Case Study: Recognizing Related Faces

Defining our goal:

Do you have your father's nose sitting on you?

Blood relatives often share facial features. Now researchers at Northeastern University want to improve their algorithm for facial image classification to bridge the gap between research and other familial markers like DNA results. This technology remains largely unseen in practice for a couple of reasons:

1. Existing image databases for kinship recognition tasks aren't large enough to capture and reflect the true data distributions of the families of the world.
2. Many hidden factors affect familial facial relationships, so a more discriminant model is needed than the computer vision algorithms used most often for higher-level categorizations (e.g. facial recognition or object classification).

So, we will be building a complex model by determining if two people are blood-related or not based solely on images of their faces.

Credits: Kaggle

Data

The data can be downloaded from the given link:

https://www.kaggle.com/c/recognizing-faces-in-the-wild/data (https://www.kaggle.com/c/recognizing-faces-in-the-wild/data)

We will be using data given by Families In the Wild (FIW), the largest and most comprehensive image database for automatic kinship recognition.

FIW's dataset is obtained from publicly available images from celebrities. For more information about their labeling process, please visit their database page.

File Description:

The folder 'train' consists of subfolders of families with names (F0123), then these family folder contains subfolders for individuals (MIDx). Images in the same MIDx folder belong to the same person. Images in the same F0123 folder belong to the same family.

The folder 'test' contains images of faces that need to be tested with some another random image to be kin related or not.

The file 'train_relationships.csv' shown below contains training labels. Remember, not every individual in a family shares a kinship relationship.

For example, a mother and father are kin to their children, but not to each other.

Type of Problem:

It is a binary classification problem.We will solve it using deep learning approach.

Performance Metric:

We will be using roc-auc score to finaly get score of how our model is performing.

Importing required libraries:

In [0]:
```
#%tensorflow_version 2.x
import pandas as pd
import numpy as np
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from PIL import Image
import os
from random import choice, sample
import cv2
from imageio import imread
from keras.preprocessing.text import Tokenizer, one_hot
from keras.preprocessing.sequence import pad_sequences
from keras.models import Model, load_model
from keras import regularizers
from keras.layers import Input, Embedding, LSTM, Dropout, BatchNormalization,D
ense, concatenate, Flatten, Conv1D
from keras.optimizers import RMSprop, Adam
import warnings
warnings.filterwarnings("ignore")
%matplotlib inline
```

```
Using TensorFlow backend.
```

The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.
We recommend you upgrade (https://www.tensorflow.org/guide/migrate) now or ensure your notebook will continue to use TensorFlow 1.x via the `%tensorflow_version 1.x` magic: more info (https://colab.research.google.com/notebooks/tensorflow_version.ipynb).

In [0]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client
_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&
redirect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%3Aoob&scope=email%20https%3A%2F%2F
www.googleapis.com%2Fauth%2Fdocs.test%20https%3A%2F%2Fwww.googleapis.com%2Fau
th%2Fdrive%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive.photos.readonly%
20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fpeopleapi.readonly&response_type=
code

Enter your authorization code:
··········
Mounted at /content/drive
```

In [0]:
```
%cd /content/drive/My Drive/recognizing-faces-in-the-wild
```

```
/content/drive/My Drive/recognizing-faces-in-the-wild
```

In [0]:
```
#installing keras_vggface model
!pip install git+https://github.com/rcmalli/keras-vggface.git
```

```
Collecting git+https://github.com/rcmalli/keras-vggface.git
  Cloning https://github.com/rcmalli/keras-vggface.git to /tmp/pip-req-build-
pg_vu2w5
  Running command git clone -q https://github.com/rcmalli/keras-vggface.git /
tmp/pip-req-build-pg_vu2w5
Requirement already satisfied: numpy>=1.9.1 in /usr/local/lib/python3.6/dist-
packages (from keras-vggface==0.6) (1.17.4)
Requirement already satisfied: scipy>=0.14 in /usr/local/lib/python3.6/dist-p
ackages (from keras-vggface==0.6) (1.3.2)
Requirement already satisfied: h5py in /usr/local/lib/python3.6/dist-packages
(from keras-vggface==0.6) (2.8.0)
Requirement already satisfied: pillow in /usr/local/lib/python3.6/dist-packag
es (from keras-vggface==0.6) (4.3.0)
Requirement already satisfied: keras in /usr/local/lib/python3.6/dist-package
s (from keras-vggface==0.6) (2.2.5)
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.6/dist-pa
ckages (from keras-vggface==0.6) (1.12.0)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.6/dist-packag
es (from keras-vggface==0.6) (3.13)
Requirement already satisfied: olefile in /usr/local/lib/python3.6/dist-packa
ges (from pillow->keras-vggface==0.6) (0.46)
Requirement already satisfied: keras-applications>=1.0.8 in /usr/local/lib/py
thon3.6/dist-packages (from keras->keras-vggface==0.6) (1.0.8)
Requirement already satisfied: keras-preprocessing>=1.1.0 in /usr/local/lib/p
ython3.6/dist-packages (from keras->keras-vggface==0.6) (1.1.0)
Building wheels for collected packages: keras-vggface
  Building wheel for keras-vggface (setup.py) ... done
  Created wheel for keras-vggface: filename=keras_vggface-0.6-cp36-none-any.w
hl size=8311 sha256=3e13991b1d9292c329775e1db98c441bb53c4ef319bf6715c8ea33efc
95ff90b
  Stored in directory: /tmp/pip-ephem-wheel-cache-eddayukp/wheels/36/07/46/06
c25ce8e9cd396dabe151ea1d8a2bc28dafcb11321c1f3a6d
Successfully built keras-vggface
Installing collected packages: keras-vggface
Successfully installed keras-vggface-0.6
```

In [0]:
```
from keras_vggface.vggface import VGGFace
from glob import glob
from keras import backend as K
from keras.preprocessing import image
from keras.layers import Input, Dense, Flatten, GlobalMaxPool2D, GlobalAvgPool
2D, Concatenate, Multiply, Dropout, Subtract, Add, Conv2D, Lambda, Reshape
from collections import defaultdict
from keras_vggface.utils import preprocess_input
```

EDA

Diving into the data folders and analyzing the train_relationship.csv file, I found some hiccups.

Ex: In the train_relationship.csv file there is a relation between 'F0039/MID1' and 'F0039/MID3', but there is no such folder for 'F0039/MID3' in the train folder.

I can see some similar issues because of the absence of the following folders

F0039/MID4

F0041/MID5

F0041/MID7

F0051/MID5

… and more.

One of the simple solutions to the above problem is to ignore these empty directories and only consider the ones which are available to us.

In [0]:
```
TRAIN_BASE = 'train'
families = sorted(os.listdir(TRAIN_BASE))
print('We have {} families in the dataset'.format(len(families)))
print(families[:5])
```

