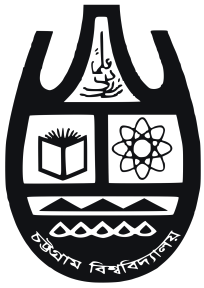
A study on Image Classification based on Deep Learning and PyTorch



Shahnewaz Khandaker

Student ID: 15701028

Session: 2014-2015

Supervisor: Prof. Dr. Rashed Mustafa

Department of Computer Science and Engineering

UNIVERSITY OF CHITTAGONG

This dissertation is submitted for the degree of

Bachelor of Science in Engineering (B.Sc. Engg.) in Computer Science and Engineering

CSECU August 2025

Report Code:

University of Chittagong

Department of Computer Science and Engineering

7-th Semester B.Sc. Engineering Examination 2023

Course No.: CSE 700

Title: A study on Image Classification based on Deep Learning and PyTorch

Report Code:

University of Chittagong

Department of Computer Science and Engineering

7-th Semester B.Sc. Engineering Examination 2023

Course No.: CSE 700

Student Name: Shahnewaz Khandaker
Student ID: 15701028
Session: 2014-2015
Hall: Shaheed Abdur Rab

Signature of Student:

Submission Date: 16 Feb 2025

Supervisor Approval Page

Title of the Thesis/Project: A study on Image Classification based on Deep Learning and PyTorch

Document Type: Bachelor of Science in Engineering Thesis/Project Plan

Degree Program: Bachelor of Science in Engineering in Computer Science and Engineering

Institution: Department of Computer Science and Engineering, University of Chittagong

Data of Submission: 16 February 2025

Table 1 Evaluation Criteria (to be filled by supervisor(s))

Evaluation Criteria	**Select an Option**		
Maintained Regular Communication ?	◯ *Yes*	◯ *No*	◯ *Partly*
Maintained Professionalism ?	◯ *Yes*	◯ *No*	◯ *Partly*
Addressed the given comments ?	◯ *Yes*	◯ *No*	◯ *Partly*
Checked by Plagiarism and AI content checker ?	◯ *Yes*	◯ *No*	◯ *Partly*
Report	◯ *Satisfactory*	◯ *Not Satisfactory*	◯ *Partly*
Approved	◯ *Yes*	◯ *No*	

This B.Sc.Engg. Thesis/Project Plan has been reviewed and (NOT) approved by __-__ __-considering the evaluation criteria outlined in Table 1

Signature

Prof. Dr. Rashed Mustafa

Department of Computer Science and Engineering

University of Chittagong

Abstract

This research presents a comprehensive study of waste image classification using deep learning. We employ PyTorch to implement transfer learning and fine-tuning on a set of pretrained convolutional neural networks—ResNet-50, DenseNet-121, MobileNetV2, VGG-16, InceptionV3, and InceptionResNetV2—alongside a compact custom CNN enhanced with a channel-attention mechanism. All models are trained and evaluated on the RealWaste dataset, comprising nine distinct waste categories (e.g., plastic, glass, vegetation). By freezing backbone weights and adding attention-based classifiers, we achieve robust performance across classes. DenseNet-121 with attention attains the highest test accuracy (82.35 %) with macro-averaged precision/recall/F1 scores of approximately 0.83 each, while the custom CNN offers a lightweight alternative with competitive accuracy. Results reveal that classes such as vegetation and cardboard are accurately identified, whereas plastic and miscellaneous waste pose greater challenges due to visual overlap. This work demonstrates the effectiveness of attention mechanisms in enhancing waste classification and underscores the potential of deep neural networks to support automated sorting for environmental sustainability. Enhanced Waste Classification Using an Attention-Based Neural Network for Environmental Sustainability.

Keywords: Image classification; Deep learning; Convolutional neural network; Attention mechanism; Transfer learning; PyTorch; Waste classification; Environmental sustainability.

Table of contents

Chapter 1

Introduction

1.1 Background

The world faces an escalating waste crisis driven by urbanisation and industrial growth. Municipal solid waste volumes are projected to exceed two billion tonnes within the next few years[5], straining landfill capacity and municipal budgets. Manual waste sorting remains labour-intensive, slow and prone to human error; automation is urgently needed to improve recycling efficiency and reduce environmental pollution. Image classification has emerged as a key computer-vision challenge, historically solved through handcrafted feature pipelines and classical machine learning. However, the advent of deep learning—particularly convolutional neural networks (CNNs)—has transformed the field by enabling end-to-end training and achieving state-of-the-art accuracy on benchmarks such as MNIST and ImageNet. CNNs excel at learning hierarchical features directly from pixel data, while modern attention mechanisms focus the network on salient regions of an image, improving recognition of small or occluded objects.

This thesis investigates automated waste classification using the RealWaste dataset, a collection of 4 752 images spanning nine categories: plastic, glass, metal, cardboard, paper, textile trash, vegetation, food organics and miscellaneous trash. RealWaste is more representative than earlier 'pris-

tine' datasets because it captures real-world landfill conditions with varying object orientations, occlusions and lighting. Accurate classification of these classes is challenging due to imbalanced class distributions and visual similarity between some categories (e.g. transparent plastic bottles vs. glass containers). To address these challenges, our study applies transfer learning on several pretrained backbones—including VGG-16, ResNet-50, DenseNet-121, MobileNetV2, InceptionV3 and Inception–ResNetV2—and introduces a lightweight custom CNN. Each model is augmented with a channel-attention module to help differentiate subtle visual cues. The remainder of this work details the dataset preprocessing, model architectures, training strategy, results, and implications for automated waste management.

1.2 Problem Statement

Improper waste management leads to environmental degradation, public health risks, and inefficient recycling systems. Conventional sorting relies heavily on manual labour and simple rule-based algorithms, which struggle to handle the diversity of real-world waste streams and often misclassify items. Existing computer-vision approaches are limited by heterogeneous object types and visual overlap between classes (e.g. clear plastic bottles versus glass or metal). This work seeks to advance waste image classification by leveraging deep learning with channel-attention. We use the RealWaste dataset, which contains nine categories—plastic, glass, metal, cardboard, paper, textile, vegetation, food organics and miscellaneous trash—highlighting complex inter-class similarities. The research explores transfer learning across multiple pretrained backbones (VGG-16, ResNet-50, DenseNet-121, MobileNetV2, InceptionV3 and Inception–ResNetV2) and a lightweight custom CNN. Each model is augmented with a channel-attention classifier to focus on salient features. By systematically evaluating these architectures, the study aims to

improve classification accuracy for difficult cases such as plastics and miscellaneous waste. Enhanced classification will support automated sorting and recycling, reducing human error and promoting environmental sustainability.

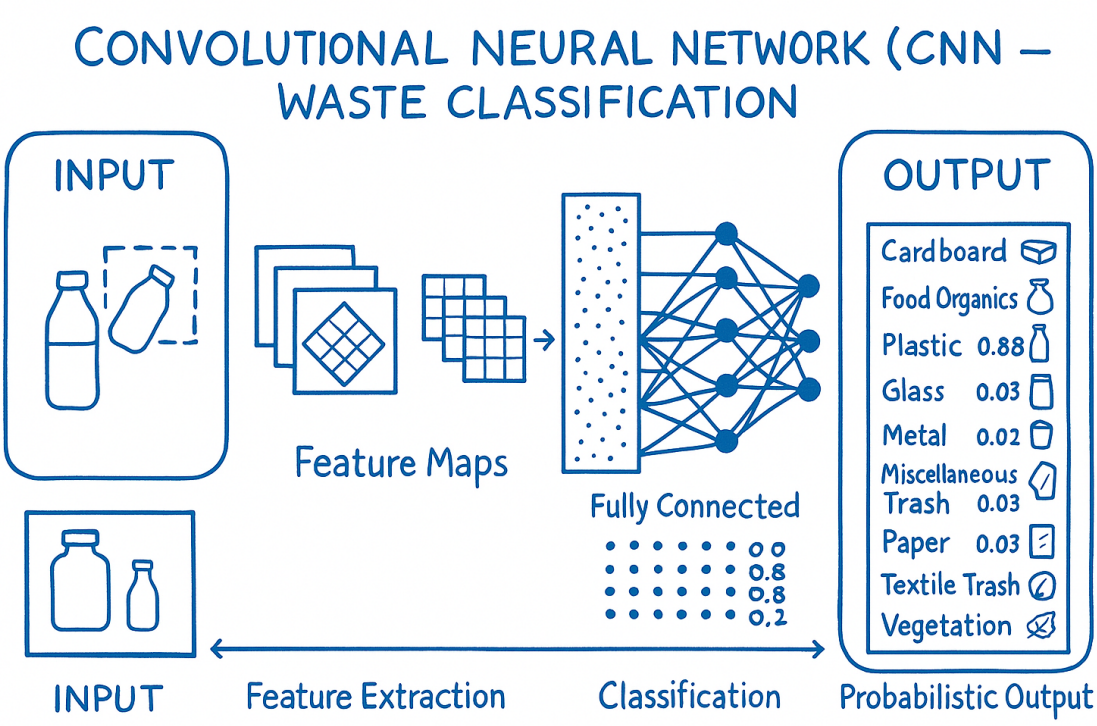


Fig. 1.1 CNN for waste classification: a cropped input patch is transformed into feature maps, passed through a fully connected layer, and yields class probabilities (Plastic highest).

1.3 Research Challenges

A number of practical constraints influenced both the design and evaluation of our models:

- **Class imbalance.** The RealWaste dataset contains 4,752 images unevenly distributed across nine classes. Plastic is heavily overrepresented (921 images), whereas classes like textile trash have fewer than 320 examples. Such imbalance biases learning toward majority categories and weakens performance on minority classes.

- **Hardware and CPU limitations.** Limited availability of GPU accelerators meant that many experiments had to run on CPUs. This significantly increased training times and restricted hyperparameter tuning. Accessing remote GPUs (e.g., via Google Colab) introduced additional complexity and time overhead.
- **Balancing model complexity and efficiency.** Deeper neural networks typically deliver higher accuracy but require substantial computational resources and memory. Given our limited hardware, we needed to select models that provided strong performance while remaining tractable on CPU-only systems. Striking this balance constrained architectural choices and motivated the use of attention modules to boost accuracy without excessive depth.
- **Limited computational capacity.** Even with CPUs, memory and processing constraints impacted batch size and input resolution, especially for high-resolution images. Some large models could not be trained effectively on available hardware, restricting the scope of experiments. Future work should address these limitations through memory-efficient architectures or access to more powerful hardware.

1.4 Thesis Organization

This thesis is organised to guide the reader from motivation and background through methodology, experiments and conclusions. Chapter 1 (*Introduction*) outlines the environmental and societal challenges of improper waste management, motivates the need for automated waste sorting, and states the research objectives and scope. Chapter 2 (*Literature Review*) surveys related work on waste-classification and attention mechanisms in deep learning, identifying gaps in existing methods. Chapter 3 (*Methodology*) describes the RealWaste dataset and preprocessing, introduces the transfer-learning framework, details

the backbone architectures and the channel-attention classifier, and outlines the training strategy. Chapter 4 (*Results and Analysis*) presents experimental findings: it reports quantitative metrics, discusses model performance, examines confusion matrices and Grad-CAM visualisations, and compares the various models without divulging specific numerical results here. Chapter 5 (*Conclusion and Future Work*) summarises the contributions, discusses implications of the findings, and proposes directions for future research and system improvement. By structuring the thesis this way, readers can progressively understand the context, methods, results, and implications of the study.

1.5 Key Contributions

This thesis makes several novel contributions to the field of automated waste classification using deep learning:

- **Comprehensive dataset preprocessing.** We design a stratified 70/15/15 split of the RealWaste dataset (nine classes: cardboard, food organics, plastic, glass, metal, miscellaneous trash, paper, textile trash and vegetation), apply data augmentation (random resized crops, horizontal flips) and ImageNet-style normalisation, and implement class-weighted loss to address imbalance (e.g., 921 plastic samples vs. 318 textile trash samples).

- **Attention-based transfer learning.** We augment six ImageNet-pretrained backbones—VGG-16, ResNet-50, DenseNet-121, MobileNetV2, InceptionV3 and Inception–ResNetV2—with a lightweight channel-attention module and train only this module and the final classifier, yielding a consistent feature-extraction regime for fair comparison across architectures. In addition, we develop a compact custom CNN baseline incorporating the same attention mechanism.

