Major Project AI

October 13, 2022

[2]:

```
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense, Dropout, Conv2D, MaxPool2D, Flatten
from keras import layers,models
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from keras.preprocessing import image
import matplotlib.pyplot as plt
import matplotlib.image as nping
import matplotlib.pyplot as plt
%matplotlib inline
import random
from tensorflow.keras.applications import InceptionV3
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import BatchNormalization, Conv2D, MaxPooling2D, Activation, Flatten, Dropout, Dense
from tensorflow.keras.preprocessing.image import load_img,img_to_array
```

[3]:

```
#Accesing Google drive
```

[4]:

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

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

[5]:

```
!unzip /content/drive/MyDrive/drive-download-20220930T072955Z-001.zip
```

```
Archive:  /content/drive/MyDrive/drive-download-20220930T072955Z-001.zip
  inflating: Validation.zip
  inflating: Training.zip
```

[ ]:

```
!unzip /content/Training.zip
```

[ ]:

```
!unzip /content/Validation.zip
```

[8]:

```
#Training Data
```

[9]:
```
epochs=50
lr=1e-3
batch_size=128
data=[]
labels=[]
```

[10]:
```
size=224
```

[11]:
```
train_datagen=ImageDataGenerator(horizontal_flip=True,width_shift_range=0.
 ↪4,height_shift_range=0.4,zoom_range=0.3,rotation_range=20,rescale=1/255)
```

[12]:
```
test_gen=ImageDataGenerator(rescale=1/255)
```

[13]:
```
target_size=(size,size)
target_size
```

[13]:
```
(224, 224)
```

[14]:
```
train_generator=train_datagen.flow_from_directory(directory="/content/
 ↪Training",target_size=target_size,batch_size=batch_size,class_mode="binary")
```

```
Found 47009 images belonging to 2 classes.
```

[15]:
```
validation_generator=test_gen.flow_from_directory(directory="/content/
 ↪Validation",target_size=target_size,batch_size=batch_size,class_mode="binary")
```

```
Found 11649 images belonging to 2 classes.
```

[16]:
```
train_generator.class_indices
```

[16]:
```
{'female': 0, 'male': 1}
```

[17]:
```
len(train_generator.classes)
```

[17]:
```
47009
```

[18]:
```
train_generator.class_mode
```

[18]:
```
'binary'
```

[19]:
```
x,y=train_generator.next()
```

[20]:
```
x[0].shape
```

[20]:
```
(224, 224, 3)
```

[21]:
```
x[0]
```

```
[21]: array([[[0.30090436, 0.13831134, 0.03921569],
              [0.3019608 , 0.13725491, 0.03921569],
              [0.33054796, 0.16686305, 0.06576092],
              …,
              [0.86274517, 0.68235296, 0.5803922 ],
              [0.8523215 , 0.67609876, 0.5762227 ],
              [0.8431373 , 0.67058825, 0.57254905]],

             [[0.32226366, 0.16057092, 0.06170106],
              [0.34837404, 0.18366814, 0.08562892],
              [0.37564322, 0.21150817, 0.11175646],
              …,
              [0.86639935, 0.6860072 , 0.58550805],
              [0.8590764 , 0.6819534 , 0.5828881 ],
              [0.84743124, 0.67488223, 0.576843  ]],

             [[0.33736065, 0.1765682 , 0.07852898],
              [0.38823533, 0.22352943, 0.1254902 ],
              [0.39282152, 0.2282363 , 0.12983501],
              …,
              [0.882353  , 0.7019608 , 0.60784316],
              [0.8740618 , 0.69603854, 0.60073644],
              [0.854902  , 0.68235296, 0.58431375]],

             …,

             [[0.7607844 , 0.6       , 0.52156866],
              [0.7607844 , 0.6       , 0.52156866],
              [0.7607844 , 0.6       , 0.52156866],
              …,
              [0.47450984, 0.19215688, 0.09411766],
              [0.50408185, 0.22172889, 0.12368967],
              [0.6420583 , 0.35970533, 0.26166612]],

             [[0.7607844 , 0.6       , 0.52156866],
              [0.7607844 , 0.6       , 0.52156866],
              [0.7607844 , 0.6       , 0.52156866],
              …,
              [0.47450984, 0.19215688, 0.09411766],
              [0.4833754 , 0.20102243, 0.10298321],
              [0.62135184, 0.33899888, 0.24095964]],

             [[0.7607844 , 0.6       , 0.52156866],
              [0.7607844 , 0.6       , 0.52156866],
              [0.7607844 , 0.6       , 0.52156866],
              …,
              [0.47450984, 0.19215688, 0.09411766],
              [0.47450984, 0.19215688, 0.09411766],
              [0.6006453 , 0.3182924 , 0.22025318]]], dtype=float32)
```

