Edge AI Prototype: Recyclable Item Classification

Overview

This project demonstrates a workflow for developing **Edge AI** applications using TensorFlow and TensorFlow Lite. It focuses on training a lightweight Convolutional Neural Network (CNN) to classify images (using Fashion MNIST as a proxy for recyclable items) and deploying it to an optimized, quantized format suitable for edge devices like Raspberry Pi or mobile phones.

Project Structure

- **edge_ai_prototype.py**: The main Python script that handles data loading, model training, conversion to TFLite, and simulated edge inference.
- **recycling_model_quantized.tflite**: The output file generated by the script. This is the optimized model ready for deployment.

Prerequisites

Ensure you have Python installed along with the following libraries:

pip install tensorflow numpy

How to Run

Execute the script directly from your terminal:

python edge_ai_prototype.py

Workflow Explanation

The script performs the following 5 steps, mirroring a real-world Edge AI lifecycle:

1. **Dataset Preparation**:
    - Loads the dataset (Fashion MNIST).
    - Normalizes pixel values (0-1) and reshapes images to match CNN input requirements (28, 28, 1).
2. **Model Definition**:
    - Constructs a "lightweight" CNN model using Conv2D and MaxPooling2D layers.
    - Designed with fewer parameters to ensure low latency on resource-constrained devices.
3. **Training**:
    - Trains the model using the Adam optimizer and Sparse Categorical Crossentropy loss.
    - *Note: The script defaults to 1 epoch for demonstration speed.*
4. **TFLite Conversion (The "Edge" Step)**:
    - Converts the standard Keras model into a .tflite file.

    - **Quantization**: Applies tf.lite.Optimize.DEFAULT to convert 32-bit floating-point weights to integers. This significantly reduces model size (often by 4x) and speeds up inference on edge hardware without major accuracy loss.
5. **Simulated Edge Inference**:
    - Uses the tf.lite.Interpreter to run predictions using *only* the .tflite file.
    - This simulates how a Raspberry Pi or microcontroller would load and run the model locally, independent of the original training environment.

Expected Output

Upon running, you will see logs indicating the training progress, the file size of the optimized model (typically < 100KB), and a final prediction comparing the Edge model's output to the true label.

...
[Step 4] Converting to TensorFlow Lite with Quantization...
Success! Model saved to disk.
Final Edge Model Size: 82.50 KB (Optimized for IoT Devices)

[Step 5] Simulating On-Device Inference...
Test Image True Label: 9
Edge Model Prediction: 9
Inference complete.