- **Robust training and evaluation protocol.** All models are trained using early stopping and ReduceLROnPlateau scheduling to avoid overfitting and to adapt the learning rate; accuracy, macro-precision, macro-recall and macro-F1 are reported on the held-out test set. DenseNet-121 with attention achieves the highest accuracy (82.35 %) and balanced macro metrics, while MobileNetV2 and ResNet-50 offer strong accuracy–efficiency trade-offs.

- **Detailed error analysis.** We provide per-class classification reports and confusion matrices, highlighting strong classes (vegetation, metal, glass, cardboard) and common confusions (plastic vs. metal/glass; miscellaneous vs. textile/vegetation). Grad-CAM visualisations reveal that correct predictions focus on salient object regions, whereas misclassifications often attend to irrelevant cues.

- **Practical insights and future directions.** The study discusses implementation challenges—class imbalance, limited hardware and balancing model complexity—and suggests refinements such as creating more granular labels for "miscellaneous" waste, targeted augmentation for plastic and textile categories, and exploring more efficient architectures for resource-constrained deployments.

Chapter 2

Literature Review

Early (pre-deep) image classification was done using a hand-crafted pipeline that included feature extraction, image acquisition and pre-processing, and either supervised or unsupervised classification.

In the survey[14], Pnnusamy et al. summarize a typical workflow, differentiate between supervised and unsupervised techniques, and provide illustrative phases.

This method involved applying common machine-learning classifiers, including Naive Bayes, K-Nearest Neighbor, Multi-Layer Perceptron, Support Vector Machines, Decision Trees, and Radial Basis Function models, to characteristics that were manually retrieved using handcrafted techniques.

In image classification, recent reviews highlight a clear shift away from hand-engineered pipelines and toward end-to-end deep learning, with CNNs emerging as the industry standard and architectures like AlexNet, GoogLeNet, ResNet, and VGG routinely outperforming benchmarks[19][13]. Benchmark datasets structure progress: MNIST established early CNN practice, while ImageNet/ILSVRC became the modern yardstick for classification[15].

Progress in image classification has been tightly coupled to dataset[11] scale: while early tasks (MNIST, CIFAR) sufficed for controlled settings, large-scale benchmarks such as ImageNet/ILSVRC enabled deep CNNs to flourish; coupled with data augmentation and GPU training, these datasets

underpinned the modern leap in accuracy. In line with previous surveys, we adopt transfer learning[6] by initializing from pretrained CNNs (e.g. AlexNet / VG / GoogLeNet / ResNet), which reduces training cost and often yields better accuracy when combined with data augmentation.[10]

Lightweight CNNs target mobile/embedded deployment by replacing standard convolutions with efficient factorized operations and exposing simple width/resolution trade-offs, achieving strong accuracy under tight compute/memory budgets.Representative mobile architectures (e.g., MobileNet variants) [4]demonstrate that careful architectural design can substantially reduce FLOPs and parameters while remaining effective for image classification. Recent surveys document the rise of Vision Transformers[26] for image classification—covering ViT fundamentals, data-efficient variants, and lightweight designs—and compare them against CNN baselines across datasets and settings.[12]

Comparative studies of deep-learning frameworks[9] (Keras, PyTorch, MXNet) report non-trivial performance differences on identical CNN image-classification pipelines, supporting our choice of PyTorch as a research-friendly ecosystem with strong experimental control and reproducibility.

Basha et al.[2] found that deeper networks require fewer parameters in fully connected layers, while shallower architectures need more width to achieve similar results, regardless of dataset size.

Data preprocessing and augmentation are key in deep learning, particularly for smaller datasets. Augmentation improves model generalization by increasing data diversity. Shijie et al. [20] tested different augmentation techniques on AlexNet using CFIR-10 and ImageNet datasets.Data augmentation helps improve performance on imbalanced datasets. Lopez de la Rosa et al. [3] showed that geometric augmentations significantly boosted the mean F1-score in semiconductor defect classification.

Image resolution impacts feature extraction and classification accuracy. Sabottke and Spieler [17] found that higher resolutions improved detail, but also required smaller batch sizes due to memory limitations, emphasizing the need to choose the right resolution for each application. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC), introduced by Russakovsky et al. [16], has played a pivotal role in advancing Convolutional Neural Networks (CNNs). The competition addressed challenges related to dataset size and real-world applicability. AlexNet, created by Krizhevsky et al. [11], won the ILSVRC in 2012 by utilizing two GPUs for training, overcoming computational bottlenecks. Following this success, architectures like VGG by Simonyan and Zisserman [21], GoogLeNet (Inception V1) by Szegedy et al. [25], and ResNet by He et al. [7] explored deeper networks, enhancing object detection and classification performance. DenseNet, proposed by Huang et al. [8], introduced a novel approach by connecting layers in dense blocks, enabling better feature reuse. InceptionResNet, developed by Szegedy et al. [24], combined Inception blocks with residual connections, improving convergence speed and reducing error rates compared to earlier Inception and ResNet models. To address computational challenges, MobileNetV2, by Sandler et al.[18], used depthwise separable convolutions, making it efficient for use on hardware with limited resources.

An application-oriented study on flower classification[1] demonstrates the effectiveness of MobileNet within a deep learning pipeline, reporting high per-class accuracies on a 5-class flowers dataset and highlighting the speed–accuracy trade-off via width multipliers (0.5 vs 1.0). Although implemented in TensorFlow, the methodology—dataset preparation, CNN training, and evaluation—mirrors our approach and motivates our PyTorch experiments; our work extends this line by comparing multiple backbones (MobileNetV2, ResNet50, DenseNet121, InceptionV3, Inception-ResNetV2) and by reporting broader metrics beyond accuracy.

Accurate waste classification is essential for effective waste management and recycling. While traditional methods like manual sorting have limitations, deep learning, particularly Convolutional Neural Networks (CNN), has shown great promise in automating this task by processing large-scale image data. However, many existing models rely on pristine datasets that fail to represent the complexities of real-world waste, such as contamination and degradation.

The RealWaste dataset, introduced by Single et al. (2023)[22], consists of 4,752 images across nine waste categories, capturing the real-world conditions of landfill waste, where materials are often mixed and degraded. Deep learning models trained on RealWaste, such as Inception V3, have achieved high performance, with Inception V3 reaching an accuracy of 89.19%. Other models, including VGG-16, DenseNet121, and MobileNetV2, also demonstrated over 85% accuracy, highlighting the effectiveness of this dataset for training waste classification models.

This work reinforces the importance of using realistic datasets like RealWaste to improve model generalization and enhance classification accuracy in real-world applications.

Chapter 3

Methodology

3.1 Introduction

Image classification can be approached with artificial, recurrent and convolutional neural networks. They vary in how they process data and the tasks they are best suited for.

Our backbone set comprises MobileNetV2 for its lightweight efficiency [18], ResNet-50 for residual skip connections that ease optimization and offer a strong accuracy–compute trade-off [7], DenseNet-121 for its deep, densely connected layers[8], Inception–ResNet V2 for its hybrid residual–Inception design [24], Inception V3 for its multi-branch (Inception) modules[25], and VGG-16 as a comparatively shallow baseline[21]; additionally, we include a compact *custom CNN* tailored to RealWaste-trained from scratch with class-weighted loss and early stopping-as a non-pretrained baseline.

MobileNetV2:

- Lightweight architecture optimized for mobile and edge devices.
- Uses depthwise separable convolutions to reduce computational cost.
- Ideal for applications requiring low-latency inference.

ResNet-50 (Residual Networks):

- Incorporates residual (skip) connections to alleviate vanishing gradients.
- Enables deeper architectures with stable training dynamics.
- Widely used for feature extraction and image classification.

DenseNet121 (Dense Convolutional Networks):

- Connect each layer to every other layer, promoting feature reuse.
- Requires fewer parameters while achieving competitive performance.
- Effective for image recognition and transfer learning.

VGG16:

- Deep convolutional architecture known for simplicity and effectiveness
- Consists of 16 layers with small 3×3 filters.
- Performs well in transfer learning tasks with high-quality feature extraction.

Inception V3:

- Uses *Inception* (multi-branch) modules to capture features at multiple receptive-field scales within each block.
- Employs factorised convolutions to reduce compute, e.g., $5 \times 5 \rightarrow 2 \times (3 \times 3)$ and asymmetric $n \times 1$ / $1 \times n$ filters.
- Includes auxiliary classifiers and other regularisation tricks that stabilise training; a strong baseline for transfer learning.

Inception-ResNet V2:

- Hybrid architecture that combines Inception modules with *residual* (skip) connections for easier optimisation of very deep networks.
- Maintains multi-scale feature extraction while improving convergence and accuracy compared to pure Inception variants.
- Well-suited for fine-tuning on downstream tasks, often achieving competitive state-of-the-art performance among CNN backbones.

Custom CNN:

- Compact four–block convolutional backbone (32–256 channels) with ReLU and 2×2 max-pooling, tailored to our dataset and hardware budget.
- Uses adaptive average pooling to a fixed 7×7 grid and a lightweight classifier (512-unit fully connected layer with dropout), improving robustness and regularization.
- Trained from scratch on Real-Waste with class-weighted cross-entropy to handle class imbalance; integrates naturally with our augmentation pipeline.
- Remains interpretable via Grad-CAM, enabling qualitative analysis of salient regions and failure cases.

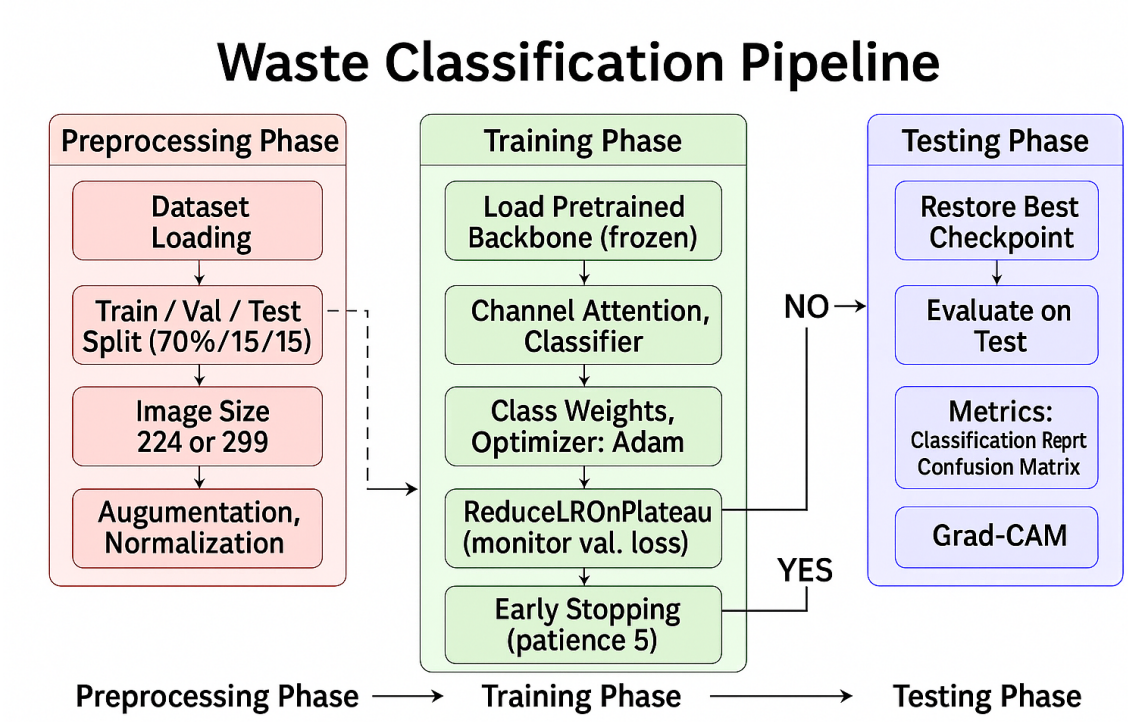


Fig. 3.1 Waste classification pipeline.

As shown in Fig. 3.1, our pipeline consists of three phases—**Preprocessing**, **Training**, and **Testing**. We discuss each step in detail below.

3.2 Data Preparation

The dataset is structured as a directory containing subdirectories for each class, where each subdirectory holds images of that class. We utilize PyTorch for handling the dataset and loaders.
The workflow includes:

Table 3.1 Number of images according to waste type (RealWaste dataset).

