**A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning**

1. **Introduction**

Rice is a staple food consumed globally, and identifying its type plays a crucial role in quality control, packaging, and supply chain management. This document presents a simple yet illustrative application of transfer learning using MobileNetV2 for classifying synthetic images of three rice types: Basmati, Jasmine, and Arborio. Though synthetic data is used here, the approach is applicable to real-world datasets with minimal changes.

1. **Dependencies and Imports**import os import numpy as np from PIL import Image 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 from tensorflow.keras.optimizers import Adam import matplotlib.pyplot as plt
2. **Synthetic Dataset Creation**

**Directory Structure and Class Labels**

BASE\_DIR = '/content/synthetic\_rice\_data'

CLASSES = ['Basmati', 'Jasmine', 'Arborio']

NUM\_TRAIN = 20

NUM\_VAL = 5

IMG\_SIZE = 96

**Create Directories and Generate Synthetic Images** def make\_dir\_structure():

for mode in ['train', 'val']: for cls in CLASSES: os.makedirs(os.path.join(BASE\_DIR, mode, cls), exist\_ok=True)

def create\_synthetic\_images(): for mode, count in [('train', NUM\_TRAIN), ('val', NUM\_VAL)]:

for label, cls in enumerate(CLASSES): for i in range(count):

arr = np.random.rand(IMG\_SIZE, IMG\_SIZE, 3) \* (label+1) / len(CLASSES) img = Image.fromarray(np.uint8(arr \* 255)) img.save(os.path.join(BASE\_DIR, mode, cls, f'{cls}\_{i}.png'))

make\_dir\_structure() create\_synthetic\_images()

1. **Data Generators**train\_dir = os.path.join(BASE\_DIR, 'train') val\_dir = os.path.join(BASE\_DIR, 'val')

train\_datagen = ImageDataGenerator(rescale=1./255) val\_datagen = ImageDataGenerator(rescale=1./255)

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

train\_dir, target\_size=(IMG\_SIZE, IMG\_SIZE), batch\_size=4, class\_mode='categorical'

)

val\_gen = val\_datagen.flow\_from\_directory( val\_dir, target\_size=(IMG\_SIZE, IMG\_SIZE), batch\_size=4, class\_mode='categorical'

)

1. **Transfer Learning Model: MobileNetV2** base\_model = MobileNetV2(weights='imagenet', include\_top=False, input\_shape=(IMG\_SIZE, IMG\_SIZE, 3)) base\_model.trainable = False

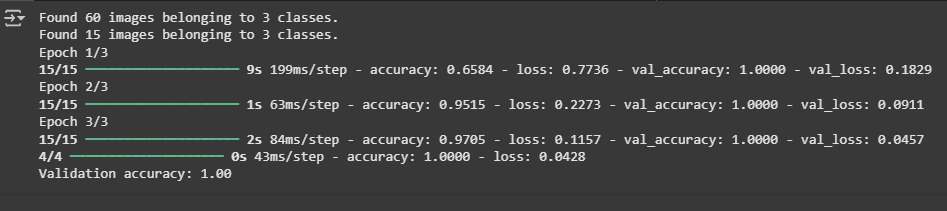
x = GlobalAveragePooling2D()(base\_model.output) x = Dropout(0.2)(x) output = Dense(len(CLASSES), activation='softmax')(x) model = Model(inputs=base\_model.input, outputs=output)

model.compile(optimizer=Adam(1e-3), loss='categorical\_crossentropy', metrics=['accuracy'])

1. **Model Training**model.fit(train\_gen, validation\_data=val\_gen, epochs=3)
2. **Evaluation**val\_gen.reset() loss, acc = model.evaluate(val\_gen) print(f'Validation accuracy: {acc:.2f}')
3. **Conclusion**

This exercise demonstrates the power and efficiency of transfer learning for image classification tasks. With a small synthetic dataset and a lightweight model like MobileNetV2, it's possible to achieve reasonable classification performance quickly. For production-level use, this approach can be scaled with real datasets and fine-tuning of the base model.

**9.Output**

****