```
We have 470 families in the dataset
['F0002', 'F0005', 'F0009', 'F0010', 'F0016']
```

In [0]:
```
all_images = glob(TRAIN_BASE + "*/*/*/*.jpg")
```

In [0]:
```
#folders with name F09 will be our validation dataset and the rest will be in
 train dataset
val_families = "F09"
train_images = [x for x in all_images if val_families not in x]
val_images = [x for x in all_images if val_families in x]
```

In [0]:
```
ppl = [x.split("/")[-3] + "/" + x.split("/")[-2] for x in all_images]
```

In [0]:
```
#preparing train and test dataset
train_person_to_images_map = defaultdict(list)

for x in train_images:
    train_person_to_images_map[x.split("/")[-3] + "/" + x.split("/")[-2]].appe
nd(x)

val_person_to_images_map = defaultdict(list)

for x in val_images:
    val_person_to_images_map[x.split("/")[-3] + "/" + x.split("/")[-2]].append
(x)

relationships = pd.read_csv('train_relationships.csv')
relationships = list(zip(relationships.p1.values, relationships.p2.values))
relationships = [x for x in relationships if x[0] in ppl and x[1] in ppl]

train = [x for x in relationships if val_families not in x[0]]
val = [x for x in relationships if val_families in x[0]]
```

Visualizing the dataset:

In [0]:
```
rel=pd.read_csv('train_relationships.csv')

def load_img(PATH):
    return np.array(Image.open(PATH))

def plot_relations(df, BASE='train', rows=1, titles=None):
    tdf = df[:rows]
    tdf1 = tdf.p1
    tdf2 = tdf.p2
    figsize=(5,3*rows)
    f = plt.figure(figsize=figsize)
    x = 0
    for i in range(rows):
        sp = f.add_subplot(rows, 2, x+1)
        sp.axis('Off')
        x+=1
        image_path = os.path.join(BASE,tdf1[i])
        im = os.listdir(image_path)[-1]
        sp.set_title(tdf1[i], fontsize=16)
        plt.imshow(load_img(os.path.join(image_path, im)))
        sp = f.add_subplot(rows, 2, x+1)
        x+=1
        sp.axis('Off')
        image_path = os.path.join(BASE,tdf2[i])
        im = os.listdir(image_path)[-1]
        sp.set_title(tdf2[i], fontsize=16)
        plt.imshow(load_img(os.path.join(image_path, im)))

plot_relations(rel, rows=10)
```

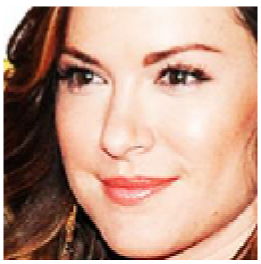
F0002/MID1

F0002/MID3

F0002/MID2

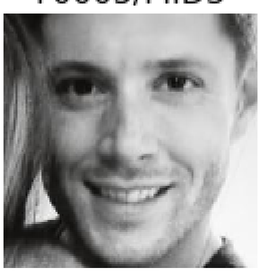
F0002/MID3

F0005/MID1

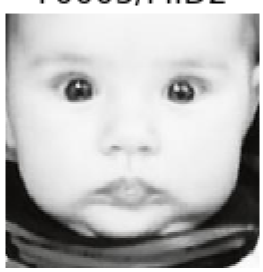


F0005/MID2

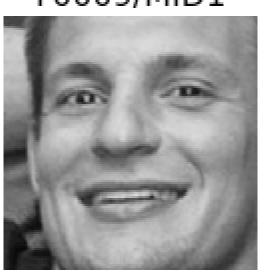
F0005/MID3



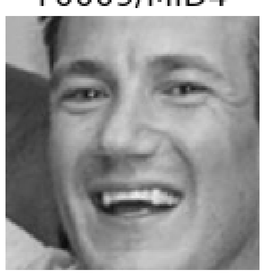
F0005/MID2



F0009/MID1

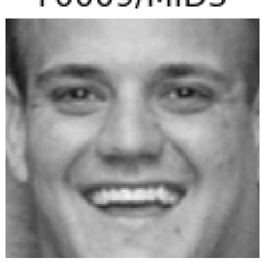


F0009/MID4



F0009/MID1

F0009/MID3




F0009/MID1
F0009/MID2
F0009/MID1
F0009/MID6
F0009/MID2
F0009/MID4
F0009/MID2
F0009/MID6

In [0]:
```
#Image preprocessing step
def prewhiten(x):
    """This function takes the image and applies stadardization as preproceesi
ng step"""
    if x.ndim == 4:
        axis = (1, 2, 3)
        size = x[0].size
    elif x.ndim == 3:
        axis = (0, 1, 2)
        size = x.size
    else:
        raise ValueError('Dimension should be 3 or 4')

    mean = np.mean(x, axis=axis, keepdims=True)
    std = np.std(x, axis=axis, keepdims=True)
    std_adj = np.maximum(std, 1.0/np.sqrt(size))
    y = (x - mean) / std_adj
    return y