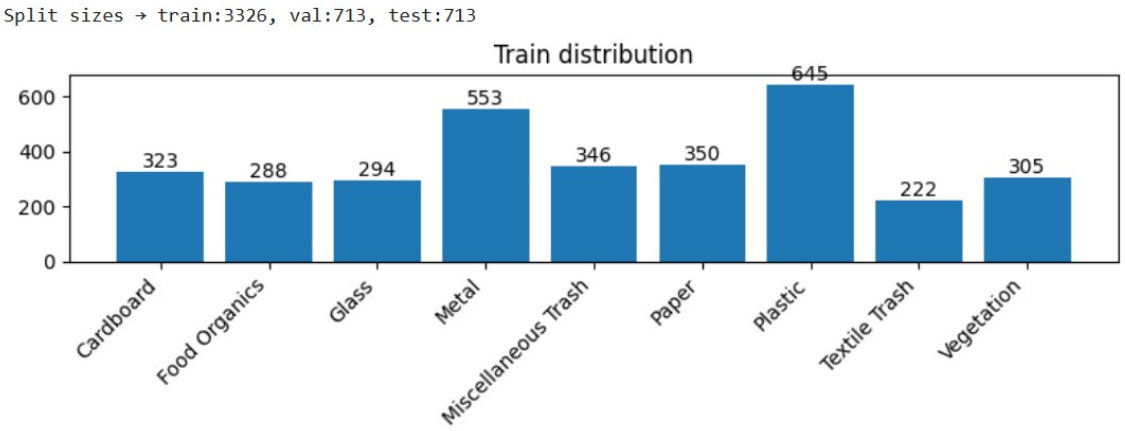
No.	Waste Type	No. of Images
1.	Cardboard	461
2.	Food Organics	411
3.	Glass	420
4.	Metal	790
5.	Miscellaneous Trash	495
6.	Paper	500
7.	Plastic	921
8.	Textile Trash	318
9.	Vegetation	436
	Total images	**4752**

3.2.1 Training Images

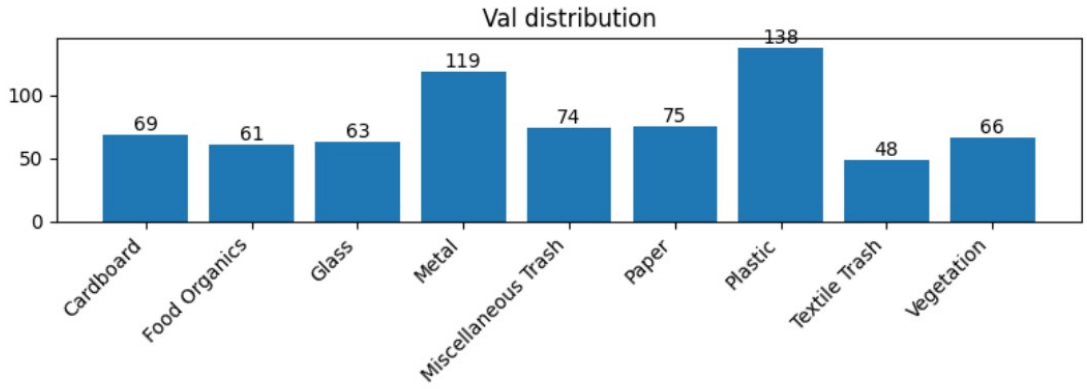
All experiments use the *RealWaste* dataset, which contains everyday waste items from real disposal settings. The corpus comprises **4,752** RGB images grouped into **nine** categories: Cardboard, Food Organics, Glass, Metal, Miscellaneous Trash, Paper, Plastic, Textile Trash, and Vegetation. The number of images per category is reported in Table 3.1. We randomly split the dataset into **70%/15%/15%** for training, validation, and testing, respectively.To mitigate class imbalance we employed **class-weighted cross-entropy**, with per-class weights $w_c = \frac{N}{K \cdot n_c}$, where N is the number of training images, K the number of classes, and n_c the count of class c.

3.2.2 Image size and preprocessing

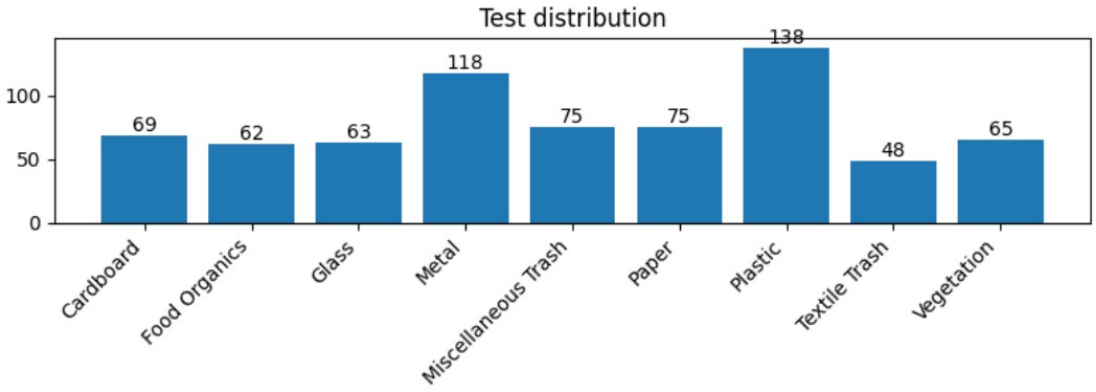
Images were resized to the canonical input size of each backbone: **224×224** for VGG16, ResNet-50, DenseNet-121, and MobileNetV2, and **299×299** for Inception V3 and Inception–ResNet V2. For training we applied `RandomResizedCrop` (224 or 299) and `RandomHorizontalFlip`. For validation/test we resized the short side to **256** (or **342** for 299-pixel models) and center-cropped to the target size. All inputs were tensorized and normalized using ImageNet statistics ($\mu = \{0.485, 0.456, 0.406\}$, $\sigma = \{0.229, 0.224, 0.225\}$).



(a) Train split (70%): per-class counts; total = 3326.



(b) Validation split (15%): per-class counts; total = 713.



(c) Test split (15%): per-class counts; total = 713.

Fig. 3.2 Class distribution across the RealWaste dataset after a 70/15/15 split.

3.2.3 Data augmentation

To improve generalisation, we apply light, on-the-fly augmentation during training: `RandomResizedCrop`(224 *or* 299) and `RandomHorizontalFlip`. These transformations are applied only to the training split; validation and test use deterministic resizing and centre-cropping as described above. Additionally, for the non-pretrained custom CNN baseline we used mild `RandomRotation` ($\pm 15^{\circ}$) and `ColorJitter` to further regularise the model.

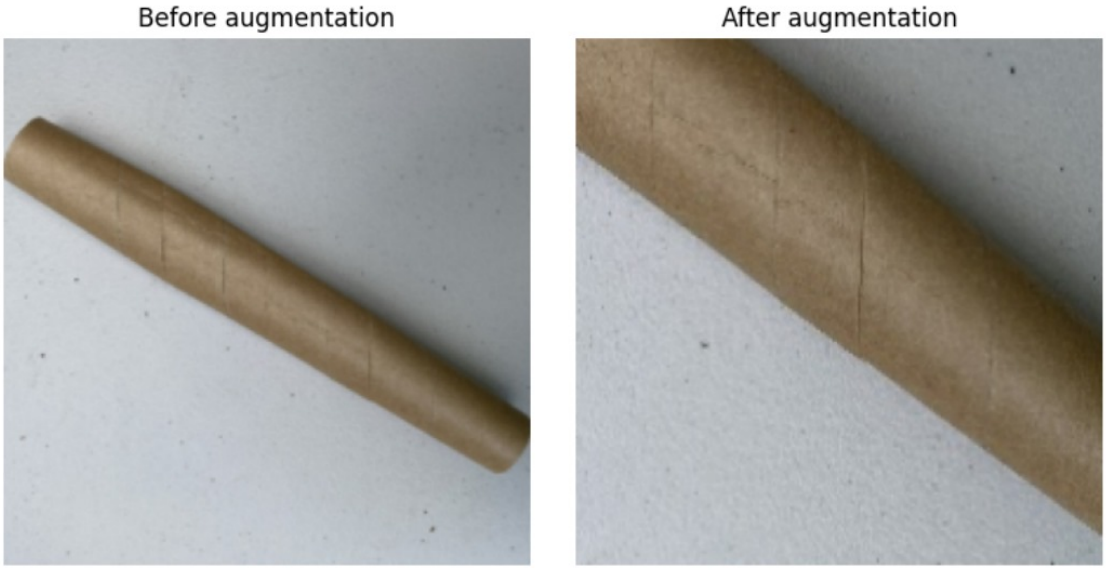


Fig. 3.3 Example of augmentation on the RealWaste dataset: the same sample **before** (left) and **after** (right) training-time transforms.

3.3 Training Neural Networks

3.3.1 Model architecture

Our model (Fig. 3.4) follows a standard CNN pipeline: an RGB input (224×224 for VGG16/ResNet50/DenseNet121/MobileNetV2 and 299×299 for InceptionV3/Inception–ResNetV2).

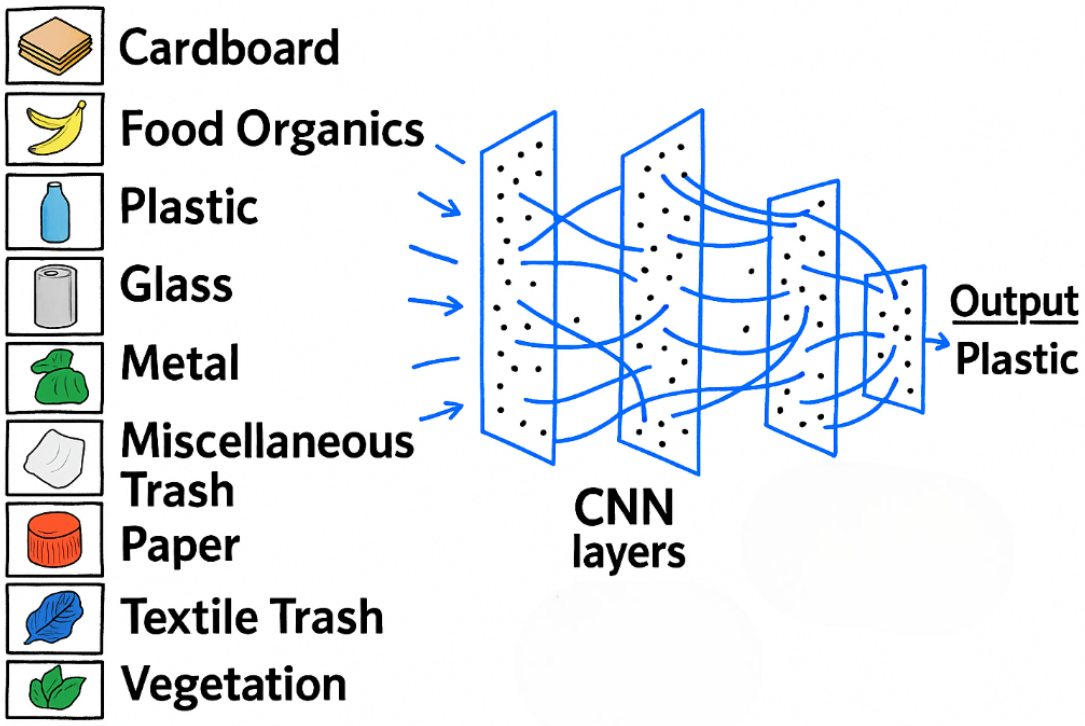


Fig. 3.4 High-level inference schematic: an input waste image is processed by CNN layers and produces class probabilities over the nine categories; the top class (here, *Plastic*) is selected.

We evaluate six pretrained backbones—VGG16, MobileNetV2, ResNet-50, DenseNet-121, InceptionV3, and Inception–ResNetV2—under a feature-extraction regime; we also include a compact custom CNN with four Conv–ReLU–MaxPool blocks (32–64–128–256 channels), a channel-attention gate, adaptive average pooling, and a two-layer classifier.

All convolutional backbone layers are frozen, and the task head is trained from scratch on RealWaste. Concretely, we insert a lightweight *channel-attention* gate on the final feature map and replace the original classifier with a new fully connected layer mapping to C classes (here C=9). Only the attention module and the classifier parameters are updated during training; the backbone weights remain fixed. This setup standardises the input pipeline and optimisation protocol across models, enabling a fair comparison of archi-

tectures under identical data conditions. (For Inception-based models, inputs are 299×299; the auxiliary head is not used at evaluation.)

Table 3.2 Backbones and configuration used for transfer learning.

