Mobile Dataset Deep Learning Classifier

# 📌 Introduction

This project explores a deep learning-based approach to classify different mobile brands using image data. Leveraging Convolutional Neural Networks (CNNs) built using Keras and TensorFlow, the model is trained to distinguish between popular mobile phone brands such as Apple, Samsung, Oppo, etc.

# 📁 Dataset

The dataset comprises labeled images of various mobile brands. Each brand is stored in a separate folder within the dataset directory. Example brands include:  
- Apple  
- Samsung  
- Oppo  
- Vivo  
- Xiaomi

# ⚙️ Data Preprocessing

1. Images were resized to a consistent dimension (e.g., 224x224).  
2. Data augmentation techniques such as rotation, flipping, zooming, and brightness adjustment were applied using Keras ImageDataGenerator to enhance model generalization.  
3. Normalization was applied to scale pixel values to the range [0, 1].

# 🧠 Model Architecture

The CNN model includes the following layers:  
- Convolutional layers with ReLU activation  
- MaxPooling layers to reduce spatial dimensions  
- Dropout layers to reduce overfitting  
- Dense layers leading to the final softmax output  
  
The model was compiled using:  
- Loss Function: categorical\_crossentropy  
- Optimizer: Adam  
- Metrics: Accuracy

# 📊 Training Details

The model was trained for a number of epochs (e.g., 15-25) with a validation split to monitor performance. Accuracy and loss were plotted for both training and validation sets to evaluate performance over epochs.

# ✅ Results

The trained model achieved a high classification accuracy on the validation set. A confusion matrix was also plotted to visualize prediction correctness across all classes.  
Additional metrics used:  
- Accuracy  
- Validation Accuracy  
- Loss  
- Confusion Matrix

# 🚀 How to Run

1. Clone this repository.  
2. Install dependencies using:  
 pip install tensorflow keras matplotlib numpy  
3. Run the notebook `mobile\_dataset.ipynb` in Jupyter Notebook or Colab.  
4. Upload your dataset in the expected folder structure (each class in separate folders).  
5. Train and evaluate the model.

# 📄 License

This project is licensed under the MIT License.