# Dharmsinh Desai University

**Academic Year: 2023-24**

**Department: Institute of Management &**

**Information Science**

**Subject: Python Programming**

# Topic: Waste Identication

**Name:**

|  |  |
|---|---|
| **MA057 – Jaymin Valaki D.** | **Submitted to Dr. Narayan Joshi** |
| **ID No: 22MAPOG067** | **MCA Department** |

**Professor Sign:** **Student Sign:**

## Contents

## 1. Introduction

The Waste Identification project is an innovative solution that uses machine learning to classify different types of waste. The project aims to contribute to environmental sustainability by automating the process of waste sorting and management. This automation not only increases efficiency but also reduces human error in waste classification. By leveraging the power of machine learning, the project provides a scalable solution that can handle a large volume of waste items.

The system is designed to be adaptable, capable of learning from new data and improving over time. This adaptability makes it a robust solution for waste management, capable of keeping up with the ever-changing landscape of waste types and disposal methods.

## 2. Problem Statement

The problem of waste management is a significant global issue. Incorrect disposal and poor segregation of waste contribute to environmental pollution. This project addresses this problem by developing a system that can accurately identify and classify different types of waste, such as plastic, e-waste, bio waste, and paper. The goal is to improve waste management processes and promote recycling and proper disposal practices, thereby reducing the environmental impact of waste.

The challenge lies in the diversity and complexity of waste materials, which require sophisticated identification methods to ensure accurate classification. By using machine learning, this project aims to overcome these challenges and provide a reliable tool for efficient and effective waste management.

# 3. Project Overview

The Waste Identification project is a comprehensive system that leverages the power of machine learning to classify waste items into one of four categories: plastic, e-waste, bio waste, or paper. The system is designed to handle unlabelled data, making it versatile and adaptable for real-world applications.

## Step 1: Data Preprocessing

- The first step in the project is data preprocessing. This involves loading images of waste items from a specified directory. These images are then pre-processed for feature extraction. The preprocessing steps include resizing the images to a uniform size and normalizing the pixel values.

## Step 2: Feature Extraction

- The next step is featuring extraction. The project uses a pre-trained VGG16 model for this purpose. VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition". The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. In our case, the model is used as a feature extractor where the last fully connected layer is removed and the output from the convolutional layers is flattened into a single vector, which serves as input to the clustering algorithm.

## Step 3: Clustering

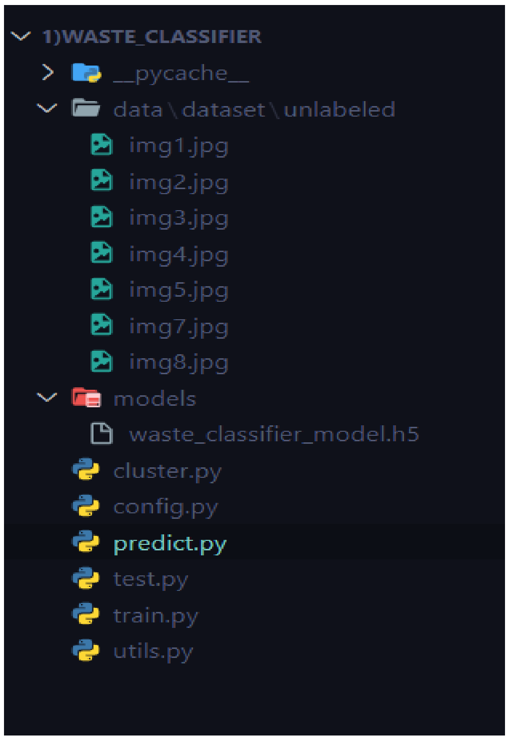
- After feature extraction, the features are clustered using the KMeans algorithm from sklearn. KMeans is an unsupervised learning algorithm that groups data based on their similarity. It works by initializing "k" centroids randomly, assigning each data point to the nearest centroid, and re-computing the centroid as the mean of all data points assigned to it. The process is repeated until the centroids no longer change significantly.