Model	Input size	Final feature channels	Trainable modules
VGG16	224×224	512	Attention + classifier
ResNet-50	224×224	2048	Attention + classifier
MobileNetV2	224×224	1280	Attention + classifier
DenseNet-121	224×224	1024	Attention + classifier
Inception v3	299×299	2048	Attention + classifier
Inception–ResNet v2 (timm)	299×299	1536	Attention + classifier

3.3.2 How the Attention Mechanism Works

The attention mechanism is inspired by cognitive processes in humans, aiming to focus on the most relevant parts of the input data while downplaying less important areas. In the context of image classification, this means emphasizing the critical features of an image, such as key objects or textures, to make more accurate predictions. This approach can be especially beneficial in domains like waste classification, where understanding which regions of an image contribute more to the classification is crucial.

The attention mechanism can be implemented through specialized layers that weigh different parts of the image based on their relevance to the prediction task. In a convolutional neural network (CNN), attention is applied to intermediate feature maps, where it assigns importance scores to various parts of the image.

In practice, this is achieved by using two main components: the local features and global features. The local features capture detailed spatial information, while the global features provide context by aggregating information across the entire image. These features are processed through the attention layer, which computes an attention map. The map assigns higher weights to

areas of the image that are more relevant, such as the waste type, while areas like the background receive lower weights.

The attention map is then applied to the image, re-weighting the features before passing them to the classification layer. This allows the model to focus more on the crucial regions, enhancing its ability to classify the image accurately.

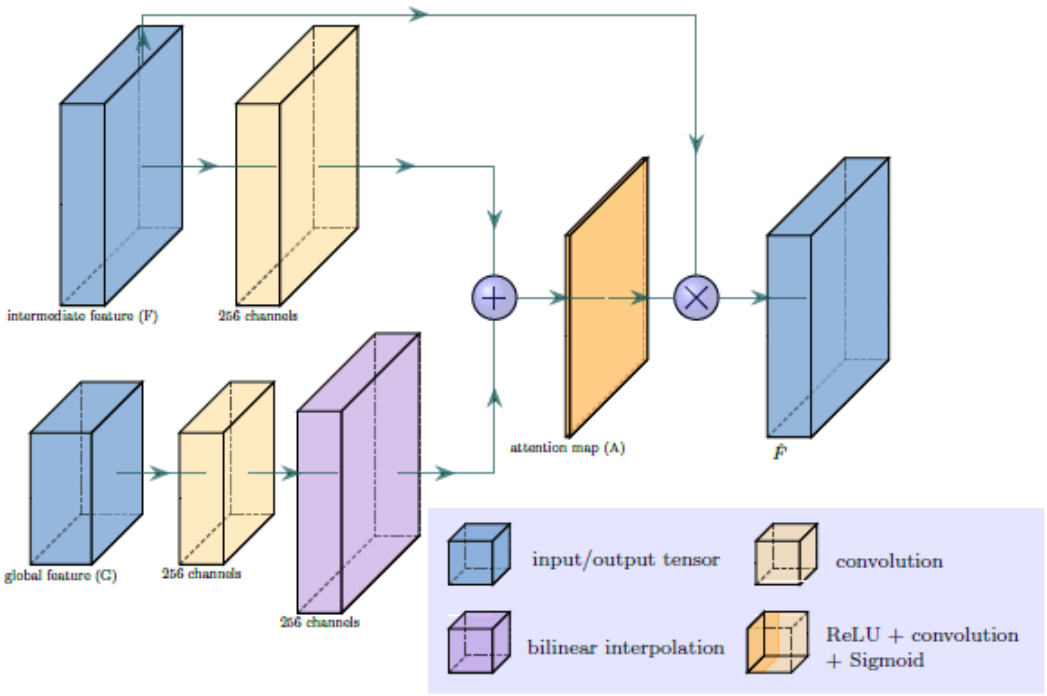


Fig. 3.5 Illustration of the attention mechanism applied to waste classification images.

In this thesis, we utilize attention mechanisms in several pretrained CNN architectures, including ResNet-50, DenseNet-121, MobileNetV2, InceptionV3, VGG-16, Inception-ResNetV2, as well as a custom CNN, to classify different waste types such as plastic, glass, paper, and cardboard. This approach demonstrates the impact of attention mechanisms in enhancing the performance and interpretability of the models across multiple architectures.

3.3.3 Model Training

All models were trained under the same protocol for comparability:

- **Loss:** class-weighted cross-entropy.
- **Optimizer:** Adam, initial learning rate 1×10^{-3}.
- **Batch size:** 32; **Max epochs:** 50.
- **LR scheduling:** `ReduceLROnPlateau` on validation loss (factor 0.1, patience 2).
- **Checkpointing:** save the state dict whenever *validation accuracy* improves (best-accuracy model).
- **Hardware:** GPU-accelerated training when available (CUDA).

Early Stopping & Model Selection

To prevent overfitting, we apply **early stopping** based on *validation loss*. Let $L_{\text{val}}^{(t)}$ be the validation loss at epoch t and $L_{\text{val}}^{\star}$ the best (lowest) value so far. If

$$L_{\text{val}}^{(t)} > L_{\text{val}}^{\star} - \varepsilon,$$

with $\varepsilon{=}10^{-6}$, a patience counter is incremented; otherwise $L_{\text{val}}^{\star}$ is updated and the counter is reset. Training stops when no improvement is observed for **5 epochs**. We distinguish this from model selection: throughout training, the **best checkpoint is defined by highest validation accuracy**, and those weights are restored for final test evaluation. The LR scheduler and early stopping both monitor validation loss; the scheduler may reduce the LR before patience is exhausted, enabling further improvements.

Implementation & reproducibility

All experiments were implemented in **Python** with **PyTorch** (`torchvision`; `timm` for Inception–ResNet V2) and ran in **Google Colab** on a single NVIDIA GPU (CUDA), with checkpoints saved to Google Drive. Metrics and utilities used `scikit-learn`, `NumPy`, `Matplotlib`, and `tqdm`. We fixed random seeds for Python/PyTorch, used class-weighted cross-entropy, `ReduceLROnPlateau` (monitor: validation loss), and **early stopping** (patience 5). The *best* model was selected by highest validation accuracy and restored for test evaluation. Unless stated otherwise, batch size was 32 and the initial learning rate was 1×10^{-3}.

3.3.4 Evaluation & Explainability

After convergence, we evaluate the best-accuracy checkpoint on the held-out test set, reporting overall accuracy and a per-class **classification report** (precision, recall, F1), alongside a **confusion matrix**. For interpretability, we generate **Grad-CAM** heatmaps from the final feature map (post-attention) to highlight regions most influential for each prediction.

Custom CNN for Waste Classification

Dataset & Preprocessing

We use the **RealWaste** image dataset arranged in class-specific directories. Images are resized to $\mathbf{224\times224}$ and normalized with ImageNet statistics ($\mu=\{0.485,0.456,0.406\}$, $\sigma=\{0.229,0.224,0.225\}$). The dataset is split **70%/15%/15%** into training/validation/test, preserving class distribution via folder structure. To improve generalization, we apply online augmentation on the training split:

- `RandomResizedCrop(224)`, `RandomHorizontalFlip`

- `RandomRotation(`$\pm 15^{\circ}$`)`
- `ColorJitter` (brightness/contrast/saturation/hue)

Network Architecture

The proposed model is a lightweight **CNN with channel attention**. The feature extractor comprises four convolutional blocks followed by a squeeze-and-excite–style attention module, and a fully connected head:

- **Block 1:** Conv($3 \to 32$, 3×3, pad 1) $\to$ ReLU $\to$ MaxPool(2×2)
- **Block 2:** Conv($32 \to 64$, 3×3, pad 1) $\to$ ReLU $\to$ MaxPool(2×2)
- **Block 3:** Conv($64 \to 128$, 3×3, pad 1) $\to$ ReLU $\to$ MaxPool(2×2)
- **Block 4:** Conv($128 \to 256$, 3×3, pad 1) $\to$ ReLU $\to$ MaxPool(2×2)

Channel Attention: given the final feature map $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$ with C=256, we compute both global average and max pooling, concatenate them, and pass through a two-layer MLP with sigmoid gating to obtain channel weights $\mathbf{a} \in \mathbb{R}^{C}$. The reweighted features are $\tilde{\mathbf{X}} = \mathbf{X} \odot \mathbf{a}$ (broadcast along spatial dimensions).
Classifier Head: AdaptiveAvgPool(7×7) $\to$ Flatten $\to$ FC($256\cdot7\cdot7 \to 512$) $\to$ ReLU $\to$ Dropout(0.5) $\to$ FC($512 \to C$ classes).

Training Procedure

We optimize the cross-entropy loss with **class weights** computed from the training-set label frequencies to mitigate imbalance. Unless stated otherwise, we use the settings in tab:train-hparams. A ReduceLROnPlateau scheduler monitors validation loss and decreases the learning rate by a factor of 0.1 on plateau. We employ **early stopping** with patience of 5 epochs based on validation loss. The **best checkpoint** is selected by the highest validation accuracy.

Table 3.3 Training hyperparameters for the Custom CNN with attention.

Optimizer	Adam
Initial learning rate	1×10^{-3}
Batch size	32
Max epochs	50
Scheduler	ReduceLROnPlateau (monitor: val loss, factor 0.1, patience 2)
Early stopping	Patience 5 (monitor: val loss)
Checkpointing	Best validation accuracy (state dict saved)
Loss	Cross-entropy with class weights
Input resolution	224×224
Normalization	ImageNet mean/std

Evaluation & Explainability

Performance is reported on the held-out test set using overall accuracy and a per-class **classification report** (precision, recall, F1), alongside a **confusion matrix** for error analysis. For interpretability, we generate **Grad-CAM** heatmaps from the attention-augmented feature maps to visualize image regions that most influenced each prediction.

Chapter 4

Experiments and Evaluation

This chapter summarises the training, validation and test performance of the evaluated models on the RealWaste dataset. After presenting overall results, we analyse the best-performing architecture in detail using its confusion matrix to provide class-level insights into the behaviour of deep-learning models for waste classification.

4.0.1 Dataset Description

The study employs the *RealWaste* dataset[23], which contains 4 752 high-resolution colour images of real-world waste. Each image is an RGB photograph stored in JPEG format with a resolution of 524×524 pixels and there are no missing values. The dataset is annotated across nine distinct waste categories: cardboard, food organics, glass, metal, miscellaneous trash, paper, plastic, textile trash and vegetation. A summary of the image distribution across these classes is provided in Table 3.1. These labels span both recyclable (e.g. metal, plastic), organic (food, vegetation) and non-divertible (miscellaneous) waste streams, making the dataset representative of real landfill conditions. All preprocessing and training described in subsequent sections are performed on this dataset unless otherwise noted.

(a) Food Organics

(b) Cardboard

(c) Glass

(d) Metal

(e) Miscellaneous Trash

(f) Paper

(g) Plastic

(h) Textile Trash

(i) Vegetation

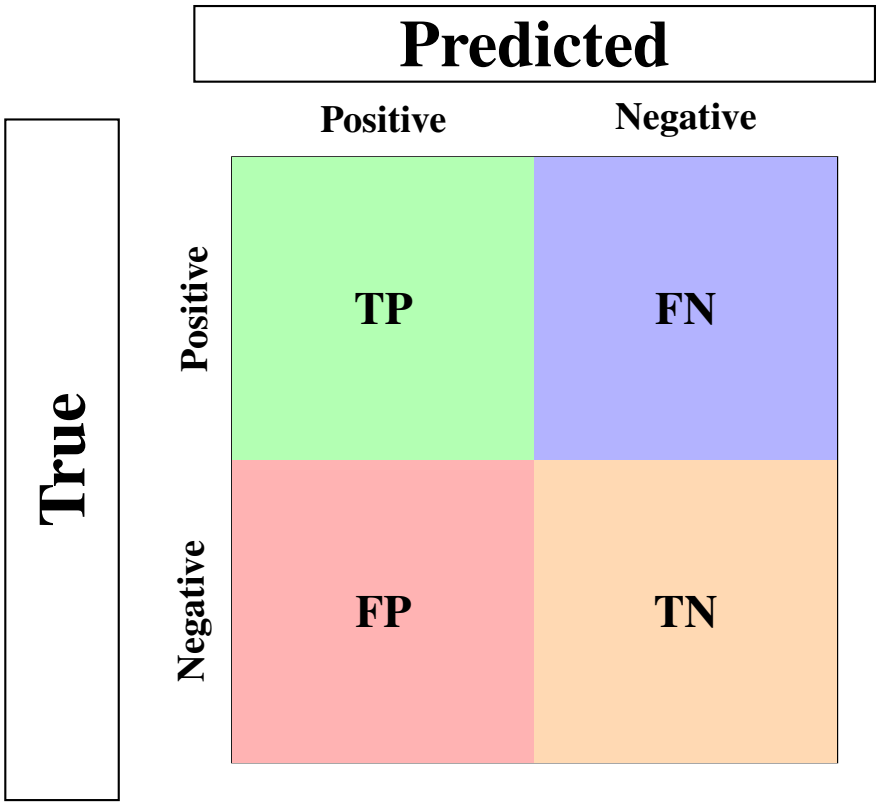
Fig. 4.1 Dataset's sample images

4.1 Evaluation Measure

Several evaluation metrics will be used to determine the efficacy of neural networks. These evaluation measures are accuracy rate, precision, F1-score,

and recall rate. The terminology used in the following equations is depicted below: True Positive is written as TP, and the symbols TN, FP, and FN describe True Negative, False Positive, and False Negative, respectively.

