Bahria University,
Karachi Campus

LAB EXPERIMENT NO.
10

LIST OF TASKS

TASK NO	OBJECTIVE
1.	Imagine you're a farmer facing the constant threat of plant diseases, which can sweep through your fields and ruin your entire crop. To help tackle this issue, there's a smart tool powered by a type of artificial intelligence known as a Convolutional Neural Network (CNN). This innovative tool works by analyzing photos of plants. It's trained to spot the subtle signs of various common diseases, such as rust, blight, or mildew, as well as confirming when plants are healthy. This means that instead of guessing or waiting for visible damage, you get an early warning system. With quick and accurate detection, you can act fast, treating the affected plants and protecting your crop from further harm. This use of technology not only saves time and resources but also gives you peace of mind, knowing you're doing all you can to keep your plants healthy and thriving.

Submitted On:
6/7/2024

TASK NO 1: Develop a chatbot using Python and NLTK that can handle and respond to user queries by understanding their context, even if the queries do not exactly match the entries in the chatbot's knowledge base.

```
import warnings
warnings.filterwarnings("ignore")
import tensorflow as tf
import matplotlib.pyplot as plt
tf.compat.v1.set_random_seed(0)
from tensorflow import keras
import numpy as np
np.random.seed(0)
import itertools
from keras.preprocessing.image import
image_dataset_from_directory
from tensorflow.keras.layers.experimental.preprocessing import
Rescaling
from sklearn.metrics import precision_score, accuracy_score,
recall_score, confusion_matrix, ConfusionMatrixDisplay
train_gen = image_dataset_from_directory(directory="../input/new-
plant-diseases-dataset/New Plant Diseases
Dataset(Augmented)/New Plant Diseases
Dataset(Augmented)/train",image_size=(256, 256))
test_gen = image_dataset_from_directory(directory="../input/new-
plant-diseases-dataset/New Plant Diseases
Dataset(Augmented)/New Plant Diseases
Dataset(Augmented)/valid",image_size=(256, 256))
rescale = Rescaling(scale=1.0/255)
train_gen = train_gen.map(lambda
image,label:(rescale(image),label))
test_gen  = test_gen.map(lambda image,label:(rescale(image),label))
model = keras.Sequential()
model.add(keras.layers.Conv2D(32,(3,3),activation="relu",padding
="same",input_shape=(256,256,3)))
model.add(keras.layers.Conv2D(32,(3,3),activation="relu",padding
="same"))
model.add(keras.layers.MaxPooling2D(3,3))
model.add(keras.layers.Conv2D(64,(3,3),activation="relu",padding
="same"))
model.add(keras.layers.Conv2D(64,(3,3),activation="relu",padding
="same"))
model.add(keras.layers.MaxPooling2D(3,3))
model.add(keras.layers.Conv2D(128,(3,3),activation="relu",padding
="same"))
model.add(keras.layers.Conv2D(128,(3,3),activation="relu",padding
="same"))
model.add(keras.layers.MaxPooling2D(3,3))
model.add(keras.layers.Conv2D(256,(3,3),activation="relu",padding
="same"))
model.add(keras.layers.Conv2D(256,(3,3),activation="relu",padding
="same"))
model.add(keras.layers.Conv2D(512,(5,5),activation="relu",padding
="same"))
model.add(keras.layers.Conv2D(512,(5,5),activation="relu",padding
="same"))
model.add(keras.layers.Flatten())
model.add(keras.layers.Dense(1568,activation="relu"))
model.add(keras.layers.Dropout(0.5))
model.add(keras.layers.Dense(38,activation="softmax"))
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(optimizer=opt,loss="sparse_categorical_crossentrop
y",metrics=['accuracy'])
history = model.fit_generator(train_gen,
validation_data=test_gen,epochs = 10)
print("Train Accuracy  : {:.2f}
%".format(history.history['accuracy'][-1]*100))
print("Test Accuracy   : {:.2f} %".format(accuracy_score(labels,
predictions) * 100))
print("Precision Score : {:.2f} %".format(precision_score(labels,
predictions, average='micro') * 100))
print("Recall Score    : {:.2f} %".format(recall_score(labels,
predictions, average='micro') * 100)
plt.figure(figsize= (20,5))
cm = confusion_matrix(labels, predictions)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                              display_labels=list(range(1,39)))
fig, ax = plt.subplots(figsize=(15,15))
disp.plot(ax=ax,colorbar= False,cmap = 'YlGnBu')
plt.title("Confusion Matrix")
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show(
```

```
Train Accuracy  : 98.29 %
Test Accuracy   : 96.77 %
Precision Score : 96.77 %
Recall Score    : 96.77 %
```