption and Production**: By enhancing waste reduction and recycling efficiency.
  - **Good Health and Well-being**: Through efficient waste sorting and recycling.

frontier tech leaders

# Data

frontier tech leaders

# Data

## Source(s) of the dataset

- **Aggregated Sources:** The dataset was constructed by merging multiple publicly available datasets to ensure diversity and robustness.
    - Kaggle Sources: Waste Segregation Image Dataset, Waste Classification Dataset, and Garbage Classification Dataset.

- **Storage & Volume:** The consolidated dataset is stored in Google Drive and consists of over 4,000 images categorized into 6 initial classes (Glass, Metal, Organic, Paper, Plastic, Trash).



# Data

**Preprocessing steps**

- **Resizing:** All images were resized to a uniform dimension of 224 X 224 pixels to match the input requirements of the VGG16 model.
- **Interpolation:** Nearest-neighbor interpolation was applied during resizing to preserve image details.
- **Normalization**: Pixel values were rescaled from the range [0, 255] to [0, 1] to aid model convergence.
- **Encoding & Conversion:** Target labels were converted to categorical vectors using One-Hot Encoding, and image lists were converted to NumPy arrays for efficient computation

# Data

**Handling Missing Values & Outliers**

- Addressing Class Imbalance (Outliers):
    - **Class Removal:** The 'Trash' category was removed during refinement due to ambiguity and imbalance.
    - **Down-sampling:** The 'Paper' class was manually down-sampled to match the size of other categories, preventing the model from developing a statistical bias toward the most frequent class.
    - **Data Augmentation:** To fix scarcity in other classes, ImageDataGenerator was used to create synthetic variations (rotations, shifts, flips).

# Model

# Model Selection and Training/Testing

1. Model Architecture & Training Details

- Base Model: Selected VGG16 (pre-trained on ImageNet) for its proven ability to extract rich features like edges and textures.
- Transfer Learning Approach:
  - Feature Extraction: The VGG16 convolutional base was initially frozen to retain learned weights.
  - Custom Classification Head: Added a Flatten layer followed by a Dense output layer with Softmax (for multi-class) or Sigmoid (for binary) activation.
- Refinement (Fine-Tuning):
  - Unfreezing: The final convolutional block (Block 5) was unfrozen to adapt high-level features to specific waste patterns
  - Retraining: The model was re-trained with a significantly lower learning rate to avoid destroying pre-trained weights.

# Model Comparison (Initial vs. Refined)

1. Binary Classification Model (Recycle vs. Non-Recycle)

- Initial Model: 81.0% Accuracy. Showed bias toward non-recyclables with lower recall for the recycle class.

- Refined Model: 84.7% Accuracy.

    - Key Improvement: Recall for "Recycle" class improved by 7% (0.74  to 0.81), significantly reducing false negatives.

2. Multi-Class Classification Model (5 Materials)

- Initial Model: 73% Accuracy. Struggled with visually similar classes like Plastic.
- Refined Model: 75% Accuracy.
    - Key Improvement: Plastic recall increased from 0.44 to 0.51; Metal precision improved from 0.62 to 0.69.

    - Impact: Fine-tuning successfully created clearer, distinct features for difficult material types.

# Result

# Evaluation Results

*Fig. The 'Training vs Validation Accuracy' and 'Loss' for Binary Classification (Recycle or Non-recycle)*

