Team ID : **PNT2022TMID14992**

Date : 27 October 2022

= ASSIGNMENT_3 :- (Narendran N.D)

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

     Mounted at /content/drive
```

```
#Extracting Data
!unzip ”/content/drive/MyDrive/Colab Notebooks/Flowers-Dataset ( Splitted ).zip‟
```

- Image Augmentation

```
#Import req. Lib.
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
#Augmentation On Training Variable
train_datagen = ImageDataGenerator(rescale= 1./255,
                zoom_range=0.2,
                horizontal_flip =True)
```

```
#Augmentation On Training Variable
test_datagen = ImageDataGenerator(rescale= 1./255)
```

```
#Augmentation On Training Variable
ftrain = train_datagen.flow_from_directory('/content/Flowers-Dataset ( Splitted )/Training',
                                           target_size=(64,64),
                                           class_mode='categorical',
                                           batch_size=100)

     Found 4086 images belonging to 5 classes.
```

```
#Augmentation On Training Variable
ftest   test_datagen.flow_from_directory('/content/Flowers-Dataset ( Splitted )/Testing',
                                         target_size=(64,64),
                                         class_mode='categorical',
                                         batch_size=100)

     Found 231 images belonging to 5 classes.
```

- Creating The Model :

Adding Layers :

```
#Import req. Lib.
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense
```

```
# Build a CNN Block:
model = Sequential() #intializing sequential model
model.add(Convolution2D(32,(3,3),activation='relu', input_shape=(64,64,3))) #convolution laye
model.add(MaxPooling2D(pool_size=(2, 2))) #Maxpooling layer
model.add(Flatten()) #Flatten layer
model.add(Dense(400,activation='relu')) #Hidden Layer 1
model.add(Dense(200,activation='relu')) #Hidden Layer 2
model.add(Dense(5,activation='softmax')) #0utput Layer
```

Compiling :

```
# Compiling The Model...
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
```

Fit / Train The Model :

```
#Train Model:
model.fit_generator(ftrain,
                    steps_per_epoch=len(ftrain),
                    epochs=10,
                    validation_data=ftest,
                    validation_steps=len(ftest))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: UserWarning: ’Model.fit

Epoch 1/10
41/41 [==============================] - 30s 711ms/step - loss: 1.3702 - accuracy: 0.43
Epoch 2/10
41/41 [==============================] - 29s 70lms/step - loss: 1.0398 - accuracy: 0.59:
Epoch 3/10
41/41 [==============================] - 29s 70lms/step - loss: 0.9675 - accuracy: 0.62:
Epoch 4/10
41/41 [==============================] - 29s 701ms/step - loss: 0.8853 - accuracy: 0.66a
Epoch 5/10
41/41 [==============================] - 29s 703ms/step - loss: 0.8395 - accuracy: 0.68
Epoch 6/10
```

```
41/41 [==============================] - 29s 701ms/step - loss: 0.7740 - accuracy: 0.691
Epoch 7/10
41/41 [==============================] - 29s 701ms/step - loss: 0.7467 - accuracy: 0.711
Epoch 8/10
41/41 [==============================] - 29s 697ms/step - loss: 0.6988 - accuracy: 0.73:
Epoch 9/10
41/41 [==============================] - 29s 700ms/step - loss: 0.6728 - accuracy: 0.74
Epoch 10/10
41/41 [==============================] - 30s 721ms/step - loss: 0.6166 - accuracy: 0.76i
<keras.callbacks.History at 0x7fd7e3c5eb90>
```

Saving The Model :

```
#Save Model
model.save('flowers.h5')
```

- Testing The Model :

```
#Import req. Lib.
from tensorflow.keras.preprocessing import image
import numpy as np
```

```
#Test1ng No 1 : -
img   image.load_img('/content/Flowers-Dataset ( Splitted )/Testing/daisy/34275662120_7757a15
f = image.img_to_array(img) #Convertinng image to array
f = np.expand_dims(f,axis=0) #Expanding dimensions
pred = np.argmax(model.predict(f)) #predicting higher propability index
op   ['daisy','dandelion','rose','sunflower','tulip'] #Creating List
op[pred] #List indexing with output

     'daisy'
```

```
#Testing No 2 :-
img   image.load_img('/content/Flowers-Dataset ( Splitted )/Testing/sunflower/14121915990_4b7
f = image.img_to_array(img) #Convertinng image to array
f = np.expand_dims(f,axis=0) #Expanding dimensions
pred = np.argmax(model.predict(f)) #predicting higher propability index
op   ['daisy','dandelion','rose','sunflower','tulip'] #Creating List
op[pred] #List indexing with output

     'sunflower'
```

```
#Test1ng No 3 : -
img   image.load_img('/content/Flowers-Dataset ( Splitted )/Testing/tulip/19425920580_cdc8f49
f = image.img_to_array(img) #Convertinng image to array
```

```
f = np.expand_dims(f,axis=0) #Expanding dimensions
pred = np.argmax(model.predict(f)) #predicting higher propability index
op   ['daisy','dandelion','rose','sunflower','tulip'] #Creating List
op[pred] #List indexing with output
```

```
     'tulip'
```

For the above three tests performed the Model has predicted the images correctly..!