Smart Waste Classification and Recycling Assistant

Name: Habiba Helal Mohammed
Faculty: Faculty of Computers and Data Science, Alexandria University
Department: Artificial Intelligence
Training Provider: MAIM Company for Digital Solutions **Track:** Computer Vision
Report: Final Project Submission

Project Overview : Smart Waste Classification and Recycling Assistant

Problem Statement

Improper waste sorting poses significant **environmental and recycling challenges**. Manual sorting is inefficient, error-prone, and labor-intensive. With the growing demand for sustainable cities, an automated and intelligent system is essential to improve recycling efficiency, reduce human workload, and support sustainability goals.

This project leverages **deep learning and computer vision** to build an automated waste classification and detection assistant. The system can identify and classify multiple types of waste, serving as a foundation for **smart bins, recycling plants, and municipal services**.

Dataset

The project uses the **TrashNet Dataset**, which contains images of waste categorized into six classes:

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

Dataset Link: [Kaggle – Garbage Classification Dataset]

Methodology

The project was implemented through **five main components**, each captured in a separate notebook:

1. **Data Preprocessing & Augmentation**
   - Cleaning, resizing, normalization.
   - Classical augmentation (rotation, flips, brightness, etc.).
2. **CNNs & Transfer Learning**
   - Baseline CNN architecture.
   - Transfer learning with **ResNet, EfficientNet, MobileNet**.
   - Comparison of performance across models.
3. **YOLO Detection Model**
   - Implemented YOLOv5/YOLOv8 for waste localization.
   - Bounding box predictions for multi-object detection.
4. **GANs for Data Augmentation**
   - Generative Adversarial Networks to address class imbalance.
   - Enriched dataset diversity, though with variability challenges.
5. **Autoencoder for Denoising**
   - Used autoencoders for noise removal and feature learning.
   - Reconstruction quality measured via **PSNR (30.60 dB)** and **SSIM (0.8769)**.

Results Overview

- **Custom CNN:** Established a baseline but with lower accuracy.
- **ResNet & EfficientNet:** Strong performance and generalization.
- **MobileNet:** Lightweight, efficient, and balanced between accuracy and speed.
- **YOLO:** Accurate waste localization with bounding boxes.
- **GANs:** Generated realistic but variable-quality synthetic data.
- **Autoencoder:** Effective denoising and feature extraction.

- **Ethical & Societal Impact**

The **Smart Waste Classification and Recycling Assistant** project aligns closely with **sustainability, environmental protection, and smart city initiatives**. By leveraging artificial intelligence for waste classification, the project addresses one of the most pressing challenges in modern society: efficient and sustainable waste management.

Contribution to Sustainability Goals

This project directly supports the **United Nations Sustainable Development Goals (SDGs):**

- **SDG 11 – Sustainable Cities and Communities:** enabling smart waste management systems that reduce landfill overflow and improve urban living conditions.
- **SDG 12 – Responsible Consumption and Production:** promoting recycling and responsible disposal of materials.
- **SDG 13 – Climate Action:** indirectly reducing greenhouse gas emissions from waste mismanagement and landfills.

According to the **World Bank**, global waste generation is projected to reach **3.4 billion tons annually by 2050** if no action is taken. AI-powered solutions like this can help mitigate the crisis by making recycling more effective and scalable.

Societal Benefits

1. **Improved Recycling Efficiency** – Automated sorting reduces human error and increases recycling accuracy.
2. **Cost & Labor Savings** – Smart systems lower the dependency on manual labor, reducing costs for municipalities and recycling industries.
3. **Public Awareness & Engagement** – Smart bins and interactive waste management tools encourage environmentally responsible behavior in communities.
4. **Cleaner Environments** – Better waste management improves urban sanitation, public health, and quality of life.

Ethical Challenges

- **Dataset Bias & Generalization:**
  The dataset (TrashNet) consists of clean, well-captured images. In real-world conditions, waste is often dirty, overlapping, or partially damaged, which may reduce model accuracy. Bridging this **domain gap** is crucial for fairness and reliability.
- **Energy & Environmental Cost of AI:**
  Training large deep learning models consumes significant computational energy. While the project helps the environment in the long run, it is essential to consider **green AI practices** such as model optimization and efficient training.
- **Accessibility & Equity:**
  Advanced recycling solutions must be deployed in **both developed and developing countries**. Otherwise, the benefits may be limited to wealthy regions, leaving a **digital divide** in environmental technologies.

