**Soft Computing Digital Assignment**

**22MCA0121 Varun Tiwari**

**22MCA0133 Priyansh Das**

**22MCA0139 Rajat Singh**

**22MCA0153 Austin Johnson**

**22MCA0162 Hrishikesh S G**

**Code (All together):**

# code cell 1

import os

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

from tensorflow.keras.applications import ResNet50, VGG16

# code cell 2

data\_directory = r'archive\train'

train\_datagen = ImageDataGenerator(rescale=1./255, validation\_split=0.3)

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

    data\_directory,

    target\_size=(224, 224),

    batch\_size=32,

    class\_mode='categorical',

    subset='training'

)

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

    data\_directory,

    target\_size=(224, 224),

    batch\_size=32,

    class\_mode='categorical',

    subset='validation'

)

# code cell 3

alexnet\_model = Sequential()

alexnet\_model.add(Conv2D(96, kernel\_size=(11, 11), strides=(4, 4), activation='relu', input\_shape=(224, 224, 3)))

alexnet\_model.add(MaxPooling2D(pool\_size=(3, 3), strides=(2, 2)))

alexnet\_model.add(Conv2D(256, kernel\_size=(5, 5), activation='relu'))

alexnet\_model.add(MaxPooling2D(pool\_size=(3, 3), strides=(2, 2)))

alexnet\_model.add(Conv2D(384, kernel\_size=(3, 3), activation='relu'))

alexnet\_model.add(Conv2D(384, kernel\_size=(3, 3), activation='relu'))

alexnet\_model.add(Conv2D(256, kernel\_size=(3, 3), activation='relu'))

alexnet\_model.add(MaxPooling2D(pool\_size=(3, 3), strides=(2, 2)))

alexnet\_model.add(Flatten())

alexnet\_model.add(Dense(4096, activation='relu'))

alexnet\_model.add(Dropout(0.5))

alexnet\_model.add(Dense(4096, activation='relu'))

alexnet\_model.add(Dropout(0.5))

alexnet\_model.add(Dense(5, activation='softmax'))

alexnet\_model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

alexnet\_history = alexnet\_model.fit(train\_generator, validation\_data=validation\_generator, epochs=3)

# code cell 4

# Save the trained model

alexnet\_model.save('alexnet\_model.h5')

# code cell 5

# Load the saved model

saved\_model = tf.keras.models.load\_model('alexnet\_model.h5')

# Test the model on new data

test\_directory =  r'archive\test'

test\_datagen = ImageDataGenerator(rescale=1./255)

test\_generator = test\_datagen.flow\_from\_directory(

    test\_directory,

    target\_size=(224, 224),

    batch\_size=32,

    class\_mode='categorical',

    shuffle=False

)

# Evaluate the saved model

scores = saved\_model.evaluate(test\_generator)

print("Test Loss:", scores[0])

print("Test Accuracy:", scores[1])

# code cell 6

# Make predictions using the saved model

predictions = saved\_model.predict(test\_generator)

predictions

# code cell 7

resnet\_model = Sequential()

resnet\_model.add(ResNet50(include\_top=False, pooling='avg', weights='imagenet'))

resnet\_model.add(Dense(5, activation='softmax'))

resnet\_model.layers[0].trainable = False

resnet\_model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

resnet\_history = resnet\_model.fit(train\_generator, validation\_data=validation\_generator, epochs=3)

# code cell 8

resnet\_model.save('resnet\_model.h5')

# code cell 9

# Load the saved model

saved\_model = tf.keras.models.load\_model('resnet\_model.h5')

# Test the model on new data

test\_directory =  r'archive\test'

test\_datagen = ImageDataGenerator(rescale=1./255)

test\_generator = test\_datagen.flow\_from\_directory(

    test\_directory,

    target\_size=(224, 224),

    batch\_size=32,

    class\_mode='categorical',

    shuffle=False

)

# Evaluate the saved model

scores = saved\_model.evaluate(test\_generator)

print("Test Loss:", scores[0])

print("Test Accuracy:", scores[1])

# code cell 10

vgg\_model = Sequential()

vgg\_model.add(VGG16(include\_top=False, pooling='avg', weights='imagenet'))

vgg\_model.add(Dense(5, activation='softmax'))

vgg\_model.layers[0].trainable = False

vgg\_model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

vgg\_history = vgg\_model.fit(train\_generator, validation\_data=validation\_generator, epochs=3)

# code cell 11

vgg\_model.save('vgg\_model.h5')

# code cell 12

# Load the saved model

saved\_model = tf.keras.models.load\_model('vgg\_model.h5')

# Test the model on new data

test\_directory =  r'archive\test'

test\_datagen = ImageDataGenerator(rescale=1./255)

test\_generator = test\_datagen.flow\_from\_directory(

    test\_directory,

    target\_size=(224, 224),

    batch\_size=32,

    class\_mode='categorical',

    shuffle=False

)

# Evaluate the saved model

scores = saved\_model.evaluate(test\_generator)

print("Test Loss:", scores[0])

print("Test Accuracy:", scores[1])

**Jupyter Notebook screenshots (code and outputs)**

**A screenshot of a computer program

Description automatically generated with low confidence**

**A screenshot of a computer program

Description automatically generated with medium confidence**

**A screenshot of a computer code

Description automatically generated with low confidence**

**A screenshot of a computer program

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer code

Description automatically generated with low confidence**

**A screenshot of a computer program

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with low confidence**

**A screenshot of a computer program

Description automatically generated with medium confidence**

**Conclusion**

**This assignment involves the comparative study of AlexNet, ResNet and VGGNet models on Rice Image Dataset. From the above analysis (code) it was observed that:**

* **AlexNet model: gives a accuracy of 95.99% with a loss of 9%**
* **ResNet model: gives a accuracy of 92% with a loss of 15%**
* **VGGNet model: gives a accuracy of 98% with a loss of 8.3%**

**The number of epochs used is 3. This was done to quickly train the model as the dataset was big. For the first model, each epoch took 20 minutes, for the second it took 40 minutes and for the third it took 60 minutes.**

**If the number of epochs are increased than the loss can be decreased significantly**

**At the end, it is observed that the VGGNet model gives comparatively higher accuracy and has less test loss than the other 2 models. So VGGNet deep learning model best fits for this Rice Image Dataset.**