

```

from google.colab import drive
drive.mount('/content/drive')

Mounted at /content/drive

ls

drive sample_data

cd '/content/drive/My Drive/PLANT DISEASE RECOGNITION'

/content/drive/My Drive/PLANT DISEASE RECOGNITION

mkdir config datasets models

mkdir: cannot create directory 'config': File exists
mkdir: cannot create directory 'datasets': File exists
mkdir: cannot create directory 'models': File exists

cd config

/content/drive/My Drive/PLANT DISEASE RECOGNITION/config

# Change directory to the previously created 'config' folder

# Upload the downloaded json file from your computer to Google drive
#from google.colab import files
#files.upload()

cd '/content/drive/My Drive/PLANT DISEASE RECOGNITION/datasets'

/content/drive/My Drive/PLANT DISEASE RECOGNITION/datasets

import os
os.environ['KAGGLE_CONFIG_DIR'] = "/content/drive/My Drive/PLANT DISEASE RECOGNITION/config"

!kaggle datasets download -d vipooooo1/new-plant-diseases-dataset

new-plant-diseases-dataset.zip: Skipping, found more recently modified local copy (use --force to force download)

ls

sample_data

#Unzipping the zip files to extract the dataset folder and deleting the zip files
!unzip *.zip && rm *.zip

Archive: new-plant-diseases-dataset.zip
replace New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Apple_scab/00075aa8-d81a-4184-8541-b69

```



```

for (root,dirs,files) in os.walk('.', topdown = True):
    print(root, dirs)

. ['New Plant Diseases Dataset(Augmented)', 'new plant diseases dataset(augmented)', 'test']
./New Plant Diseases Dataset(Augmented) ['New Plant Diseases Dataset(Augmented)']
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented) ['train', 'valid']
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train ['Apple___Apple_scab', 'Apple___Black_rot', 'Ap
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Apple_scab []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Black_rot []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Cedar_apple_rust []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Blueberry___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Cherry_(including_sour)___Powdery_mildew []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Cherry_(including_sour)___healthy []

```

```
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Corn_(maize)___Cercospora_leaf_spot Gray_leaf_s
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Corn_(maize)___Common_rust_ []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Corn_(maize)___Northern_Leaf_Blight []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Corn_(maize)___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Grape___Black_rot []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Grape___Esca_(Black_Measles) []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Grape___Leaf_blight_(Isariopsis_Leaf_Spot) []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Grape___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Orange___Haunglongbing_(Citrus_greening) []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Peach___Bacterial_spot []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Peach___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Pepper,_bell___Bacterial_spot []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Pepper,_bell___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Potato___Early_blight []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Potato___Late_blight []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Potato___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Raspberry___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Soybean___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Squash___Powdery_mildew []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Strawberry___Leaf_scorch []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Strawberry___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___Bacterial_spot []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___Early_blight []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___Late_blight []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___Leaf_Mold []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___Septoria_leaf_spot []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___Spider_mites Two-spotted_spider_mite []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___Target_Spot []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___Tomato_Yellow_Leaf_Curl_Virus []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___Tomato_mosaic_virus []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/train/Tomato___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid ['Apple___Apple_scab', 'Apple___Black_rot', 'Ap
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Apple___Apple_scab []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Apple___Black_rot []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Apple___Cedar_apple_rust []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Apple___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Blueberry___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Cherry_(including_sour)___Powdery_mildew []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Cherry_(including_sour)___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Corn_(maize)___Cercospora_leaf_spot Gray_leaf_s
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Corn_(maize)___Common_rust_ []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Corn_(maize)___Northern_Leaf_Blight []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Corn_(maize)___healthy []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Grape___Black_rot []
./New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid/Grape___Esca_(Black_Measles) []
```

```
base_dir = './New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)'
os.listdir(base_dir)
```

```
['train', 'valid']
```

```
len(os.listdir(os.path.join(base_dir, 'train')))
```

```
len(os.listdir(os.path.join(base_dir, 'valid')))
```

▼ Importing the required libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```

import cv2
import PIL
import tensorflow as tf
from tensorflow.python import keras
import warnings
import argparse
warnings.filterwarnings('ignore')
from keras.preprocessing.image import ImageDataGenerator, img_to_array, load_img
from sklearn.preprocessing import LabelBinarizer, StandardScaler
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical

```

```

fig, axes = plt.subplots(1, 5, figsize=(18, 18))
images = os.listdir(os.path.join(base_dir, 'train/Apple__Black_rot'))
for _ in range(5):
    ax = axes[_]
    image_path = base_dir + '/train/Apple__Black_rot/' + images[_]
    img = cv2.cvtColor(cv2.imread(image_path), cv2.COLOR_BGR2RGB)
    ax.imshow(img)
    ax.axis('off')
plt.show()

```

▼ Data Augmentation transformations for the train dataset

```

train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   fill_mode='nearest')

# Data augmentation for validation dataset
validation_datagen = ImageDataGenerator(rescale = 1./255)

```

▼ Loading the images from their directories