#https://stackoverflow.com/questions/41032551/how-to-compute-receiving-operati
ng-characteristic-roc-and-auc-in-keras

import tensorflow as tf
from sklearn.metrics import roc_auc_score


def auc(y_true, y_pred):
    auc = tf.metrics.auc(y_true, y_pred)[1]
    K.get_session().run(tf.local_variables_initializer())
    return auc
```

In [0]:
```
#Loading facenet model
model_path = 'keras-facenet/model/facenet_keras.h5'
facenet_model = load_model(model_path)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:541: The name tf.placeholder is deprecated. Please use
tf.compat.v1.placeholder instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. Please
use tf.random.uniform instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:66: The name tf.get_default_graph is deprecated. Please
use tf.compat.v1.get_default_graph instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:190: The name tf.get_default_session is deprecated. Ple
ase use tf.compat.v1.get_default_session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:197: The name tf.ConfigProto is deprecated. Please use
tf.compat.v1.ConfigProto instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:203: The name tf.Session is deprecated. Please use tf.c
ompat.v1.Session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:207: The name tf.global_variables is deprecated. Please
use tf.compat.v1.global_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:216: The name tf.is_variable_initialized is deprecated.
Please use tf.compat.v1.is_variable_initialized instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:223: The name tf.variables_initializer is deprecated. P
lease use tf.compat.v1.variables_initializer instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:2041: The name tf.nn.fused_batch_norm is deprecated. Pl
ease use tf.compat.v1.nn.fused_batch_norm instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:148: The name tf.placeholder_with_default is deprecate
d. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:4267: The name tf.nn.max_pool is deprecated. Please use
tf.nn.max_pool2d instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:3733: calling dropout (from tensorflow.python.ops.nn_op
s) with keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - k
eep_prob`.
```

Building deep learning models.

Instead of creating one best single model, we will use the power of ensebling models to get better results.

We will use 4 different models for this problem and finally took the avg of these 4 to test it on final test dataset.

Model 1:

1-For the first model we will create face embeddings using facenet and vgg16 architecture.

2-We have face embedding by facnet for image 1 and image 2 and face embedding by vgg16 for image 1 and image 2.

3- We will blend these embeddings using airthmetic operations to capture more features for the face in the image.

In [0]:
```
#Facenet architecture will take image of size 160 x 160
IMG_SIZE_FN = 160
#Facenet architecture will take image of size 224 x 224
IMG_SIZE_VGG = 224
```

In [0]:
```
#We will train full network except the last 3 layers
for layer in facenet_model.layers[:-3]:
    layer.trainable = True

#We will train full network except the last 3 layers
vgg_model = VGGFace(model='resnet50', include_top=False)
for layer in vgg_model.layers[:-3]:
    layer.trainable = True
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:4271: The name tf.nn.avg_pool is deprecated. Please use
tf.nn.avg_pool2d instead.

Downloading data from https://github.com/rcmalli/keras-vggface/releases/downl
oad/v2.0/rcmalli_vggface_tf_notop_resnet50.h5
94699520/94694792 [==============================] - 1s 0us/step
```

In [0]:
```
def read_img_fn(path):
    """this function will read image from specified path and convert it into s
ize of 160 for facenet"""
    img = cv2.imread(path)
    img = cv2.resize(img,(IMG_SIZE_FN,IMG_SIZE_FN))
    img = np.array(img).astype(np.float)
    return prewhiten(img)

def read_img_vgg(path):
    """this function will read image from specified path and convert it into s
ize of 224 for VGGFace"""
    img = cv2.imread(path)
    img = cv2.resize(img,(IMG_SIZE_VGG,IMG_SIZE_VGG))
    img = np.array(img).astype(np.float)
    return preprocess_input(img, version=2)


def gen(list_tuples, person_to_images_map, batch_size=16):
    """generator funtion will generate images in the right format while traini
ng the model """
    ppl = list(person_to_images_map.keys())
    while True:
        batch_tuples = sample(list_tuples, batch_size // 2)
        labels = [1] * len(batch_tuples)
        while len(batch_tuples) < batch_size:
            p1 = choice(ppl)
            p2 = choice(ppl)

            if p1 != p2 and (p1, p2) not in list_tuples and (p2, p1) not in li
st_tuples:
                batch_tuples.append((p1, p2))
                labels.append(0)

        for x in batch_tuples:
            if not len(person_to_images_map[x[0]]):
                print(x[0])

        X1 = [choice(person_to_images_map[x[0]]) for x in batch_tuples]
        X1_FN = np.array([read_img_fn(x) for x in X1])
        X1_VGG = np.array([read_img_vgg(x) for x in X1])

        X2 = [choice(person_to_images_map[x[1]]) for x in batch_tuples]
        X2_FN = np.array([read_img_fn(x) for x in X2])
        X2_VGG = np.array([read_img_vgg(x) for x in X2])

        yield [X1_FN, X2_FN, X1_VGG, X2_VGG], labels
```

In [0]:
```
valx=gen(val, val_person_to_images_map, batch_size=100)
```

In [0]:
```
for i in valx:
    valx=i
    break
```

In [0]:
```
#this model takes four inputs
input_1 = Input(shape=(IMG_SIZE_FN, IMG_SIZE_FN, 3))        #facenet for Image 1
input_2 = Input(shape=(IMG_SIZE_FN, IMG_SIZE_FN, 3))        #facenet for image 2
input_3 = Input(shape=(IMG_SIZE_VGG, IMG_SIZE_VGG, 3))      #VGG for image 1
input_4 = Input(shape=(IMG_SIZE_VGG, IMG_SIZE_VGG, 3))      #VGG for image 2

fn_1 = facenet_model(input_1)
fn_2 = facenet_model(input_2)
vgg_1 = vgg_model(input_3)
vgg_2 = vgg_model(input_4)

x1 = Reshape((1, 1 ,128))(fn_1)   #reshaping image array for global max pool layer
x2 = Reshape((1, 1 ,128))(fn_2)
x1 = Concatenate(axis=-1)([GlobalMaxPool2D()(x1), GlobalAvgPool2D()(x1)])
x2 = Concatenate(axis=-1)([GlobalMaxPool2D()(x2), GlobalAvgPool2D()(x2)])

