MobileNet implementation on Cellphone

```
import tensorflow as tf
from tensorflow.keras import Sequential,Model
from tensorflow.keras.layers import Dense,Dropout,Flatten,BatchNormalization
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from keras.applications import MobileNet
from keras.applications.mobilenet import preprocess_input
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

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

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

```
img_width=224
img_height=224
batch_size=10
```

```
# Initializing training dataset
train_data_dir="/content/drive/My Drive/cell_phone/training"
datagen = ImageDataGenerator(rescale=1./255)
train_generator = datagen.flow_from_directory(directory=train_data_dir,
                                              target_size = (img_width, img_height),
                                              class_mode = 'binary',
                                              batch_size=batch_size)
```

```
Found 245 images belonging to 2 classes.
```

```
# Initializing validation dataset
validation_dir="/content/drive/My Drive/cell_phone/val"
val_generator=datagen.flow_from_directory(validation_dir,target_size=(img_width,img_height),classes=['cat','dog'],batch_size
```

```
Found 0 images belonging to 2 classes.
```

```
mobile=tf.keras.applications.mobilenet.MobileNet()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilenet/mobilenet_1_0_224_tf.h5
17227776/17225924 [==============================] - 0s 0us/step
```

```
mobile.summary()
```

```
_________________________________________________________________
conv_pw_1 (Conv2D)           (None, 112, 112, 64)      2048
_________________________________________________________________
conv_pw_1_bn (BatchNormaliza (None, 112, 112, 64)      256
_________________________________________________________________
conv_pw_1_relu (ReLU)        (None, 112, 112, 64)      0
_________________________________________________________________
conv_pad_2 (ZeroPadding2D)   (None, 113, 113, 64)      0
_________________________________________________________________
conv_dw_2 (DepthwiseConv2D)  (None, 56, 56, 64)        576
_________________________________________________________________
conv_dw_2_bn (BatchNormaliza (None, 56, 56, 64)        256
_________________________________________________________________
conv_dw_2_relu (ReLU)        (None, 56, 56, 64)        0
_________________________________________________________________
conv_pw_2 (Conv2D)           (None, 56, 56, 128)       8192
_________________________________________________________________
conv_pw_2_bn (BatchNormaliza (None, 56, 56, 128)       512
_________________________________________________________________
conv_pw_2_relu (ReLU)        (None, 56, 56, 128)       0
_________________________________________________________________
conv_dw_3 (DepthwiseConv2D)  (None, 56, 56, 128)       1152
_________________________________________________________________
conv_dw_3_bn (BatchNormaliza (None, 56, 56, 128)       512
_________________________________________________________________
conv_dw_3_relu (ReLU)        (None, 56, 56, 128)       0
```

```
conv_pw_3 (Conv2D)           (None, 56, 56, 128)       16384
_________________________________________________________________
conv_pw_3_bn (BatchNormaliza (None, 56, 56, 128)       512
_________________________________________________________________
conv_pw_3_relu (ReLU)        (None, 56, 56, 128)       0
_________________________________________________________________
conv_pad_4 (ZeroPadding2D)   (None, 57, 57, 128)       0
_________________________________________________________________
conv_dw_4 (DepthwiseConv2D)  (None, 28, 28, 128)       1152
_________________________________________________________________
conv_dw_4_bn (BatchNormaliza (None, 28, 28, 128)       512
_________________________________________________________________
conv_dw_4_relu (ReLU)        (None, 28, 28, 128)       0
_________________________________________________________________
conv_pw_4 (Conv2D)           (None, 28, 28, 256)       32768
_________________________________________________________________
conv_pw_4_bn (BatchNormaliza (None, 28, 28, 256)       1024
_________________________________________________________________
conv_pw_4_relu (ReLU)        (None, 28, 28, 256)       0
_________________________________________________________________
conv_dw_5 (DepthwiseConv2D)  (None, 28, 28, 256)       2304
_________________________________________________________________
conv_dw_5_bn (BatchNormaliza (None, 28, 28, 256)       1024
_________________________________________________________________
conv_dw_5_relu (ReLU)        (None, 28, 28, 256)       0
_________________________________________________________________
conv_pw_5 (Conv2D)           (None, 28, 28, 256)       65536
_________________________________________________________________
conv_pw_5_bn (BatchNormaliza (None, 28, 28, 256)       1024
_________________________________________________________________
conv_pw_5_relu (ReLU)        (None, 28, 28, 256)       0
_________________________________________________________________
conv_pad_6 (ZeroPadding2D)   (None, 29, 29, 256)       0
```

```
# This says that not to use dense values mentioned in VGG (Not to use VGG layers)
for layers in mobile.layers:
  layers.trainable=False


# Training the model
model=Sequential()
model.add(mobile)
```

```
model.add(Flatten())
model.add(Dense(128,activation='relu',))
model.add(Dropout(0.5))
model.add(BatchNormalization())
model.add(Dense(1,activation="sigmoid"))