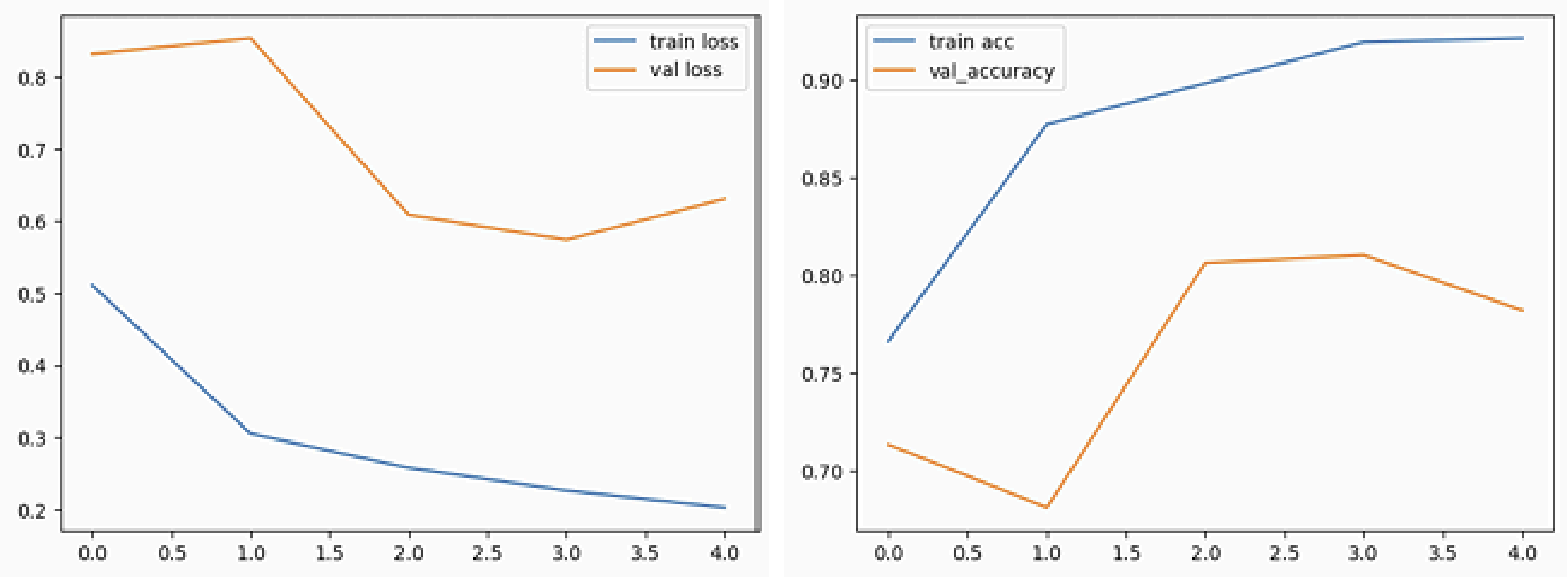
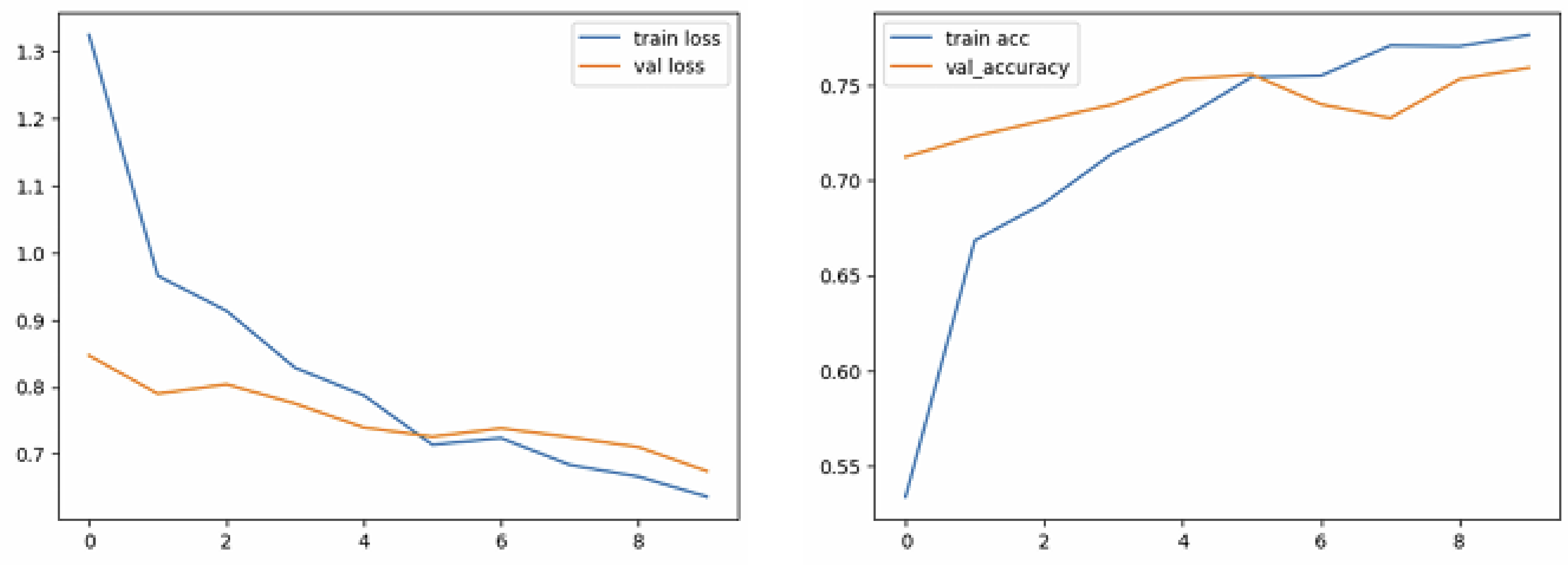