## Step 4: Prediction

- Finally, based on these clusters, predictions are made for each image. These predictions are then compared with expected results to evaluate the performance of the model.
- The prediction phase is a critical part of the project. After the features have been extracted and clustered, the model uses these clusters to make predictions for each image. The model assigns each image to the cluster whose centroid is nearest to the image's feature vector. The index of this cluster is then used as the predicted class for the image.
- To evaluate the performance of the model, these predictions are compared with the expected results. This comparison allows us to calculate the accuracy of the model, which is defined as the percentage of images that were correctly classified.

The project demonstrates how machine learning can be applied to solve real-world problems and highlights the potential of AI in contributing to environmental sustainability. By automating waste classification, it not only makes waste management more efficient but also contributes to reducing environmental pollution.

## Folder Structure:



```
1)WASTE_CLASSIFIER
  __pycache__
  data\dataset\unlabeled
    img1.jpg
    img2.jpg
    img3.jpg
    img4.jpg
    img5.jpg
    img7.jpg
    img8.jpg
  models
    waste_classifier_model.h5
  cluster.py
  config.py
  predict.py
  test.py
  train.py
  utils.py
```

# 4. Library

The project leverages several powerful Python libraries, each contributing unique capabilities that make the development of this complex AI system possible:

- **TensorFlow**: TensorFlow is an open-source machine learning library developed by Google. It provides a comprehensive ecosystem of tools, libraries, and community resources that allows researchers to push the state-of-the-art in ML, and developers to easily build and deploy ML-powered applications. In this project, TensorFlow is used for building and training the machine learning model. It provides high-level APIs that greatly simplify the process of creating neural networks. It also supports a wide range of neural network architectures and provides tools for visualizing and debugging models.

- **Keras**: Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research. In this project, Keras is used for leveraging the pre-trained VGG16 model and for image preprocessing. It provides simple methods for loading and preprocessing image data.

- **Scikit-learn (sklearn):** Scikit-learn is a free software machine learning library for Python. It features various classification, regression, and clustering algorithms, including support vector machines, random forests, gradient boosting, k-means, etc., and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. In this project, sklearn’s KMeans clustering algorithm is used to group similar features together.

- **NumPy**: NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. In this project, NumPy is used for numerical computations and handling arrays of features extracted from images.

These libraries collectively provide a robust framework for implementing machine learning models in Python. They abstract away many of the complex details of machine learning algorithms and allow developers to focus on the high-level design of their models.

## 5. System Flow

The system architecture consists of the following components:

- **Data Preprocessing**: This involves loading and preprocessing images from a directory using the Python Imaging Library (PIL). The images are resized and normalized to prepare them for feature extraction.

- **Feature Extraction**: The VGG16 model, pre-trained on ImageNet, is used to extract features from the pre-processed images. This deep learning model has been proven effective in image classification tasks.

- **Clustering:** Once features are extracted, they are clustered using the KMeans algorithm from sklearn. KMeans is an unsupervised learning method that groups similar features together, aiding in the classification process. It works by initializing “k” centroids randomly, assigning each data point to the nearest centroid, and re-computing the centroid as the mean of all data points assigned to it. The process is repeated until the centroids no longer change significantly.

- **Prediction:** The final step in the system flow is prediction. Here, the model predicts the class of each image based on the extracted features and clusters. These predictions are then used for further analysis or action, such as sorting waste items into their respective categories or providing feedback on waste disposal practices. This step brings together all previous steps and produces actionable outputs.