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



history=model.fit_generator(generator=train_generator, steps_per_epoch=len(train_generator), epochs = 30,
                            validation_data=val_generator, validation_steps=len(val_generator)
                            , verbose = 2)
```

```
WARNING:tensorflow:From <ipython-input-12-624fd0d036d0>:3: Model.fit_generator (from tensorflow.python.keras.engine
Instructions for updating:
Please use Model.fit, which supports generators.
Epoch 1/30
25/25 - 111s - loss: 0.4701 - accuracy: 0.7796
Epoch 2/30
25/25 - 9s - loss: 0.2606 - accuracy: 0.8980
Epoch 3/30
25/25 - 9s - loss: 0.1737 - accuracy: 0.9551
Epoch 4/30
25/25 - 9s - loss: 0.2366 - accuracy: 0.9184
Epoch 5/30
25/25 - 9s - loss: 0.1381 - accuracy: 0.9510
Epoch 6/30
25/25 - 9s - loss: 0.1014 - accuracy: 0.9755
Epoch 7/30
25/25 - 9s - loss: 0.0886 - accuracy: 0.9714
Epoch 8/30
25/25 - 9s - loss: 0.0999 - accuracy: 0.9755
Epoch 9/30
25/25 - 9s - loss: 0.0627 - accuracy: 0.9918
Epoch 10/30
25/25 - 9s - loss: 0.0859 - accuracy: 0.9714
Epoch 11/30
25/25 - 9s - loss: 0.0678 - accuracy: 0.9755
Epoch 12/30
```

```
25/25 - 9s - loss: 0.0766 - accuracy: 0.9714
Epoch 13/30
25/25 - 9s - loss: 0.0487 - accuracy: 0.9837
Epoch 14/30
25/25 - 9s - loss: 0.0731 - accuracy: 0.9755
Epoch 15/30
25/25 - 9s - loss: 0.0868 - accuracy: 0.9673
Epoch 16/30
25/25 - 9s - loss: 0.0707 - accuracy: 0.9755
Epoch 17/30
25/25 - 9s - loss: 0.0729 - accuracy: 0.9755
Epoch 18/30
25/25 - 10s - loss: 0.0551 - accuracy: 0.9837
Epoch 19/30
25/25 - 9s - loss: 0.0420 - accuracy: 0.9837
Epoch 20/30
25/25 - 9s - loss: 0.0591 - accuracy: 0.9796
Epoch 21/30
25/25 - 9s - loss: 0.0697 - accuracy: 0.9837
Epoch 22/30
25/25 - 9s - loss: 0.0604 - accuracy: 0.9837
Epoch 23/30
25/25 - 9s - loss: 0.0572 - accuracy: 0.9714
Epoch 24/30
25/25 - 9s - loss: 0.0836 - accuracy: 0.9796
Epoch 25/30
25/25 - 9s - loss: 0.0544 - accuracy: 0.9878
Epoch 26/30
25/25 - 9s - loss: 0.0274 - accuracy: 0.9959
Epoch 27/30
25/25 - 9s - loss: 0.0782 - accuracy: 0.9796
Epoch 28/30
```

```
from tensorflow.keras.preprocessing import image
```

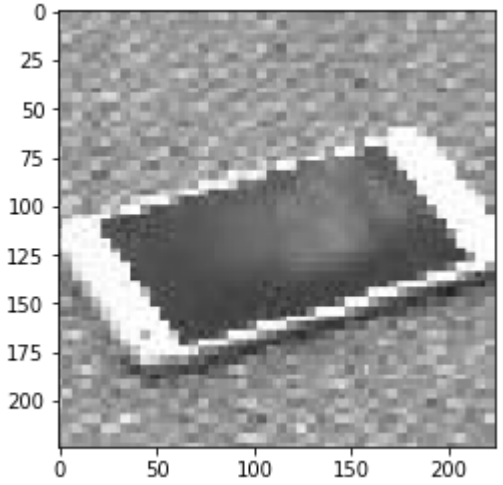
```
img=image.load_img("/content/img130.png",target_size=(img_width,img_height))
plt.imshow(img)
img=image.img_to_array(img)
```

```
img=img/255.0
img = np.expand_dims(img, axis=0)
img_class = np.argmax(model.predict(img),axis=1)

if(model.predict(img)<=0.5):
  print('cellphone=NO')

else:
  print('cellphone=YES')
```

```
cellphone=YES
```




```
img=image.load_img("/content/img82.png",target_size=(img_width,img_height))
plt.imshow(img)
img=image.img_to_array(img)
img=img/255.0
img = np.expand_dims(img, axis=0)

if(model.predict(img)<=0.5):
  print('cellphone=NO')

else:
  print('cellphone=YES')
```

```
cellphone=NO
```