```

BATCH_SIZE = 32

train_set_from_dir = train_datagen.flow_from_directory(os.path.join(base_dir, 'train'),
                                                       target_size = (224, 224),
                                                       batch_size = BATCH_SIZE,
                                                       class_mode = 'categorical'
                                                       )

validation_set_from_dir = validation_datagen.flow_from_directory(os.path.join(base_dir, 'valid'),
                                                                target_size = (224, 224),
                                                                batch_size = BATCH_SIZE,
                                                                class_mode = 'categorical'
                                                                )

Found 43014 images belonging to 38 classes.
Found 0 images belonging to 0 classes.

classes_dict = train_set_from_dir.class_indices

img = train_set_from_dir.filepaths[np.random.random_integers(low=0, high=train_set_from_dir.samples)]
img = cv2.imread(img)
plt.imshow(img)

```

```
import json
with open('/content/drive/My Drive/PLANT DISEASE RECOGNITION/class_indices.json','w') as f:
    json.dump(classes_dict, f)
```

▼ Transfer learning with MobileNet architecture

```
from keras import Input, Model
from keras.applications import MobileNet
from keras.layers import Flatten, Dense, Dropout
from keras.layers import GlobalAveragePooling2D
from keras.layers import BatchNormalization
from keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau
from keras.models import Model, model_from_json
from tensorflow.keras.optimizers import Adam
from sklearn.metrics import classification_report, roc_auc_score
```

```
base_model = MobileNet(
    #Load weights into the pre-trained MobileNet model
    weights="imagenet",
    input_shape=(224, 224, 3),
    #Exclude the ImageNet classifier at the top of the model
    include_top=False
)
```

Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilenet/mobilenet_1_0_224_tf_no_top.h5
17225924/17225924 [=====] - 0s 0us/step

```
base_model.summary()
```

Model: "mobilenet_1.00_224"

Layer (type)	Output Shape	Param #
=====		
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
conv1 (Conv2D)	(None, 112, 112, 32)	864
conv1_bn (BatchNormalizati on)	(None, 112, 112, 32)	128
conv1_relu (ReLU)	(None, 112, 112, 32)	0
conv_dw_1 (DepthwiseConv2D)	(None, 112, 112, 32)	288
)		

```
conv_dw_1_bn (BatchNormali (None, 112, 112, 32) 128
zation)

conv_dw_1_relu (ReLU) (None, 112, 112, 32) 0

conv_pw_1 (Conv2D) (None, 112, 112, 64) 2048

conv_pw_1_bn (BatchNormali (None, 112, 112, 64) 256
zation)

conv_pw_1_relu (ReLU) (None, 112, 112, 64) 0

conv_pad_2 (ZeroPadding2D) (None, 113, 113, 64) 0

conv_dw_2 (DepthwiseConv2D (None, 56, 56, 64) 576
)

conv_dw_2_bn (BatchNormali (None, 56, 56, 64) 256
zation)

conv_dw_2_relu (ReLU) (None, 56, 56, 64) 0

conv_pw_2 (Conv2D) (None, 56, 56, 128) 8192

conv_pw_2_bn (BatchNormali (None, 56, 56, 128) 512
zation)

conv_pw_2_relu (ReLU) (None, 56, 56, 128) 0

conv_dw_3 (DepthwiseConv2D (None, 56, 56, 128) 1152
)

conv_dw_3_bn (BatchNormali (None, 56, 56, 128) 512
zation)

conv_dw_3_relu (ReLU) (None, 56, 56, 128) 0

conv_pw_3 (Conv2D) (None, 56, 56, 128) 16384

conv_pw_3_bn (BatchNormali (None, 56, 56, 128) 512
zation)

head_model = base_model.output
head_model = GlobalAveragePooling2D()(head_model)
# Regularization by applying DropOut
head_model = Dropout(0.2)(head_model)
outputs = Dense(28, activation="softmax")(head_model)
mobilenet_model = Model(base_model.input, outputs, name='pretrained_mobilenet' )

for layer in mobilenet_model.layers:
    layer.trainable = False

# or if we want to set the first 20 layers of the network to be non-trainable
for layer in mobilenet_model.layers[:20]:
    layer.trainable=False
for layer in mobilenet_model.layers[20:]:
    layer.trainable=True

# Compiling the model with the optimizer and loss function

mobilenet_model.compile(optimizer = Adam(),
                        loss = 'categorical_crossentropy',
                        metrics = ['accuracy']
)

mobilenet_model.summary()
for idx, layer in enumerate(mobilenet_model.layers):
    print(idx, layer.name, layer.trainable)

