**ASSIGNMENT-3**

from tensorflow.keras.preprocessing.image import ImageDataGenerator

train\_datagen = ImageDataGenerator(rescale= 1./255,horizontal\_flip = True,vertical\_flip = True,zoom\_range = 0.2) test\_datagen = ImageDataGenerator(rescale= 1./255)

x\_train =

train\_datagen.flow\_from\_directory(r"C:\Users\LonelyDinesh\

Desktop\data\_for\_ibm\Flowers-Dataset\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"C:\Users\LonelyDinesh\

Desktop\data\_for\_ibm\Flowers-Dataset\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}

from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Dense from tensorflow.keras.layers import

Convolution2D,MaxPooling2D,Flatten 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()) 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



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

model.add(Dense(5,activation='softmax')) len(x\_train) 180

model.compile(loss='categorical\_crossentropy',optimizer='adam',metric s

=['accuracy'])

model.fit(x\_train,steps\_per\_epoch=len(x\_train),validation\_data=x\_test , validation\_steps=len(x\_test),epochs=10)

Epoch 1/10

180/180 [==============================] - 33s 183ms/step - loss:

1.3003 - accuracy: 0.4691 - val\_loss: 1.1679 - val\_accuracy: 0.5342

Epoch 2/10

180/180 [==============================] - 28s 157ms/step - loss:

1.0616 - accuracy: 0.5812 - val\_loss: 1.0829 - val\_accuracy: 0.5800

Epoch 3/10

180/180 [==============================] - 28s 157ms/step - loss:

0.9799 - accuracy: 0.6185 - val\_loss: 1.1128 - val\_accuracy: 0.5821

Epoch 4/10

180/180 [==============================] - 29s 161ms/step - loss:

0.9217 - accuracy: 0.6366 - val\_loss: 0.9303 - val\_accuracy: 0.6386

Epoch 5/10

180/180 [==============================] - 28s 158ms/step - loss:

0.8893 - accuracy: 0.6583 - val\_loss: 0.8627 - val\_accuracy: 0.6650

Epoch 6/10

180/180 [==============================] - 29s 162ms/step - loss:

0.8509 - accuracy: 0.6755 - val\_loss: 0.8262 - val\_accuracy: 0.6880

Epoch 7/10

180/180 [==============================] - 30s 169ms/step - loss:

0.8274 - accuracy: 0.6755 - val\_loss: 0.8372 - val\_accuracy: 0.6796

Epoch 8/10

180/180 [==============================] - 30s 166ms/step - loss:

0.7923 - accuracy: 0.6965 - val\_loss: 0.8437 - val\_accuracy: 0.6734

Epoch 9/10

180/180 [==============================] - 28s 157ms/step - loss:

0.7745 - accuracy: 0.7072 - val\_loss: 0.6995 - val\_accuracy: 0.7306

Epoch 10/10

180/180 [==============================] - 28s 158ms/step - loss:

0.7363 - accuracy: 0.7192 - val\_loss: 0.7278 - val\_accuracy: 0.7278

<keras.callbacks.History at

0x16061cf68f0> model.save('IBM\_flowers.h5') pwd

'C:\\Users\\jass\_q3mm6nk\\Desktop\\data\_for\_ibm'

import numpy as np

from tensorflow.keras.models import load\_model from tensorflow.keras.preprocessing import image model=load\_model('IBM\_flowers.h5')

img=image.load\_img(r'C:\Users\maris\_q3mm6nk\Desktop\data\_for\_ib m\ Flowers-Dataset\flowers\rose/394990940\_7af082cf8d\_n.jpg') img



img=image.load\_img(r'C:\Users\maris\_q3mm6nk\Desktop\data\_for\_ib m\ Flowers-Dataset\flowers\rose/

394990940\_7af082cf8d\_n.jpg',target\_size=(64,64)) img



x=image.img\_to\_array(img

) x

|  |  |
| --- | --- |
| array([[[ 4., 14., | 3.], |
| [ 4., 15., | 0.], |
| [ 7., 10., ..., | 3.], |
| [ 1., 1., | 1.], |
| [ 1., 1., | 1.], |
| [ 3., 3., | 3.]], |

[[21., 37., 8.], [ 7., 18., 1.],

[ 5., 11., 1.], ...,

[ 1., 1., 3.], [ 1., 1., 1.],

[ 2., 2., 2.]],

[[15., 34., 4.], [ 5., 18., 0.],

[ 6., 14., 3.], ...,

[ 1., 2., 4.], [ 0., 0., 0.],

[ 1., 1., 1.]],

...,

[[ 7., 11., 10.], [ 7., 16., 15.],

[17., 23., 21.], ...,

[ 1., 1., 1.], [ 2., 2., 2.],

[ 0., 0., 0.]],

[[ 9., 18., 15.], [ 2., 7., 3.],

[ 5., 11., 7.], ...,

[ 0., 0., 0.], [ 1., 1., 1.],

[ 1., 1., 1.]],

[[18., 26., 28.], [ 0., 10., 2.], [ 8., 14., 10.], ...,

[ 2., 6., 9.],

[ 1., 1., 1.],

[ 1., 1., 1.]]], dtype=float32)

x=np.expand\_dims(x,axis=0

) x

|  |  |  |  |
| --- | --- | --- | --- |
| array([[[[ 4., 14., | | 3.], | |
| [ 4., 15., | | 0.], | |
| [ 7., 10., ..., | | 3.], | |
| [ 1., 1., | | 1.], | |
| [ 1., 1., | | 1.], | |
| [ 3., 3., | | 3.]], | |
| [[21., 37., | | 8.], | |
| [ 7., 18., | | 1.], | |
| [ 5., 11., ...,  [ 1., 1., | | 1.],  3.], | |
| [ 1., 1., | | 1.], | |
| [ 2., 2., | | 2.]], | |
| [[15., 34., | | 4.], | |
| [ 5., 18., | | 0.], | |
| [ 6., 14., ...,  [ 1., 2., | | 3.],  4.], | |
| [ 0., 0., | | 0.], | |
| [ 1., 1.,  ..., | | 1.]], | |
| [[ 7., 11., 10.], [ 7., 16., 15.],  [17., 23., 21.], ...,  [ 1., 1., 1.], [ 2., 2., 2.],  [ 0., 0., 0.]],  [[ 9., 18., 15.], [ 2., 7., 3.],  [ 5., 11., 7.], ...,  [ 0., 0., 0.], [ 1., 1., 1.],  [ 1., 1., 1.]],  [[18., 26., 28.], [ 0., 10., 2.], [ 8., 14., 10.], ...,  [ 2., 6., 9.],  [ 1., 1., 1.],  [ 1., 1., 1.]]]], dtype=float32) | | | |

y=np.argmax(model.predict(x),axis=1 ) y

1/1 [==============================] - 0s 74ms/step array([2], dtype=int64) x\_train.class\_indices

{'daisy': 0, 'dandelion': 1, 'rose': 2, 'sunflower': 3, 'tulip': 4} index=['daisy','dandelion','rose','sunflower','tulip'] index[y[0] ] 'rose'