```
!unzip '/content/drive/MyDrive/Flowers-Dataset.zip'
```

```
  inflating: flowers/tulip/16702117379_c25bff70e9.jpg
  inflating: flowers/tulip/16702188449_3dacce90b2_m.jpg
  inflating: flowers/tulip/16711791713_e54bc9c1af_n.jpg
  inflating: flowers/tulip/16717320956_d4b00807f2.jpg
  inflating: flowers/tulip/16732302779_8aa56f255d_n.jpg
  inflating: flowers/tulip/16751015081_af2ef77c9a_n.jpg
  inflating: flowers/tulip/16754984282_3a801bfa50_n.jpg
  inflating: flowers/tulip/16755061382_f6531150c1_n.jpg
  inflating: flowers/tulip/16755090222_d6797d51db_n.jpg
  inflating: flowers/tulip/16756198195_13b54e70b3_n.jpg
  inflating: flowers/tulip/16756239775_75e84a3d50_n.jpg
  inflating: flowers/tulip/16765283686_0315ae00a8.jpg
  inflating: flowers/tulip/16862349256_0a1f91ab53.jpg
  inflating: flowers/tulip/16862351376_f0fcc6fc91_n.jpg
  inflating: flowers/tulip/16862374316_4135908d4c_m.jpg
  inflating: flowers/tulip/16862422576_5226e8d1d0.jpg
  inflating: flowers/tulip/16904202259_8f45d045c3_m.jpg
  inflating: flowers/tulip/16907559551_05ded87fb2_n.jpg
  inflating: flowers/tulip/16930105456_8b826dc4a8_n.jpg
  inflating: flowers/tulip/16930121391_a4092ecf00_n.jpg
  inflating: flowers/tulip/16937554595_3e1de22f9c.jpg
  inflating: flowers/tulip/16938892686_3613ea68e8_n.jpg
  inflating: flowers/tulip/16951623209_00fb7ec1b1_n.jpg
  inflating: flowers/tulip/16986144192_55e0e6c152.jpg
  inflating: flowers/tulip/17012955700_7141d29eee.jpg
  inflating: flowers/tulip/17066862602_7530f21efe.jpg
  inflating: flowers/tulip/17066864992_1cbc4fc908.jpg
  inflating: flowers/tulip/17078576150_6f272ce73f_n.jpg
  inflating: flowers/tulip/17078716890_68e0723389_n.jpg
  inflating: flowers/tulip/17094167287_865840060d_n.jpg
  inflating: flowers/tulip/17104364030_ee31ee279b_n.jpg
  inflating: flowers/tulip/17105644339_e301a8a6be_n.jpg
  inflating: flowers/tulip/17113203493_735185295f_n.jpg
  inflating: flowers/tulip/17146928665_600fa3a1f1_n.jpg
  inflating: flowers/tulip/17159349572_c0c51599f7_n.jpg
  inflating: flowers/tulip/17159816388_deafbebdb0.jpg
  inflating: flowers/tulip/17165583356_38cb1f231d_n.jpg
  inflating: flowers/tulip/17167151059_a53bfe0b02.jpg
  inflating: flowers/tulip/17189456156_6fc1067831.jpg
  inflating: flowers/tulip/17189526216_fa24dd541a_n.jpg
  inflating: flowers/tulip/17198868382_697b23c715_n.jpg
  inflating: flowers/tulip/17202535346_ab828e779b.jpg
  inflating: flowers/tulip/17224410762_402455ed8f.jpg
  inflating: flowers/tulip/17282288501_e8738c9cfb_n.jpg
  inflating: flowers/tulip/17291451621_0e39f08b9e_n.jpg
  inflating: flowers/tulip/17291908295_dc7d45ae9b_n.jpg
  inflating: flowers/tulip/17291930305_79deae6a90_n.jpg
  inflating: flowers/tulip/17295127995_62eff434fe_n.jpg
  inflating: flowers/tulip/17309017866_b6b290b970_n.jpg
  inflating: flowers/tulip/17318339476_54479b6660_n.jpg
  inflating: flowers/tulip/17324469461_2b318aff8d_m.jpg
  inflating: flowers/tulip/17408197905_829c4d7940_m.jpg
  inflating: flowers/tulip/17469578564_35a8360f58.jpg
  inflating: flowers/tulip/17545306218_5bf0ee38cd_n.jpg
  inflating: flowers/tulip/175686816_067a8cb4c5.jpg
  inflating: flowers/tulip/176458518_f81d4bff8e.jpg
  inflating: flowers/tulip/17706953166_cc21734d26_n.jpg
```

```
inflating: flowers/tulip/17719248689_cfd5d2f228_n.jpg
inflating: flowers/tulip/17720403638_94cfcd8d5c_n.jpg
```

```
#Data Augmentation

from tensorflow.keras.preprocessing.image import ImageDataGenerator


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


test_datagen = ImageDataGenerator(rescale=1./255)


xtrain = train_datagen.flow_from_directory('/content/flowers',
                                           target_size=(64,64),
                                           class_mode='categorical',
                                           batch_size=100)
```

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

```
xtest = test_datagen.flow_from_directory('/content/flowers',
                                         target_size=(64,64),
                                         class_mode='categorical',
                                         batch_size=100)
```

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

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


model = Sequential() # Initializing sequential model
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3))) # convolution l
model.add(MaxPooling2D(pool_size=(2, 2))) # Max pooling layer
model.add(Flatten()) # Flatten layer
model.add(Dense(300,activation='relu')) # Hidden layer 1
model.add(Dense(150,activation='relu')) # Hidden layer 2
model.add(Dense(5,activation='softmax')) # Output layer


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


model.fit_generator(xtrain,
                    steps_per_epoch=len(xtrain),
                    epochs=10,
                    validation_data=xtest,
                    validation_steps=len(xtest))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: UserWarning: `Model.f
  """
Epoch 1/10
44/44 [==============================] - 22s 501ms/step - loss: 1.0921 - accuracy: 0
```

```
Epoch 2/10
44/44 [==============================] - 21s 488ms/step - loss: 1.0125 - accuracy: 0
Epoch 3/10
44/44 [==============================] - 21s 492ms/step - loss: 0.9534 - accuracy: 0
Epoch 4/10
44/44 [==============================] - 21s 489ms/step - loss: 0.8870 - accuracy: 0
Epoch 5/10
44/44 [==============================] - 21s 485ms/step - loss: 0.8340 - accuracy: 0
Epoch 6/10
44/44 [==============================] - 21s 488ms/step - loss: 0.8059 - accuracy: 0
Epoch 7/10
44/44 [==============================] - 21s 492ms/step - loss: 0.7731 - accuracy: 0
Epoch 8/10
44/44 [==============================] - 21s 487ms/step - loss: 0.7264 - accuracy: 0
Epoch 9/10
44/44 [==============================] - 21s 486ms/step - loss: 0.6962 - accuracy: 0
Epoch 10/10
44/44 [==============================] - 23s 521ms/step - loss: 0.6635 - accuracy: 0
<keras.callbacks.History at 0x7efef0382790>
```

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

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

```
img = image.load_img('/content/flowers/sunflower/1022552002_2b93faf9e7_n.jpg',target_size=
x = image.img_to_array(img) # Converting image into array
x = np.expand_dims(x,axis=0) # expanding Dimensions
pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
op = ['daisy','dandelion','rose','sunflower','tulip'] # Creating list
op[pred] # List indexing with output
```

```
'sunflower'
```

```
img = image.load_img('/content/flowers/daisy/10391248763_1d16681106_n.jpg',target_size=(64
x = image.img_to_array(img) # Converting image into array
x = np.expand_dims(x,axis=0) # expanding Dimensions
pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
op = ['daisy','dandelion','rose','sunflower','tulip'] # Creating list
op[pred] # List indexing with output
```

```
'tulip'
```

```
xtrain.class_indices
```

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

```
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
```

```
early_stop = EarlyStopping(monitor='val_accuracy',
                           patience=5)
```

```
lr = ReduceLROnPlateau(monitor='val_accuaracy',
                       factor=0.5,
                       min_lr=0.00001)

callback = [early_stop,lr]


model.fit_generator(xtrain,
                    steps_per_epoch=len(xtrain),
                    epochs=100,
                    callbacks=callback,
                    validation_data=xtest,
                    validation_steps=len(xtest))
```

```
[==============================] - 22s 516ms/step - loss: 0.1127 - accuracy: 0.960
7/100
[==============================] - ETA: 0s - loss: 0.1473 - accuracy: 0.9527WARNIN
[==============================] - 21s 489ms/step - loss: 0.1473 - accuracy: 0.952
8/100
[==============================] - ETA: 0s - loss: 0.1246 - accuracy: 0.9604WARNIN
[==============================] - 21s 492ms/step - loss: 0.1246 - accuracy: 0.960
9/100
[==============================] - ETA: 0s - loss: 0.1305 - accuracy: 0.9571WARNIN
[==============================] - 22s 494ms/step - loss: 0.1305 - accuracy: 0.957
10/100
[==============================] - ETA: 0s - loss: 0.1020 - accuracy: 0.9641WARNIN
[==============================] - 22s 496ms/step - loss: 0.1020 - accuracy: 0.964
11/100
[==============================] - ETA: 0s - loss: 0.1144 - accuracy: 0.9632WARNIN
[==============================] - 21s 486ms/step - loss: 0.1144 - accuracy: 0.963
12/100
[==============================>.] - ETA: 0s - loss: 0.1151 - accuracy: 0.9642WARNIN
[==============================] - 21s 490ms/step - loss: 0.1152 - accuracy: 0.964
13/100
[==============================] - ETA: 0s - loss: 0.1265 - accuracy: 0.9583WARNIN
[==============================] - 21s 489ms/step - loss: 0.1265 - accuracy: 0.958
14/100
[==============================] - ETA: 0s - loss: 0.1079 - accuracy: 0.9671WARNIN

[==============================] - 21s 490ms/step - loss: 0.1079 - accuracy: 0.967
15/100
[==============================] - ETA: 0s - loss: 0.1282 - accuracy: 0.9569WARNIN
[==============================] - 23s 523ms/step - loss: 0.1282 - accuracy: 0.956
16/100
[==============================] - ETA: 0s - loss: 0.1027 - accuracy: 0.9683WARNIN
[==============================] - 21s 492ms/step - loss: 0.1027 - accuracy: 0.968
17/100
[==============================] - ETA: 0s - loss: 0.1047 - accuracy: 0.9673WARNIN
[==============================] - 21s 491ms/step - loss: 0.1047 - accuracy: 0.967
18/100
[==============================] - ETA: 0s - loss: 0.0935 - accuracy: 0.9685WARNIN
[==============================] - 21s 492ms/step - loss: 0.0935 - accuracy: 0.968
19/100
[==============================] - ETA: 0s - loss: 0.0881 - accuracy: 0.9706WARNIN
[==============================] - 21s 491ms/step - loss: 0.0881 - accuracy: 0.970
20/100
[==============================] - ETA: 0s - loss: 0.1094 - accuracy: 0.9618WARNIN
[==============================] - 21s 490ms/step - loss: 0.1094 - accuracy: 0.961
21/100
```

```
21/100
[==============================] - ETA: 0s - loss: 0.0951 - accuracy: 0.9708WARNIN
[==============================] - 21s 487ms/step - loss: 0.0951 - accuracy: 0.970
22/100
[==============================] - ETA: 0s - loss: 0.0869 - accuracy: 0.9685WARNIN
[==============================] - 21s 489ms/step - loss: 0.0869 - accuracy: 0.968
23/100
[==============================] - ETA: 0s - loss: 0.0694 - accuracy: 0.9768WARNIN
[==============================] - 22s 499ms/step - loss: 0.0694 - accuracy: 0.976
24/100
[==============================] - ETA: 0s - loss: 0.0612 - accuracy: 0.9810WARNIN
[==============================] - 23s 518ms/step - loss: 0.0612 - accuracy: 0.981
.callbacks.History at 0x7efef0166050>
```

```
img = image.load_img('/content/flowers/sunflower/13095941995_9a66faa713_n.jpg',target_size
x = image.img_to_array(img) # Converting image into array
x = np.expand_dims(x,axis=0) # expanding Dimensions
pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
op = ['daisy','dandelion','rose','sunflower','tulip'] # Creating list
op[pred] # List indexing with output
```

```
'sunflower'
```

```
img = image.load_img('/content/flowers/tulip/10163955604_ae0b830975_n.jpg',target_size=(64
x = image.img_to_array(img) # Converting image into array
x = np.expand_dims(x,axis=0) # expanding Dimensions
pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
op = ['daisy','dandelion','rose','sunflower','tulip'] # Creating list
op[pred] # List indexing with output
```

```
'tulip'
```

```
img = image.load_img('/content/pexels-pixabay-39669.jpg',target_size=(64,64)) # Reading im
x = image.img_to_array(img) # Converting image into array
x = np.expand_dims(x,axis=0) # expanding Dimensions
pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
op = ['daisy','dandelion','rose','sunflower','tulip'] # Creating list
op[pred] # List indexing with output
```

```
'daisy'
```

```
img = image.load_img('/content/sunflower.jpg',target_size=(64,64)) # Reading image
x = image.img_to_array(img) # Converting image into array
x = np.expand_dims(x,axis=0) # expanding Dimensions
pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
op = ['daisy','dandelion','rose','sunflower','tulip'] # Creating list
op[pred] # List indexing with output
```

```
'sunflower'
```

Colab paid products - Cancel contracts here