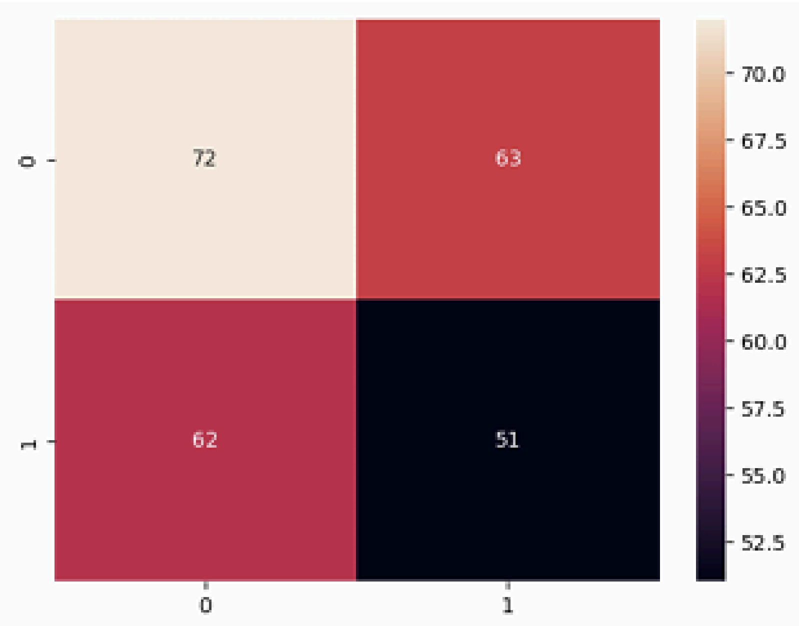

*Fig. The 'Training vs Validation Accuracy' and 'Loss' for Multi Classification*



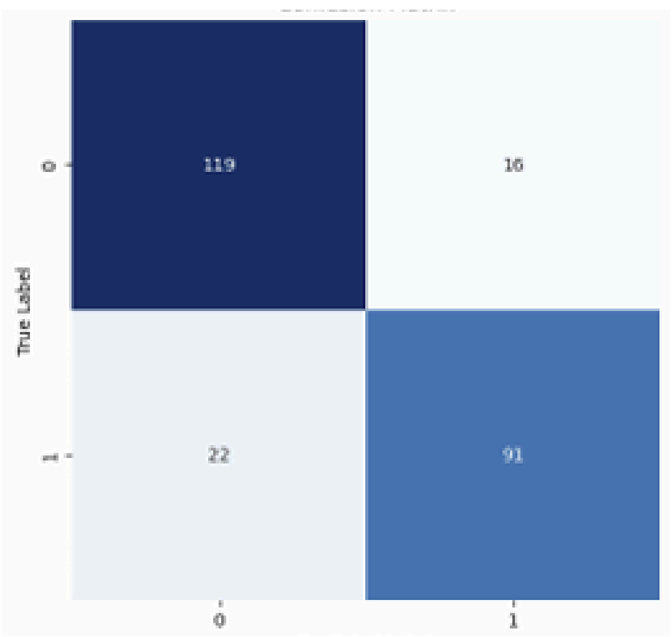
# Evaluation Results

**Model 1: Binary Classification (Recycle vs. Non-Recycle)**

**Before Fine-Tune**

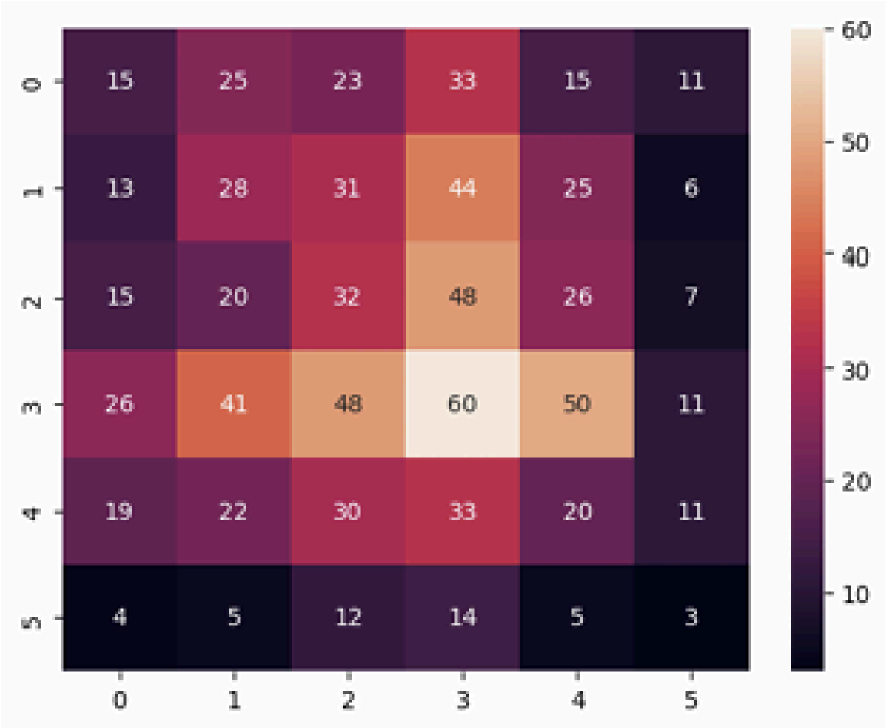



