

Assignment -3

Python Programming

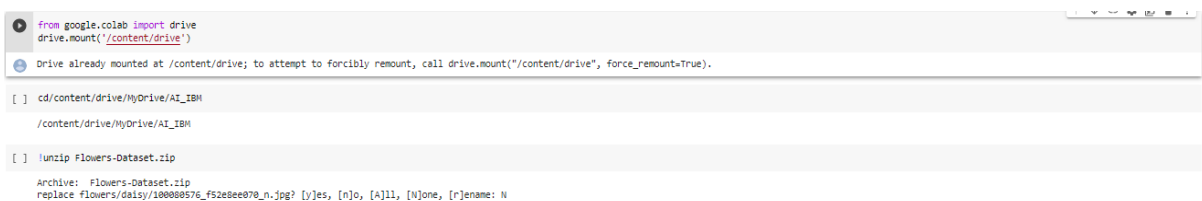
Assignment Date	7 September 2022
Student Name	Ravindran E
Student Roll Number	412519104108
Maximum Marks	2 Marks

Question-1:

Download dataset

Solution:

```
from google.colab import drive
drive.mount('/content/drive')
cd/content/drive/MyDrive/AI_IBM
!unzip Flowers-Dataset.zip
```



```
from google.colab import drive
drive.mount('/content/drive')
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[ ] cd/content/drive/MyDrive/AI_IBM
/content/drive/MyDrive/AI_IBM

[ ] !unzip Flowers-Dataset.zip
Archive:  Flowers-Dataset.zip
replace flowers/daisy/1808080576_fs2e8ee070_n.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename: N
```

Question-2:

Image Augmentation

Solution:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)

test_datagen=ImageDataGenerator(rescale=1./255)

x_train=train_datagen.flow_from_directory(r"/content/drive/MyDrive/AI_IBM/flowers",target_size=(64,64),class_mode='categorical',batch_size=24)

x_train.class_indices
```

```
[ ] from tensorflow.keras.preprocessing.image import ImageDataGenerator

[ ] train_datagen=ImageDataGenerator(rescale=1./255, zoom_range=0.2, horizontal_flip=True, vertical_flip=False)

[ ] test_datagen=ImageDataGenerator(rescale=1./255)

[ ] x_train=train_datagen.flow_from_directory(r"/content/drive/MyDrive/AI_IBM/flowers",target_size=(64,64),class_mode='categorical',batch_size=24)
Found 4317 images belonging to 5 classes.

[ ] x_test=test_datagen.flow_from_directory(r"/content/drive/MyDrive/AI_IBM/flowers",target_size=(64,64),class_mode='categorical',batch_size=24)
Found 4317 images belonging to 5 classes.

[ ] x_train.class_indices
{'daisy': 0, 'dandelion': 1, 'rose': 2, 'sunflower': 3, 'tulip': 4}
```

Question 3

Create Model

Solution:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Convolution2D, MaxPooling2D, Flatten
```

Step -3 Initializing CNN And Create Model

```
[ ] from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Convolution2D, MaxPooling2D, Flatten
```

Question 4

Add Layers

Solution:

```
model=Sequential()
```

4.1 Input Layers (Convolution ,MaxPooling,Flatten)

```
model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation='relu'))
```

```
model.add(MaxPooling2D(pool_size=(2,2)))
```

```
model.add(Flatten())
```

```
model.summary()
```

Step -4 Add layers

```
[ ] model=Sequential()
```

4.1 Input Layers (Convolution ,MaxPooling,Flatten)

```
[ ] model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation='relu'))
```

```
[ ] model.add(MaxPooling2D(pool_size=(2,2)))
```

```
[ ] model.add(Flatten())
```

```
[ ] model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
flatten (Flatten)	(None, 30752)	0
Total params: 896		
Trainable params: 896		
Non-trainable params: 0		

4.2 Hidden Layers

```
model.add(Dense(300,activation='relu'))
```

```
model.add(Dense(150,activation='relu'))
```

Step -7 Test The model

```
ls
flowers/ Flowers_classification_model1.hs Flowers-Dataset.zip

[ ] import numpy as np
    from tensorflow.keras.models import load_model
    from tensorflow.keras.preprocessing import image

[ ] # Load the model
    model=load_model('Flowers_classification_model1.hs')

[ ] img=image.load_img(r"/content/drive/MyDrive/AI_IBH/flowers/s3.jpg",target_size=(64,64))
    x=image.img_to_array(img)
    x=np.expand_dims(x,axis=0)
    y=np.argmax(model.predict(x),axis=1)
    # x_train.class_indices
    index=['daisy','dandelion','rose','sunflower','tulip']
    index[y[0]]

'daisy'
```

** percent of accuracy with this model** Team ID : PNT2022TMID03893

4.3 Output Layers

```
model.add(Dense(5,activation='softmax'))
```

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

```
len(x_train)
```

4.3 Output Layers

```
[ ] model.add(Dense(5,activation='softmax'))

[ ] model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

[ ] len(x_train)