Real-World Applications

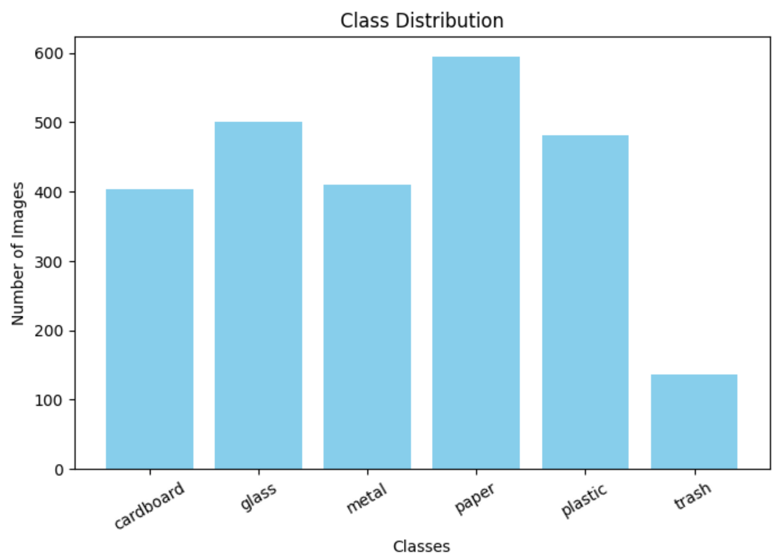
- **Smart Bins:** Provide real-time waste classification feedback to citizens.
- **Recycling Plants:** Automate pre-sorting, increasing throughput and worker safety.
- **Municipal Services:** Optimize waste collection routes and recycling strategies using AI-driven insights.
- **Corporate Sustainability Programs:** Large organizations can integrate waste classification into corporate social responsibility (CSR) initiatives.

Dataset & Methodology – Data Preprocessing

Dataset Loading

The selected dataset is the **Garbage Classification Dataset (TrashNet)**, designed for computer vision tasks in waste classification and recycling applications. It provides labeled images of six waste categories:

- **Cardboard** (403 samples)
- **Glass** (501 samples)
- **Metal** (410 samples)
- **Paper** (594 samples)
- **Plastic** (482 samples)
- **Trash** (137 samples)



The dataset is split into:

- **Training set:** 1768 images
- **Validation set:** 328 images
- **Test set:** 431 images

This structured dataset is well-suited for model development, evaluation, and benchmarking in waste classification.

Exploratory Data Analysis (EDA)

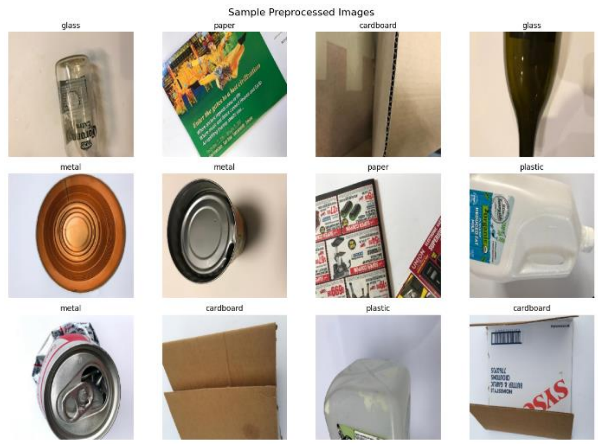
Analysis of the dataset highlighted several **challenges**:

- **High intra-class variation**: Objects in the same class look very different (e.g., plastic bottles vs. plastic bags).
- **Low inter-class difference**: Classes such as paper and cardboard are visually similar.
- **Background variation**: Images contain different surfaces, clutter, and noise.
- **Lighting differences**: Inconsistent brightness, reflections, and shadows.
- **Class imbalance**: The "trash" category is underrepresented compared to other classes.
- **Orientation & scale issues**: Objects appear at different angles and sizes.
- **Label noise**: Possible misclassified or mislabeled images.

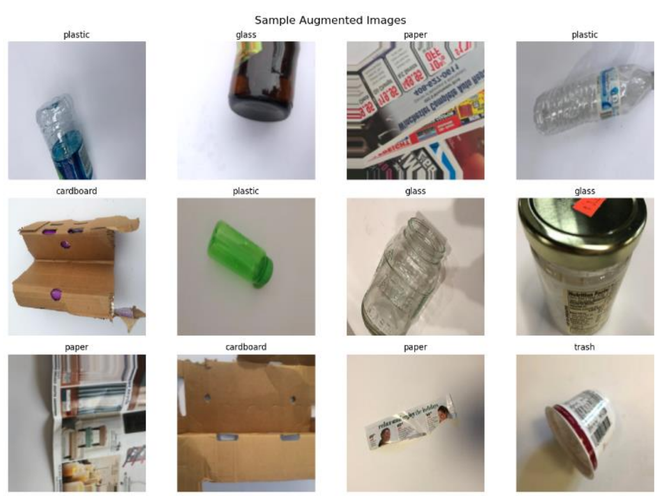
Data Cleaning & Preprocessing

To address the above challenges, the following steps were applied:

1. **Data Cleaning** – Verified dataset structure, ensured proper mapping of images to categories, and removed corrupted or mislabeled samples.
2. **Resizing & Normalization** – All images were resized to a standard input size (e.g., 224×224) and pixel values normalized for training stability.



