Material Stream Identification System (MSI)

1. Introduction

The rapid increase in post-consumer waste has made efficient recycling a global challenge. Manual waste sorting is slow, error-prone, and expensive. Automated Material Stream Identification (MSI) systems aim to solve this problem by using computer vision and machine learning to recognize different waste materials from images.

In this project, we designed and implemented an end-to-end MSI system that classifies waste images into material categories such as **glass, paper, cardboard, plastic, metal, trash**, and an additional **unknown** class. The system follows the complete machine learning pipeline: data preprocessing, data augmentation, feature extraction, classifier training, evaluation, and real-time deployment.

---

2. Project Objectives

The main objectives of this project are: - Convert raw images into meaningful numerical feature vectors. - Train and compare two classical machine learning classifiers: - Support Vector Machine (SVM) - k-Nearest Neighbors (k-NN) - Achieve high classification accuracy ($\geq$ 0.85) on validation data. - Handle uncertain or out-of-distribution inputs using an **Unknown** class. - Deploy the best-performing model in a real-time camera-based application.

---

3. Dataset Description

3.1 Material Classes

The dataset contains images belonging to the following classes:

ID	Class Name	Description
0	Glass	Bottles and jars made of glass
1	Paper	Newspapers, office paper
2	Cardboard	Thick layered paper materials
3	Plastic	Bottles, packaging, films
4	Metal	Aluminum cans, metal objects
5	Trash	Non-recyclable waste
6	Unknown	Blurred or unseen objects

3.2 Dataset Structure

- Images are organized into folders, one folder per class.
- Each image has a one-to-one mapping with its class label.
- The dataset is split into **80% training** and **20% testing** using stratified sampling to preserve class balance.

---

4. Data Augmentation

4.1 Why Data Augmentation?

The original dataset is relatively small and imbalanced across classes. Training directly on it may lead to **overfitting**. Data augmentation artificially increases dataset size and diversity, helping the model generalize better to new images.

4.2 Applied Augmentation Techniques

We applied extensive image augmentation using OpenCV, increasing the training data by **28×**, far exceeding the required 30% increase.

The following techniques were used: - Horizontal and vertical flipping - Image rotation (±15°, ±30°, 45°) - Brightness and contrast adjustment - HSV color modification (saturation and value) - Random cropping and resizing - Gaussian blur and sharpening - Gaussian noise addition - Perspective transformation - CLAHE (adaptive histogram equalization)

Justification: These augmentations simulate real-world conditions such as different lighting, camera angles, motion blur, and object positions.

---

5. Feature Extraction Using CNN

5.1 Why Use CNN Features?

Traditional features (e.g., color histograms, edges) are limited in capturing complex patterns. Convolutional Neural Networks (CNNs) automatically learn **high-level visual features** such as textures, shapes, and object structures.

Instead of training a CNN from scratch, we use **transfer learning**, which leverages a pretrained model trained on ImageNet.

5.2 Why EfficientNetB3?

We selected **EfficientNetB3** for feature extraction because: - It provides an excellent balance between accuracy and computational cost. - It uses compound scaling (depth, width, resolution) efficiently. - It outperforms older models like MobileNetV2 in feature quality.

5.3 Feature Vector Generation

- Input images are resized to **224×224**.
- Images are passed through EfficientNetB3 without the classification head.
- A **Global Average Pooling** layer converts feature maps into a **1D feature vector (1536 dimensions)**.
- Dropout (0.3) is applied to improve robustness.

Each image is converted into a fixed-length numerical vector suitable for classical ML classifiers.

---

6. Classifier Models

6.1 Support Vector Machine (SVM)

- Kernel: Radial Basis Function (RBF)
- Parameters: C = 10, gamma = 'scale'

Why SVM? - Effective in high-dimensional feature spaces. - Robust to overfitting when properly regularized. - Works well with CNN-extracted features.

6.2 k-Nearest Neighbors (k-NN)

- Number of neighbors: k = 6
- Weighting: Distance-based

Why k-NN? - Simple and intuitive classifier. - No training phase (instance-based learning). - Useful as a baseline for comparison.

6.3 Feature Scaling

All feature vectors are standardized using **StandardScaler**, which is essential for distance-based classifiers like SVM and k-NN.

---

7. Handling the Unknown Class

To handle uncertain predictions: - Confidence thresholds are applied to classifier outputs. - If the prediction confidence is below a predefined threshold, the object is labeled as **Unknown**.

This mechanism prevents forced misclassification of unseen or low-quality inputs.

---

8. Experimental Results

8.1 Dataset Statistics

- Training images: 1492 (before augmentation)

- Augmented training samples: 41,776
- Test images: 373
- Feature dimension: 1536

8.2 Classification Accuracy

Classifier	Accuracy
SVM (RBF)	**91.69%**
k-NN	88.20%

Observation: - SVM achieved the highest accuracy and better generalization. - k-NN performed well but was slower and more sensitive to noise.

---

9. Single Image Testing and Output

The trained system supports single-image prediction: - Input image is resized and preprocessed. - CNN features are extracted. - Features are scaled and passed to SVM and k-NN classifiers.

Example Output:

- **SVM Prediction:** Plastic (Confidence: 94%)
- **k-NN Prediction:** Plastic (Confidence: 91%)

If confidence < threshold → Output: **Unknown Object**

---

10. Real-Time System Deployment

- Live camera frames are captured using OpenCV.
- Each frame is processed in real-time.
- The predicted material label is displayed on the screen.

Performance: - Stable real-time inference - Fast feature extraction using GPU acceleration

---

11. Conclusion

In this project, we successfully built a complete Material Stream Identification system. By combining CNN-based feature extraction with classical machine learning classifiers, we achieved high accuracy and robust performance.

Key achievements: - Effective data augmentation strategy - High-quality CNN feature extraction using EfficientNetB3 - SVM classifier achieving over 91% accuracy - Real-time deployment with Unknown class handling

This system demonstrates a practical and scalable approach for automated waste sorting and recycling applications.
