Pattern Sense: Classifying Fabric Patterns using Deep Learning

# 1. Project Overview

This project automates the classification of fabric patterns (like silk, cotton, etc.) using a deep learning model (ResNet50). It enables users to upload fabric images via a web interface, which the model classifies into one of the predefined categories.

# 2. Project Components

- Model Training (train.py)

- Testing Script (test.py)

- Web Interface with Flask (app.py)

# 3. Dataset Structure

Your dataset is organized into three directories for training, validation, and testing:

dataset034/

├── train/

│ ├── cotton/

│ ├── silk/

│ └── ...

├── val/

└── test/

# 4. Model Building & Training (train.py)

Step-by-step:

1. Image Preprocessing:

- Used ImageDataGenerator with preprocess\_input from ResNet.

- Applied augmentation (rotation, flip, zoom, etc.) to training data.

2. Model Architecture:

- Based on ResNet50, with the top layers removed (include\_top=False).

- Added custom layers: GlobalAveragePooling2D, Dense(512, ReLU), Dropout(0.5), Dense(num\_classes, softmax).

3. Training Phases:

- Phase 1: Freeze base layers, train custom layers.

- Phase 2: Unfreeze last 30 ResNet layers and fine-tune.

4. Callbacks Used:

- EarlyStopping, ReduceLROnPlateau, ModelCheckpoint

5. Model Saving:

- Saved as Model\_CNN.h5

6. Testing Predictions:

- Printed predictions for 10 test images.

# 5. Image Testing Script (test.py)

Step-by-step:

1. Loads trained model fabric\_model\_resnet50.h5

2. Resizes image to 224x224, rescales it

3. Predicts label and confidence using model

4. Displays prediction using matplotlib

# 6. Flask Web App (app.py)

Step-by-step:

1. Flask setup with upload folder 'uploads'

2. Loads trained model fabric\_model\_resnet50.h5

3. Routes:

- '/' → Home page (index.html)

- '/predict' → Accepts and processes uploaded image

- '/uploads/<filename>' → Serves image

4. Image is resized, normalized, and predicted

5. Returns result.html with image, prediction and confidence

# 7. Output Example

Example:

- Predicted Fabric: Silk

- Confidence: 95.87%

- Uploaded image is displayed alongside prediction.

# 8. Technologies Used

| Component | Technology |

|----------------|---------------------------|

| Model | TensorFlow + Keras (ResNet50) |

| Web App | Flask (Python) |

| Frontend | HTML (Templates) |

| Data Augmentation | ImageDataGenerator |

| Model Evaluation | Accuracy, Confidence |

# 9. Deployment Suggestions

Use gunicorn or waitress for production server.

Host on Heroku, Render, or AWS EC2.

Optionally integrate with React or Vue frontend.

# 10. Future Enhancements

- Add webcam capture support

- Include misclassified image analysis

- Add training graphs and model metrics in UI

- Try other architectures like EfficientNet, MobileNet