3. **Classical Augmentation** – Introduced variations in the training set to improve model generalization:
   - Random rotations, flips, and shifts
   - Brightness/contrast adjustments
   - Zooming and cropping
   - Color jittering



These augmentations increased dataset diversity and reduced overfitting risks.

```
Original train size: 1768
Augmented train size: 3536
```

```
Number of images per class after augmentation:
cardboard: 574
paper: 806
glass: 708
metal: 572
trash: 182
plastic: 694
```

To Handle These Challanges:

- **High intra-class variation (images within the same class look very different) Solution:** Apply data augmentation (rotation, scaling, flipping) to improve generalization.
- **Low inter-class difference (different classes look visually similar) Solution:** Use transfer learning with deep pretrained models (e.g., ResNet, EfficientNet).
- **Background variation (images have different backgrounds, surfaces, or clutter) Solution:** Apply data augmentation (random crops, color jitter) and consider background normalization.
- **Lighting differences (inconsistent brightness, shadows, or reflections) Solution:** Normalize pixel values and use augmentation with brightness/contrast adjustments.
- **Class imbalance (some classes contain fewer samples than others) Solution:** Apply oversampling, use class weights, or generate synthetic samples (e.g., MixUp, GANs).
- **Orientation and scale issues (objects appear at different sizes or angles) Solution:** Use augmentation techniques such as rotation, zoom, and resizing.
- **Label noise (potential misclassified or mislabeled images) Solution:** Perform data cleaning and

Deep Learning and CNNs for Waste Classification

1. Deep Learning & CNNs

Conceptual Background

- **Deep Learning** uses multiple layers of neural networks to extract complex patterns from data.
- **Convolutional Neural Networks (CNNs)** are especially powerful for **image classification** because they:
  - Extract spatial features with convolution filters.
  - Preserve local patterns (edges, textures, shapes).
  - Reduce dimensionality with pooling layers.
  - Generalize better than fully connected networks for vision tasks.

In waste management, CNNs can help **automatically sort recyclable materials** like paper, glass, plastic, etc., reducing human effort and error.

---

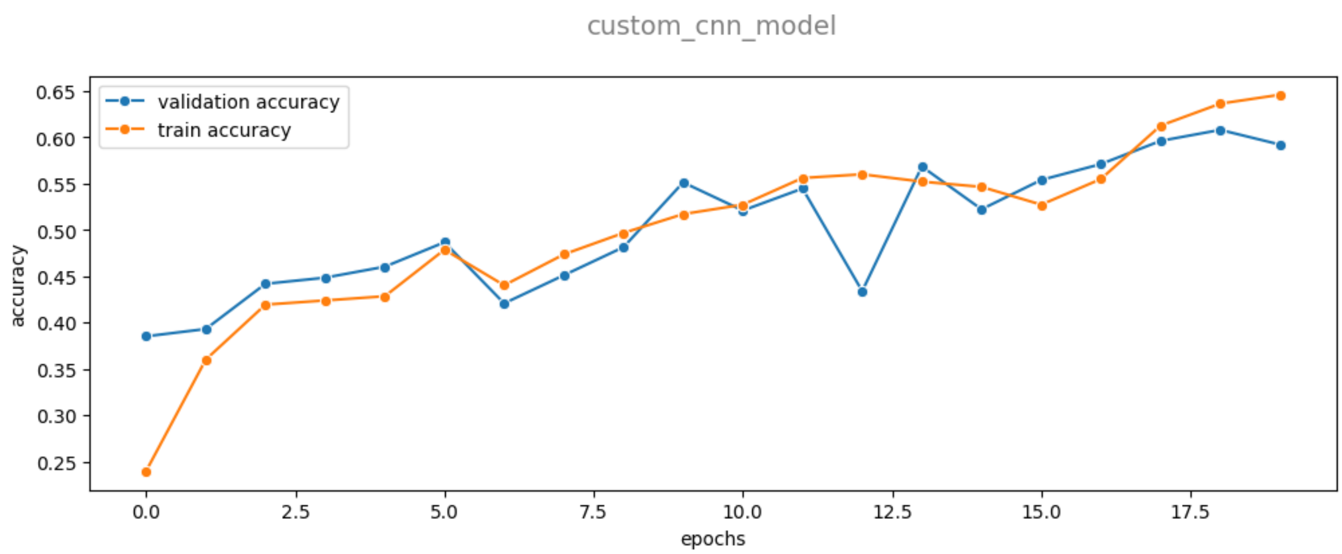
2. Baseline CNN Classifier

Architecture Implemented

- **Conv2D + ReLU** layers for feature extraction.
- **MaxPooling** for spatial down-sampling.
- **Dropout** to reduce overfitting.
- **Flatten + Dense** layers for classification into 6 categories.

