```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
import keras
from keras.models import Sequential
from keras.layers import Dense, Conv2D, MaxPool2D , Flatten
from keras.preprocessing.image import ImageDataGenerator
from keras.applications.vgg16 import VGG16
from keras.models import Model
from keras.optimizers import Adam
from keras.callbacks import ModelCheckpoint, EarlyStopping
from PIL import Image
import matplotlib.pyplot as plt


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

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

```
traindata = r"/content/drive/MyDrive/dataset/train"

testdata = r"/content/drive/MyDrive/dataset/test"
train_datagen = ImageDataGenerator(
    rotation_range=40,
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    traindata,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical')

validation_generator = test_datagen.flow_from_directory(
    testdata,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical')
```

```
Found 840 images belonging to 2 classes.
Found 188 images belonging to 2 classes.
```

```
model = Sequential()
model.add(tf.keras.applications.vgg16.VGG16(
    include_top=False,
```

✓ 1s completed at 1:18 AM

```
    input_tensor=None,
    input_shape=None,
    pooling='avg',
    classes=2,
    classifier_activation='softmax'
))
model.add(Flatten())
model.add(Dense(units=4096,activation="relu"))
model.add(Dense(units=4096,activation="relu"))
model.add(Dense(units=2, activation="softmax"))



opt = Adam(lr=0.0001)
model.compile(optimizer=opt, loss=keras.losses.categorical_crossentropy, metrics

for layer in model.layers[0].layers:
    layer.trainable = False

model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-appli
58889256/58889256 [==============================] - 5s 0us/step
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 512)               14714688

 flatten (Flatten)           (None, 512)               0

 dense (Dense)               (None, 4096)              2101248

 dense_1 (Dense)             (None, 4096)              16781312

 dense_2 (Dense)             (None, 2)                 8194

=================================================================
Total params: 33,605,442
Trainable params: 18,890,754
Non-trainable params: 14,714,688
_________________________________________________________________
/usr/local/lib/python3.8/dist-packages/keras/optimizers/optimizer_v2/adam.p
  super(Adam, self).__init__(name, **kwargs)
```

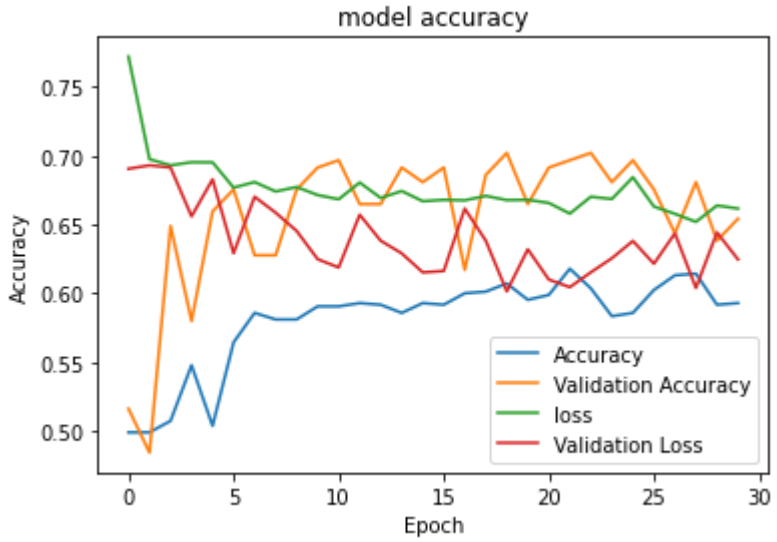
```
hist = model.fit(train_generator , validation_data = validation_generator, epoch
```

```
Epoch 1/30
27/27 [==============================] - 644s 23s/step - loss: 0.7722 - acc
Epoch 2/30
27/27 [==============================] - 18s 651ms/step - loss: 0.6975 - ac
Epoch 3/30
27/27 [==============================] - 19s 693ms/step - loss: 0.6931 - ac
Epoch 4/30
```