# 6. Source Code

- **Cluster.py**

```
from sklearn.cluster import KMeans

from keras.applications.vgg16 import VGG16

from keras.preprocessing import image

from keras.applications.vgg16 import preprocess_input

import numpy as np

import os

from config import UNLABELED_DATA_DIR

# This function loads and preprocesses images from a directory.

def load_and_preprocess_images(data_dir, img_size=(224, 224)):

    image_list = os.listdir(data_dir)

    images = []

    for img_name in image_list:

        # Skip non-image files.

        if not img_name.lower().endswith(('.png', '.jpg', '.jpeg')):

            continue

        img_path = os.path.join(data_dir, img_name)


        try:

            # Load the image and preprocess it.

            img = image.load_img(img_path, target_size=img_size)

            x = image.img_to_array(img)

            x = np.expand_dims(x, axis=0)

            x = preprocess_input(x)

            images.append(x)
        except Exception as e:
            print(f"Can't open {img_name}. Error: {e}")

    # If no valid images are found, return None.
    if images:
        return np.vstack(images)
    else:
        return None

# Load the VGG16 model with pre-trained ImageNet weights.
model = VGG16(weights='imagenet', include_top=False)

# Specify the directory containing your unlabeled images.
images = load_and_preprocess_images(UNLABELED_DATA_DIR)

if images is None:
    print("No valid images found.")
else:
    # Use the VGG16 model to extract features from the images.
    features = model.predict(images)
    features = features.reshape(features.shape[0], -1)

    # Cluster the features using KMeans.
    kmeans = KMeans(n_clusters=4, random_state=0).fit(features)

    # Print the cluster labels for each image.
    print(kmeans.labels_)
```

- **Prediction.py**

```
import tensorflow as tf

import numpy as np

from collections import Counter

from utils import load_and_preprocess_data

from test import expected_result

from config import UNLABELED_DATA_DIR, MODEL_PATH


def calculate_total_percentage(counts, total):

    percentages = {}

    for class_index, count in counts.items():

        percentages[class_index] = (count / total) * 100

    return sum(percentages.values())


model = tf.keras.models.load_model('models/waste_classifier_model.h5')

class_labels = ['plastic', 'e-waste', 'bio waste', 'paper']

images, _ = load_and_preprocess_data(UNLABELED_DATA_DIR)

predictions = model.predict(images)

predicted_classes = np.argmax(predictions, axis=1)

counts = Counter(predicted_classes)


for class_index, count in counts.items():

    print(f"{class_labels[class_index]}: {count} images")


correct_predictions = sum (p == e for p, e in zip (predicted_classes,
expected_result))

total_predictions = len(predicted_classes)

accuracy = (correct_predictions / total_predictions) * 100

print (f"Accuracy: {accuracy}%")
```

# 7. Conclusion

The Waste Identification project demonstrates the potential of machine learning in addressing real-world problems, specifically in the area of waste management. By automating the process of waste classification, the system not only increases efficiency but also contributes to environmental sustainability.

The project successfully uses a pre-trained VGG16 model to extract features from images of waste items and classifies them into one of four categories: plastic, e-waste, bio waste, or paper. The system’s ability to handle unlabelled data makes it versatile and adaptable for real-world applications.

However, like any machine learning model, its performance is dependent on the quality and diversity of the training data. Therefore, continuous improvement and updating of the model with new data are crucial for maintaining its accuracy.

In terms of future enhancements, integrating this system into a mobile application would make it more accessible and user-friendly. Users could classify waste in real-time using their mobile devices, contributing to better waste management practices.

Overall, this project is a step forward in leveraging technology to create a more sustainable environment. It highlights how AI can be used as a tool to drive positive change and address global issues.

# 8. Future Enhancement

The current system can be further enhanced by integrating it into a mobile application such as **WasteX**. This would allow users to classify waste in real-time using their mobile devices, making it even more accessible and user-friendly. The application could use the device's camera to capture images of waste items, which would then be classified by the system. Additionally, incorporating real-time feedback mechanisms could help improve the accuracy of predictions over time.

Please note that this is a high-level overview of the project. For detailed implementation and code-related queries, please refer to the respective Python scripts (cluster.py, utils.py, predict.py). Each script contains comments explaining its functionality in detail.