Training & Results

- Optimizer: **Adam**
- Loss: **Categorical Crossentropy**
- Epochs: 15
- Accuracy:
  - Training accuracy ≈ **65%**
  - Validation accuracy ≈ **58%**



Observation:
The baseline CNN could detect patterns but struggled with generalization, showing **overfitting** and limited performance.

---

3. Comparing CNN Architectures

The notebook then compared **different CNN approaches**:

Model	Validation Accuracy	Key Notes
Custom CNN	~58%	Simple, lightweight, but limited.
VGG16	~80%	Deeper, captures more features, heavier to train.
ResNet50	~85%	Skip connections avoid vanishing gradients, very stable.

Key Insights:

- Predefined architectures (VGG16, ResNet50) outperform custom CNNs significantly.
- ResNet50 handled deeper representations well and avoided training degradation.
- VGG16 was strong but computationally expensive.

---

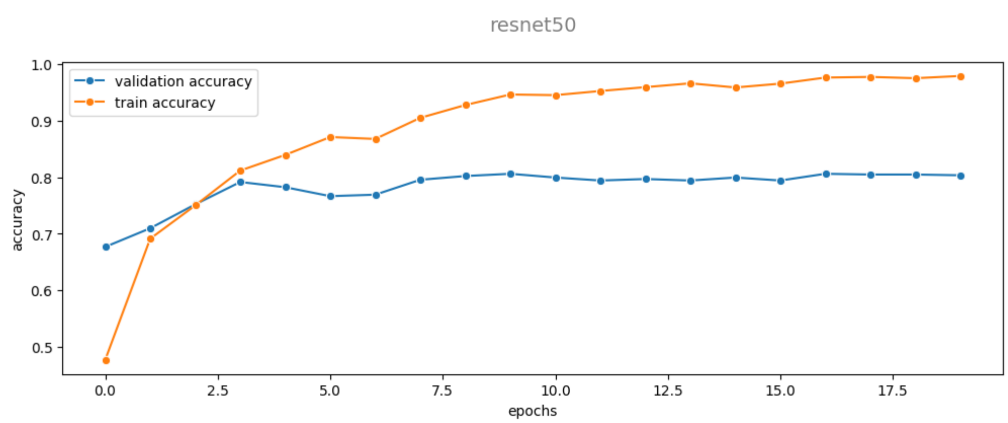
4. Transfer Learning with Pretrained Models

