Import tensorflow as tf

From tensorflow.keras.preprocessing.image import ImageDataGenerator

From tensorflow.keras.applications import MobileNetV2

From tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout

From tensorflow.keras.models import Model

Import matplotlib.pyplot as plt

Import numpy as np

Import os

# ==== CONFIG ====

IMG\_SIZE = (224, 224)

BATCH\_SIZE = 32

EPOCHS = 10

DATA\_DIR = “fabric\_dataset” # adjust to your dataset folder

NUM\_CLASSES = 4 # e.g., stripes, plaid, floral, plain

# ==== DATA GENERATORS ====

Train\_datagen = ImageDataGenerator(

Rescale=1./255,

Rotation\_range=20,

Zoom\_range=0.2,

Horizontal\_flip=True

)

Val\_datagen = ImageDataGenerator(

Rescale=1./255

)

Train\_generator = train\_datagen.flow\_from\_directory(

Os.path.join(DATA\_DIR, “train”),

Target\_size=IMG\_SIZE,

Batch\_size=BATCH\_SIZE,

Class\_mode=”categorical”

)

Val\_generator = val\_datagen.flow\_from\_directory(

Os.path.join(DATA\_DIR, “val”),

Target\_size=IMG\_SIZE,

Batch\_size=BATCH\_SIZE,

Class\_mode=”categorical”

)

# ==== MODEL ====

Base\_model = MobileNetV2(weights=”imagenet”, include\_top=False, input\_shape=(224, 224, 3))

Base\_model.trainable = False # freeze backbone

X = base\_model.output

X = GlobalAveragePooling2D()(x)

X = Dropout(0.3)(x)

Predictions = Dense(NUM\_CLASSES, activation=”softmax”)(x)

Model = Model(inputs=base\_model.input, outputs=predictions)

Model.compile(

Optimizer=tf.keras.optimizers.Adam(),

Loss=”categorical\_crossentropy”,

Metrics=[“accuracy”]

)

Model.summary()

# ==== TRAINING ====

History = model.fit(

Train\_generator,

Epochs=EPOCHS,

Validation\_data=val\_generator

)

# ==== VISUALIZE RESULTS ====

Plt.figure(figsize=(8, 4))

Plt.plot(history.history[“accuracy”], label=”Train Accuracy”)

Plt.plot(history.history[“val\_accuracy”], label=”Val Accuracy”)

Plt.xlabel(“Epoch”)

Plt.ylabel(“Accuracy”)

Plt.legend()

Plt.title(“Training and Validation Accuracy”)

Plt.show()

Plt.figure(figsize=(8, 4))

Plt.plot(history.history[“loss”], label=”Train Loss”)

Plt.plot(history.history[“val\_loss”], label=”Val Loss”)

Plt.xlabel(“Epoch”)

Plt.ylabel(“Loss”)

Plt.legend()

Plt.title(“Training and Validation Loss”)

Plt.show()

# ==== EXAMPLE PREDICTION ====

From tensorflow.keras.preprocessing import image

# Change the path to a real image in your validation set

Img\_path = os.path.join(DATA\_DIR, “val/stripes/example.jpg”) # adjust this path

Img = image.load\_img(img\_path, target\_size=IMG\_SIZE)

X = image.img\_to\_array(img) / 255.0

X = np.expand\_dims(x, axis=0)

Pred = model.predict(x)

Class\_idx = np.argmax(pred)

Class\_labels = list(train\_generator.class\_indices.keys())

Print(“Predicted class:”, class\_labels[class\_idx])