#For simple, stateless custom operations, we can use Lambda layers
#the below 4 lamda functions will calcluate the square of each input image
lambda_1 = Lambda(lambda tensor  : K.square(tensor))(fn_1)
lambda_2 = Lambda(lambda tensor  : K.square(tensor))(fn_2)
lambda_3 = Lambda(lambda tensor  : K.square(tensor))(vgg_1)
lambda_4 = Lambda(lambda tensor  : K.square(tensor))(vgg_2)

added_facenet = Add()([x1, x2])     #this function will add two images image 1 image 2 given by facenet architecture
added_vgg = Add()([vgg_1, vgg_2])    #this function will add two images image 3 image 4 given by VGG architecture
subtract_fn = Subtract()([x1,x2])    #this function will subtract two images image 1 image 2 given by facenet architecture
subtract_vgg = Subtract()([vgg_1,vgg_2])   #this function will subtract two images image 3 image 4 given by VGG architecture
subtract_fn2 = Subtract()([x2,x1])    #this function will subtract two images image 2 image 1 given by facenet architecture
subtract_vgg2 = Subtract()([vgg_2,vgg_1])   #this function will subtract two images image 4 image 3 given by VGG architecture
prduct_fn1 = Multiply()([x1,x2])    #this function will multiply two images image 1 image 2 given by facenet architecture
prduct_vgg1 = Multiply()([vgg_1,vgg_2])   #this function will multiply two images image 3 image 4 given by VGG architecture
sqrt_fn1 = Add()([lambda_1,lambda_2])        # this function implements x1^2 + x2^2 where x1 and x2 are image by facenet
sqrt_vgg1 = Add()([lambda_3,lambda_4])       # this function implements vgg_1^2 + vgg_2^2 where vgg_1 and vgg_2 are image by VGG
sqrt_fn2 = Lambda(lambda tensor  : K.sign(tensor)*K.sqrt(K.abs(tensor)+1e-9))(prduct_fn1) #squre_root of sqrt_fn1
sqrt_vgg2 = Lambda(lambda tensor  : K.sign(tensor)*K.sqrt(K.abs(tensor)+1e-9))(prduct_vgg1) #squre_root of sqrt_vgg1


added_vgg = Conv2D(128 , [1,1] )(added_vgg)
subtract_vgg = Conv2D(128 , [1,1] )(subtract_vgg)
subtract_vgg2 = Conv2D(128 , [1,1] )(subtract_vgg2)
prduct_vgg1 = Conv2D(128 , [1,1] )(prduct_vgg1)
```

```
sqrt_vgg1 = Conv2D(128 , [1,1] )(sqrt_vgg1)
sqrt_vgg2 = Conv2D(128 , [1,1] )(sqrt_vgg2)

#finally concatenating all the above featues for final layer which is to be in
puted to the dense layers.
concatenated= Concatenate(axis=-1)([Flatten()(added_vgg), (added_facenet), Fla
tten()(subtract_vgg), (subtract_fn),
                                   Flatten()(subtract_vgg2), (subtract_fn2), F
latten()(prduct_vgg1), (prduct_fn1),
                                   Flatten()(sqrt_vgg1), (sqrt_fn1), Flatten()
(sqrt_vgg2), (sqrt_fn2)])

concatenated= Dense(500, activation="relu")(concatenated)
concatenated= Dropout(0.1)(concatenated)
concatenated= Dense(100, activation="relu")(concatenated)
concatenated= Dropout(0.1)(concatenated)
concatenated= Dense(25, activation="relu")(concatenated)
concatenated= Dropout(0.1)(concatenated)
out = Dense(1, activation="sigmoid")(concatenated) #output sigmoid layer

#defining the model
model = Model([input_1, input_2, input_3, input_4], out)
```

In [0]:
```
model.compile(loss="binary_crossentropy", metrics=[auc], optimizer=Adam(1e-5))

model.summary()
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimize
rs.py:793: The name tf.train.Optimizer is deprecated. Please use tf.compat.v
1.train.Optimizer instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:3657: The name tf.log is deprecated. Please use tf.mat
h.log instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_cor
e/python/ops/nn_impl.py:183: where (from tensorflow.python.ops.array_ops) is
deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_cor
e/python/ops/metrics_impl.py:808: div (from tensorflow.python.ops.math_ops) i
s deprecated and will be removed in a future version.
Instructions for updating:
Deprecated in favor of operator or tf.math.divide.
Model: "model_1"
_____________________________________________________________________________
_____________________
Layer (type)                    Output Shape         Param #     Connected to
=============================================================================
=====================
input_2 (InputLayer)            (None, 160, 160, 3)  0
_____________________________________________________________________________
_____________________
input_3 (InputLayer)            (None, 160, 160, 3)  0
_____________________________________________________________________________
_____________________
input_4 (InputLayer)            (None, 224, 224, 3)  0
_____________________________________________________________________________
_____________________
input_5 (InputLayer)            (None, 224, 224, 3)  0
_____________________________________________________________________________
_____________________
inception_resnet_v1 (Model)     (None, 128)          22808144    input_2[0]
[0]
                                                                 input_3[0]
[0]
_____________________________________________________________________________
_____________________
vggface_resnet50 (Model)        multiple             23561152    input_4[0]
[0]
                                                                 input_5[0]
[0]
_____________________________________________________________________________
_____________________
reshape_1 (Reshape)             (None, 1, 1, 128)    0           inception_re
snet_v1[1][0]
_____________________________________________________________________________
_____________________
reshape_2 (Reshape)             (None, 1, 1, 128)    0           inception_re
snet_v1[2][0]
_____________________________________________________________________________
_____________________
global_max_pooling2d_1 (GlobalM (None, 128)          0           reshape_1[0]
```

