Assignment 3

Build CNN Model for Classification of Flowers - PNT2022TMID00519

1)Download the Dataset and Unzip the file

In [13]:
```
!unzip "/content/Flowers-Dataset.zip"
```

```
Archive:  /content/Flowers-Dataset.zip
  inflating: flowers/daisy/100080576_f52e8ee070_n.jpg
  inflating: flowers/daisy/10140303196_b88d3d6cec.jpg
  inflating: flowers/daisy/10172379554_b296050f82_n.jpg
  inflating: flowers/daisy/10172567486_2748826a8b.jpg
  inflating: flowers/daisy/10172636503_21bededa75_n.jpg
  inflating: flowers/daisy/102841525_bd6628ae3c.jpg
  inflating: flowers/daisy/10300722094_28fa978807_n.jpg
  inflating: flowers/daisy/1031799732_e7f4008c03.jpg
  inflating: flowers/daisy/10391248763_1d16681106_n.jpg
  inflating: flowers/daisy/10437754174_22ec990b77_m.jpg
  inflating: flowers/daisy/10437770546_8bb6f7bdd3_m.jpg
  inflating: flowers/daisy/10437929963_bc13eebe0c.jpg
  inflating: flowers/daisy/10466290366_cc72e33532.jpg
  inflating: flowers/daisy/10466558316_a7198b87e2.jpg
  inflating: flowers/daisy/10555749515_13a12a026e.jpg
  inflating: flowers/daisy/10555815624_dc211569b0.jpg
  inflating: flowers/daisy/10555826524_423eb8bf71_n.jpg
  inflating: flowers/daisy/10559679065_50d2b16f6d.jpg
```

2)Image Augmentation

In [14]:
```
# Import required lib

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

In [15]:
```
# Creating augmentation on training variable
train_datagen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,
                                 horizontal_flip=True,
                                 vertical_flip=False)
```

In [16]:
```
#Creating augmentation on testing variable
test_datagen=ImageDataGenerator(rescale=1./255)
```

In [33]:
```
pip install split-folders #Seprating the Train and Test Data
```

```
Looking in indexes: https://pypi.org/simple, (https://pypi.org/simple,) http
s://us-python.pkg.dev/colab-wheels/public/simple/ (https://us-python.pkg.dev/co
lab-wheels/public/simple/)
Collecting split-folders
  Downloading split_folders-0.5.1-py3-none-any.whl (8.4 kB)
Installing collected packages: split-folders
Successfully installed split-folders-0.5.1
```

In [34]:
```
import splitfolders
input_folder= "/content/flowers"
```

In [37]:
```
splitfolders.ratio(input_folder,output='/content/flowers',
                   ratio=(.8,0,.2),
                   group_prefix=None)
```

```
Copying files: 4317 files [00:01, 3113.78 files/s]
```

In [38]:
```
x_train=train_datagen.flow_from_directory("/content/flowers/test",
                                          target_size=(64,64),
                                          class_mode='categorical',
                                          batch_size=19)
```

```
Found 865 images belonging to 8 classes.
```

In [42]:
```
x_test=test_datagen.flow_from_directory("/content/flowers/train",
                                        target_size=(64,64),
                                        class_mode='categorical',
                                        batch_size=19)
```

```
Found 3452 images belonging to 8 classes.
```

In [45]:
```
x_train.class_indices
```

Out[45]:
```
{'daisy': 0,
 'dandelion': 1,
 'rose': 2,
 'sunflower': 3,
 'test': 4,
 'train': 5,
 'tulip': 6,
 'val': 7}
```

3)Create Model

In [47]:
```
# Importing required lib

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

In [48]:
```
model=Sequential()
```

4)Add Layers (Convolution,MaxPooling,Flatten,Dense-(HiddenLayers),Output)

In [50]:
```
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3))) # Conv
model.add(MaxPooling2D(pool_size=(2,2))) # Max pooling layer
model.add(Flatten()) # Flatten layer
```

In [51]:
```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_6 (Conv2D)           (None, 62, 62, 32)        896

 conv2d_7 (Conv2D)           (None, 60, 60, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 30, 30, 32)       0
 2D)

 flatten_1 (Flatten)         (None, 28800)             0

=================================================================
Total params: 10,144
Trainable params: 10,144
Non-trainable params: 0
_________________________________________________________________
```

In [52]:
```
model.add(Dense(300,activation='relu')) # Hidden layer 1
model.add(Dense(150,activation='relu')) # Hidden layer 2
model.add(Dense(4,activation='softmax')) # Output layer
```

5)Compile The Model

In [81]:
```
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy
```

In [54]:
```
len(x_train)
len(x_test)
```

Out[54]:
```
182
```

In [55]:
```
1238/24
```

Out[55]:
```
51.583333333333336
```

In [56]:
```
326/24
```

Out[56]:
```
13.583333333333334
```

6)Fit The Model

In [ ]:
```
model.fit_generator(x_train,steps_per_epoch=len(x_train),
                    validation_data=x_test,
                    validation_steps=len(x_test),
                    epochs=20)
```

7)Save The Model

In [61]:
```
model.save('Flowers.h5')
```

8)Test The Model

In [62]:
```
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```

In [65]:
```
model.save('flowers.h5')
```

In [88]:
```
img1 = image.load_img('/content/flowers/rose/10090824183_d02c613f10_m.jpg') # Rec
img1 # Visualize the image
```

Out[88]:

In [68]:
```
x=image.img_to_array(img)
x  # Converting image to array
```

Out[68]:
```
array([[[ 6., 15.,  0.],
        [ 6., 15.,  0.],
        [ 8., 17.,  0.],
        ...,
        [31., 51., 24.],
        [32., 52., 25.],
        [33., 53., 26.]],

       [[14., 22.,  7.],
        [14., 22.,  7.],
        [13., 21.,  6.],
        ...,
        [30., 46., 20.],
        [33., 49., 23.],
        [35., 51., 25.]],

       [[15., 23., 12.],
        [15., 23., 12.],
        [14., 22., 11.],
        ...,
        [30., 42., 20.],
        [33., 45., 23.],
        [36., 48., 26.]],

       ...,

       [[27., 30., 19.],
        [18., 24., 14.],
        [13., 20., 12.],
        ...,
        [ 3., 13.,  4.],
        [ 1.,  8.,  0.],
        [ 0.,  5.,  0.]],

       [[28., 30., 19.],
        [24., 27., 18.],
        [16., 23., 15.],
        ...,
        [ 2., 12.,  3.],
        [ 2.,  9.,  1.],
        [ 2.,  7.,  0.]],

       [[19., 19.,  9.],
        [24., 25., 17.],
        [24., 29., 22.],
        ...,
        [ 2., 12.,  1.],
        [ 3., 10.,  2.],
        [ 4., 11.,  3.]]], dtype=float32)
```

```
In [101]: x = np.expand_dims(x,axis=0)
          x # Expanding dimensions

Out[101]: array([[[[[ 32.,  23.,  18.],
                    [ 39.,  28.,  22.],
                    [ 43.,  28.,  21.],
                    ...,
                    [ 31.,  21.,  12.],
                    [ 39.,  25.,  16.],
                    [ 34.,  21.,  13.]],

                   [[ 31.,  21.,  19.],
                    [ 40.,  30.,  21.],
                    [ 48.,  29.,  23.],
                    ...,
                    [ 33.,  20.,  11.],
                    [ 42.,  25.,  17.],
                    [ 35.,  20.,  13.]],

                   [[ 38.,  24.,  21.],
                    [ 42.,  29.,  23.],
                    [ 43.,  28.,  21.],
                    ...,
                    [ 43.,  26.,  16.],
                    [ 48.,  30.,  20.],
                    [ 42.,  23.,  16.]],

                   ...,

                   [[ 53.,  33.,  24.],
                    [ 50.,  24.,  11.],
                    [ 48.,  34.,  21.],
                    ...,
                    [ 70.,  19.,   2.],
                    [ 74.,  22.,   9.],
                    [ 57.,  18.,   3.]],

                   [[ 49.,  30.,  16.],
                    [ 66.,  34.,  19.],
                    [ 76.,  54.,  33.],
                    ...,
                    [ 15.,   0.,   0.],
                    [ 64.,  20.,   7.],
                    [ 52.,  19.,   2.]],

                   [[ 47.,  24.,  16.],
                    [ 52.,  29.,  15.],
                    [ 40.,  14.,   0.],
                    ...,
                    [117.,  65.,  25.],
                    [128.,  53.,  30.],
                    [100.,  33.,  14.]]]]], dtype=float32)
```

In [100]:
```
img=image.load_img("/content/flowers/test/rose/12243069253_e512464095_n.jpg",targ
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]]
```

Out[100]:
```
'rose'
```