```
27/27 [==============================] - 18s 659ms/step - loss: 0.6954 - ac
Epoch 5/30
27/27 [==============================] - 18s 657ms/step - loss: 0.6953 - ac
Epoch 6/30
27/27 [==============================] - 18s 652ms/step - loss: 0.6769 - ac
Epoch 7/30
27/27 [==============================] - 18s 659ms/step - loss: 0.6809 - ac
Epoch 8/30
27/27 [==============================] - 18s 658ms/step - loss: 0.6741 - ac
Epoch 9/30
27/27 [==============================] - 18s 663ms/step - loss: 0.6773 - ac
Epoch 10/30
27/27 [==============================] - 18s 654ms/step - loss: 0.6715 - ac
Epoch 11/30
27/27 [==============================] - 18s 656ms/step - loss: 0.6684 - ac
Epoch 12/30
27/27 [==============================] - 18s 660ms/step - loss: 0.6806 - ac
Epoch 13/30
27/27 [==============================] - 19s 695ms/step - loss: 0.6694 - ac
Epoch 14/30
27/27 [==============================] - 18s 656ms/step - loss: 0.6744 - ac
Epoch 15/30
27/27 [==============================] - 18s 655ms/step - loss: 0.6670 - ac
Epoch 16/30
27/27 [==============================] - 18s 650ms/step - loss: 0.6680 - ac
Epoch 17/30
27/27 [==============================] - 18s 651ms/step - loss: 0.6676 - ac
Epoch 18/30
27/27 [==============================] - 18s 653ms/step - loss: 0.6709 - ac
Epoch 19/30
27/27 [==============================] - 18s 657ms/step - loss: 0.6678 - ac
Epoch 20/30
27/27 [==============================] - 18s 656ms/step - loss: 0.6680 - ac
Epoch 21/30
27/27 [==============================] - 18s 658ms/step - loss: 0.6657 - ac
Epoch 22/30
27/27 [==============================] - 18s 658ms/step - loss: 0.6580 - ac
Epoch 23/30
27/27 [==============================] - 18s 655ms/step - loss: 0.6702 - ac
Epoch 24/30
27/27 [==============================] - 18s 658ms/step - loss: 0.6686 - ac
Epoch 25/30
27/27 [==============================] - 18s 656ms/step - loss: 0.6844 - ac
Epoch 26/30
27/27 [==============================] - 18s 652ms/step - loss: 0.6631 - ac
Epoch 27/30
27/27 [==============================] - 18s 655ms/step - loss: 0.6578 - ac
Epoch 28/30
27/27 [==============================] - 18s 656ms/step - loss: 0.6520 - ac
Epoch 29/30
27/27 [==============================] - 18s 664ms/step - loss: 0.6639 - ac
```

```
plt.plot(hist.history["accuracy"])
plt.plot(hist.history['val_accuracy'])
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title("model accuracy")
plt.ylabel("Accuracy")
```

```
plt.xlabel("Epoch")
plt.legend(["Accuracy","Validation Accuracy","loss","Validation Loss"])
plt.show()
```




```
model.save('/content/drive/MyDrive/model/elephant_model_vggid.h5')
model.save_weights('/content/drive/MyDrive/model/elephant_weights_vggid.h5')
width, height = 224, 224
model_p = '/content/drive/MyDrive/model/elephant_model_vggid.h5'
weight_p = '/content/drive/MyDrive/model/elephant_weights_vggid.h5'

model = tf.keras.models.load_model(model_p)
model.load_weights(weight_p)
def fpredict(file):
    if not os.path.isfile(file):
        return "Error: file does not exist."
    try:
        test_image = tf.keras.utils.load_img(file, target_size=(width,height))
        test_image = tf.keras.utils.img_to_array(test_image)
        test_image = np.expand_dims(test_image, axis=0)
    except Exception as e:
        return f"Error: {e}"
    if not 'model' in globals():
        return "Error: model is not defined."
    try:
        array = model.predict(test_image)
        result = array[0]
    except Exception as e:
        return f"Error: {e}"
    if result[0] >= 0.5 :
        an = 'African Elephant'
    elif result[1]>= 0.5:
        an = 'Asian Elephant'
    else:
        an = "Not sure"
    return an
```

```
WARNING:tensorflow:Error in loading the saved optimizer state. As a result,
```

```
fpredict('/content/drive/MyDrive/elephsnt.jpg')
```

```
1/1 [==============================] - 1s 988ms/step
'Asian Elephant'
```