[22]:

```
#Building Model for Gender prediction
```

[23]:

```
model=Sequential()
model.add(InceptionV3(include_top=False,pooling="avg",weights="imagenet"))
model.add(Flatten())

model.add(BatchNormalization())
model.add(Dense(2048,activation="relu"))
model.add(BatchNormalization())

model.add(Dense(1024,activation="relu"))
model.add(BatchNormalization())

model.add(Dense(1,activation="sigmoid"))

model.layers[0].trainable=False
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applicatio
ns/inception_v3/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5
87910968/87910968 [==============================] - 0s 0us/step
```

[24]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 inception_v3 (Functional)   (None, 2048)              21802784

 flatten (Flatten)           (None, 2048)              0

 batch_normalization_94 (Bat  (None, 2048)             8192
 chNormalization)

 dense (Dense)               (None, 2048)              4196352

 batch_normalization_95 (Bat  (None, 2048)             8192
 chNormalization)

 dense_1 (Dense)             (None, 1024)              2098176

 batch_normalization_96 (Bat  (None, 1024)             4096
 chNormalization)
 dense_2 (Dense)             (None, 1)                 1025

=================================================================
Total params: 28,118,817
Trainable params: 6,305,793
Non-trainable params: 21,813,024
_________________________________________________________________
```

[25]:
```
model.compile(optimizer="adam",loss="binary_crossentropy",metrics=["accuracy"])
```

[26]:
```
len((train_generator.filenames)),batch_size,len((train_generator.filenames))//
 ↪batch_size
```

[26]:
```
(47009, 128, 367)
```

[27]:
```
model.fit(train_generator,steps_per_epoch=len(train_generator.filenames)//
 ↪batch_size,epochs=5,validation_data=validation_generator,validation_steps=len(validation_ge
 ↪filenames)//batch_size)
```

```
Epoch 1/5
367/367 [==============================] - 544s 1s/step - loss: 0.3815 -
accuracy: 0.8494 - val_loss: 0.2023 - val_accuracy: 0.9211
Epoch 2/5
367/367 [==============================] - 527s 1s/step - loss: 0.3098 -
accuracy: 0.8694 - val_loss: 0.2060 - val_accuracy: 0.9208
Epoch 3/5
367/367 [==============================] - 523s 1s/step - loss: 0.2972 -
accuracy: 0.8744 - val_loss: 0.2039 - val_accuracy: 0.9246
Epoch 4/5
367/367 [==============================] - 527s 1s/step - loss: 0.2920 -
accuracy: 0.8781 - val_loss: 0.2094 - val_accuracy: 0.9172
Epoch 5/5
367/367 [==============================] - 526s 1s/step - loss: 0.2814 -
accuracy: 0.8834 - val_loss: 0.2456 - val_accuracy: 0.9015
```

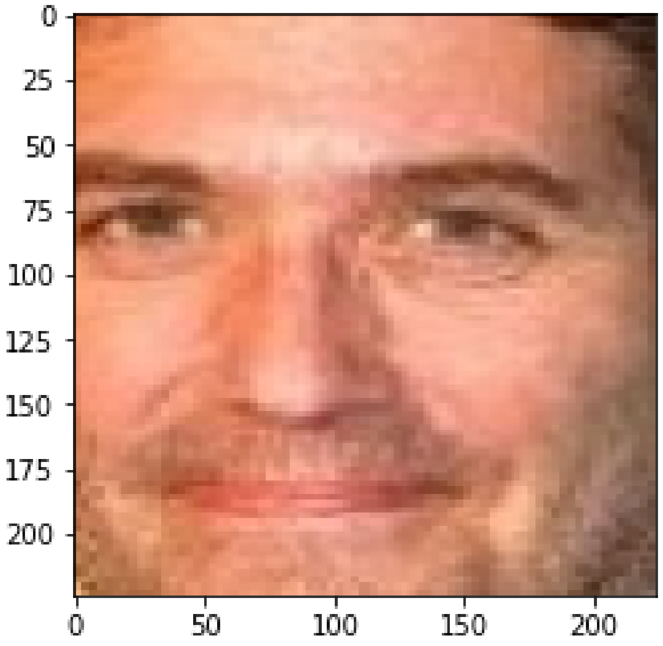
[27]:
```
<keras.callbacks.History at 0x7f820e76d610>
```

[29]:
```
#Testing model by passing a random image
```

[28]:
```
img_path="/content/Training/male/090648.jpg.jpg"
```

[29]:
```
img=load_img(img_path,target_size=(size,size,3))
plt.imshow(img)
```

[29]:
```
<matplotlib.image.AxesImage at 0x7f817fe9eb10>
```



[30]:

```
img=img_to_array(img)
img
```

[30]:

```
array([[[203., 108.,  64.],
         [203., 108.,  64.],
         [203., 108.,  64.],
         …,
         [ 47.,  28.,  21.],
         [ 47.,  28.,  21.],
         [ 47.,  28.,  21.]],

        [[203., 108.,  64.],
         [203., 108.,  64.],
         [203., 108.,  64.],
         …,
         [ 47.,  28.,  21.],
         [ 47.,  28.,  21.],
         [ 47.,  28.,  21.]],

        [[184.,  87.,  44.],
         [184.,  87.,  44.],
         [184.,  87.,  44.],
         …,
         [ 40.,  21.,  14.],
         [ 40.,  21.,  14.],
         [ 40.,  21.,  14.]],

        …,

        [[224., 168., 119.],
         [224., 168., 119.],
         [224., 168., 119.],
         …,
         [107.,  83.,  71.],
         [107.,  83.,  71.],
         [107.,  83.,  71.]],

        [[244., 186., 138.],
         [244., 186., 138.],
         [244., 186., 138.],
         …,
         [132., 110.,  99.],
         [132., 110.,  99.],
         [132., 110.,  99.]],

        [[244., 186., 138.],
         [244., 186., 138.],
         [244., 186., 138.],
         …,
         [132., 110.,  99.],
         [132., 110.,  99.],
         [132., 110.,  99.]]], dtype=float32)
```

[31]:
```
img=img/255.0
img=img.reshape(1,size,size,3)
```

[32]:
```
img.shape
```

[32]:
```
(1, 224, 224, 3)
```

[33]:
```
res=model.predict(img)
res
```

```
1/1 [==============================] - 2s 2s/step
```

[33]:
```
array([[0.9878735]], dtype=float32)
```

[34]:
```
train_generator.class_indices
```

[34]:
```
{'female': 0, 'male': 1}
```

[35]:

```
if res[0][0]<=0.5:
  prediction="women"
else:
  prediction="men"
print("prediction:",prediction)
```

```
prediction: men
```

[36]:

```
#predicting whole Dataset 1st validation data set
```

[37]:

```
import matplotlib.pyplot as plt
import matplotlib.image as nping
import matplotlib.pyplot as plt
import os
import random
import numpy as np
import cv2
import pandas as pd
from tensorflow.keras.preprocessing.image import load_img,img_to_array
directroy=r"/content/Validation"
Catogeries=["female","male"]
```

[38]:

```
size=224
data=[]
i=0
data=pd.read_csv("/content/work.csv")
l_pred=list()
l_imgid=list()
d={}
```

[ ]:

```
for catogery in Catogeries:
  folder=os.path.join(directroy,catogery)
  for img in os.listdir(folder):
    img_path=os.path.join(folder,img)
    img_arr=cv2.imread(img_path)
    img_arr=cv2.resize(img_arr,(size,size))
    img_arr=img_to_array(img_arr)
    img_arr=img_arr/255.0
    img_arr=img_arr.reshape(1,size,size,3)
    res=model.predict(img_arr)
    i=i+1
    if res[0][0]<0.5:
      l_imgid.append(i)
      l_pred.append("Women")
    else:
      l_imgid.append(i)
      l_pred.append("Men")
```

[40]:
```
len(l_imgid)
```

[40]:
```
11649
```

[41]:
```
#saving prediction to csv file
```

[42]:
```
d=pd.DataFrame({"Image_id":l_imgid,"prediction":l_pred})
d.to_csv("/content/work.csv")
s=pd.read_csv("/content/work.csv")
s
```

[42]:
```
       Unnamed: 0  Image_id prediction
0               0         1      Women
1               1         2      Women
2               2         3        Men
3               3         4      Women
4               4         5      Women
…             …         …         …
11644       11644     11645        Men
11645       11645     11646        Men
11646       11646     11647        Men
11647       11647     11648        Men
11648       11648     11649        Men

[11649 rows x 3 columns]
```

[43]:
```
s.head(50)
```

[43]:
```
    Unnamed: 0  Image_id prediction
0            0         1      Women
1            1         2      Women
2            2         3        Men
3            3         4      Women
4            4         5      Women
5            5         6      Women
6            6         7        Men
7            7         8      Women
8            8         9      Women
9            9        10      Women
10          10        11      Women
11          11        12      Women
12          12        13      Women
13          13        14      Women
14          14        15      Women
15          15        16      Women
16          16        17      Women
17          17        18      Women
```

```
18          18        19     Women
19          19        20     Women
20          20        21       Men
21          21        22     Women
22          22        23     Women
23          23        24     Women
24          24        25     Women
25          25        26     Women
26          26        27     Women
27          27        28     Women
28          28        29     Women
29          29        30     Women
30          30        31     Women
31          31        32     Women
32          32        33     Women
33          33        34     Women
34          34        35     Women
35          35        36     Women
36          36        37     Women
37          37        38     Women
38          38        39       Men
39          39        40     Women
40          40        41     Women
41          41        42     Women
42          42        43     Women
43          43        44     Women
44          44        45     Women
45          45        46     Women
46          46        47     Women
47          47        48     Women
48          48        49     Women
49          49        50     Women
```

[44]:

```
s.tail(50)
```

[44]:

```
       Unnamed: 0  Image_id prediction
11599       11599     11600        Men
11600       11600     11601        Men
11601       11601     11602        Men
11602       11602     11603        Men
11603       11603     11604        Men
11604       11604     11605        Men
11605       11605     11606        Men
11606       11606     11607        Men
11607       11607     11608        Men
11608       11608     11609        Men
11609       11609     11610        Men
```

```
11610       11610     11611        Men
11611       11611     11612        Men
11612       11612     11613        Men
11613       11613     11614        Men
11614       11614     11615        Men
11615       11615     11616        Men
11616       11616     11617      Women
11617       11617     11618        Men
11618       11618     11619        Men
11619       11619     11620        Men
11620       11620     11621        Men
11621       11621     11622        Men
11622       11622     11623        Men
11623       11623     11624        Men
11624       11624     11625        Men
11625       11625     11626        Men
11626       11626     11627        Men
11627       11627     11628        Men
11628       11628     11629        Men
11629       11629     11630        Men
11630       11630     11631        Men
11631       11631     11632        Men
11632       11632     11633        Men
11633       11633     11634        Men
11634       11634     11635        Men
11635       11635     11636        Men
11636       11636     11637        Men
11637       11637     11638        Men
11638       11638     11639        Men
11639       11639     11640      Women
11640       11640     11641        Men
11641       11641     11642        Men
11642       11642     11643        Men
11643       11643     11644        Men
11644       11644     11645        Men
11645       11645     11646        Men
11646       11646     11647        Men
11647       11647     11648        Men
11648       11648     11649        Men
```

[45]:

```
#predicting whole training data set
```

[46]:

```
directroy=r"/content/Training"
Catogeries=["female","male"]
```

[47]:

```
size=224
data=[]
```

```
i=0
data=pd.read_csv("/content/training_pred.csv")
l_pred=list()
l_imgid=list()
d={}
```

[ ]:

