import os

import numpy as np

import matplotlib.pyplot as plt

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.applications import ResNet50

from tensorflow.keras.models import Model

from tensorflow.keras.layers import Dense, Flatten, Dropout, GlobalAveragePooling2D

from tensorflow.keras.optimizers import Adam

from sklearn.metrics import classification\_report, confusion\_matrix

# Dataset path

DATASET\_PATH = "rice\_dataset"

IMG\_SIZE = (224, 224)

BATCH\_SIZE = 32

EPOCHS = 10

# 1. Data Augmentation & Generators

train\_datagen = ImageDataGenerator(

rescale=1./255,

rotation\_range=30,

zoom\_range=0.2,

horizontal\_flip=True,

validation\_split=0.2

)

train\_gen = train\_datagen.flow\_from\_directory(

DATASET\_PATH,

target\_size=IMG\_SIZE,

batch\_size=BATCH\_SIZE,

class\_mode='categorical',

subset='training'

)

val\_gen = train\_datagen.flow\_from\_directory(

DATASET\_PATH,

target\_size=IMG\_SIZE,

batch\_size=BATCH\_SIZE,

class\_mode='categorical',

subset='validation'

)

# 2. Load Pretrained ResNet50 without top layer

base\_model = ResNet50(weights='imagenet', include\_top=False, input\_shape=IMG\_SIZE + (3,))

base\_model.trainable = False # Freeze base

# 3. Add custom layers

x = base\_model.output

x = GlobalAveragePooling2D()(x)

x = Dense(256, activation='relu')(x)

x = Dropout(0.3)(x)

predictions = Dense(train\_gen.num\_classes, activation='softmax')(x)

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

# 4. Compile Model

model.compile(optimizer=Adam(1e-4), loss='categorical\_crossentropy', metrics=['accuracy'])

# 5. Train Model

history = model.fit(

train\_gen,

epochs=EPOCHS,

validation\_data=val\_gen

)

# 6. Evaluate

val\_gen.reset()

Y\_pred = model.predict(val\_gen, verbose=1)

y\_pred = np.argmax(Y\_pred, axis=1)

print("Classification Report:")

print(classification\_report(val\_gen.classes, y\_pred, target\_names=list(val\_gen.class\_indices.keys())))

# 7. Save model

model.save("grainpalette\_rice\_model.h5")

print("Model saved as grainpalette\_rice\_model.h5")