Assignment -3

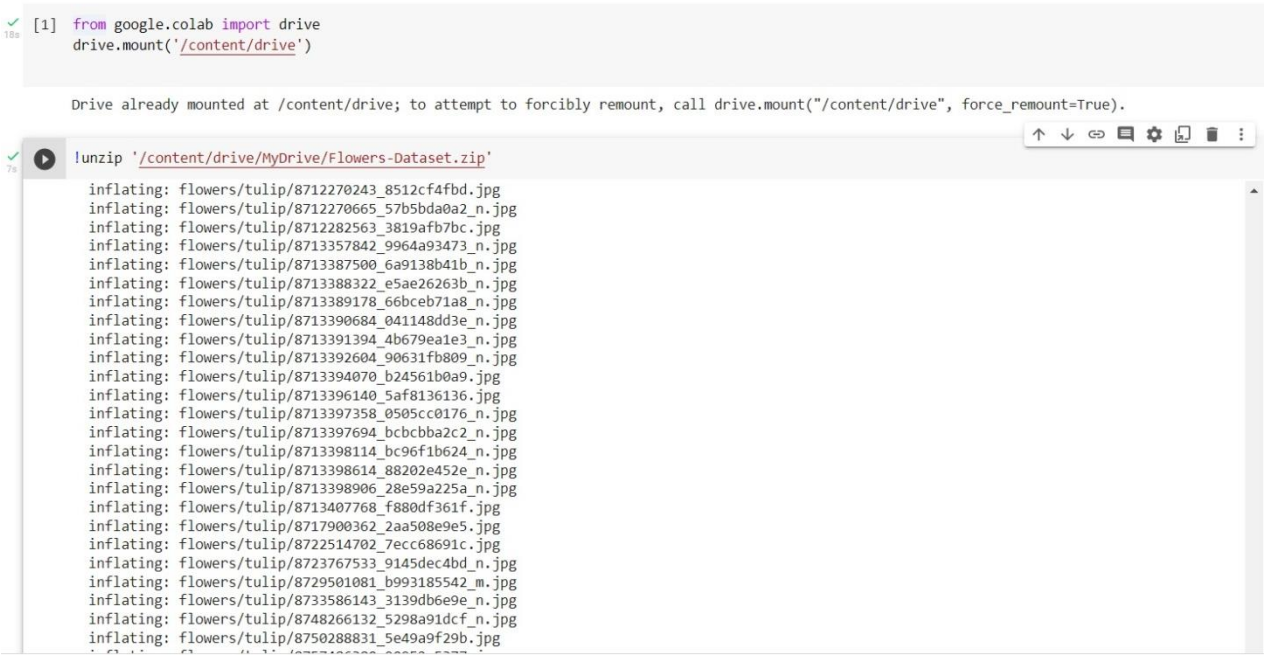
Python Programming

Assignment Date	6 October 2022
Student Name	T.Valarmathi
Student Roll Number	912419104035
Maximum Marks	2 Marks

Load dataset from drive

Solution:

```
from google.colab import drive
drive.mount('/content/drive')
!unzip '/content/drive/MyDrive/FlowersDataset.zip'
```



Question-1:

Image Augmentation

Solution:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_data = ImageDataGenerator(rescale= 1./255,
                                horizontal_flip =True,
                                vertical_flip = True,
                                zoom_range = 0.2)

test_data = ImageDataGenerator(rescale= 1./255)
```

```
Flower_train = train_data.flow_from_directory('/content/flowers',
                                              target_size = (64,64),
                                              class_mode ="categorical",
                                              batch_size = 28)
```

```
Flower_test = test_data.flow_from_directory('/content/flowers',
                                            target_size = (64,64),
                                            class_mode = "categorical",
                                            batch_size = 28)
```

Image Augmentation

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

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

    test_data = ImageDataGenerator(rescale= 1./255)
```

```
[4] Flower_train = train_data.flow_from_directory('/content/flowers',
                                                  target_size = (64,64),
                                                  class_mode ="categorical",
                                                  batch_size = 28)

    Found 4317 images belonging to 5 classes.
```

```
    Flower_test = test_data.flow_from_directory('/content/flowers',
                                                target_size = (64,64),
                                                class_mode = "categorical",
                                                batch_size = 28)

    Found 4317 images belonging to 5 classes.
```

Question-2:

Create Model

Solution:

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
model=Sequential()
```

Create Model

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

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

Question-3:

Add Layers (Convolution,MaxPooling,Flatten,Dense-(Hidden Layers),Output)

Solution:

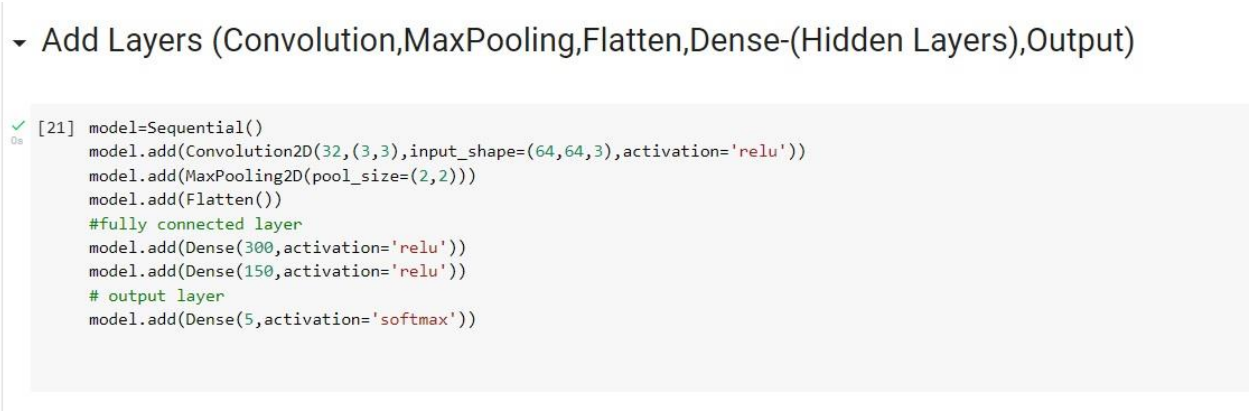
```
model=Sequential()
model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())

#fully connected layer

model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))

# output layer

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



Question-4:

Compile The Model

Solution:

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

Question-5:

Fit The Model

Solution:

```
model.fit(Flower_train,steps_per_epoch=len(Flower_train),validation_data=Flower_test,validation_steps=len(Flower_test),epochs=10)
```

```
model.fit(Flower_train,steps_per_epoch=len(Flower_train),validation_data=Flower_test,
validation_steps=len(Flower_test),epochs=10)

Epoch 1/10
155/155 [==============================] - 53s 340ms/step - loss: 1.3213 - accuracy: 0.4536 - val_loss: 1.3018 - val_accuracy: 0.4693
Epoch 2/10
155/155 [==============================] - 51s 330ms/step - loss: 1.0754 - accuracy: 0.5735 - val_loss: 1.2419 - val_accuracy: 0.5319
Epoch 3/10
155/155 [==============================] - 50s 326ms/step - loss: 0.9799 - accuracy: 0.6097 - val_loss: 1.0259 - val_accuracy: 0.5981
Epoch 4/10
155/155 [==============================] - 50s 325ms/step - loss: 0.9354 - accuracy: 0.6349 - val_loss: 0.9737 - val_accuracy: 0.6254
Epoch 5/10
155/155 [==============================] - 50s 324ms/step - loss: 0.9171 - accuracy: 0.6412 - val_loss: 0.9706 - val_accuracy: 0.6331
Epoch 6/10
155/155 [==============================] - 50s 325ms/step - loss: 0.8665 - accuracy: 0.6646 - val_loss: 0.9757 - val_accuracy: 0.6342
Epoch 7/10
155/155 [==============================] - 52s 339ms/step - loss: 0.8256 - accuracy: 0.6773 - val_loss: 0.8668 - val_accuracy: 0.6595
Epoch 8/10
155/155 [==============================] - 50s 325ms/step - loss: 0.8027 - accuracy: 0.6861 - val_loss: 0.7986 - val_accuracy: 0.6912
Epoch 9/10
155/155 [==============================] - 50s 325ms/step - loss: 0.7861 - accuracy: 0.6972 - val_loss: 0.8013 - val_accuracy: 0.6891
Epoch 10/10
155/155 [==============================] - 53s 340ms/step - loss: 0.7483 - accuracy: 0.7070 - val_loss: 0.6732 - val_accuracy: 0.7380
<keras.callbacks.History at 0x7f2721189f50>
```

Question-6:

Save The Model

Solution:

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

```
Save The Model

[11] model.save('Flower.h5')
```

Question-7:

Test The Model

Solution:

```
import numpy as np
from tensorflow.keras.preprocessing import image
Rose = image.load_img('/content/flowers/tulip/10163955604_ae0b830975_n.jpg'

,target_size=(200,210))

print(Rose)
```

Test the model

```
[3] import numpy as np
    from tensorflow.keras.preprocessing import image
```

```
tulip = image.load_img('/content/flowers/tulip/10163955604_ae0b830975_n.jpg',target_size=(200,200))
```

```
[ ] tulip
```

Solution:

```
array = image.img_to_array(tulip)
array
```

```
array = image.img_to_array(tulip)
array
```

```
array([[[ 82., 108., 133.],
        [ 27.,  27.,   0.],
        [  0.,   5.,   0.],
        ...,
        [  1.,   9.,   0.],
        [  6.,   7.,   0.],
        [  7.,   0.,   5.]],

       [[ 67.,  92.,  60.],
        [ 22.,  24.,  19.],
        [ 53.,  40.,  31.],
        ...,
        [ 11.,   0.,   8.],
        [  3.,   0.,   4.],
        [  0.,   5.,   0.]],

       [[ 31.,  29.,  17.],
        [ 47.,  62.,   0.],
        [ 41.,  57.,  18.],
        ...,
        [ 10.,  10.,  12.],
        [  3.,   3.,  13.],
        [  7.,   8.,   0.]],
```

Solution:

```
array = np.expand_dims(array,axis=0)
array
```

```
array = np.expand_dims(array,axis=0)
array
```

```
array([[[[ 82., 108., 133.],
         [ 27.,  27.,   0.],
         [  0.,   5.,   0.],
         ...,
         [  1.,   9.,   0.],
         [  6.,   7.,   0.],
         [  7.,   0.,   5.]],

        [[ 67.,  92.,  60.],
         [ 22.,  24.,  19.],
         [ 53.,  40.,  31.],
         ...,
         [ 11.,   0.,   8.],
         [  3.,   0.,   4.],
         [  0.,   5.,   0.]],

        [[ 31.,  29.,  17.],
         [ 47.,  62.,   0.],
         [ 41.,  57.,  18.],
         ...,
         [ 10.,  10.,  12.],
         [  3.,   3.,  13.],
         [  7.,   8.,   0.]],

        ...,

        [[ 30.,  43.,  60.],
```

Solution:

```
Flower_train.class_indices

op = ['daisy','dandelion','rose','sunflower','tulip']
pred = np.argmax(model.predict(x))
op[pred]

dandelion = image.load_img('/content/flowers/dandelion/10043234166_e6dd
915111_n.jpg',target_size=(64,64))
x = image.img_to_array(dandelion)
x = np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x))
op[pred]
```

```
[20] op = ['daisy','dandelion','rose','sunflower','tulip']
     pred = np.argmax(model.predict(x))
     op[pred]
```

```
1/1 [==============================] - 0s 28ms/step
'daisy'
```

```
dandelion = image.load_img('/content/flowers/dandelion/10043234166_e6dd915111_n.jpg',target_size=(64,64))
x = image.img_to_array(dandelion)
x = np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x))
op[pred]
```

```
1/1 [==============================] - 0s 112ms/step
'daisy'
```