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

```
inflating: flowers/tulip/4579079143_f65b59dd9f.jpg
inflating: flowers/tulip/4579128789_1561575458_n.jpg
inflating: flowers/tulip/4580206494_9386c81ed8_n.jpg
inflating: flowers/tulip/4582198748_20fa7caaa1.jpg
inflating: flowers/tulip/4587872443_a86c692cb8.jpg
inflating: flowers/tulip/4588904196_3c5825c7f4.jpg
inflating: flowers/tulip/4589624702_b6baa83699_m.jpg
inflating: flowers/tulip/4590702749_e1df8e0c1b.jpg
inflating: flowers/tulip/4590703575_6371c0a186_n.jpg
inflating: flowers/tulip/4591323356_030d8b6967_m.jpg
inflating: flowers/tulip/4599815420_8ee42c2382.jpg
inflating: flowers/tulip/4602809199_d3030cef01_m.jpg
inflating: flowers/tulip/4604238410_bcec9da4a0_n.jpg
inflating: flowers/tulip/4604272150_0c92385530_n.jpg
inflating: flowers/tulip/4612075317_91eefff68c_n.jpg
inflating: flowers/tulip/4624404489_11e10fcd33_n.jpg
inflating: flowers/tulip/4644110077_ff252cd7c4.jpg
inflating: flowers/tulip/466409031_4c10294db5_m.jpg
inflating: flowers/tulip/467702445_b8676f60fb_n.jpg
inflating: flowers/tulip/4679869990_7c5f28f2fe_n.jpg
inflating: flowers/tulip/4681062529_36186617d9.jpg
inflating: flowers/tulip/471298577_cc7558bcf1.jpg
inflating: flowers/tulip/478765271_6a8ca1cfa1_m.jpg
inflating: flowers/tulip/480228053_513791d474.jpg
inflating: flowers/tulip/4838669164_ffb6f67139.jpg
inflating: flowers/tulip/483880052_19fdb26a9f.jpg
inflating: flowers/tulip/485266837_671def8627.jpg
inflating: flowers/tulip/485415743_eeb5d7c1a5.jpg
inflating: flowers/tulip/4890786831_91bb82a9e4_n.jpg
inflating: flowers/tulip/489506904_9b68ba211c.jpg
inflating: flowers/tulip/490541142_c37e2b4191_n.jpg
inflating: flowers/tulip/4945315538_97bdd873c4.jpg
inflating: flowers/tulip/4955884820_7e4ce4d7e5_m.jpg
inflating: flowers/tulip/497305666_b5d4348826_n.jpg
inflating: flowers/tulip/5012813078_99fb977616_n.jpg
inflating: flowers/tulip/503770507_f397245a6a.jpg
inflating: flowers/tulip/5043225469_0aa23f3c8f_n.jpg
inflating: flowers/tulip/510698601_9f61d6f8d8.jpg
inflating: flowers/tulip/518256494_368a72db37.jpg
inflating: flowers/tulip/5208680166_c4372477ef_n.jpg
inflating: flowers/tulip/5388013398_09a8a0f166_m.jpg
inflating: flowers/tulip/5417115048_3b78d6c875_n.jpg
inflating: flowers/tulip/5430796647_f21b7b0fea.jpg
inflating: flowers/tulip/5433747333_869a2a172d_m.jpg
inflating: flowers/tulip/5443985113_54e9f608b3_n.jpg
inflating: flowers/tulip/5470898169_52a5ab876c_n.jpg
inflating: flowers/tulip/5524946579_307dc74476.jpg
inflating: flowers/tulip/5529939805_1679b014e1_n.jpg
inflating: flowers/tulip/5543457754_89c44c88de_n.jpg
inflating: flowers/tulip/5546723510_39a5a10d3a_n.jpg
inflating: flowers/tulip/5552198702_35856ed8ec.jpg
inflating: flowers/tulip/5565089564_a30c318f44.jpg
inflating: flowers/tulip/5574219476_1f46775487_n.jpg
inflating: flowers/tulip/5603625247_e4ff1828af_m.jpg
inflating: flowers/tulip/5628970369_54eb9ed31c_n.jpg
inflating: flowers/tulip/5631861819_f0eb39a357_m.jpg
inflating: flowers/tulip/5632006303_15acd2cf1e_n.jpg
```

```
        inflating: flowers/tulip/5633266048_4f4bfb2cf1_n.jpg
        inflating: flowers/tulip/5634767665_0ae724774d.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))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: UserWarning: `Mode

Epoch 1/100
44/44 [==============================] - ETA: 0s - loss: 0.2357 - accuracy: 0.9159
44/44 [==============================] - 22s 497ms/step - loss: 0.2357 - accuracy:
Epoch 2/100
44/44 [==============================] - ETA: 0s - loss: 0.1924 - accuracy: 0.9321
44/44 [==============================] - 21s 490ms/step - loss: 0.1924 - accuracy:
Epoch 3/100
44/44 [==============================] - ETA: 0s - loss: 0.1733 - accuracy: 0.9379
44/44 [==============================] - 21s 491ms/step - loss: 0.1733 - accuracy:
Epoch 4/100
44/44 [==============================] - ETA: 0s - loss: 0.1507 - accuracy: 0.9488
44/44 [==============================] - 21s 491ms/step - loss: 0.1507 - accuracy:
Epoch 5/100
44/44 [==============================] - ETA: 0s - loss: 0.1350 - accuracy: 0.9562
44/44 [==============================] - 21s 492ms/step - loss: 0.1350 - accuracy:
Epoch 6/100
44/44 [==============================] - ETA: 0s - loss: 0.1127 - accuracy: 0.9602
44/44 [==============================] - 22s 516ms/step - loss: 0.1127 - accuracy:
Epoch 7/100
44/44 [==============================] - ETA: 0s - loss: 0.1473 - accuracy: 0.9527
44/44 [==============================] - 21s 489ms/step - loss: 0.1473 - accuracy:
Epoch 8/100
44/44 [==============================] - ETA: 0s - loss: 0.1246 - accuracy: 0.9604
44/44 [==============================] - 21s 492ms/step - loss: 0.1246 - accuracy:
Epoch 9/100
44/44 [==============================] - ETA: 0s - loss: 0.1305 - accuracy: 0.9571
44/44 [==============================] - 22s 494ms/step - loss: 0.1305 - accuracy:
Epoch 10/100
44/44 [==============================] - ETA: 0s - loss: 0.1020 - accuracy: 0.9641
44/44 [==============================] - 22s 496ms/step - loss: 0.1020 - accuracy:
Epoch 11/100
44/44 [==============================] - ETA: 0s - loss: 0.1144 - accuracy: 0.9632
44/44 [==============================] - 21s 486ms/step - loss: 0.1144 - accuracy:
Epoch 12/100
43/44 [============================>.] - ETA: 0s - loss: 0.1151 - accuracy: 0.9642
44/44 [==============================] - 21s 490ms/step - loss: 0.1152 - accuracy:
Epoch 13/100
44/44 [==============================] - ETA: 0s - loss: 0.1265 - accuracy: 0.9583
44/44 [==============================] - 21s 489ms/step - loss: 0.1265 - accuracy:
Epoch 14/100
44/44 [==============================] - ETA: 0s - loss: 0.1079 - accuracy: 0.9671
```

```
44/44 [==============================] - 21s 490ms/step - loss: 0.1079 - accuracy:
Epoch 15/100
44/44 [==============================] - ETA: 0s - loss: 0.1282 - accuracy: 0.9569
44/44 [==============================] - 23s 523ms/step - loss: 0.1282 - accuracy:
Epoch 16/100
44/44 [==============================] - ETA: 0s - loss: 0.1027 - accuracy: 0.9683
44/44 [==============================] - 21s 492ms/step - loss: 0.1027 - accuracy:
Epoch 17/100
44/44 [==============================] - ETA: 0s - loss: 0.1047 - accuracy: 0.9673
44/44 [==============================] - 21s 491ms/step - loss: 0.1047 - accuracy:
Epoch 18/100
44/44 [==============================] - ETA: 0s - loss: 0.0935 - accuracy: 0.9685
44/44 [==============================] - 21s 492ms/step - loss: 0.0935 - accuracy:
```

```
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'
```