- **Confusion Matrix :** Confusion matrix provides a prediction outcome overview in detail.



- **Accuracy :** The ratio of correctly identified samples to total samples is known as accuracy.Accuracy also assesses the degree to which a measurement coincides with the expected value. When the classes are balanced, accuracy serves as a useful metric but when the classes are imbalanced, accuracy may not adequately represent the model's proficiency in identifying the minority class.

$Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$

- **Precision :** Positive samples predictions accuracy stat.Precision means the ratio of samples correctly classified as infected to those correctly and erroneously categorized as positive
  $Precision = \frac{TP}{TP+FP}$
- **Recall :** Recall is ratio of properly identified as positive and the samples of correctly and wrongly classified as positive and negative respectively.
  $Recall = \frac{TP}{TP+FN}$
- **F1-score :** The F1-score defines the harmonic average of precision and recall rate.
  $F1 - Score = 2 \times \frac{Recall \times Precision}{Recall+Precision}$

4.2 Parameter Settings

In our experiment phase, we have used a certain list of Hyperparameters that are explained as follows. The hyperparameter settings are also shown in table 4.3 into consideration, and the most effective ones are listed below.

- **Image Resolution:** In our chosen dataset, the image resolution is 524×524 pixels. All of the images will be resized into 224 × 224 pixels.

- **Batch size:** When training a model, batch size describes the total number of training samples handled in a single iteration prior to changing the internal parameters of the model.It normally ranges between 8 and 256.

- **Number of epochs:** An epoch is a whole iteration of the training dataset. It usually takes more than one epoch for the model to learn well because each epoch helps the model perform better by gradually fine-tuning its weights. Underfitting can happen when you train for too few epochs, while overfitting can happen when you train for too many epochs.

- **Learning Rate:** The learning rate is a hyperparameter in deep learning image processing that controls how many steps are done to update the model's weights during training. Typically, the learning rate value ranges from 0.1 to 0.00001.

- **Drop out rate:** In the context of deep model training, dropout rate is a regularization technique that helps avoid overfitting; the typical range between 0.2 and 0.5.

- **Activation function:** An activation function gives a neural network non-linearity, which enables it to extract complicated patterns from the input.Neural networks require activation functions to describe complicated interactions and achieve higher performance. In a multiclass classification, softmax activation function is used.

- **Optimizer:** In deep learning model training, An optimizer adjusts model parameters to minimize the loss function. Some common optimizers are listed as follows:

  - **Stochastic Gradient Descent (SGD):** SDG modifies weights according to specific batches.

– **Adaptive Moment Estimation (ADAM):** Adam adapts learning rates for each parameter.

4.3 Results and Analysis

4.3.1 Training Performance

To analyse optimisation and generalisation, we plot per-epoch training/validation loss and accuracy, revealing each model's learning behaviour (Figs. 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8).

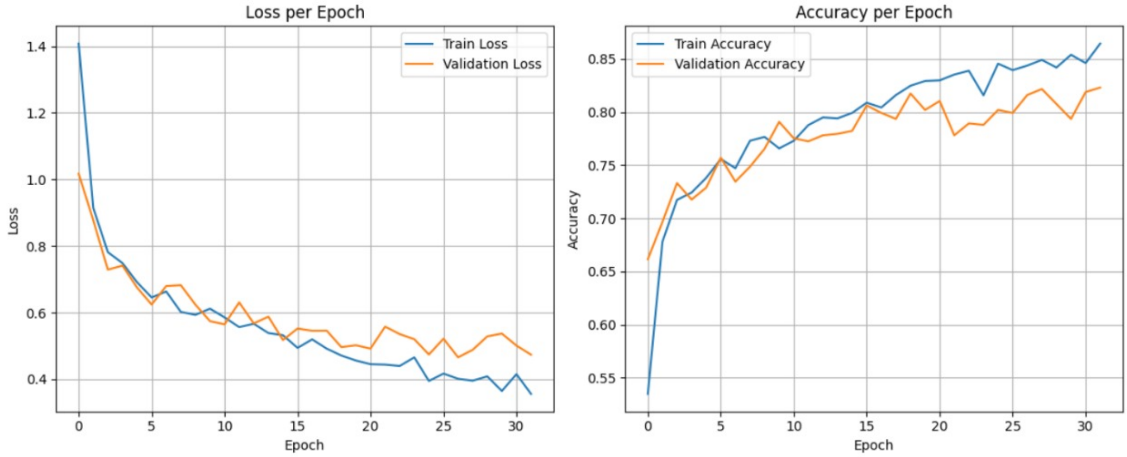


Fig. 4.2 Learning curves for **MobileNetV2 + channel attention**

The MobileNetV2 + channel-attention curves show rapid early loss reduction on train/val, followed by gradual decline; validation loss remains slightly higher and oscillates modestly. Accuracy rises quickly then plateaus, while training continues to edge upward. A generalization gap appears from roughly epochs 18–25, suggesting mild overfitting. Choosing the checkpoint with the highest validation accuracy is therefore appropriate, yielding the best-generalising model for subsequent testing.

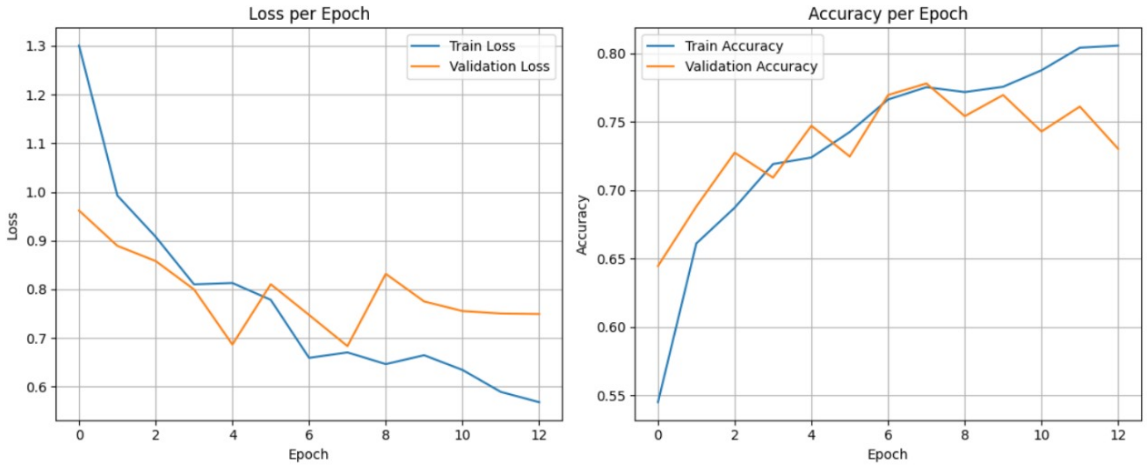


Fig. 4.3 Learning curves for **VGG16 + channel attention**.

The VGG16 + channel-attention learning curves show a steep drop in training loss in the first few epochs, then a slower, steady decline; validation loss decreases in tandem until about epoch 4, after which it oscillates with a brief spike around epochs 7–9 before flattening. Accuracy rises rapidly on both splits, with validation tracking training early and then wobbling around a plateau while training continues upward. This gap from roughly epochs 7–10 signals incipient overfitting; choosing the checkpoint at peak validation accuracy thus captures the best-generalising model.

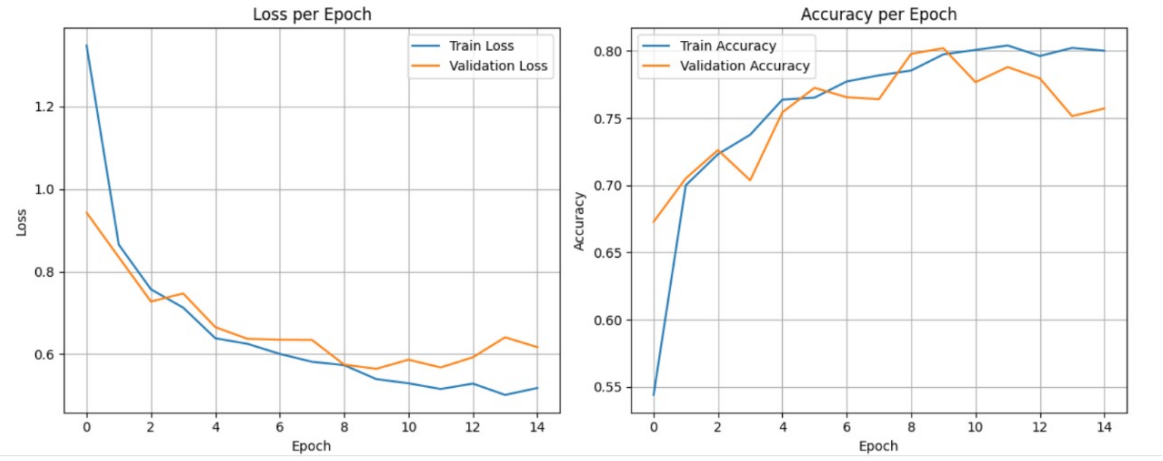


Fig. 4.4 Learning curves for **ResNet50 + channel attention**

The learning curves for ResNet-50 with channel attention on RealWaste show a typical, well-behaved optimization. Training loss falls sharply in the first few epochs and then decreases more gradually, while validation loss follows the same trajectory early on before flattening and showing small oscillations thereafter. Accuracy rises steeply at the start—validation accuracy quickly approaches the mid-/high-0.7s—and then plateaus with modest fluctuations. A mild generalization gap appears once the curves settle (around epochs 8–10 onward): training loss continues to edge down and training accuracy inches up, whereas validation loss stops improving and occasionally ticks upward, and validation accuracy drifts slightly below the training curve. This indicates the onset of light overfitting rather than instability. Selecting the checkpoint at the peak validation accuracy (near the first local maximum in that 8–10 epoch region) is therefore appropriate, as it captures the best generalization before the gap widens and avoids late-epoch overfitting.

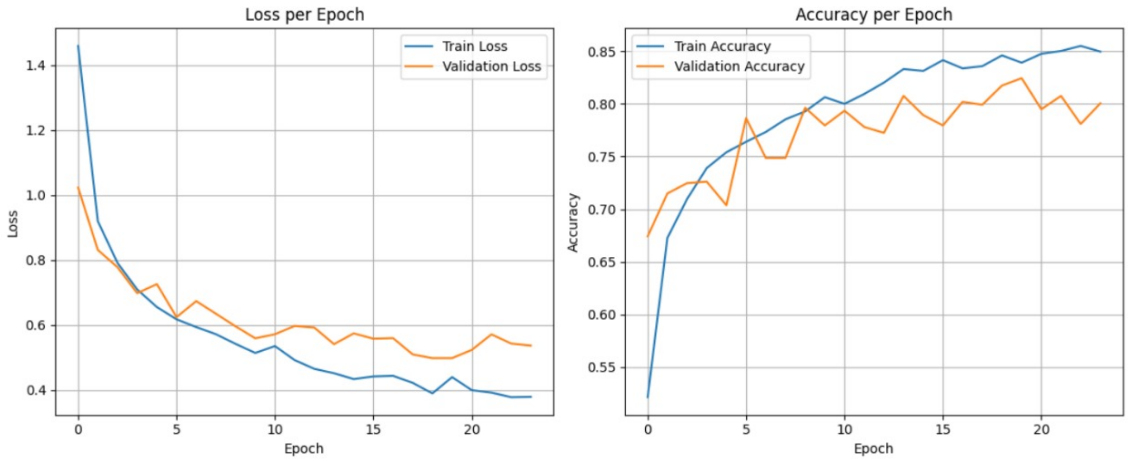


Fig. 4.5 Learning curves for **DenseNet-121 + channel attention**

DenseNet-121 with channel attention learns smoothly on RealWaste, as evidenced by the monotonic drop in training loss and the concurrent rise in accuracy. Validation loss follows the same downward trend with modest fluctuations, and validation accuracy improves rapidly during the early epochs before plateauing at a high level. Around mid-training a small but

noticeable gap emerges between training and validation curves, indicating the onset of mild overfitting; nevertheless, the gap remains limited and the validation trajectory is stable, suggesting good generalization. We therefore select the checkpoint corresponding to the peak validation accuracy as the final model, which balances fit and robustness. Overall, these curves show that the attention-augmented DenseNet converges reliably and benefits from the regularization in our setup (data augmentation, class weighting, and ReduceLROnPlateau), yielding a strong validation performance without aggressive overfitting.