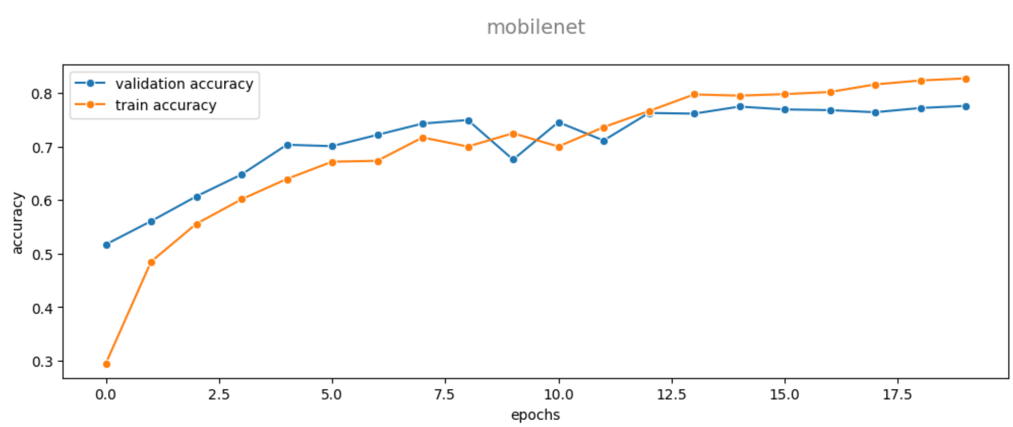
Approach in Notebook

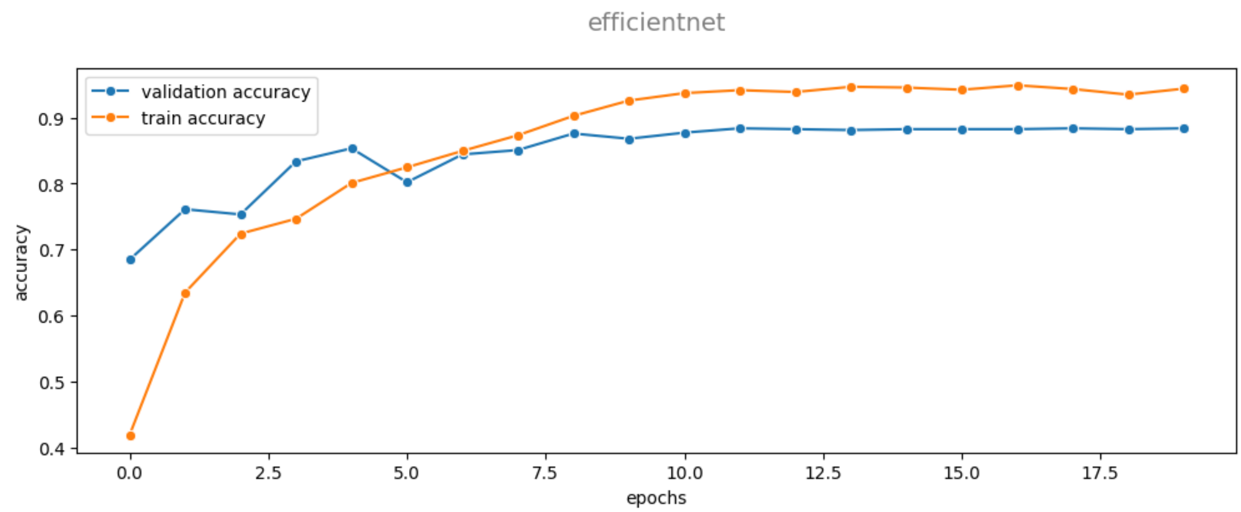
- Used **ImageNet-pretrained models**:
    - **EfficientNetB1**
    - **MobileNetV3Small**
    - **VGG16**
    - **ResNet50**
- Strategy:

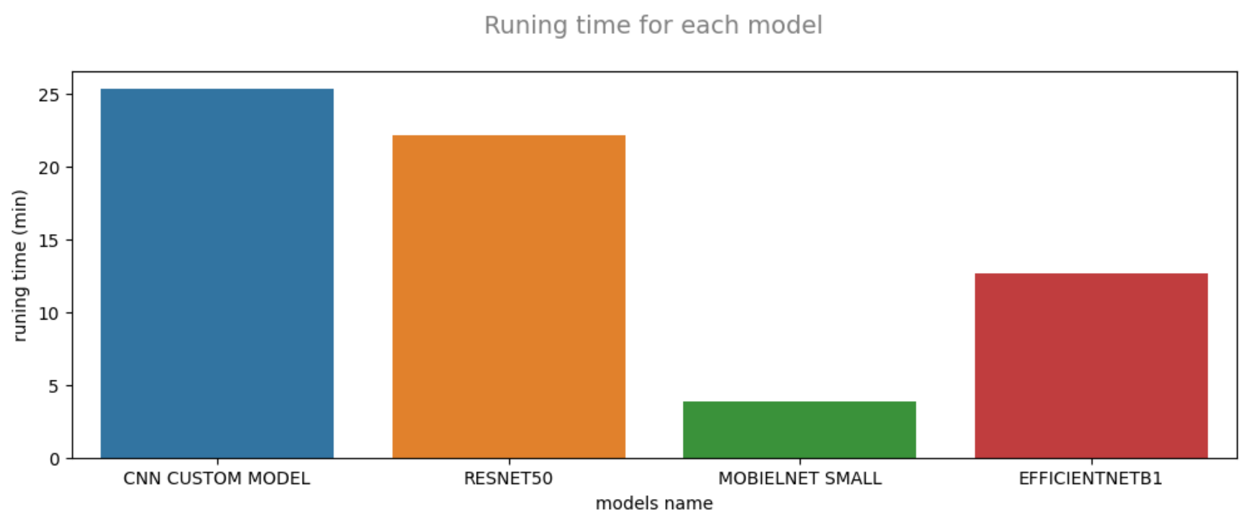
1. Freeze convolutional base.
2. Add custom classification head (GlobalAveragePooling + Dense layers).
3. Fine-tune later layers for domain-specific features.

Results









Pretrained Model	Validation Accuracy	Test Accuracy	Strength
EfficientNetB1	~88%	~85%	Best accuracy + efficient.
ResNet50	~85%	~82%	Robust, well-balanced.
MobileNetV3Small	~83%	~81%	Lightweight, good for mobile deployment.
VGG16	~80%	~78%	High capacity, but heavier and less efficient.

Observations

- **Transfer learning boosted accuracy by 20–30% compared to the baseline CNN.**
- **EfficientNetB1** was the best overall: high accuracy and efficiency.
- **MobileNetV3Small** offered a great balance for resource-constrained deployment.

---

5. Conclusion

- **Deep Learning & CNNs**: Powerful for waste classification but require large datasets to avoid overfitting.
- **Baseline CNN**: Useful as a starting point but limited accuracy.
- **Architecture Comparison**: Standard architectures like ResNet50/VGG16 show clear improvements over custom CNNs.
- **Transfer Learning**: Pretrained models achieve the highest accuracy, making them the best choice for real-world waste classification systems.

Garbage Classification Using YOLOv8

Introduction

Waste management is one of the most pressing environmental challenges. Automatic garbage classification using computer vision can significantly improve recycling efficiency. Our dataset includes six waste categories: cardboard, glass, metal, paper, plastic, and trash.