Model: "pretrained_mobilenet"
```

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
conv1 (Conv2D)	(None, 112, 112, 32)	864
conv1_bn (BatchNormalizati	(None, 112, 112, 32)	128
on)		

conv1_relu (ReLU)	(None, 112, 112, 32)	0
conv_dw_1 (DepthwiseConv2D)	(None, 112, 112, 32)	288
conv_dw_1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv_dw_1_relu (ReLU)	(None, 112, 112, 32)	0
conv_pw_1 (Conv2D)	(None, 112, 112, 64)	2048
conv_pw_1_bn (BatchNormalization)	(None, 112, 112, 64)	256
conv_pw_1_relu (ReLU)	(None, 112, 112, 64)	0
conv_pad_2 (ZeroPadding2D)	(None, 113, 113, 64)	0
conv_dw_2 (DepthwiseConv2D)	(None, 56, 56, 64)	576
conv_dw_2_bn (BatchNormalization)	(None, 56, 56, 64)	256
conv_dw_2_relu (ReLU)	(None, 56, 56, 64)	0
conv_pw_2 (Conv2D)	(None, 56, 56, 128)	8192
conv_pw_2_bn (BatchNormalization)	(None, 56, 56, 128)	512
conv_pw_2_relu (ReLU)	(None, 56, 56, 128)	0
conv_dw_3 (DepthwiseConv2D)	(None, 56, 56, 128)	1152
conv_dw_3_bn (BatchNormalization)	(None, 56, 56, 128)	512
conv_dw_3_relu (ReLU)	(None, 56, 56, 128)	0
conv_pw_3 (Conv2D)	(None, 56, 56, 128)	16384
conv_pw_3_bn (BatchNormalization)	(None, 56, 56, 128)	512

```
## Setting up callbacks for our model
```

```
callbacks = [
    ModelCheckpoint('content/drive/My Drive/PLANT DISEASE RECOGNITION/checkpoints/mobilenet_plantdiseases.h5', save_best_only=True, monitor='val_loss', patience=2, verbose=1),
    EarlyStopping(monitor='val_loss', patience=2, verbose=1),
    ReduceLROnPlateau(factor=0.1, patience=10, min_lr=0.00001, verbose=1)
]
```

```
history = mobilenet_model.fit(train_set_from_dir,
                              validation_data = valid_set_from_dir,
                              epochs = 10,
                              # Use 128 random batches for training set
                              steps_per_epoch = 128, # 128 x 32 = 2**12 random samples
                              # Use 64 random batches for training validation set
                              validation_steps = 100, # 100 x 32 = 3200 random samples
                              callbacks = callbacks
                              )

n = 6
plt.figure(figsize = (8,5))
plt.plot(np.arange(1,n+1), history.history['loss'], label = 'train_loss')
plt.plot(np.arange(1,n+1), history.history['val_loss'], label = 'val_loss')
plt.plot(np.arange(1,n+1), history.history['accuracy'], label = 'train_accuracy')
plt.plot(np.arange(1,n+1), history.history['val_accuracy'], label = 'val_accuracy')

plt.grid(True)
plt.legend(loc = "best")
plt.savefig('/content/drive/My Drive/PLANT DISEASE RECOGNITION/performance.jpg')
plt.show()
```

```

# Evaluate the model on the Validation dataset
results = mobilenet_model.evaluate(validation_set_from_dir)

550/550 [=====] - 59s 107ms/step

print("Validation Loss :-", results[0])
print("="*30)
print("Validation Accuracy :-", results[1])

Validation Loss :- 0.014583874493837357
=====
Validation Accuracy :- 0.8529478907585144

cd '/content/drive/My Drive/PLANT DISEASE RECOGNITION'

/content/drive/My Drive/PLANT DISEASE RECOGNITION

mkdir models

# Save model in HDF5 format
mobilenet_model.save('models/mobilenet_model.h5')

# Testing on a random image from the test images directory
from PIL import Image
np.random.seed(200)
idx = np.random.randint(30)
test_images_dir = os.path.join('datasets/test', 'test')
test1 = Image.open(os.path.join(test_images_dir, os.listdir(test_images_dir)[idx]))

26

plt.imshow(test1)
plt.title(os.listdir(test_images_dir)[idx])

test1 = test1.resize((224,224))
test1_scaled = np.expand_dims(np.asarray(test1), axis = 0) / 255
predictions = mobilenet_model.predict(test1_scaled)
print(predictions)

[[[3.2511131e-12 2.0627213e-12 1.0672722e-08 7.4512084e-11 9.7505737e-10
 2.5782879e-07 3.5203795e-11 7.8878709e-10 3.4717043e-10 1.1837048e-10
 1.6650042e-11 1.1119714e-09 3.0714025e-07 8.4619423e-08 2.6696884e-10
 4.2097961e-09 1.6930435e-12 7.7451039e-11 8.5175400e-10 3.9144616e-12
 4.2878266e-13 1.6123290e-10 3.3550971e-11 2.9634718e-08 7.4205871e-11
 9.7984865e-10 7.2316797e-10 1.9204359e-08 3.5286668e-08 7.3232462e-07
 1.6105752e-05 1.7480016e-06 1.6131581e-08 3.9582712e-08 2.0550729e-07
 9.9997985e-01 5.7835371e-07 5.9569259e-11]]]

```



```
classes_dict = train_set_from_dir.class_indices
classes_dict = { v:k for (k,v) in classes_dict.items() }
classes_dict[np.argmax(predictions)]
```

Accurate !

► Converting model to Tensorflow js

[] ↳ 7 cells hidden