IBM ASSIGNMENT - 3

CNN MODEL

In [1]:

```
from zipfile import ZipFile
```

In [2]:

```
fl = "Flowers-Dataset.zip"
```

In [3]:

```
with ZipFile(fl, 'r') as zip:
    zip.printdir()
```

```
File Name                                             Modified
Size
flowers/daisy/100080576_f52e8ee070_n.jpg      2021-07-16 16:01:08
26797
flowers/daisy/10140303196_b88d3d6cec.jpg      2021-07-16 16:01:08       1
17247
flowers/daisy/10172379554_b296050f82_n.jpg    2021-07-16 16:01:08
36410
flowers/daisy/10172567486_2748826a8b.jpg      2021-07-16 16:01:08       1
02862
flowers/daisy/10172636503_21bededa75_n.jpg    2021-07-16 16:01:08
27419
flowers/daisy/102841525_bd6628ae3c.jpg        2021-07-16 16:01:08       1
32803
flowers/daisy/10300722094_28fa978807_n.jpg    2021-07-16 16:01:08
29941
flowers/daisy/1031799732_e7f4008c03.jpg       2021-07-16 16:01:08       1
02618
flowers/daisy/10391248763_1d16681106_n.jpg    2021-07-16 16:01:08
```

- IMAGE AUGMENTATION

In [4]:

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

In [5]:

```
train_datagen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,horizontal_flip=True,vertica
```

In [6]:

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

In [7]:

```
pip install split-folders
```

```
Requirement already satisfied: split-folders in c:\users\dell\anaconda3\lib
\site-packages (0.5.1)
Note: you may need to restart the kernel to use updated packages.
```

In [8]:

```
import splitfolders
```

In [9]:

```
input_folder='C:Downloads\Flowers'
```

In [10]:

```
splitfolders.ratio(input_folder,output='/content/drive/MyDrive/flowersdataset',ratio=(.8,0,
```

```
Copying files: 3584 files [00:12, 287.93 files/s]
```

In [11]:

```
x_train=train_datagen.flow_from_directory(r"/content/drive/MyDrive/flowersdataset/train",ta
```

```
Found 3452 images belonging to 5 classes.
```

In [12]:

```
x_test=test_datagen.flow_from_directory(r"/content/drive/MyDrive/flowersdataset/test",targe
```

```
Found 865 images belonging to 5 classes.
```

In [13]:

```
x_train.class_indices
print(x_train)
```

```
<keras.preprocessing.image.DirectoryIterator object at 0x00000194B9BDC070>
```

- CREATE MODEL

In [14]:

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

In [15]:

```
model=Sequential()
```

- ADDING LAYERS

1. Adding Convolution Layer

In [16]:

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

2. Adding Max pooling

In [17]:

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

3. Adding Flatten Layer

In [18]:

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

In [19]:

```
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. Hidden Layers

In [20]:

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

5. Output Layer

In [21]:

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

- COMPILE THE MODEL

In [22]:

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

Out[22]:

144

In [23]:

```
print(len(x_train))
```

144

- FIT THE MODEL

In [ ]:

```

```

In [24]:

```
model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation_
```

```
C:\Users\Dell\AppData\Local\Temp/ipykernel_1712/2428895824.py:1: UserWarnin
g: `Model.fit_generator` is deprecated and will be removed in a future versi
on. Please use `Model.fit`, which supports generators.
  model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x
_test,validation_steps=len(x_test),epochs=20)
```

```
Epoch 1/20
144/144 [==============================] - 68s 461ms/step - loss: 1.3001 -
accuracy: 0.4563 - val_loss: 1.1174 - val_accuracy: 0.5468
Epoch 2/20
144/144 [==============================] - 37s 256ms/step - loss: 1.0532 -
accuracy: 0.5794 - val_loss: 1.0999 - val_accuracy: 0.5723
Epoch 3/20
144/144 [==============================] - 38s 265ms/step - loss: 0.9379 -
accuracy: 0.6364 - val_loss: 1.1032 - val_accuracy: 0.5861
Epoch 4/20
144/144 [==============================] - 39s 269ms/step - loss: 0.8892 -
accuracy: 0.6544 - val_loss: 0.9482 - val_accuracy: 0.6451
Epoch 5/20
144/144 [==============================] - 37s 255ms/step - loss: 0.8200 -
accuracy: 0.6834 - val_loss: 0.9185 - val_accuracy: 0.6659
Epoch 6/20
144/144 [==============================] - 39s 267ms/step - loss: 0.7673 -
accuracy: 0.7089 - val_loss: 0.9445 - val_accuracy: 0.6370
Epoch 7/20
144/144 [==============================] - 40s 277ms/step - loss: 0.7264 -
accuracy: 0.7210 - val_loss: 1.0336 - val_accuracy: 0.6289
Epoch 8/20
144/144 [==============================] - 41s 286ms/step - loss: 0.6742 -
accuracy: 0.7410 - val_loss: 0.9891 - val_accuracy: 0.6451
Epoch 9/20
144/144 [==============================] - 39s 271ms/step - loss: 0.6495 -
accuracy: 0.7517 - val_loss: 1.0629 - val_accuracy: 0.6405
Epoch 10/20
144/144 [==============================] - 36s 248ms/step - loss: 0.6026 -
accuracy: 0.7729 - val_loss: 1.0400 - val_accuracy: 0.6462
Epoch 11/20
144/144 [==============================] - 35s 240ms/step - loss: 0.5418 -
accuracy: 0.7894 - val_loss: 1.0212 - val_accuracy: 0.6590
Epoch 12/20
144/144 [==============================] - 41s 287ms/step - loss: 0.5069 -
accuracy: 0.8149 - val_loss: 1.0110 - val_accuracy: 0.6578
Epoch 13/20
144/144 [==============================] - 35s 242ms/step - loss: 0.4764 -
accuracy: 0.8242 - val_loss: 1.2612 - val_accuracy: 0.6405
Epoch 14/20
144/144 [==============================] - 37s 254ms/step - loss: 0.4737 -
accuracy: 0.8244 - val_loss: 1.0247 - val_accuracy: 0.6879
Epoch 15/20
144/144 [==============================] - 35s 246ms/step - loss: 0.4016 -
accuracy: 0.8520 - val_loss: 1.0349 - val_accuracy: 0.6855
Epoch 16/20
144/144 [==============================] - 34s 234ms/step - loss: 0.3920 -
accuracy: 0.8583 - val_loss: 1.1563 - val_accuracy: 0.6844
Epoch 17/20
144/144 [==============================] - 32s 220ms/step - loss: 0.3791 -
accuracy: 0.8583 - val_loss: 1.0277 - val_accuracy: 0.7017
```

```
Epoch 18/20
144/144 [==============================] - 32s 223ms/step - loss: 0.3084 -
accuracy: 0.8824 - val_loss: 1.1081 - val_accuracy: 0.6913
Epoch 19/20
144/144 [==============================] - 37s 258ms/step - loss: 0.2939 -
accuracy: 0.8922 - val_loss: 1.2515 - val_accuracy: 0.6832
Epoch 20/20
144/144 [==============================] - 34s 237ms/step - loss: 0.2829 -
accuracy: 0.8960 - val_loss: 1.1719 - val_accuracy: 0.6902
```

Out[24]:

```
<keras.callbacks.History at 0x194ba0daa30>
```

- SAVE THE MODEL

In [25]:

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

- TEST THE MODEL

In [28]:

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

In [29]:

```
img=image.load_img(r"/content/drive/MyDrive/flowersdataset/test/daisy/3706420943_66f3214862
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','dandellion','rose','sunflower','tulip']
index[y[0]]
```

```
1/1 [==============================] - 0s 148ms/step
```

Out[29]:

```
'daisy'
```

In [ ]:

In [ ]:

In [ ]: