Transforming Waste Management with Transfer Learning

Introduction

The exponential increase in waste generation worldwide has made waste management one of the most critical environmental challenges. Traditional manual waste segregation methods are labor-intensive, error-prone, and inefficient. To address this, we propose a deep learning-based smart classification system using transfer learning, capable of recognizing and categorizing waste into different types.

Objective

To develop a machine learning model using transfer learning that can classify waste images into predefined categories, thus facilitating smart bins, robotics automation, or city-level smart waste tracking systems.

Tools & Technologies

- Language: Python
- Frameworks: TensorFlow, Keras
- Libraries: OpenCV, NumPy, Matplotlib, Scikit-learn
- Model: MobileNetV2 (pretrained on ImageNet)
- Environment: Jupyter Notebook / VSCode / Google Colab

Dataset

We use a custom dataset based on categories:

```
dataset/
├── organic/
├── recyclable/
├── hazardous/
└── general/
```

Each folder should contain images labeled as per the type.

Methodology

1. Data Collection & Labeling
2. Data Preprocessing
3. Transfer Learning
4. Training & Validation
5. Evaluation

6. Deployment (Optional)

Data Preprocessing & Loading

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

IMG_SIZE = 224
BATCH_SIZE = 32
data_dir = 'dataset/'

datagen = ImageDataGenerator(rescale=1./255, validation_split=0.2,
                             rotation_range=25, horizontal_flip=True,
zoom_range=0.2)

train_gen = datagen.flow_from_directory(data_dir, target_size=(IMG_SIZE,
IMG_SIZE),
                                        batch_size=BATCH_SIZE,
class_mode='categorical', subset='training')

val_gen = datagen.flow_from_directory(data_dir, target_size=(IMG_SIZE,
IMG_SIZE),
                                      batch_size=BATCH_SIZE,
class_mode='categorical', subset='validation')
```

Load and Modify Pre-trained Model

```
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.keras.optimizers import Adam

base_model = MobileNetV2(input_shape=(IMG_SIZE, IMG_SIZE, 3),
                         include_top=False, weights='imagenet')
base_model.trainable = False

x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(128, activation='relu')(x)
predictions = Dense(train_gen.num_classes, activation='softmax')(x)

model = Model(inputs=base_model.input, outputs=predictions)
model.compile(optimizer=Adam(), loss='categorical_crossentropy',
metrics=['accuracy'])
```

Train the Model

```
from tensorflow.keras.callbacks import EarlyStopping

early_stop = EarlyStopping(monitor='val_loss', patience=3)
history = model.fit(train_gen, validation_data=val_gen, epochs=10,
callbacks=[early_stop])
```

Visualize Accuracy and Loss

```
import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Model Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

Evaluate the Model

```
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np

val_gen.reset()
preds = model.predict(val_gen)
y_pred = np.argmax(preds, axis=1)

print("Classification Report:")
print(classification_report(val_gen.classes, y_pred,
target_names=list(val_gen.class_indices.keys())))
```

Expected Output

```
Training Accuracy:
Epoch 1/10 - accuracy: 0.84 - val_accuracy: 0.80
Epoch 2/10 - accuracy: 0.91 - val_accuracy: 0.87

Classification Report:
              precision    recall  f1-score   support
    general       0.90      0.85      0.87        40
```

```
   hazardous      0.92      0.91      0.91         35
     organic      0.88      0.89      0.88         45
  recyclable      0.93      0.95      0.94         50
accuracy                             0.90        170
```

Model Saving

```
model.save('waste_classifier_model.h5')
```

Applications

- Smart Bins
- City Waste Monitoring
- Robotic Waste Segregators
- Educational Tools

Future Enhancements

- Use object detection
- Fine-tune more layers
- Create a mobile/web app
- Implement real-time video classification