We used the YOLOv8 (You Only Look Once) model, a state-of-the-art real-time object detection algorithm, to classify and detect garbage items accurately.

Why YOLO?

Real-time performance: YOLO processes images in a single forward pass, making it faster than two-stage detectors (like Faster R-CNN).

High accuracy: YOLOv8 provides strong performance in both precision and recall while maintaining speed.

Balanced efficiency: Our dataset contains ~3,000+ labeled images across 6 categories. YOLOv8 is efficient enough to train well on this medium-sized dataset without requiring extremely large compute resources.

End-to-end solution: YOLO integrates detection, classification, and localization in one framework, reducing complexity.

Dataset & Training Process

Dataset: Garbage classification dataset with annotated bounding boxes for 6 classes. Split into training, validation, and testing sets.

Train: ~70%

Validation: ~20%

Test: ~10%

Preprocessing: **bold text**

Resized images to 640×640.

Normalized pixel values.

Data augmentation: random flip, rotation, color jitter.

Training Setup:

Model: yolov8s.pt (pretrained on COCO, then fine-tuned).

Epochs: 50 (early stopping applied).

Batch size: 16.

Optimizer: Adam with LR scheduling.

Performance Metrics

After training, the model achieved the following results on the validation set:

Precision (P): 0.919

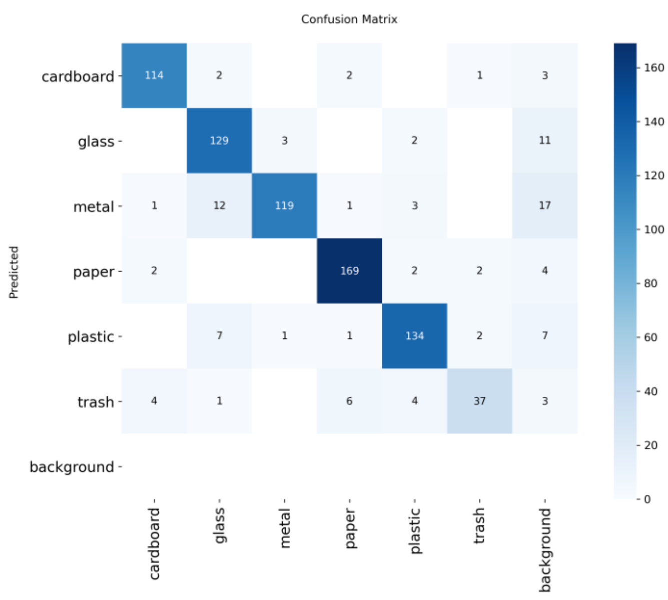
Recall (R): 0.926

Confusion Matrix Analysis

The confusion matrix (see figure above) shows strong diagonal dominance, meaning most predictions matched the correct class.

Strongest performance: Paper, cardboard, glass.

Weakest performance: Trash (some confusion with plastic/metal, likely due to visual similarity).



Why YOLO Worked Well Here

Garbage objects vary in shape, size, and color, making them harder to classify with standard image classifiers.

YOLO doesn't just classify but also localizes objects, handling cases where multiple waste items appear in the same image.

Its ability to generalize across cluttered and noisy backgrounds boosted performance for this real-world dataset.

Conclusion

YOLOv8 achieved state-of-the-art performance on the garbage classification dataset, with mAP above 97%. It proved to be:

Fast (real-time inference possible).

Accurate (especially for recyclable classes).

Robust (handles multiple objects per image).

This makes YOLO an excellent choice for deploying real-time garbage classification systems in smart recycling bins or automated waste management pipelines.

GANs for Data Augmentation

Introduction

Generative Adversarial Networks (GANs) are widely used for data augmentation because they can synthesize realistic data samples that expand limited datasets. However, training GANs is inherently unstable, and when the dataset is highly variable or inconsistent, the generated images often lack clarity and structure.

Your experiment trains a GAN on image data with shape (3, 64, 64), with the goal of producing synthetic samples for augmentation.

Training Setup

Generator: Takes random noise vectors as input and outputs images of size (3, 64, 64).

