Assignment_3

October 7, 2022

Suvetha.M
732119104075

#Unzipping

[23]:
```
#!unzip '/content/drive/MyDrive/Flowers-Dataset.zip'
```

[3]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

#Data Augmentation

[5]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

gen_train = ImageDataGenerator(rescale=1./255,
                               zoom_range=0.2,
                               horizontal_flip=True)
gen_test = ImageDataGenerator(rescale=1./255)
```

[6]:
```
xtrain = gen_train.flow_from_directory('/content/flowers',
                                       target_size=(64,64),
                                       class_mode='categorical',
                                       batch_size=100)
```

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

#Train

[7]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Dense, Flatten
from keras.callbacks import EarlyStopping, ReduceLROnPlateau
```

[8]:
```
model = Sequential()
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
model.add(MaxPooling2D((2,2)))
model.add(Flatten())

model.add(Dense(400,activation='relu'))
```

```
model.add(Dense(200,activation='relu'))
model.add(Dense(100,activation='relu'))
model.add(Dense(5,activation='softmax'))
```

[9]:
```
model.
  ↪compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
```

[10]:
```
early_stopping = EarlyStopping(monitor='accuracy',
                        patience=3)
reduce_lr = ReduceLROnPlateau(monitor='accuracy',
                        patience=5,
                        factor=0.5,min_lr=0.00001)

callback = [reduce_lr,early_stopping]
```

[11]:
```
model.fit_generator(xtrain,
                    steps_per_epoch = len(xtrain),
                    callbacks=callback,
                    epochs=100)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: UserWarning:
`Model.fit_generator` is deprecated and will be removed in a future version.
Please use `Model.fit`, which supports generators.
  after removing the cwd from sys.path.

Epoch 1/100
44/44 [==============================] - 28s 607ms/step - loss: 1.6189 -
accuracy: 0.3595 - lr: 0.0010
Epoch 2/100
44/44 [==============================] - 30s 685ms/step - loss: 1.1551 -
accuracy: 0.5210 - lr: 0.0010
Epoch 3/100
44/44 [==============================] - 27s 614ms/step - loss: 1.0560 -
accuracy: 0.5793 - lr: 0.0010
Epoch 4/100
44/44 [==============================] - 27s 607ms/step - loss: 0.9934 -
accuracy: 0.6164 - lr: 0.0010
Epoch 5/100
44/44 [==============================] - 27s 600ms/step - loss: 0.9360 -
accuracy: 0.6328 - lr: 0.0010
Epoch 6/100
44/44 [==============================] - 26s 593ms/step - loss: 0.8735 -
accuracy: 0.6648 - lr: 0.0010
Epoch 7/100
44/44 [==============================] - 28s 620ms/step - loss: 0.8501 -
accuracy: 0.6752 - lr: 0.0010
Epoch 8/100
44/44 [==============================] - 27s 602ms/step - loss: 0.8115 -
```

```
accuracy: 0.6924 - lr: 0.0010
Epoch 9/100
44/44 [==============================] - 26s 610ms/step - loss: 0.7763 -
accuracy: 0.7072 - lr: 0.0010
Epoch 10/100
44/44 [==============================] - 27s 598ms/step - loss: 0.7650 -
accuracy: 0.7111 - lr: 0.0010
Epoch 11/100
44/44 [==============================] - 28s 642ms/step - loss: 0.7170 -
accuracy: 0.7239 - lr: 0.0010
Epoch 12/100
44/44 [==============================] - 27s 601ms/step - loss: 0.6939 -
accuracy: 0.7399 - lr: 0.0010
Epoch 13/100
44/44 [==============================] - 27s 597ms/step - loss: 0.6370 -
accuracy: 0.7672 - lr: 0.0010
Epoch 14/100
44/44 [==============================] - 27s 599ms/step - loss: 0.6580 -
accuracy: 0.7526 - lr: 0.0010
Epoch 15/100
44/44 [==============================] - 27s 599ms/step - loss: 0.6004 -
accuracy: 0.7677 - lr: 0.0010
Epoch 16/100
44/44 [==============================] - 28s 643ms/step - loss: 0.5704 -
accuracy: 0.7846 - lr: 0.0010
Epoch 17/100
44/44 [==============================] - 27s 607ms/step - loss: 0.5222 -
accuracy: 0.8094 - lr: 0.0010
Epoch 18/100
44/44 [==============================] - 27s 601ms/step - loss: 0.5483 -
accuracy: 0.8001 - lr: 0.0010
Epoch 19/100
44/44 [==============================] - 27s 600ms/step - loss: 0.5111 -
accuracy: 0.8184 - lr: 0.0010
Epoch 20/100
44/44 [==============================] - 29s 644ms/step - loss: 0.4638 -
accuracy: 0.8304 - lr: 0.0010
Epoch 21/100
44/44 [==============================] - 27s 597ms/step - loss: 0.4379 -
accuracy: 0.8473 - lr: 0.0010
Epoch 22/100
44/44 [==============================] - 26s 596ms/step - loss: 0.4238 -
accuracy: 0.8446 - lr: 0.0010
Epoch 23/100
44/44 [==============================] - 27s 598ms/step - loss: 0.4012 -
accuracy: 0.8559 - lr: 0.0010
Epoch 24/100
44/44 [==============================] - 27s 599ms/step - loss: 0.3751 -
```

```
accuracy: 0.8687 - lr: 0.0010
Epoch 25/100
44/44 [==============================] - 27s 596ms/step - loss: 0.3922 -
accuracy: 0.8594 - lr: 0.0010
Epoch 26/100
44/44 [==============================] - 26s 597ms/step - loss: 0.3392 -
accuracy: 0.8816 - lr: 0.0010
Epoch 27/100
44/44 [==============================] - 27s 598ms/step - loss: 0.3358 -
accuracy: 0.8802 - lr: 0.0010
Epoch 28/100
44/44 [==============================] - 27s 599ms/step - loss: 0.3087 -
accuracy: 0.8930 - lr: 0.0010
Epoch 29/100
44/44 [==============================] - 27s 601ms/step - loss: 0.2769 -
accuracy: 0.9064 - lr: 0.0010
Epoch 30/100
44/44 [==============================] - 27s 597ms/step - loss: 0.2946 -
accuracy: 0.8937 - lr: 0.0010
Epoch 31/100
44/44 [==============================] - 27s 595ms/step - loss: 0.2377 -
accuracy: 0.9192 - lr: 0.0010
Epoch 32/100
44/44 [==============================] - 27s 598ms/step - loss: 0.2304 -
accuracy: 0.9194 - lr: 0.0010
Epoch 33/100
44/44 [==============================] - 27s 596ms/step - loss: 0.2121 -
accuracy: 0.9243 - lr: 0.0010
Epoch 34/100
44/44 [==============================] - 28s 646ms/step - loss: 0.2368 -
accuracy: 0.9175 - lr: 0.0010
Epoch 35/100
44/44 [==============================] - 26s 597ms/step - loss: 0.2262 -
accuracy: 0.9266 - lr: 0.0010
Epoch 36/100
44/44 [==============================] - 26s 597ms/step - loss: 0.2251 -
accuracy: 0.9236 - lr: 0.0010
Epoch 37/100
44/44 [==============================] - 27s 599ms/step - loss: 0.2059 -
accuracy: 0.9300 - lr: 0.0010
Epoch 38/100
44/44 [==============================] - 27s 603ms/step - loss: 0.2164 -
accuracy: 0.9238 - lr: 0.0010
Epoch 39/100
44/44 [==============================] - 28s 620ms/step - loss: 0.1694 -
accuracy: 0.9407 - lr: 0.0010
Epoch 40/100
44/44 [==============================] - 27s 597ms/step - loss: 0.2033 -
```

```
accuracy: 0.9300 - lr: 0.0010
Epoch 41/100
44/44 [==============================] - 27s 596ms/step - loss: 0.1562 -
accuracy: 0.9486 - lr: 0.0010
Epoch 42/100
44/44 [==============================] - 27s 596ms/step - loss: 0.1399 -
accuracy: 0.9534 - lr: 0.0010
Epoch 43/100
44/44 [==============================] - 26s 594ms/step - loss: 0.1536 -
accuracy: 0.9488 - lr: 0.0010
Epoch 44/100
44/44 [==============================] - 26s 595ms/step - loss: 0.1296 -
accuracy: 0.9569 - lr: 0.0010
Epoch 45/100
44/44 [==============================] - 26s 595ms/step - loss: 0.1556 -
accuracy: 0.9465 - lr: 0.0010
Epoch 46/100
44/44 [==============================] - 26s 597ms/step - loss: 0.1449 -
accuracy: 0.9518 - lr: 0.0010
Epoch 47/100
44/44 [==============================] - 29s 659ms/step - loss: 0.1180 -
accuracy: 0.9613 - lr: 0.0010
Epoch 48/100
44/44 [==============================] - 27s 605ms/step - loss: 0.1233 -
accuracy: 0.9592 - lr: 0.0010
Epoch 49/100
44/44 [==============================] - 27s 596ms/step - loss: 0.1386 -
accuracy: 0.9525 - lr: 0.0010
Epoch 50/100
44/44 [==============================] - 26s 596ms/step - loss: 0.1039 -
accuracy: 0.9685 - lr: 0.0010
Epoch 51/100
44/44 [==============================] - 26s 594ms/step - loss: 0.1281 -
accuracy: 0.9576 - lr: 0.0010
Epoch 52/100
44/44 [==============================] - 28s 639ms/step - loss: 0.1345 -
accuracy: 0.9553 - lr: 0.0010
Epoch 53/100
44/44 [==============================] - 27s 598ms/step - loss: 0.1024 -
accuracy: 0.9673 - lr: 0.0010
```

[11]:
```
<keras.callbacks.History at 0x7f09e3218cd0>
```

[12]:
```
model.save('flower_cnn.h5')
```

#Test

[13]:
```
import numpy as np
from tensorflow.keras.preprocessing import image
```

[14]:
```
tulip = image.load_img('/content/download.jpg',target_size=(64,64))
tulip
```

[14]:

[15]:
```
s = image.img_to_array(tulip)
s
```

[15]:
```
array([[[255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        …,
        [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.]],

       [[255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        …,
        [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.]],

       [[255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        …,
        [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.]],

       …,

       [[255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        …,
```

```
        [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.]],

       [[255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        …,
        [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.]],

       [[255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        …,
        [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.]]], dtype=float32)
```

[16]:
```
s= np.expand_dims(s,axis= 0)
s
```

[16]:
```
array([[[[255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.],
         …,
         [255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.]],

        [[255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.],
         …,
         [255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.]],

        [[255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.],
         …,
         [255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.]],
```

```
       …,

       [[255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        …,
        [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.]],

       [[255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        …,
        [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.]],

       [[255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        …,
        [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.]]], dtype=float32)
```

[17]:
```
p = list(xtrain.class_indices.keys())
p
```

[17]:
```
['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']
```

[20]:
```
p[np.argmax(model.predict(s))]
```

[20]:
```
'tulip'
```