Date: 10/10/2022

Team ID:PNT2022TMID38674.

Title:AI Based Discourse for Banking Industry

Assignment 3

Build CNN Model for Classification of Flowers

1. Download the Dataset

In [41]:
```
pwd
```
Out[41]:
```
'/content/drive/MyDrive'
```

> **Load the Image Dataset**

In [ ]:
```
ls
```
```
drive/  sample_data/
```

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```
```
Mounted at /content/drive
```

> **Un-zip the Folder**

In [ ]:
```
cd /content/drive/MyDrive
```
```
/content/drive/MyDrive
```

In [77]:
```
!unzip Flowers-Dataset.zip
```
```
Archive:  Flowers-Dataset.zip
replace flowers/daisy/100080576_f52e8ee070_n.jpg? [y]es, [n]o, [A]ll, [N]one, [r]enam
e: N
```

In [ ]:
```
pwd
```
Out[ ]:
```
'/content/drive/MyDrive'
```

2. Image Augmentation

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

```
In [ ]: train_datagen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,horizontal_flip=True,ve
```

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

```
In [ ]: pwd
```

```
Out[ ]: '/content/drive/MyDrive'
```

```
In [ ]: x_train=train_datagen.flow_from_directory(r"/content/drive/MyDrive/flowers",target_siz
```

```
Found 4317 images belonging to 5 classes.
```

```
In [ ]: x_test=test_datagen.flow_from_directory(r"//content/drive/MyDrive/flowers",target_size
```

```
Found 4317 images belonging to 5 classes.
```

```
In [ ]: x_train.class_indices
```

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

> **CNN**

3. Create Model

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

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

4. Add Layers(Convolution, MaxPooling, Flatten)

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

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

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

```
In [ ]: 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
_________________________________________________________________
```

In [ ]:

```
32*(3*3*3+1)
```

Out[ ]:

```
896
```

Dense - (Hidden Layers)

In [ ]:

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

Output Layers

In [ ]:

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

5. Compile the model

In [ ]:

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

In [ ]:

```
len(x_train)
```

Out[ ]:

```
180
```

In [ ]:

```
4317/24
```

Out[ ]:

```
179.875
```

6. Fit the Model

In [ ]:

```
model.fit(x_train, epochs = 5, validation_data=x_test, steps_per_epoch=len(x_train), v
Epoch 1/5
180/180 [==============================] - 711s 4s/step - loss: 1.6647 - accuracy: 0.
2201 - val_loss: 1.6395 - val_accuracy: 0.2437
Epoch 2/5
180/180 [==============================] - 65s 362ms/step - loss: 1.6257 - accuracy:
0.2409 - val_loss: 1.6142 - val_accuracy: 0.2437
Epoch 3/5
180/180 [==============================] - 66s 366ms/step - loss: 1.6083 - accuracy:
0.2437 - val_loss: 1.6034 - val_accuracy: 0.2437
Epoch 4/5
180/180 [==============================] - 65s 361ms/step - loss: 1.6015 - accuracy:
0.2437 - val_loss: 1.5998 - val_accuracy: 0.2437
Epoch 5/5
180/180 [==============================] - 65s 360ms/step - loss: 1.5994 - accuracy:
0.2432 - val_loss: 1.5987 - val_accuracy: 0.2437
```

Out[ ]:

```
<keras.callbacks.History at 0x7fb054985e90>
```

7. Save the Model

In [39]:

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

In [40]:

```
ls flowers/
```

```
daisy/  dandelion/  rose/  sunflower/  tulip/
```

8. Test the Model

In [42]:

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

In [43]:

```
#load the model
model=load_model('flowers.h5')
```

In [44]:

```
img=image.load_img(r"/content/drive/MyDrive/flowers/daisy/100080576_f52e8ee070_n.jpg")
```

In [45]:

```
img
```

Out[45]:

In [46]:

```
img=image.load_img(r"/content/drive/MyDrive/flowers/daisy/100080576_f52e8ee070_n.jpg"
img
```

Out[46]:

In [47]:

```
x=image.img_to_array(img)
```

In [48]:

```
x
```

```
Out[48]: array([[[141., 141., 139.],
                 [149., 149., 149.],
                 [152., 152., 154.],
                 ...,
                 [162., 161., 166.],
                 [154., 154., 152.],
                 [153., 153., 153.]],

                [[136., 135., 131.],
                 [146., 145., 143.],
                 [169., 168., 174.],
                 ...,
                 [159., 158., 163.],
                 [155., 155., 153.],
                 [149., 149., 149.]],

                [[125., 125., 117.],
                 [138., 140., 137.],
                 [152., 152., 152.],
                 ...,
                 [156., 156., 156.],
                 [157., 157., 155.],
                 [143., 142., 140.]],

                ...,

                [[ 41.,  44.,  23.],
                 [ 43.,  46.,  25.],
                 [ 49.,  51.,  37.],
                 ...,
                 [128., 124., 121.],
                 [125., 121., 118.],
                 [125., 122., 117.]],

                [[ 43.,  46.,  25.],
                 [ 43.,  46.,  25.],
                 [ 54.,  55.,  37.],
                 ...,
                 [130., 126., 125.],
                 [129., 125., 124.],
                 [127., 123., 122.]],

                [[ 44.,  47.,  26.],
                 [ 45.,  48.,  27.],
                 [ 53.,  55.,  34.],
                 ...,
                 [137., 133., 132.],
                 [133., 129., 128.],
                 [130., 126., 125.]]], dtype=float32)
```

```
In [49]: x=np.expand_dims(x,axis=0)
```

```
In [50]: x
```

Out[50]:

```
array([[[[141., 141., 139.],
         [149., 149., 149.],
         [152., 152., 154.],
         ...,
         [162., 161., 166.],
         [154., 154., 152.],
         [153., 153., 153.]],

        [[136., 135., 131.],
         [146., 145., 143.],
         [169., 168., 174.],
         ...,
         [159., 158., 163.],
         [155., 155., 153.],
         [149., 149., 149.]],

        [[125., 125., 117.],
         [138., 140., 137.],
         [152., 152., 152.],
         ...,
         [156., 156., 156.],
         [157., 157., 155.],
         [143., 142., 140.]],

        ...,

        [[ 41.,  44.,  23.],
         [ 43.,  46.,  25.],
         [ 49.,  51.,  37.],
         ...,
         [128., 124., 121.],
         [125., 121., 118.],
         [125., 122., 117.]],

        [[ 43.,  46.,  25.],
         [ 43.,  46.,  25.],
         [ 54.,  55.,  37.],
         ...,
         [130., 126., 125.],
         [129., 125., 124.],
         [127., 123., 122.]],

        [[ 44.,  47.,  26.],
         [ 45.,  48.,  27.],
         [ 53.,  55.,  34.],
         ...,
         [137., 133., 132.],
         [133., 129., 128.],
         [130., 126., 125.]]]], dtype=float32)
```

In [70]:

```
y=np.argmax(model.predict(x),axis=0)
```

In [52]:

```
y
```

Out[52]:

```
array([1])
```

In [53]:

```
x_train.class_indices
```

Out[53]:

```
{'daisy': 0, 'dandelion': 1, 'rose': 2, 'sunflower': 3, 'tulip': 4}
```

In [54]:

```
index=['daisy','dandelion','rose','sunflower']
```

In [71]:

```
index[y[0]]
```

Out[71]:

```
'daisy'
```

In [61]:

```
img=image.load_img(r"/content/drive/MyDrive/flowers/dandelion/10200780773_c6051a7d71_r
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
y=np.argmax(model.predict(x),axis=1)
index=['daisy','dandelion','rose','sunflower']
index[y[0]]
```

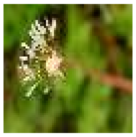
Out[61]:

```
'dandelion'
```

In [57]:

```
img
```

Out[57]:



In [74]:

```
img=image.load_img(r"/content/drive/MyDrive/flowers/rose/10503217854_e66a804309.jpg",
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
y=np.argmax(model.predict(x),axis=1)
index=['daisy','rose','dandelion','sunflower']
index[y[0]]
```

Out[74]:

```
'rose'
```

In [75]:

```
img
```

Out[75]:

In [72]:

```
img=image.load_img(r"/content/drive/MyDrive/flowers/sunflower/10386503264_e05387e1f7_m
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
y=np.argmax(model.predict(x),axis=0)
index=['sunflower','daisy','dandelion','rose']
index[y[0]]
```

Out[72]:

```
'sunflower'
```

In [60]:

```
img
```

Out[60]: