```python
import tensorflow as tf
import cv2
import random
import numpy as np
import matplotlib.pyplot as plt
import os
from tensorflow.keras.applications import *
from tensorflow.keras.models import *
from tensorflow.keras.layers import *
from tensorflow.keras.utils import load_img
```

```python
lt=[cv2.ROTATE_180,cv2.ROTATE_90_CLOCKWISE,cv2.ROTATE_90_COUNTERCLOCKWISE]
def brightness(img):
  value = random.uniform(0.5,2)
  hsv=cv2.cvtColor(img,cv2.COLOR_BGR2HSV)
  hsv=np.array(hsv,dtype=np.float64)
  hsv[:,:,1]=hsv[:,:,1]*value
  hsv[:,:,1][hsv[:,:,1]>255]=255
  hsv[:,:,2]=hsv[:,:,2]*value
  hsv[:,:,2][hsv[:,:,2]>255]=255
  hsv=np.array(hsv,dtype=np.uint8)
  img=cv2.cvtColor(np.array(hsv,dtype=np.uint8),cv2.COLOR_HSV2BGR)
  return img
```

```python
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```python
import pathlib
import glob
directory = pathlib.Path("/content/drive/My Drive/members")
resultant="/content/augmenntedimages"
items = os.listdir(directory)
classes=[]
count=0
images=[]
labels=[]
for i in items:
  i1=0
  print(i)
  classes.append(i)
  path1=f"{directory}/{i}"
  a=random.randint(4,10)
  img=cv2.imread(path1)
  img=cv2.resize(img,(224,224))
  k=i.split(".")[0]
  cv2.imwrite(f"{resultant}/{k}_{i1}.png",img)
  i1+=1
  while a!=0:
    img=cv2.rotate(img,lt[random.randint(0,2)])
    images.append(img)
    cv2.imwrite(f"{resultant}/{k}_{i1}.png",img)
    i1+=1
    labels.append(count)
    if a%2==0:
      img=brightness(img)
      images.append(img)
      cv2.imwrite(f"{resultant}/{k}_{i1}.png",img)
      i1+=1
      labels.append(count)
    a-=1
  count+=1
images=np.array(images)
labels=np.array(labels)
```

```
A.jpg
B.jpg
C.jpg
D.jpg
E.jpg
F.jpg
G.jpg
```

```
images.shape
```

```
(81, 224, 224, 3)
```

```
from keras.layers import Dense,Dropout,Flatten
from tensorflow.keras.models import *
from keras.applications.vgg16 import VGG16,preprocess_input
model=VGG16(weights="imagenet")
for i in model.layers:
  i.trainable=False
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-
applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels.h5
553467096/553467096 ━━━━━━━━━━━━━━━━━━━━ 3s 0us/step
```

```
len(model.layers)
```

```
23
```

```
model.summary()
```

```
Model: "vgg16"
```

```
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┓
━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                         ┃ Output Shape                ┃
```

```
Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇
━━━━━━━━━━━━━━━━━┩
│ input_layer (InputLayer)             │ (None, 224, 224, 3)         │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block1_conv1 (Conv2D)                │ (None, 224, 224, 64)        │
1,792 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block1_conv2 (Conv2D)                │ (None, 224, 224, 64)        │
36,928 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block1_pool (MaxPooling2D)           │ (None, 112, 112, 64)        │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block2_conv1 (Conv2D)                │ (None, 112, 112, 128)       │
73,856 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block2_conv2 (Conv2D)                │ (None, 112, 112, 128)       │
147,584 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block2_pool (MaxPooling2D)           │ (None, 56, 56, 128)         │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block3_conv1 (Conv2D)                │ (None, 56, 56, 256)         │
295,168 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block3_conv2 (Conv2D)                │ (None, 56, 56, 256)         │
590,080 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block3_conv3 (Conv2D)                │ (None, 56, 56, 256)         │
590,080 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block3_pool (MaxPooling2D)           │ (None, 28, 28, 256)         │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block4_conv1 (Conv2D)                │ (None, 28, 28, 512)         │
1,180,160 │
```

```
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block4_conv2 (Conv2D)                │ (None, 28, 28, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block4_conv3 (Conv2D)                │ (None, 28, 28, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block4_pool (MaxPooling2D)           │ (None, 14, 14, 512)         │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block5_conv1 (Conv2D)                │ (None, 14, 14, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block5_conv2 (Conv2D)                │ (None, 14, 14, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block5_conv3 (Conv2D)                │ (None, 14, 14, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block5_pool (MaxPooling2D)           │ (None, 7, 7, 512)           │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ flatten (Flatten)                    │ (None, 25088)               │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ fc1 (Dense)                          │ (None, 4096)                │
102,764,544 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ fc2 (Dense)                          │ (None, 4096)                │
16,781,312 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ predictions (Dense)                  │ (None, 1000)                │
4,097,000 │
└──────────────────────────────────────┴─────────────────────────────┴
─────────────────┘

 Total params: 138,357,544 (527.79 MB)

 Trainable params: 0 (0.00 B)
 Non-trainable params: 138,357,544 (527.79 MB)
```

```
transferVGG=Sequential()
for i in range(18):
  transferVGG.add(model.layers[i])
transferVGG.add(Flatten())
transferVGG.add(Dense(512,activation="relu"))
transferVGG.add(Dense(128,activation="relu"))
transferVGG.add(Dense(7,activation="softmax"))
transferVGG.summary()
```

```
Model: "sequential"

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳
━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                         ┃ Output Shape                ┃
Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇
━━━━━━━━━━━━━━━━━┩
│ block1_conv1 (Conv2D)                │ (None, 224, 224, 64)        │
1,792 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block1_conv2 (Conv2D)                │ (None, 224, 224, 64)        │
36,928 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block1_pool (MaxPooling2D)           │ (None, 112, 112, 64)        │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block2_conv1 (Conv2D)                │ (None, 112, 112, 128)       │
73,856 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block2_conv2 (Conv2D)                │ (None, 112, 112, 128)       │
147,584 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block2_pool (MaxPooling2D)           │ (None, 56, 56, 128)         │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block3_conv1 (Conv2D)                │ (None, 56, 56, 256)         │
295,168 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block3_conv2 (Conv2D)                │ (None, 56, 56, 256)         │
590,080 │
```

```
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block3_conv3 (Conv2D)                │ (None, 56, 56, 256)         │
590,080 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block3_pool (MaxPooling2D)           │ (None, 28, 28, 256)         │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block4_conv1 (Conv2D)                │ (None, 28, 28, 512)         │
1,180,160 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block4_conv2 (Conv2D)                │ (None, 28, 28, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block4_conv3 (Conv2D)                │ (None, 28, 28, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block4_pool (MaxPooling2D)           │ (None, 14, 14, 512)         │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block5_conv1 (Conv2D)                │ (None, 14, 14, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block5_conv2 (Conv2D)                │ (None, 14, 14, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ block5_conv3 (Conv2D)                │ (None, 14, 14, 512)         │
2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ flatten (Flatten)                    │ (None, 100352)              │
0 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ dense (Dense)                        │ (None, 512)                 │
51,380,736 │
├──────────────────────────────────────┼─────────────────────────────┼
─────────────────┤
│ dense_1 (Dense)                      │ (None, 128)                 │
65,664 │
├──────────────────────────────────────┼─────────────────────────────┼
```

```
─────────────────┤
│ dense_2 (Dense)                      │ (None, 7)                   │
903 │
└──────────────────────────────────────┴─────────────────────────────┴
─────────────────┘

 Total params: 66,161,991 (252.39 MB)

 Trainable params: 51,447,303 (196.26 MB)

 Non-trainable params: 14,714,688 (56.13 MB)
```

```
import tensorflow as tf
class myCallback(tf.keras.callbacks.Callback):
  def on_epoch_end(self,epoch,logs={}):
    print("call")
    if(logs.get('accuracy')>0.99):
      print("\nReached %2.2f%% accuracy so, stopping training!!"
%(99))
      self.model.stop_training=True
callbacks=myCallback()
transferVGG.compile(optimizer="adam",loss="sparse_categorical_crossent
ropy",metrics=["accuracy"])
transferVGG.fit(images,labels,epochs=100,callbacks=[callbacks])
```

```
Epoch 1/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 17s/step - accuracy: 0.1314 - loss:
26.4505 call
3/3 ━━━━━━━━━━━━━━━━━━━━ 60s 17s/step - accuracy: 0.1541 - loss:
28.8140
Epoch 2/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 16s/step - accuracy: 0.7894 - loss:
18.1687 call
3/3 ━━━━━━━━━━━━━━━━━━━━ 82s 16s/step - accuracy: 0.7896 - loss:
18.3494
Epoch 3/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 16s/step - accuracy: 0.9668 - loss: 0.9085
call
3/3 ━━━━━━━━━━━━━━━━━━━━ 77s 16s/step - accuracy: 0.9659 - loss:
0.8511
Epoch 4/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 17s/step - accuracy: 1.0000 - loss:
6.6687e-09 call

Reached 99.00% accuracy so, stopping training!!
3/3 ━━━━━━━━━━━━━━━━━━━━ 85s 17s/step - accuracy: 1.0000 - loss:
7.2091e-09

<keras.src.callbacks.history.History at 0x7dae18a12140>
```

```
transferVGG.evaluate(images,labels)
```

```
3/3 ━━━━━━━━━━━━━━━━━━━━ 52s 16s/step - accuracy: 0.9899 - loss:
0.2900

[0.35524889826774597, 0.9876543283462524]

def predict(i,transferVGG,labels):
  path1=f"{directory}/{i}"

  img=cv2.imread(path1)
  img=cv2.resize(img,(224,224))
  a=np.argmax(transferVGG.predict(np.array([img])))
  img=cv2.putText(img,labels[a],(25,25),cv2.FONT_HERSHEY_SIMPLEX,1,
(255,0,0),2,cv2.LINE_AA)

  plt.imshow(img)
predict(i="C.jpg",transferVGG=transferVGG,labels=classes)

1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step
```