190
```

Question 5

Train the Model

```
model.fit_generator(x_train,steps_per_epoch=len(x_train), validation_data=x_test,
validation_steps=len(x_test), epochs= 30)
```

Step-5 Train the Model

```

1 model.fit_generator(x_train,steps_per_epoch=len(x_train), validation_data=x_test, validation_steps=len(x_test), epochs= 30)
2
3 /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: 'Model.fit_generator' is deprecated and will be removed in a future version. Please use 'Model.fit', which supports generators.
4 """Entry point for launching an IPython kernel.
5
6 Epoch 1/30
7 180/180 [=====] - 539s 3s/step - loss: 1.2262 - accuracy: 0.5088 - val_loss: 1.3867 - val_accuracy: 0.4959
8 Epoch 2/30
9 180/180 [=====] - 65s 360ms/step - loss: 0.9982 - accuracy: 0.6092 - val_loss: 0.9071 - val_accuracy: 0.6518
10 Epoch 3/30
11 180/180 [=====] - 63s 350ms/step - loss: 0.9337 - accuracy: 0.6382 - val_loss: 0.8871 - val_accuracy: 0.6574
12 Epoch 4/30
13 180/180 [=====] - 64s 357ms/step - loss: 0.8495 - accuracy: 0.6778 - val_loss: 1.0011 - val_accuracy: 0.6296
14 Epoch 5/30
15 180/180 [=====] - 63s 350ms/step - loss: 0.8044 - accuracy: 0.6912 - val_loss: 0.7725 - val_accuracy: 0.7014
16 Epoch 6/30
17 180/180 [=====] - 63s 349ms/step - loss: 0.7469 - accuracy: 0.7174 - val_loss: 0.7189 - val_accuracy: 0.7320
18 Epoch 7/30
19 180/180 [=====] - 65s 359ms/step - loss: 0.6911 - accuracy: 0.7403 - val_loss: 0.6169 - val_accuracy: 0.7739
20 Epoch 8/30
21 180/180 [=====] - 63s 353ms/step - loss: 0.6428 - accuracy: 0.7545 - val_loss: 0.6712 - val_accuracy: 0.7424
22 Epoch 9/30
23 180/180 [=====] - 66s 365ms/step - loss: 0.6243 - accuracy: 0.7623 - val_loss: 0.5460 - val_accuracy: 0.8010
24 Epoch 10/30
25 180/180 [=====] - 64s 356ms/step - loss: 0.5638 - accuracy: 0.7885 - val_loss: 0.5537 - val_accuracy: 0.7820
26 Epoch 11/30
27 180/180 [=====] - 63s 350ms/step - loss: 0.5468 - accuracy: 0.7948 - val_loss: 0.4838 - val_accuracy: 0.8149
28 Epoch 12/30
29 180/180 [=====] - 64s 355ms/step - loss: 0.4855 - accuracy: 0.8175 - val_loss: 0.4960 - val_accuracy: 0.8151
30 Epoch 13/30
31 180/180 [=====] - 63s 350ms/step - loss: 0.4574 - accuracy: 0.8365 - val_loss: 0.3337 - val_accuracy: 0.8798
32 Epoch 14/30
33 180/180 [=====] - 65s 364ms/step - loss: 0.4429 - accuracy: 0.8372 - val_loss: 0.3569 - val_accuracy: 0.8694
34 Epoch 15/30
35 180/180 [=====] - 63s 352ms/step - loss: 0.4245 - accuracy: 0.8398 - val_loss: 0.4004 - val_accuracy: 0.8487

```

Activate Windows
Go to PC settings to activate Windows.

Question 6

Save The model

```
model.save('Flowers_classification_model1.h5')
```

Step-6 Save The model

```
[ ] model.save('Flowers_classification_model1.h5')
```

Question 7

Test The model

```

1 Is
2
3 import numpy as np
4
5 from tensorflow.keras.models import load_model
6
7 from tensorflow.keras.preprocessing import image
8
9 model=load_model('Flowers_classification_model1.h5')
10
11 img=image.load_img(r"/content/drive/MyDrive/AI_IBM/flowers/s3.jpg",target_size=(64
12 ,64))
13
14 x=image.img_to_array(img)
15
16 x=np.expand_dims(x,axis=0)
17
18 y=np.argmax(model.predict(x),axis=1)
19
20 index=['daisy','dandelion','rose','sunflower','tulip']
21
22 index[y[0]]

```

Step -7 Test The model

```
ls
flowers/  Flowers_classification_model1.hs  Flowers-Dataset.zip

[ ] import numpy as np
    from tensorflow.keras.models import load_model
    from tensorflow.keras.preprocessing import image

[ ] # Load the model
    model=load_model('Flowers_classification_model1.hs')

[ ] img=image.load_img(r"content/drive/MyDrive/AI_IBM/flowers/s3.jpg",target_size=(64,64))
    x=image.img_to_array(img)
    x=np.expand_dims(x,axis=0)
    y=np.argmax(model.predict(x),axis=1)
    # X_train.class_indices
    index=['daisy','dandelion','rose','sunflower','tulip']
    index[y[0]]

'daisy'
```

** percent of accuracy with this model** Team ID : PNT2022TMID03893