```
[0]
__________________________________________________________________________________________________
global_average_pooling2d_1 (Glo (None, 128)          0           reshape_1[0]
[0]
__________________________________________________________________________________________________
global_max_pooling2d_2 (GlobalM (None, 128)          0           reshape_2[0]
[0]
__________________________________________________________________________________________________
global_average_pooling2d_2 (Glo (None, 128)          0           reshape_2[0]
[0]
__________________________________________________________________________________________________
multiply_2 (Multiply)           (None, 1, 1, 2048)   0           vggface_resn
et50[1][0]
                                                                 vggface_resn
et50[2][0]
__________________________________________________________________________________________________
lambda_3 (Lambda)               (None, 1, 1, 2048)   0           vggface_resn
et50[1][0]
__________________________________________________________________________________________________
lambda_4 (Lambda)               (None, 1, 1, 2048)   0           vggface_resn
et50[2][0]
__________________________________________________________________________________________________
add_18 (Add)                    (None, 1, 1, 2048)   0           vggface_resn
et50[1][0]
                                                                 vggface_resn
et50[2][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, 256)          0           global_max_p
ooling2d_1[0][0]
                                                                 global_avera
ge_pooling2d_1[0][0]
__________________________________________________________________________________________________
concatenate_2 (Concatenate)     (None, 256)          0           global_max_p
ooling2d_2[0][0]
                                                                 global_avera
ge_pooling2d_2[0][0]
__________________________________________________________________________________________________
subtract_2 (Subtract)           (None, 1, 1, 2048)   0           vggface_resn
et50[1][0]
                                                                 vggface_resn
et50[2][0]
__________________________________________________________________________________________________
subtract_4 (Subtract)           (None, 1, 1, 2048)   0           vggface_resn
et50[2][0]
                                                                 vggface_resn
et50[1][0]
```

```
________________________________________________________________________________

add_20 (Add)                    (None, 1, 1, 2048)   0           lambda_3[0]
[0]
                                                                 lambda_4[0]
[0]
________________________________________________________________________________

lambda_6 (Lambda)               (None, 1, 1, 2048)   0           multiply_2
[0][0]
________________________________________________________________________________

conv2d_1 (Conv2D)               (None, 1, 1, 128)    262272      add_18[0][0]
________________________________________________________________________________

conv2d_2 (Conv2D)               (None, 1, 1, 128)    262272      subtract_2
[0][0]
________________________________________________________________________________

conv2d_3 (Conv2D)               (None, 1, 1, 128)    262272      subtract_4
[0][0]
________________________________________________________________________________

conv2d_4 (Conv2D)               (None, 1, 1, 128)    262272      multiply_2
[0][0]
________________________________________________________________________________

multiply_1 (Multiply)           (None, 256)          0           concatenate_
1[0][0]
                                                                 concatenate_
2[0][0]
________________________________________________________________________________

conv2d_5 (Conv2D)               (None, 1, 1, 128)    262272      add_20[0][0]
________________________________________________________________________________

lambda_1 (Lambda)               (None, 128)          0           inception_re
snet_v1[1][0]
________________________________________________________________________________

lambda_2 (Lambda)               (None, 128)          0           inception_re
snet_v1[2][0]
________________________________________________________________________________

conv2d_6 (Conv2D)               (None, 1, 1, 128)    262272      lambda_6[0]
[0]
________________________________________________________________________________

flatten_1 (Flatten)             (None, 128)          0           conv2d_1[0]
[0]
________________________________________________________________________________

add_17 (Add)                    (None, 256)          0           concatenate_
1[0][0]
                                                                 concatenate_
2[0][0]
________________________________________________________________________________
```

```
________________________________
flatten_2 (Flatten)             (None, 128)          0           conv2d_2[0]
[0]
__________________________________________________________________________________________________
________________________________
subtract_1 (Subtract)           (None, 256)          0           concatenate_
1[0][0]
                                                                 concatenate_
2[0][0]
__________________________________________________________________________________________________
________________________________
flatten_3 (Flatten)             (None, 128)          0           conv2d_3[0]
[0]
__________________________________________________________________________________________________
________________________________
subtract_3 (Subtract)           (None, 256)          0           concatenate_
2[0][0]
                                                                 concatenate_
1[0][0]
__________________________________________________________________________________________________
________________________________
flatten_4 (Flatten)             (None, 128)          0           conv2d_4[0]
[0]
__________________________________________________________________________________________________
________________________________
flatten_5 (Flatten)             (None, 128)          0           conv2d_5[0]
[0]
__________________________________________________________________________________________________
________________________________
add_19 (Add)                    (None, 128)          0           lambda_1[0]
[0]
                                                                 lambda_2[0]
[0]
__________________________________________________________________________________________________
________________________________
flatten_6 (Flatten)             (None, 128)          0           conv2d_6[0]
[0]
__________________________________________________________________________________________________
________________________________
lambda_5 (Lambda)               (None, 256)          0           multiply_1
[0][0]
__________________________________________________________________________________________________
________________________________
concatenate_3 (Concatenate)     (None, 2176)         0           flatten_1[0]
[0]
                                                                 add_17[0][0]
                                                                 flatten_2[0]
[0]
                                                                 subtract_1
[0][0]
                                                                 flatten_3[0]
[0]
                                                                 subtract_3
[0][0]
                                                                 flatten_4[0]
[0]
                                                                 multiply_1
```

```
[0][0]
                                                                flatten_5[0]
[0]
                                                                add_19[0][0]
                                                                flatten_6[0]
[0]
                                                                lambda_5[0]
[0]
____________________________________________________________________________
_____________________
dense_1 (Dense)                 (None, 500)          1088500     concatenate_
3[0][0]
____________________________________________________________________________
_____________________
dropout_1 (Dropout)             (None, 500)          0           dense_1[0]
[0]
____________________________________________________________________________
_____________________
dense_2 (Dense)                 (None, 100)          50100       dropout_1[0]
[0]
____________________________________________________________________________
_____________________
dropout_2 (Dropout)             (None, 100)          0           dense_2[0]
[0]
____________________________________________________________________________
_____________________
dense_3 (Dense)                 (None, 25)           2525        dropout_2[0]
[0]
____________________________________________________________________________
_____________________
dropout_3 (Dropout)             (None, 25)           0           dense_3[0]
[0]
____________________________________________________________________________
_____________________
dense_4 (Dense)                 (None, 1)            26          dropout_3[0]
[0]
============================================================================
=====================
Total params: 49,084,079
Trainable params: 49,002,127
Non-trainable params: 81,952
____________________________________________________________________________
_____________________
```

Training the model and saving it with name facenet_vgg.h5

In [0]:
```
import datetime
from keras.callbacks import TensorBoard,EarlyStopping, ModelCheckpoint, Reduce
LROnPlateau

# Clear any logs from previous runs
!rm -rf ./logs/

log_dir="logs"
tensorboard_callback = TensorBoard(log_dir=log_dir, histogram_freq=1)

es = EarlyStopping(monitor='val_auc', mode='max', verbose=1, patience=10)

checkpoint = ModelCheckpoint('new facevgg.h5', monitor='val_auc', verbose=1, s
ave_best_only=True, mode='max')

reduce_on_plateau = ReduceLROnPlateau(monitor="val_auc", mode="max", factor=0.
1, patience=20, verbose=1)

callbacks_list = [tensorboard_callback, checkpoint, reduce_on_plateau, es]

history = model.fit_generator(gen(train, train_person_to_images_map, batch_siz
e=16), use_multiprocessing=True,
                    validation_data=(valx[0],valx[1]), epochs=50, verbose=1,
                    workers = 4,callbacks=callbacks_list, steps_per_epoch=200)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:1033: The name tf.assign_add is deprecated. Please use
tf.compat.v1.assign_add instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:1020: The name tf.assign is deprecated. Please use tf.c
ompat.v1.assign instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1120: The name tf.summary.histogram is deprecated. Please use tf.compat.
v1.summary.histogram instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1122: The name tf.summary.merge_all is deprecated. Please use tf.compat.
v1.summary.merge_all instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1125: The name tf.summary.FileWriter is deprecated. Please use tf.compa
t.v1.summary.FileWriter instead.

Epoch 1/50
200/200 [==============================] - 967s 5s/step - loss: 1.6720 - auc:
0.5157 - val_loss: 1.5008 - val_auc: 0.5242
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1265: The name tf.Summary is deprecated. Please use tf.compat.v1.Summary
instead.


Epoch 00001: val_auc improved from -inf to 0.52424, saving model to new facev
gg.h5
Epoch 2/50
200/200 [==============================] - 237s 1s/step - loss: 0.9823 - auc:
0.5354 - val_loss: 0.9698 - val_auc: 0.5443

Epoch 00002: val_auc improved from 0.52424 to 0.54433, saving model to new fa
cevgg.h5
Epoch 3/50
200/200 [==============================] - 237s 1s/step - loss: 0.7913 - auc:
0.5507 - val_loss: 0.8984 - val_auc: 0.5566

Epoch 00003: val_auc improved from 0.54433 to 0.55657, saving model to new fa
cevgg.h5
Epoch 4/50
200/200 [==============================] - 118s 588ms/step - loss: 0.6989 - a
uc: 0.5628 - val_loss: 0.7348 - val_auc: 0.5704

Epoch 00004: val_auc improved from 0.55657 to 0.57035, saving model to new fa
cevgg.h5
Epoch 5/50
200/200 [==============================] - 131s 657ms/step - loss: 0.6563 - a
uc: 0.5790 - val_loss: 0.7023 - val_auc: 0.5872

Epoch 00005: val_auc improved from 0.57035 to 0.58725, saving model to new fa
cevgg.h5
Epoch 6/50
200/200 [==============================] - 176s 878ms/step - loss: 0.6352 - a
uc: 0.5952 - val_loss: 0.6555 - val_auc: 0.6029
```

```
Epoch 00006: val_auc improved from 0.58725 to 0.60288, saving model to new fa
cevgg.h5
Epoch 7/50
200/200 [==============================] - 108s 541ms/step - loss: 0.6173 - a
uc: 0.6108 - val_loss: 0.6117 - val_auc: 0.6176

Epoch 00007: val_auc improved from 0.60288 to 0.61759, saving model to new fa
cevgg.h5
Epoch 8/50
200/200 [==============================] - 126s 628ms/step - loss: 0.5994 - a
uc: 0.6242 - val_loss: 0.6012 - val_auc: 0.6314

Epoch 00008: val_auc improved from 0.61759 to 0.63141, saving model to new fa
cevgg.h5
Epoch 9/50
200/200 [==============================] - 124s 622ms/step - loss: 0.5797 - a
uc: 0.6382 - val_loss: 0.5609 - val_auc: 0.6450

Epoch 00009: val_auc improved from 0.63141 to 0.64505, saving model to new fa
cevgg.h5
Epoch 10/50
200/200 [==============================] - 108s 539ms/step - loss: 0.5682 - a
uc: 0.6516 - val_loss: 0.5291 - val_auc: 0.6577

Epoch 00010: val_auc improved from 0.64505 to 0.65771, saving model to new fa
cevgg.h5
Epoch 11/50
200/200 [==============================] - 109s 543ms/step - loss: 0.5514 - a
uc: 0.6638 - val_loss: 0.4869 - val_auc: 0.6698

Epoch 00011: val_auc improved from 0.65771 to 0.66977, saving model to new fa
cevgg.h5
Epoch 12/50
200/200 [==============================] - 109s 543ms/step - loss: 0.5295 - a
uc: 0.6757 - val_loss: 0.4317 - val_auc: 0.6817