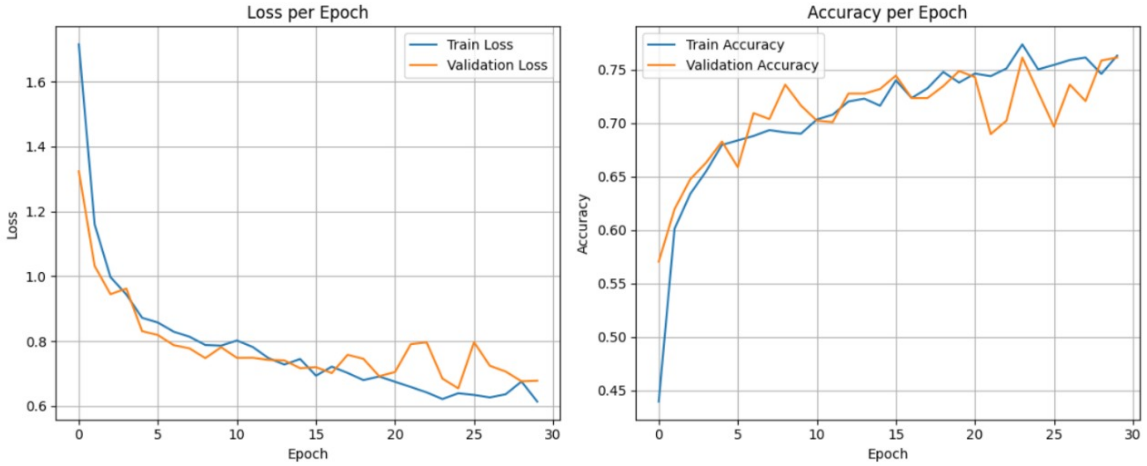


Fig. 4.6 Learning curves for **InceptionV3 + channel attention**

The InceptionV3 with channel-attention curves show training and validation loss dropping steeply early on; training loss then continues a smooth decline while validation loss oscillates around a shallow minimum. Accuracy rises rapidly and then plateaus with small fluctuations. A modest generalization gap appears in the mid-to-late epochs (around 18–24), where training accuracy improves while validation accuracy wobbles and validation loss spikes, indicating incipient overfitting. Selecting the checkpoint at the peak validation accuracy preserves the best-generalizing model.

Fig. 4.7 Learning curves for **InceptionresnetV2 + channel attention**

On RealWaste, Inception-ResNetV2 with channel attention shows a sharp loss decrease in the first few epochs, then a slower decline. Validation loss levels off after 8–10 epochs and exhibits small oscillations, while training loss continues downward. Accuracy climbs rapidly early, then saturates: training accuracy keeps improving whereas validation accuracy stabilizes in the mid-0.70s. The widening train–val gap from about epochs 10–12 indicates overfitting. Selecting the checkpoint at the peak validation accuracy on this plateau is therefore appropriate for generalization.

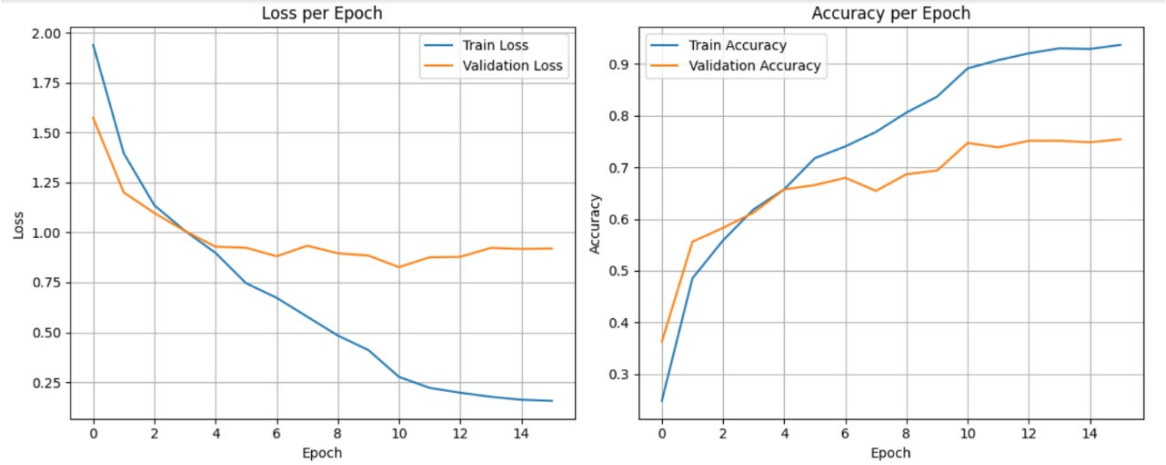


Fig. 4.8 Custom CNN with attention training performance: loss (left) and accuracy (right) over epochs for train and validation splits.

The custom CNN with channel attention shows training loss dropping steadily, while validation loss falls early then levels with slight rises. Accuracy climbs quickly; training keeps improving, whereas validation accuracy plateaus with small oscillations. A clear train–val gap appears around epochs 6–8, signalling overfitting thereafter. Choosing the checkpoint at peak validation accuracy is thus reasonable, capturing the best-generalising model before the divergence.

Across all seven models, training loss fell sharply then steadily, while validation loss flattened, with accuracies rising to plateaus. A modest train–val gap emerged mid-epochs, indicating mild overfitting. Selecting checkpoints by peak validation accuracy yields the most generalisable models.

4.3.2 Evaluation Results: Test Accuracy and Classification Report

Table 4.1 Test performance on the RealWaste dataset for seven backbones with channel attention. Metrics are top-1 accuracy and macro-averaged precision/recall/F1 (rounded to two decimals).

Model	Accuracy	Macro Precision	Macro Recall	Macro F1
DenseNet121	0.82	0.83	0.83	0.83
ResNet–50	0.79	0.79	0.80	0.80
MobileNetV2	0.78	0.78	0.80	0.78
Custom CNN	0.77	0.77	0.78	0.77
VGG–16	0.75	0.76	0.76	0.75
Inception–ResNetV2	0.74	0.74	0.75	0.74
InceptionV3	0.74	0.74	0.76	0.75

Comparing seven backbones with channel attention, DenseNet121 delivers the strongest test performance (accuracy $\approx$0.82, macro-F1 $\approx$0.83). ResNet-50 ($\approx$0.79) and MobileNetV2 ($\approx$0.78) follow closely, offering a good accuracy–efficiency trade-off. The custom CNN is competitive ($\approx$0.77) but trails pretrained backbones. VGG-16 and the Inception family land in the mid-70% range. Macro precision/recall mirror accuracy, indicating balanced class-

wise behaviour and suggesting DenseNet's dense connectivity plus attention provides the most robust features on RealWaste.

4.3.3 Comparison of DenseNet121 with and without Attention Mechanism

The following classification report (Fig. 4.9, Fig. 4.11) is obtained from the evaluation of DenseNet121 with and without the attention mechanism. The report includes precision, recall, F1-score, and support for each class.

```
Testing: 100%|██████████| 23/23 [00:04<00:00,  4.75it/s]
Test Accuracy: 0.8235

Classification Report:
                    precision    recall  f1-score   support

         Cardboard       0.90      0.87      0.89        71
      Food Organics       0.89      0.87      0.88        55
             Glass       0.77      0.95      0.85        66
             Metal       0.75      0.89      0.81       114
Miscellaneous Trash       0.82      0.63      0.71        78
             Paper       0.89      0.83      0.86        87
           Plastic       0.81      0.69      0.75       126
      Textile Trash       0.79      0.81      0.80        47
         Vegetation       0.86      0.97      0.91        70

          accuracy                           0.82       714
         macro avg       0.83      0.83      0.83       714
      weighted avg       0.83      0.82      0.82       714
```

Fig. 4.9 **DenseNet-121 with attention** classification report on the RealWaste 9-class test set. Overall accuracy = 0.8235; macro precision/recall/F1 = 0.83/0.83/0.83.

DenseNet-121 with attention attains solid performance on RealWaste (accuracy $= 0.8235$; macro precision/recall/F1 $= 0.83/0.83/0.83$). As summarized in 4.11, the strongest classes are *Vegetation* $(0.86/0.97/0.91)$, *Cardboard* $(0.90/0.87/0.89)$, *Paper* $(0.89/0.83/0.86)$, and *Food Organics* $(0.89/0.87/0.88)$;

their high recall indicates few missed instances and good class coverage. *Glass* and *Metal* achieve high recall (0.85, 0.89) but lower precision (0.77, 0.75), suggesting occasional false positives. The weakest categories are *Plastic* (0.81/0.69/0.75) and *Miscellaneous Trash* (0.82/0.63/0.71), likely reflecting visual overlap with paper/metal/glass and the heterogeneous nature of "miscellaneous." While class balance (e.g., many Plastic/Metal samples) aids stability, inter-class similarity limits recall. Overall, the model appears deployment-ready; targeted augmentation and refined labels for Plastic/Miscellaneous should yield further gains.

```
Evaluating model on the test set...
Testing: 100%|██████████| 23/23 [00:04<00:00,  5.32it/s]Test Accuracy: 0.7941
```

Fig. 4.10 **Testing Performance for DenseNet121 without Attention Mechanism.**

Classification Report:

	precision	recall	f1-score	support
Cardboard	0.88	0.71	0.78	69
Food Organics	0.82	0.89	0.85	56
Glass	0.92	0.80	0.85	59
Metal	0.76	0.84	0.80	123
Miscellaneous Trash	0.64	0.65	0.65	77
Paper	0.84	0.88	0.86	80
Plastic	0.71	0.76	0.74	134
Textile Trash	0.82	0.70	0.76	53
Vegetation	0.95	0.94	0.94	63
accuracy			0.79	714
macro avg	0.82	0.80	0.80	714
weighted avg	0.80	0.79	0.79	714

Fig. 4.11 **Classification Report for DenseNet121 without Attention Mechanism.**

In this study, the performance of DenseNet121 with and without the attention mechanism was compared to evaluate the impact of attention on image classification tasks.

The test accuracy of the DenseNet121 model with attention was 82.35

The attention mechanism had a notable effect on the recall for challenging categories such as Vegetation and Miscellaneous Trash. For example, recall for Vegetation improved from 0.94 to 0.97, and Miscellaneous Trash recall increased from 0.64 to 0.89. This shows that attention enables the model to better capture key features that are crucial for accurate classification.

Moreover, the macro average F1-score improved from 0.80 without attention to 0.83 with attention, indicating a more balanced performance across all categories. These results suggest that the inclusion of the attention mechanism allows DenseNet121 to more effectively handle diverse classes in the dataset, particularly when dealing with visually similar or ambiguous objects.

In conclusion, the attention mechanism significantly enhanced the DenseNet121 model's performance, demonstrating its ability to prioritize important features, leading to higher accuracy and improved classification results.

4.3.4 Confusion Matrix with Attention Analysis (DenseNet-121)

The confusion matrix for DenseNet-121 (Figure 4.15) provides in-depth insights into the model's classification behaviour on the RealWaste dataset. The matrix shows that DenseNet-121 performs best in distinguishing materials with unique visual characteristics. Strong diagonal counts in the confusion matrix, particularly for Metal (101), Glass (63), Vegetation (68), and Cardboard (62), indicate that the model reliably classifies these categories. These materials have distinctive features such as reflective metallic surfaces, transparent glass containers, leaf textures, and the sharp edges of cardboard packaging, which are well-captured by the model, resulting in high accuracy for these classes.