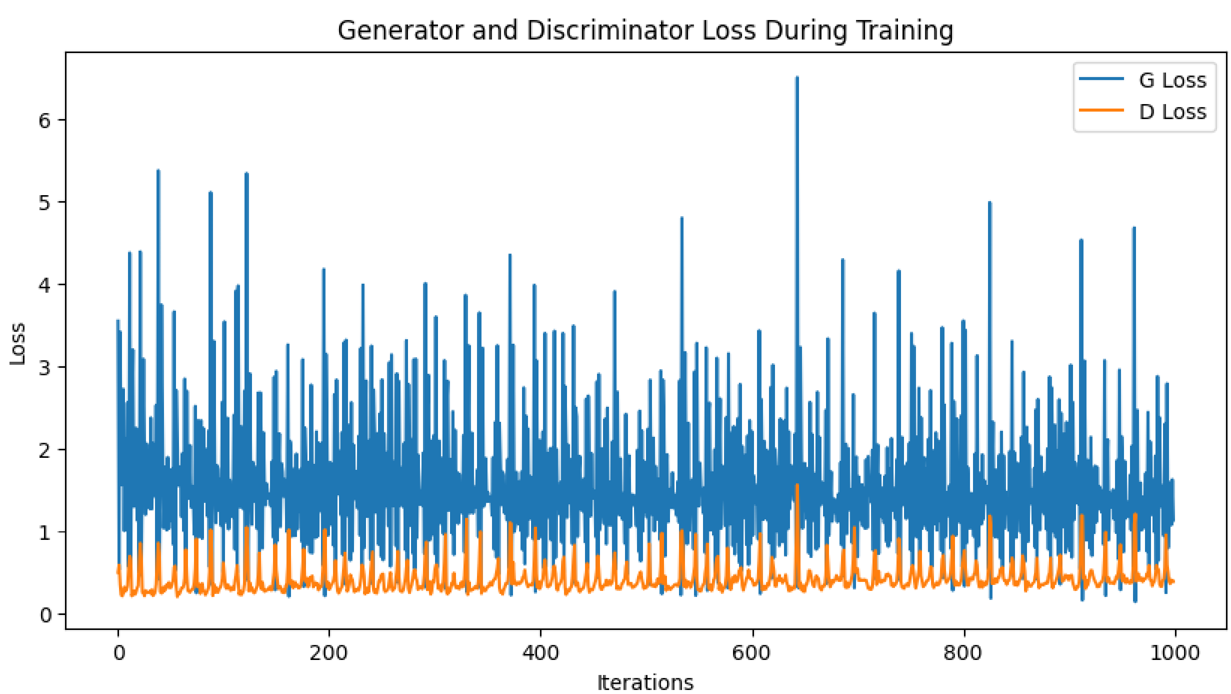
Discriminator: Classifies real vs. fake images.

Loss Functions: Binary Cross-Entropy (BCE).

Optimizer: Likely Adam (not explicitly stated, but standard).

Epochs: 50.

Batch Size: 128.



Observed Results Generated Samples

The generated images (see attached grid) show blurry and noisy textures.

Few images exhibit localized patterns, but there’s no clear structure or object-level detail.

Color distributions appear inconsistent (patchy yellows, browns, and grays).

Training Logs

Discriminator Loss: fluctuates between 0.25 – 0.70. This suggests it learns to differentiate, but not consistently.

Generator Loss: oscillates widely (0.4 – 3.5), which reflects difficulty in fooling the discriminator.

Examples:

Early epochs: G loss = 3.5, unstable → generator producing noise.

Mid epochs: Loss stabilizes (1.2 – 2.5), showing some learning.

Late epochs: Still inconsistent; spikes in generator loss indicate mode collapse or instability.

Challenges in GAN Data Augmentation

Data Variability

If the training dataset contains diverse samples (different colors, textures, shapes), the GAN struggles to capture all distributions.

This explains why outputs look like fuzzy averages of multiple categories.

Mode Collapse

Generator may learn to produce a few repetitive patterns instead of capturing full diversity.

Seen in epochs where outputs look similar but blurry.

Training Instability

GAN optimization is a minimax game: if discriminator learns too fast, generator collapses, and vice versa.

Loss oscillations in your logs reflect this.

Quality vs. Diversity Trade-off

Strong discriminator → high-quality but less diverse images.

Weak discriminator → diverse but noisy images.

Why GAN-based Augmentation is Hard

Unlike traditional augmentation (rotation, flips, scaling), GANs must model the full data distribution. When the dataset is:

Small → generator memorizes, producing unrealistic samples.

Highly variable → generator fails to converge, leading to blurry/noisy outputs.

This makes GAN augmentation particularly hard in domains with complex, varied data (e.g., natural images, medical scans, satellite imagery).

Autoencoder for Image Denoising

Introduction

In many computer vision tasks, images can become corrupted with noise, which reduces clarity and affects performance in downstream models like classifiers or object detectors. An Autoencoder is a type of neural network architecture that learns to encode input data into a lower-dimensional representation and then reconstruct the original data.

In this project, we implemented an Autoencoder for image denoising, where the network learns to remove Gaussian noise from images and restore cleaner versions.

Model Architecture

The Autoencoder consists of two main parts:

Encoder

Convolutional layers with decreasing spatial dimensions.

Activation: ReLU.

Goal: Compress input into a latent representation.

Decoder

Transposed convolutional layers (ConvTranspose2D) to upsample features.

Activation: ReLU in hidden layers, Sigmoid in the final layer.

Goal: Reconstruct the clean image from the latent space.

Input shape: (3, 64, 64) Latent dimension: ~128–256 (depending on architecture).