Epoch 00012: val_auc improved from 0.66977 to 0.68173, saving model to new fa
cevgg.h5
Epoch 13/50
200/200 [==============================] - 108s 539ms/step - loss: 0.5134 - a
uc: 0.6874 - val_loss: 0.4245 - val_auc: 0.6935

Epoch 00013: val_auc improved from 0.68173 to 0.69350, saving model to new fa
cevgg.h5
Epoch 14/50
200/200 [==============================] - 107s 534ms/step - loss: 0.5183 - a
uc: 0.6981 - val_loss: 0.4382 - val_auc: 0.7029

Epoch 00014: val_auc improved from 0.69350 to 0.70290, saving model to new fa
cevgg.h5
Epoch 15/50
200/200 [==============================] - 108s 539ms/step - loss: 0.4930 - a
uc: 0.7078 - val_loss: 0.4369 - val_auc: 0.7125

Epoch 00015: val_auc improved from 0.70290 to 0.71253, saving model to new fa
cevgg.h5
```

```
Epoch 16/50
200/200 [==============================] - 107s 536ms/step - loss: 0.4850 - a
uc: 0.7173 - val_loss: 0.4028 - val_auc: 0.7218

Epoch 00016: val_auc improved from 0.71253 to 0.72183, saving model to new fa
cevgg.h5
Epoch 17/50
200/200 [==============================] - 108s 540ms/step - loss: 0.4828 - a
uc: 0.7260 - val_loss: 0.4128 - val_auc: 0.7298

Epoch 00017: val_auc improved from 0.72183 to 0.72976, saving model to new fa
cevgg.h5
Epoch 18/50
200/200 [==============================] - 107s 537ms/step - loss: 0.4706 - a
uc: 0.7337 - val_loss: 0.4508 - val_auc: 0.7374

Epoch 00018: val_auc improved from 0.72976 to 0.73744, saving model to new fa
cevgg.h5
Epoch 19/50
200/200 [==============================] - 108s 538ms/step - loss: 0.4506 - a
uc: 0.7413 - val_loss: 0.4171 - val_auc: 0.7450

Epoch 00019: val_auc improved from 0.73744 to 0.74504, saving model to new fa
cevgg.h5
Epoch 20/50
200/200 [==============================] - 108s 540ms/step - loss: 0.4410 - a
uc: 0.7489 - val_loss: 0.4020 - val_auc: 0.7525

Epoch 00020: val_auc improved from 0.74504 to 0.75250, saving model to new fa
cevgg.h5
Epoch 21/50
200/200 [==============================] - 107s 537ms/step - loss: 0.4334 - a
uc: 0.7559 - val_loss: 0.3881 - val_auc: 0.7595

Epoch 00021: val_auc improved from 0.75250 to 0.75945, saving model to new fa
cevgg.h5
Epoch 22/50
200/200 [==============================] - 107s 533ms/step - loss: 0.4142 - a
uc: 0.7631 - val_loss: 0.3872 - val_auc: 0.7664

Epoch 00022: val_auc improved from 0.75945 to 0.76640, saving model to new fa
cevgg.h5
Epoch 23/50
200/200 [==============================] - 106s 532ms/step - loss: 0.4174 - a
uc: 0.7696 - val_loss: 0.3765 - val_auc: 0.7726

Epoch 00023: val_auc improved from 0.76640 to 0.77261, saving model to new fa
cevgg.h5
Epoch 24/50
200/200 [==============================] - 106s 532ms/step - loss: 0.4451 - a
uc: 0.7747 - val_loss: 0.3298 - val_auc: 0.7774

Epoch 00024: val_auc improved from 0.77261 to 0.77742, saving model to new fa
cevgg.h5
Epoch 25/50
200/200 [==============================] - 107s 536ms/step - loss: 0.4144 - a
uc: 0.7802 - val_loss: 0.3560 - val_auc: 0.7827
```

```
Epoch 00025: val_auc improved from 0.77742 to 0.78274, saving model to new fa
cevgg.h5
Epoch 26/50
200/200 [==============================] - 106s 530ms/step - loss: 0.3939 - a
uc: 0.7854 - val_loss: 0.3216 - val_auc: 0.7879

Epoch 00026: val_auc improved from 0.78274 to 0.78793, saving model to new fa
cevgg.h5
Epoch 27/50
200/200 [==============================] - 106s 532ms/step - loss: 0.4391 - a
uc: 0.7898 - val_loss: 0.3838 - val_auc: 0.7916

Epoch 00027: val_auc improved from 0.78793 to 0.79163, saving model to new fa
cevgg.h5
Epoch 28/50
200/200 [==============================] - 106s 531ms/step - loss: 0.3832 - a
uc: 0.7942 - val_loss: 0.3593 - val_auc: 0.7964

Epoch 00028: val_auc improved from 0.79163 to 0.79643, saving model to new fa
cevgg.h5
Epoch 29/50
200/200 [==============================] - 107s 537ms/step - loss: 0.4140 - a
uc: 0.7983 - val_loss: 0.3456 - val_auc: 0.8001

Epoch 00029: val_auc improved from 0.79643 to 0.80014, saving model to new fa
cevgg.h5
Epoch 30/50
200/200 [==============================] - 107s 536ms/step - loss: 0.3755 - a
uc: 0.8023 - val_loss: 0.3695 - val_auc: 0.8045

Epoch 00030: val_auc improved from 0.80014 to 0.80447, saving model to new fa
cevgg.h5
Epoch 31/50
200/200 [==============================] - 107s 534ms/step - loss: 0.3727 - a
uc: 0.8065 - val_loss: 0.3705 - val_auc: 0.8085

Epoch 00031: val_auc improved from 0.80447 to 0.80855, saving model to new fa
cevgg.h5
Epoch 32/50
200/200 [==============================] - 106s 531ms/step - loss: 0.3678 - a
uc: 0.8105 - val_loss: 0.3683 - val_auc: 0.8124