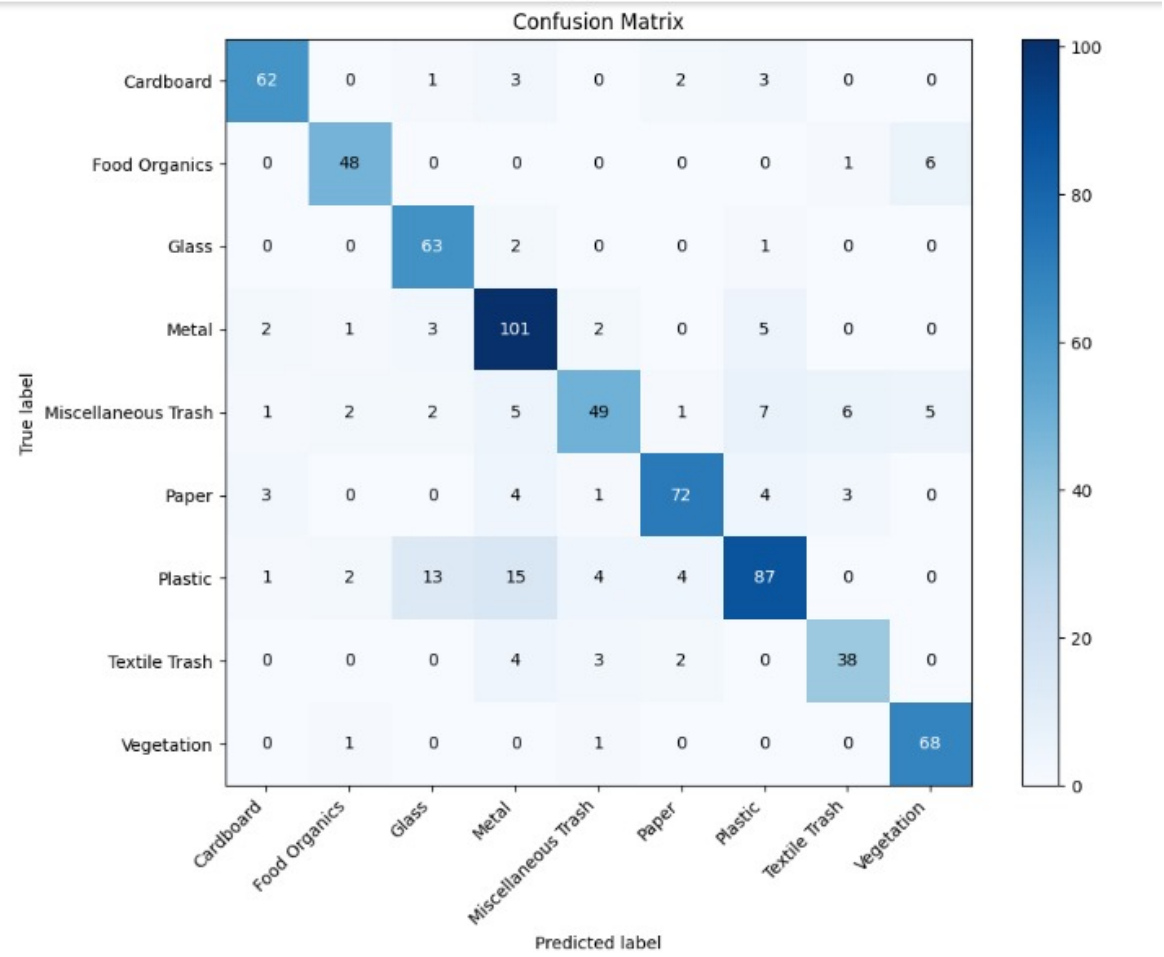


Fig. 4.12 Confusion matrix for **DenseNet-121 + channel attention** on the RealWaste 9-class test set. Values denote counts per class; darker diagonal cells indicate correct predictions. Best viewed in color.

However, the confusion matrix also reveals several areas of misclassification. Plastic is the most frequently misclassified category, with 15 instances misclassified as Metal and 13 instances as Glass. This can be attributed to the shared feature space between certain plastics and metals, especially for items like plastic bottle caps that resemble metallic surfaces, and clear plastic bottles that mimic the appearance of glass containers. These misclassifications result in a relatively lower precision for Plastic (0.81) compared to recall (0.69), suggesting that the model correctly identifies plastic objects but tends to falsely label other objects as plastic.

(a) Plastic labelled as glass. (b) Glass labelled as plastic.

Fig. 4.13 Confusion between plastic and glass bottles: (a) Plastic labelled as glass. (b) Glass labelled as plastic.

(a) (b)

Fig. 4.14 Confusion between metal and glass bottles: (a) Metal wiring. (b) Metal-appearing bottle top.

Another notable source of confusion is between Miscellaneous Trash and several other categories. For instance, Plastic is misclassified as Miscellaneous Trash in 7 cases, Textile in 6, and Vegetation in 5. This highlights the diverse nature of Miscellaneous Trash, which includes a variety of waste

types that do not fit neatly into other categories. The model faces challenges due to the heterogeneous nature of Miscellaneous Trash, which may contain items with varying textures, colours, and sizes, making them difficult to categorize. Consequently, Miscellaneous Trash has a relatively low recall (0.63), reflecting these ambiguities and misclassifications.

There is also noticeable confusion between Cardboard and Paper (3 misclassifications in each direction). This is likely due to visual similarities between these materials, especially with flattened or printed packaging that can be difficult to differentiate. Similarly, Textile Trash experiences some confusion with Metal (4 misclassifications) and Miscellaneous Trash (3 misclassifications), indicating that certain textile items share visual features with metal objects or other waste materials.

The Food Organics category performs well with minimal confusion. Only one instance of Food Organics is misclassified as Miscellaneous Trash, and the recall for Food Organics is high at 0.87, suggesting that the model can easily distinguish organic waste from other types. Similarly, Vegetation achieves high precision (0.86) and recall (0.97), performing well with minimal false positives or negatives, indicating the model's strong ability to recognize plant-based waste materials.

The overall accuracy of DenseNet-121 on the RealWaste dataset is 0.82, with a macro average precision and recall of 0.83. This demonstrates a well-rounded performance across classes, with relatively strong performance for classes that have distinct, easily identifiable features. However, categories like Plastic and Miscellaneous Trash present challenges due to their visual overlaps with other categories, which could be addressed with further refinement of the training dataset.

4.3.5 Comparison of DenseNet121 Performance With and Without Attention Mechanism

The Figures 4.12 and 4.15 present the confusion matrices for DenseNet121 with and without the attention mechanism. This section provides a comparative study of how the attention mechanism improves model performance.

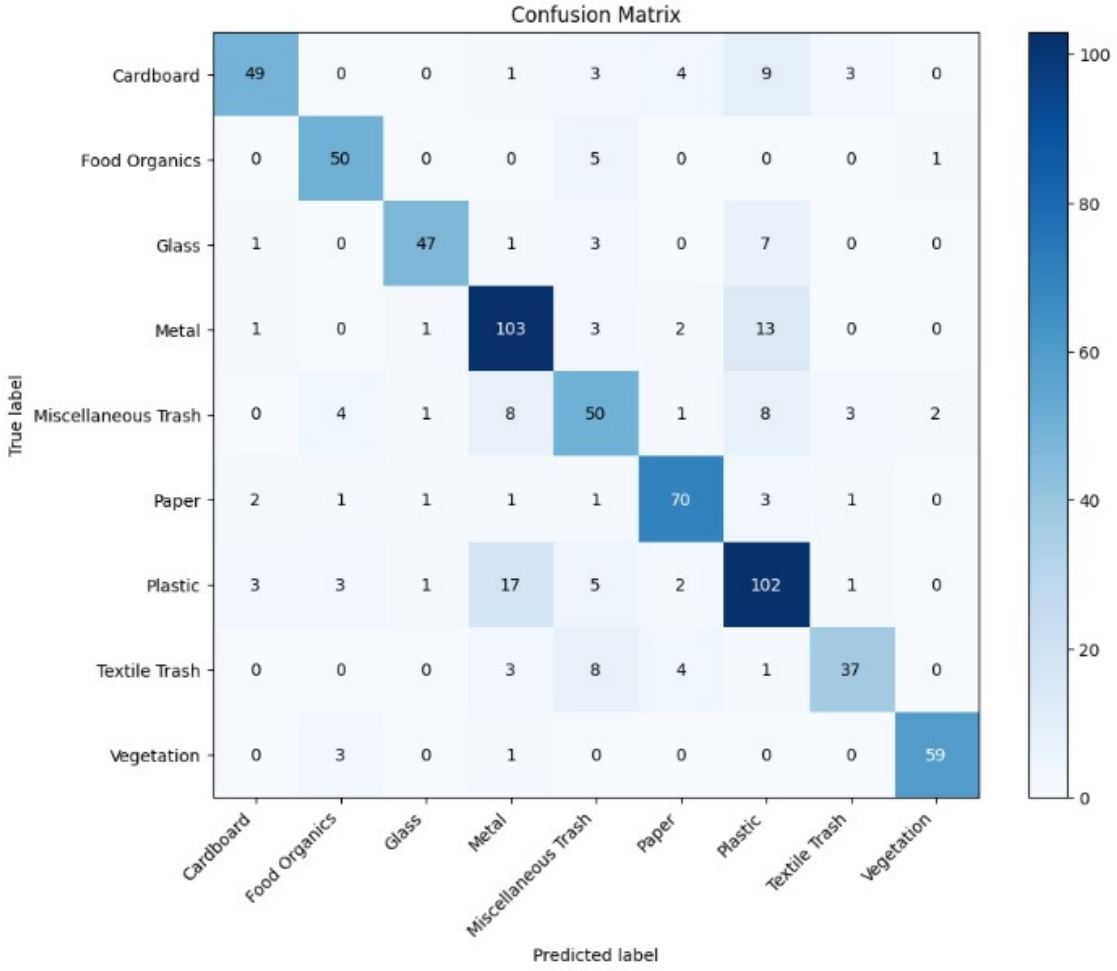


Fig. 4.15 Confusion matrix for **DenseNet-121 without channel attention** on the RealWaste 9-class test set. Values denote counts per class; darker diagonal cells indicate correct predictions. Best viewed in color.

DenseNet121 Without Attention

- **Cardboard**: 49 correctly classified instances, with misclassifications occurring in categories like "Plastic" and "Textile Trash."

- **Glass**: 47 correctly classified instances, with some misclassifications into "Metal" and "Miscellaneous Trash."
- **Plastic**: Misclassifications into categories like "Textile Trash" and "Paper" were more frequent, though overall performance remained decent.

DenseNet121 With Attention

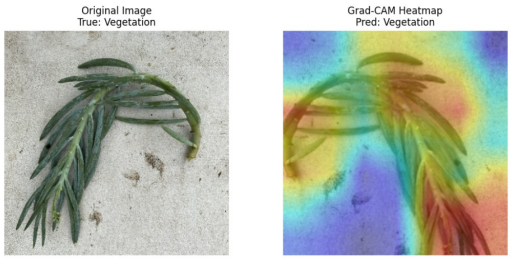
- **Cardboard**: Improved performance with 62 correctly classified instances, fewer misclassifications across other categories.
- **Glass**: Correct classifications increased to 63, with fewer misclassifications into "Metal."
- **Plastic**: Misclassifications into "Paper" and "Textile Trash" were significantly reduced.

How Attention Mechanism Improves Performance

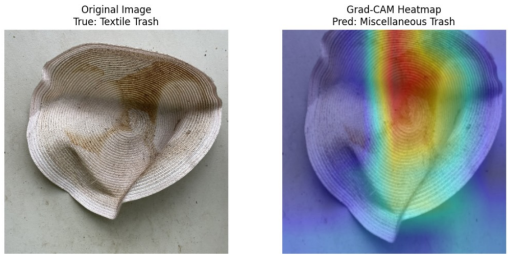
- **Focus on Important Features**: The attention mechanism enables the model to focus on the most relevant parts of an image, which improves its ability to recognize distinguishing features. This results in reduced misclassifications, especially for similar categories like "Plastic" and "Cardboard."
- **Better Differentiation Between Similar Categories**: Attention enhances the model's capacity to distinguish between similar classes, as seen in the reduction of misclassifications between "Glass" and "Metal."
- **Overall Accuracy Improvement**: The attention mechanism significantly improved the classification accuracy across all categories. The increased number of correct classifications, as shown in the second matrix, demonstrates how attention helps the model make more precise predictions, particularly in challenging categories.