**After Fine-Tune**

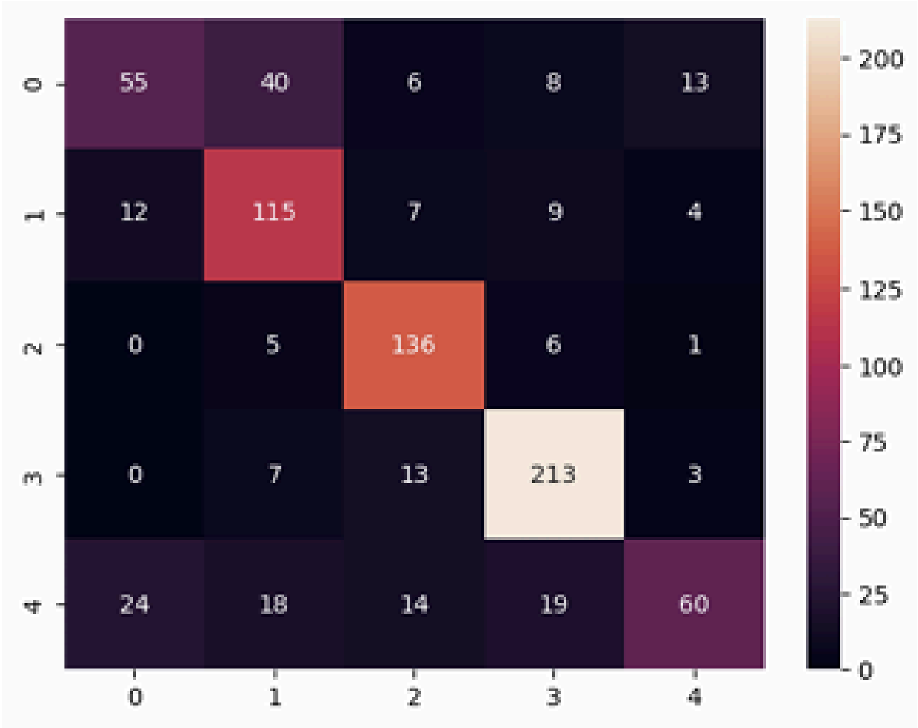



**Model 2: Multi-Class Classification (5 Classes)**

**Before Fine-Tune**




**After Fine-Tune**



# Deployment

# Deployment

Streamlit as the Core: We used Streamlit to transform the Python script (which loads the serialized model and makes predictions) into a dynamic, user-friendly web application.