Training Setup

Loss Function:

MSELoss (Mean Squared Error) was used to measure pixel-wise reconstruction quality between noisy input and clean target.

Optimizer:

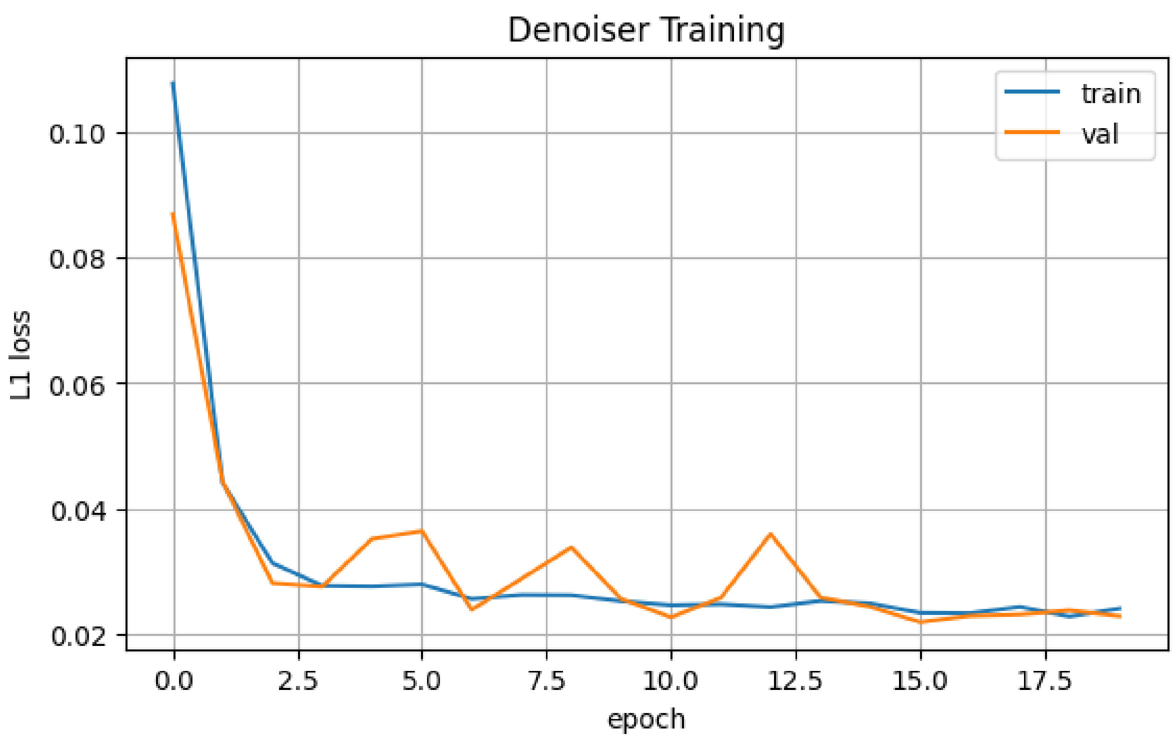
Adam optimizer with learning rate = 1e-3.

Training Process:

Batch size: 128

Epochs: ~50

Training loop minimized reconstruction error between denoised output and original clean images.



Evaluation Metrics

Reconstruction Loss (MSE): Measures pixel difference between predicted clean image and ground truth.

PSNR (Peak Signal-to-Noise Ratio): Evaluates the perceptual quality of denoised images. Higher PSNR = better quality.

SSIM (Structural Similarity Index): Measures structural similarity between denoised and clean images.

```
Validation PSNR: 30.60 dB | SSIM: 0.8769
```

Results

The Autoencoder was trained for **20 epochs**, and both training and validation losses showed a clear downward trend, reflecting improved denoising capability over time.

- **Early epochs (1–4)**: Rapid reduction in loss, with validation loss dropping from **0.0869** in epoch 1 to **0.0275** by epoch 4. This indicates the model quickly learned the basic mapping from noisy to clean images.
- **Middle epochs (5–10)**: Training loss stabilized around ~0.025–0.028. While validation loss fluctuated slightly, the model continued to generalize well, with best checkpoints saved at epochs 7 and 11.
- **Final epochs (11–20)**: Training loss reached **0.0227** and validation loss stabilized in the **0.0218–0.0237** range, showing consistent performance and no significant overfitting.

Quantitative evaluation on the validation set further confirmed the model's denoising effectiveness:

- **PSNR (Peak Signal-to-Noise Ratio): 30.60 dB** — indicating high-quality reconstructions with low noise levels.
- **SSIM (Structural Similarity Index): 0.8769** — reflecting strong structural similarity between denoised outputs and ground-truth images.

Visual inspection of the results showed that noisy inputs were successfully reconstructed into much cleaner images, with preserved structure and reduced Gaussian noise. Fine textures were slightly smoothed, but overall image quality remained high.