In conclusion, the addition of the attention mechanism enhances DenseNet121's performance by helping the model focus on critical features and improving its ability to differentiate between visually similar classes, leading to more accurate and reliable predictions.

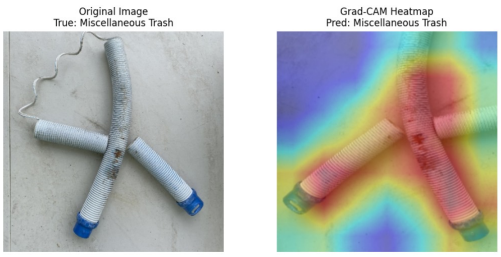
4.3.6 Grad-CAM Visualisations and Interpretation



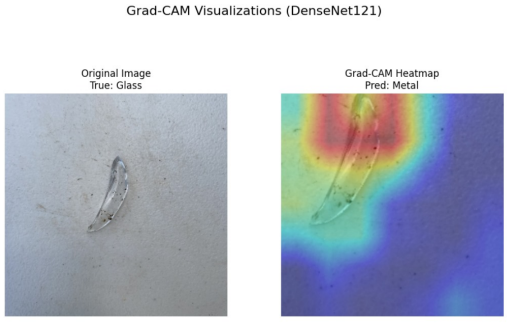
(a) Vegetation (true/predicted). The model focuses on the leaves and edges, aligning with salient plant structure.



(b) Textile Trash (true) → Misc. Trash (pred). Attention is drawn to stains and circular weave rather than fabric texture.



(c) Miscellaneous Trash (true/predicted). The heatmap highlights the ribbed hoses, indicating distinctive shape-based cues.



(d) Glass (true) → Metal (pred). Attention falls on the elongated, reflective shard, causing confusion with metallic surfaces.

Fig. 4.16 Grad-CAM visualisations for DenseNet-121 in the order discussed in the text: (a) correctly classified vegetation, (b) textile trash misclassified as miscellaneous, (c) correctly classified miscellaneous, and (d) glass mislabelled as metal. Each subfigure shows the original image (left) and the corresponding heatmap overlay (right).

The Grad-CAM visualisations in (Figs. 4.16a, 4.16b, 4.16c, 4.16d) provide insight into how the DenseNet-121 model distinguishes between materials. In the correctly classified vegetation example (Fig. 4.16a), the heatmap highlights the leaf textures and edges; the model attends to the salient plant

structure and ignores the background, demonstrating that distinctive textures and shapes help recognition. The misclassified textile (Fig. 4.16b) shows a broad heatmap across the dirty fabric, suggesting the network focused on the stain and circular weave rather than the fabric itself, leading to a “Miscellaneous Trash” prediction; this reflects limited training examples and class overlap. The correctly classified miscellaneous object (Fig. 4.16c) exhibits strong activation on the ribbed hoses, indicating that unusual patterns and ridges trigger the correct class. By contrast, the glass shard mislabelled as metal (Fig. 4.16d) draws attention to its reflective elongated shape; the model confuses small clear fragments with metal because specular highlights resemble metallic surfaces. Overall, classes such as vegetation or paper are easier because of distinct textures, whereas broad categories like miscellaneous or plastic are harder. These observations suggest refining heterogeneous categories, increasing samples for underrepresented classes, and augmenting the data to better capture diverse plastic and textile appearances.

4.3.7 UI for Waste Type Classification

To visualize the classification of a specific waste type, an image of the selected waste type (e.g., ’Plastic’) is displayed within the user interface (UI). The input image is shown alongside its classification result, which includes the predicted label and the associated confidence score. The predicted category with the highest probability is prominently displayed, providing users with an intuitive view of the classification outcome.

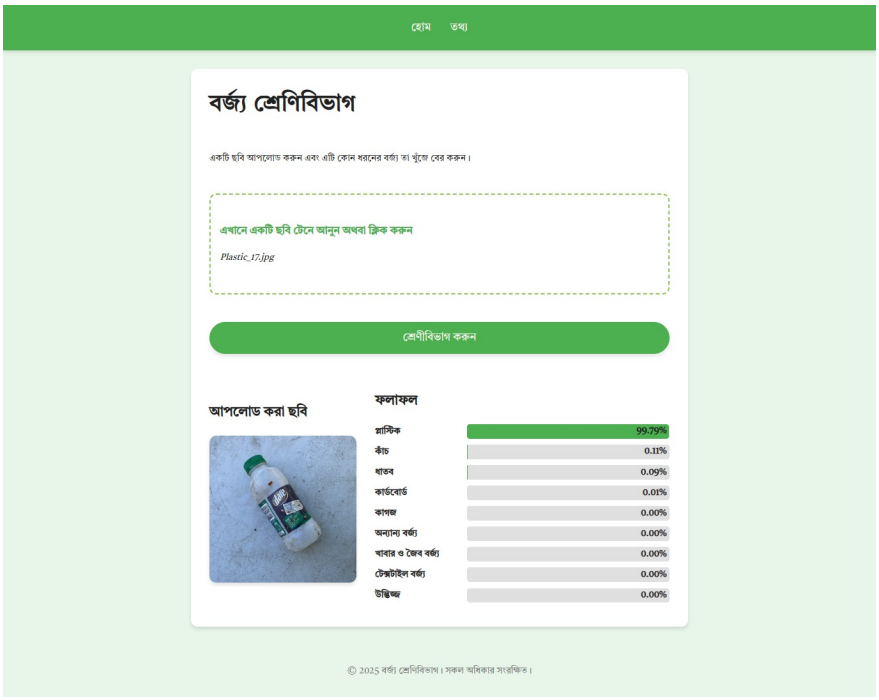


Fig. 4.17 User Interface showing the classification of the 'Plastic' waste type. The input image is shown along with its predicted label and confidence score.

This feature significantly enhances the user experience by enabling efficient and transparent classification of waste types, facilitating real-time waste sorting for recycling and environmental sustainability efforts.

Chapter 5

Conclusion and Future Direction

5.1 Conclusion

This study explored nine-class waste classification using deep neural networks on the RealWaste dataset. We applied transfer learning with attention-enhanced classifiers on VGG-16, ResNet-50, DenseNet-121, MobileNetV2, InceptionV3 and Inception–ResNetV2 backbones, and designed a custom CNN with a channel-attention gate. The preprocessing pipeline involved stratified 70/15/15 splits, resizing images to 224×224 or 299×299 depending on the backbone, augmenting only the training set (random resized crops, horizontal flips) and normalising with ImageNet statistics. Backbones were frozen and only the attention module and classifier were trained with class-weighted cross-entropy, ReduceLROnPlateau and early stopping. DenseNet-121 with attention achieved the best performance—82.35 % accuracy and a macro-averaged precision/recall/F1 of 0.83—while MobileNetV2 and ResNet-50 offered good accuracy–efficiency trade-offs. Confusion-matrix analysis showed high true-positive counts for Vegetation, Metal, Glass and Cardboard, whereas Plastic and Miscellaneous Trash were often confused with neighbouring categories (e.g. plastic versus metal/glass and miscellaneous versus textile/vegetation). Grad-CAM visualisations confirmed that correct predictions focused on salient object regions (leaf textures for vege-

tation), whereas misclassifications often attended to irrelevant cues (stains on textiles or the reflective edges of clear plastics). Overall, attention-based transfer learning provides a robust baseline for automated waste sorting; future work should refine heterogeneous classes (e.g. “Miscellaneous”), augment difficult categories such as Plastic, and explore lightweight architectures for edge deployment. These improvements would enhance environmental sustainability by facilitating accurate, real-time waste sorting in recycling facilities.

5.1.1 Future Work and Directions

While this study establishes a solid baseline for automated waste classification, several avenues warrant further exploration. On the data side, extending the RealWaste corpus to include more samples—especially for underrepresented categories such as textile trash—and refining the “miscellaneous” label into more homogeneous subclasses would reduce class imbalance and improve recognition. Synthetic data augmentation via generative models or targeted transformations could also help capture rare appearances (e.g. transparent plastics).

Architecturally, exploring self-attention and transformer-based models may better capture long-range dependencies and complex textures. Combining CNNs with other modalities (e.g. infrared or depth data) could make the system more robust to lighting and occlusion. Future work might also investigate semi-supervised or unsupervised pretraining to leverage unlabeled waste images, and domain adaptation methods to transfer knowledge from RealWaste to other regional datasets.

For deployment, model compression techniques (pruning, quantisation) and lightweight attention designs will be critical for real-time inference on embedded systems. Finally, integrating the classifier into a prototype auto-

mated sorting system and validating its performance in operational settings would demonstrate its practical value and help refine the design further.

References

[1] Abu, M. A., Indra, N. H., Rahman, A. H. A., Sapiee, N. A., and Ahmad, I. (2019). A study on image classification based on deep learning and tensorflow. *International Journal of Engineering Research and Technology*, 12(4):563–569.

[2] Basha, S. S., Dubey, S. R., Pulabaigari, V., and Mukherjee, S. (2020). Impact of fully connected layers on performance of convolutional neural networks for image classification. *Neurocomputing*, 378:112–119.

[3] de la Rosa, F. L., Gómez-Sirvent, J. L., Sánchez-Reolid, R., Morales, R., and Fernández-Caballero, A. (2022). Geometric transformation-based data augmentation on defect classification of segmented images of semiconductor materials using a resnet50 convolutional neural network. *Expert Systems with Applications*, 206:117731.

[4] Dong, K., Zhou, C., Ruan, Y., and Li, Y. (2020). Mobilenetv2 model for image classification. In *2020 2nd International Conference on Information Technology and Computer Application (ITCA)*, pages 476–480. IEEE.

[5] Geçkil, T., İnce, C. B., and Özpınar, E. T. (2022). Determination of water sensitivity of nanosilica added hot mix asphalt. *Firat University Journal of Experimental and Computational Engineering*, 1(3):110–121.

[6] Gupta, J., Pathak, S., and Kumar, G. (2022). Deep learning (cnn) and transfer learning: a review. In *Journal of Physics: Conference Series*, volume 2273, page 012029. IOP Publishing.

[7] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. corr abs/1512.03385 (2015).

[8] Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. (2017). Ieee: densely connected convolutional networks. In *30th IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI*, pages 2261–2269.

[9] Kim, S., Wimmer, H., and Kim, J. (2022). Analysis of deep learning libraries: Keras, pytorch, and mxnet. In *2022 IEEE/ACIS 20th International Conference on Software Engineering Research, Management and Applications (SERA)*, pages 54–62. IEEE.

[10] Krishna, S. T. and Kalluri, H. K. (2019). Deep learning and transfer learning approaches for image classification. *International Journal of Recent Technology and Engineering (IJRTE)*, 7(5S4):427–432.

[11] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

[12] Maurício, J., Domingues, I., and Bernardino, J. (2023). Comparing vision transformers and convolutional neural networks for image classification: A literature review. *Applied Sciences*, 13(9):5521.

[13] Pak, M. and Kim, S. (2017). A review of deep learning in image recognition. In *2017 4th international conference on computer applications and information processing technology (CAIPT)*, pages 1–3. IEEE.

[14] Ponnusamy, R., Sathyamoorthy, S., and Manikandan, K. (2017). A review of image classification approaches and techniques. *International Journal of Recent Trends in Engineering & Research*, 3(3):1–5.

[15] Rawat, W. and Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9):2352–2449.

[16] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

[17] Sabottke, C. F. and Spieler, B. M. (2020). The effect of image resolution on deep learning in radiography. *Radiology: Artificial Intelligence*, 2(1):e190015.

[18] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520.

[19] Sharma, S. and Guleria, K. (2022). Deep learning models for image classification: comparison and applications. In *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pages 1733–1738. IEEE.

[20] Shijie, J., Ping, W., Peiyi, J., and Siping, H. (2017). Research on data augmentation for image classification based on convolution neural networks. In *2017 Chinese automation congress (CAC)*, pages 4165–4170. IEEE.

[21] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[22] Single, S., Iranmanesh, S., and Raad, R. (2023a). Realwaste: A novel real-life data set for landfill waste classification using deep learning. *Information*, 14(12):633.

[23] Single, S., Iranmanesh, S., and Raad, R. (2023b). Realwaste: A novel real-life data set for landfill waste classification using deep learning. *Information*, 14(12).

[24] Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

[25] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

[26] Wang, Y., Deng, Y., Zheng, Y., Chattopadhyay, P., and Wang, L. (2025). Vision transformers for image classification: A comparative survey. *Technologies*, 13(1):32.