Technical Documentation: Garbage Classification System

1. Introduction

This project involves the creation of a garbage classification system using Convolutional Neural Networks (CNN) to classify images of garbage into various categories. The model uses real-time webcam input for predictions and is trained on a dataset of labeled garbage images.

2. System Architecture and Design

This section describes the architecture of the system, focusing on the high-level design, including both the machine learning model and the real-time webcam interface.
Architecture Overview: The system consists of three primary components:

- Dataset: A labeled dataset containing images of various garbage categories.
- CNN Model: A deep learning model to classify images based on the dataset.
- Webcam Interface: A real-time system that captures webcam frames and makes predictions using the trained model.

Data Flow:

The images are preprocessed and passed into the CNN for training.
After training, the model is deployed for real-time classification using webcam input.
Predictions are displayed on the webcam feed in real-time.
Model Design: The CNN architecture consists of several convolutional layers followed by max-pooling layers and fully connected layers. This allows the model to learn hierarchical features from the image data.

CNN Layers:
- Conv2D (32 filters, 3x3 kernel)
- MaxPooling2D (2x2 pool size)
- Conv2D (64 filters, 3x3 kernel)
- MaxPooling2D (2x2 pool size)
- Conv2D (128 filters, 3x3 kernel)
- MaxPooling2D (2x2 pool size)
- Flatten
- Dense (128 neurons)
- Dense (output layer with softmax activation)

3. Explanation of Key Components and Modules

3.1 Dataset
The dataset consists of images categorized into different classes of garbage (e.g., plastic, paper, metal, glass).
The dataset is organized into subdirectories where each subdirectory represents a different class of garbage.
The images are preprocessed using ImageDataGenerator to normalize them and split them into training and validation sets.

3.2 Image Preprocessing
Rescaling: The pixel values of the images are rescaled by a factor of 1/255.0 to normalize them to the range [0, 1].
Data Augmentation: In some cases, data augmentation techniques (rotation, zoom, shift, etc.) may be applied to the training images to enhance model generalization.

3.3 CNN Model

The CNN model is built using TensorFlow and Keras libraries. It is designed to perform multi-class classification, utilizing the softmax activation function at the output layer to classify images into multiple categories.

Layers and Activation:The model uses ReLU (Rectified Linear Unit) activations for the convolutional and dense layers. The final output layer uses softmax for multi-class classification.

3.4 Real-Time Webcam Classification
OpenCV Integration: The system uses OpenCV (cv2.VideoCapture) to access the webcam and capture frames.
Prediction: The captured frame is resized and normalized, and the model makes a prediction based on the image's content. The predicted class is displayed on the webcam feed.

4. Endpoint: Predict Garbage Type

URL: /predict
Method: POST
Description: Accepts an image and returns the predicted garbage class.
Request Body:
image: The image file to classify (in base64 encoded format).
Response:
class: The predicted class name (e.g., 'plastic', 'paper').

5. Setup and Usage Instructions

5.1 Prerequisites
Ensure you have the following installed:

- Python 3.x
- TensorFlow
- OpenCV
- Matplotlib
- NumPy
- Install dependencies using pip:
- bash
- Copy code
- pip install tensorflow opencv-python matplotlib numpy

5.2 Dataset Setup
Place your dataset in the specified directory (/Users/raushankuamar/Desktop/Garbage/dataset/Garbage classification) with each class in a separate folder.

Example directory structure:

javascript
Copy code
/Garbage classification/
    /plastic/
    /paper/
    /glass/
    /metal/

5.3 Running the Model
Train the model by executing the script. It will automatically load the dataset, train the model, and display the training progress.
Once the model is trained, it will be ready for real-time webcam predictions.
Run the webcam prediction by executing the script, and the predicted class will be displayed on the webcam window.
To exit the webcam loop, press q while the webcam window is active.

5.4 Model Evaluation
The model's performance can be evaluated using the accuracy and loss metrics printed during training. You can plot these metrics using matplotlib for visualization.

5.5 Saving and Loading the Model

To save the trained model:
python
Copy code
model.save('garbage_classifier.h5')
To load the saved model for later use:

python
Copy code
model = tf.keras.models.load_model('garbage_classifier.h5')

6. Conclusion

This project demonstrates the application of convolutional neural networks for real-time garbage classification. With further improvements such as more data augmentation, fine-tuning, and performance optimizations, this model could be used for real-world applications such as waste management systems.

7. References

- TensorFlow Documentation
- OpenCV Documentation
- Keras Documentation