Deployment Strategy

- Version Control: Project artifacts (app.py, requirements.txt, model.h5) are stored in a GitHub Repository.
- Continuous Deployment (CD): Hosting service is linked to GitHub. Pushing changes to the main branch automatically triggers redeployment.
- Outcome: A live, accessible URL for real-time model interaction, providing immediate value demonstration.

# Our App

## 🔎 Waste Classification System

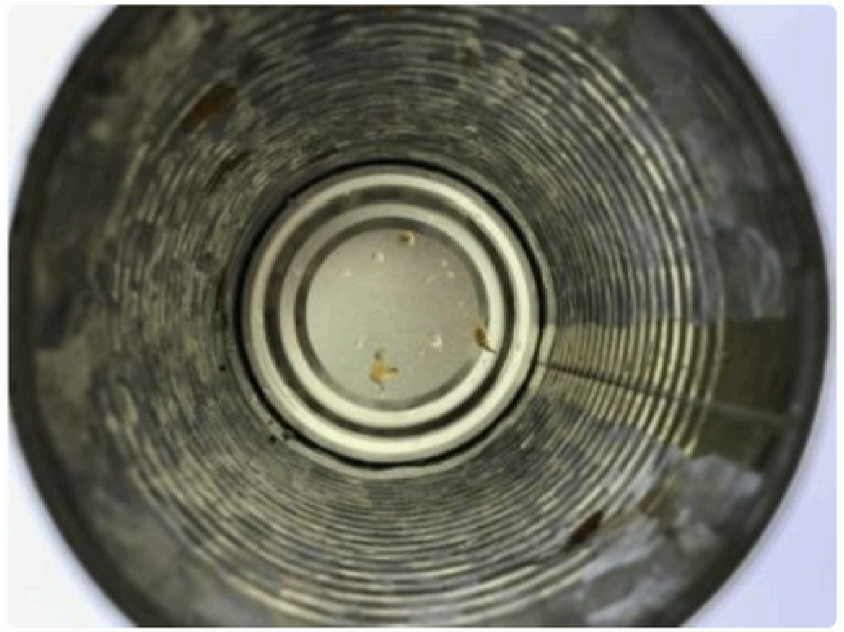
Upload a single image to detect **Recyclability** AND **Material Type** simultaneously.

Choose a waste photo...

Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG

Browse files

metal180.jpg 20.7KB



Uploaded Image

## Analysis Results

### 1. Status

Recycle

Confidence: 75.5%

### 2. Material

Metal

Confidence: 95.7%

### 💡 Decision Support & Sorting Guidance

#### ♻️ Sorting Instruction Card (SDG 12)

✅ **RECYCLE:** Rinse cans/tins (remove residue). Flatten if possible. Ensure no sharp edges.

📊 Full Technical Breakdown & Confusion Summary

### Visual Confusion Summary (Based on Testing Data)

- **Plastic** is commonly confused with **Paper** (if crumpled and white).
- **Metal** is sometimes confused with **Glass** (due to reflections/shine).
- **Organic** can be misclassified as **Non-Recycle** due to contamination features.

### Raw Prediction Probabilities

**Recyclability Probabilities:**

- Non-Recycle: 24.5%
- Recycle: 75.5%

**Material Type Probabilities:**

- Glass: 3.2%
- Metal: 95.7%
- Organic: 0.8%
- Paper: 0.0%
- Plastic: 0.3%

# Conclusion and Futurework

**Conclusion**

- Successful Deployment: Transformed local models into a production-ready, real-time web app hosted on Streamlit Cloud.
- High Performance: Refined VGG16 models achieved 84.7% (Binary) and 75% (Multi-class) accuracy, with significantly improved recall for Recyclables and Plastic through Fine-Tuning.

**Limitations**

- Scope: Hazardous Waste Classification was excluded due to time constraints regarding data collection and training.
- Resources: VGG16 is computationally heavy, requiring Git LFS for storage and caching strategies for efficient cloud deployment.
- Data: Imbalances necessitated removing the ambiguous "Trash" class and manually down-sampling "Paper" to prevent bias.

# Conclusion and Futurework

**Future Work**

- Completing the Architecture: Develop and integrate the missing Hazardous Waste classification level.
- Advanced Detection: Transition to Object Detection (e.g., YOLO) for live video streams and integrate with robotic arms for automated physical sorting
- Edge Deployment: Optimize for low-power IoT devices (e.g., Raspberry Pi) to enable decentralized "Smart Bins"

# Thank you!

frontier tech leaders