```
for catogery in Catogeries:
  folder=os.path.join(directroy,catogery)
  for img in os.listdir(folder):
    img_path=os.path.join(folder,img)
    img_arr=cv2.imread(img_path)
    img_arr=cv2.resize(img_arr,(size,size))
    img_arr=img_to_array(img_arr)
    img_arr=img_arr/255.0
    img_arr=img_arr.reshape(1,size,size,3)
    res=model.predict(img_arr)
    i=i+1
    if res[0][0]<0.5:
      l_imgid.append(i)
      l_pred.append("Women")
    else:
      l_imgid.append(i)
      l_pred.append("Men")
```

[49]:

```
#saving to csv file
```

[50]:

```
d1=pd.DataFrame({"Image_id":l_imgid,"prediction":l_pred})
d1.to_csv("/content/training_pred.csv")
s1=pd.read_csv("/content/training_pred.csv")
s1
```

[50]:

```
       Unnamed: 0  Image_id prediction
0               0         1      Women
1               1         2      Women
2               2         3      Women
3               3         4        Men
4               4         5      Women
…            …        …        …
47004       47004     47005        Men
47005       47005     47006        Men
47006       47006     47007        Men
47007       47007     47008        Men
47008       47008     47009        Men

[47009 rows x 3 columns]
```

[51]:

```
s1.head(50)
```

```
[51]:     Unnamed: 0  Image_id prediction
      0            0         1      Women
      1            1         2      Women
      2            2         3      Women
      3            3         4        Men
      4            4         5      Women
      5            5         6      Women
      6            6         7      Women
      7            7         8      Women
      8            8         9      Women
      9            9        10      Women
      10          10        11      Women
      11          11        12      Women
      12          12        13      Women
      13          13        14      Women
      14          14        15      Women
      15          15        16      Women
      16          16        17      Women
      17          17        18      Women
      18          18        19      Women
      19          19        20      Women
      20          20        21      Women
      21          21        22      Women
      22          22        23      Women
      23          23        24      Women
      24          24        25      Women
      25          25        26      Women
      26          26        27        Men
      27          27        28      Women
      28          28        29      Women
      29          29        30      Women
      30          30        31      Women
      31          31        32      Women
      32          32        33      Women
      33          33        34      Women
      34          34        35      Women
      35          35        36      Women
      36          36        37      Women
      37          37        38      Women
      38          38        39      Women
      39          39        40      Women
      40          40        41      Women
      41          41        42      Women
      42          42        43      Women
      43          43        44      Women
      44          44        45        Men
      45          45        46      Women
```

```
46          46        47         Men
47          47        48         Men
48          48        49       Women
49          49        50       Women
```

[52]:

```
s1.tail(50)
```

[52]:

```
       Unnamed: 0  Image_id prediction
46959       46959     46960        Men
46960       46960     46961        Men
46961       46961     46962        Men
46962       46962     46963        Men
46963       46963     46964        Men
46964       46964     46965        Men
46965       46965     46966        Men
46966       46966     46967        Men
46967       46967     46968        Men
46968       46968     46969        Men
46969       46969     46970        Men
46970       46970     46971        Men
46971       46971     46972        Men
46972       46972     46973        Men
46973       46973     46974        Men
46974       46974     46975      Women
46975       46975     46976        Men
46976       46976     46977        Men
46977       46977     46978        Men
46978       46978     46979        Men
46979       46979     46980        Men
46980       46980     46981        Men
46981       46981     46982        Men
46982       46982     46983        Men
46983       46983     46984        Men
46984       46984     46985        Men
46985       46985     46986      Women
46986       46986     46987        Men
46987       46987     46988      Women
46988       46988     46989        Men
46989       46989     46990        Men
46990       46990     46991        Men
46991       46991     46992        Men
46992       46992     46993        Men
46993       46993     46994        Men
46994       46994     46995        Men
46995       46995     46996        Men
46996       46996     46997        Men
46997       46997     46998        Men
```

```
46998       46998     46999      Women
46999       46999     47000        Men
47000       47000     47001        Men
47001       47001     47002        Men
47002       47002     47003        Men
47003       47003     47004        Men
47004       47004     47005        Men
47005       47005     47006        Men
47006       47006     47007        Men
47007       47007     47008        Men
47008       47008     47009        Men
```