Epoch 00032: val_auc improved from 0.80855 to 0.81243, saving model to new fa
cevgg.h5
Epoch 33/50
200/200 [==============================] - 107s 534ms/step - loss: 0.3912 - a
uc: 0.8141 - val_loss: 0.3578 - val_auc: 0.8156

Epoch 00033: val_auc improved from 0.81243 to 0.81564, saving model to new fa
cevgg.h5
Epoch 34/50
200/200 [==============================] - 106s 531ms/step - loss: 0.3749 - a
uc: 0.8173 - val_loss: 0.3946 - val_auc: 0.8188

Epoch 00034: val_auc improved from 0.81564 to 0.81883, saving model to new fa
cevgg.h5
```

```
Epoch 35/50
200/200 [==============================] - 107s 534ms/step - loss: 0.3492 - a
uc: 0.8206 - val_loss: 0.3603 - val_auc: 0.8224

Epoch 00035: val_auc improved from 0.81883 to 0.82236, saving model to new fa
cevgg.h5
Epoch 36/50
200/200 [==============================] - 106s 532ms/step - loss: 0.3799 - a
uc: 0.8238 - val_loss: 0.3809 - val_auc: 0.8252

Epoch 00036: val_auc improved from 0.82236 to 0.82515, saving model to new fa
cevgg.h5
Epoch 37/50
200/200 [==============================] - 106s 530ms/step - loss: 0.3513 - a
uc: 0.8267 - val_loss: 0.3453 - val_auc: 0.8283

Epoch 00037: val_auc improved from 0.82515 to 0.82828, saving model to new fa
cevgg.h5
Epoch 38/50
200/200 [==============================] - 105s 526ms/step - loss: 0.3497 - a
uc: 0.8298 - val_loss: 0.3628 - val_auc: 0.8312

Epoch 00038: val_auc improved from 0.82828 to 0.83123, saving model to new fa
cevgg.h5
Epoch 39/50
200/200 [==============================] - 105s 527ms/step - loss: 0.3484 - a
uc: 0.8326 - val_loss: 0.3697 - val_auc: 0.8341

Epoch 00039: val_auc improved from 0.83123 to 0.83405, saving model to new fa
cevgg.h5
Epoch 40/50
200/200 [==============================] - 106s 528ms/step - loss: 0.3365 - a
uc: 0.8356 - val_loss: 0.4313 - val_auc: 0.8369

Epoch 00040: val_auc improved from 0.83405 to 0.83689, saving model to new fa
cevgg.h5
Epoch 41/50
200/200 [==============================] - 105s 527ms/step - loss: 0.3258 - a
uc: 0.8383 - val_loss: 0.4054 - val_auc: 0.8397

Epoch 00041: val_auc improved from 0.83689 to 0.83965, saving model to new fa
cevgg.h5
Epoch 42/50
200/200 [==============================] - 105s 527ms/step - loss: 0.3319 - a
uc: 0.8409 - val_loss: 0.4018 - val_auc: 0.8423

Epoch 00042: val_auc improved from 0.83965 to 0.84229, saving model to new fa
cevgg.h5
Epoch 43/50
200/200 [==============================] - 106s 530ms/step - loss: 0.3083 - a
uc: 0.8438 - val_loss: 0.3525 - val_auc: 0.8451

Epoch 00043: val_auc improved from 0.84229 to 0.84515, saving model to new fa
cevgg.h5
Epoch 44/50
200/200 [==============================] - 107s 533ms/step - loss: 0.3286 - a
uc: 0.8463 - val_loss: 0.3366 - val_auc: 0.8475
```

```
Epoch 00044: val_auc improved from 0.84515 to 0.84752, saving model to new fa
cevgg.h5
Epoch 45/50
200/200 [==============================] - 106s 532ms/step - loss: 0.3233 - a
uc: 0.8486 - val_loss: 0.3824 - val_auc: 0.8498

Epoch 00045: val_auc improved from 0.84752 to 0.84983, saving model to new fa
cevgg.h5
Epoch 46/50
200/200 [==============================] - 106s 531ms/step - loss: 0.2975 - a
uc: 0.8512 - val_loss: 0.3952 - val_auc: 0.8524

Epoch 00046: val_auc improved from 0.84983 to 0.85237, saving model to new fa
cevgg.h5
Epoch 47/50
200/200 [==============================] - 106s 532ms/step - loss: 0.3003 - a
uc: 0.8536 - val_loss: 0.3813 - val_auc: 0.8548

Epoch 00047: val_auc improved from 0.85237 to 0.85475, saving model to new fa
cevgg.h5
Epoch 48/50
200/200 [==============================] - 107s 537ms/step - loss: 0.2988 - a
uc: 0.8559 - val_loss: 0.3766 - val_auc: 0.8570

Epoch 00048: val_auc improved from 0.85475 to 0.85702, saving model to new fa
cevgg.h5
Epoch 49/50
200/200 [==============================] - 108s 538ms/step - loss: 0.3211 - a
uc: 0.8580 - val_loss: 0.3788 - val_auc: 0.8590

Epoch 00049: val_auc improved from 0.85702 to 0.85896, saving model to new fa
cevgg.h5
Epoch 50/50
200/200 [==============================] - 107s 537ms/step - loss: 0.3101 - a
uc: 0.8599 - val_loss: 0.3888 - val_auc: 0.8609

Epoch 00050: val_auc improved from 0.85896 to 0.86088, saving model to new fa
cevgg.h5
```

Visualizing metric using tensorboard

In [0]:
```
%load_ext tensorboard
```

In [0]:
```
%tensorboard --logdir logs
```

The below function cells are used to predict the probablities when given pairs of images from final test data.

In [0]:
```
test_path="test/"

def chunker(seq, size=32):
    return (seq[pos:pos + size] for pos in range(0, len(seq), size))


from tqdm import tqdm

submission = pd.read_csv('sample_submission.csv')
```

In [0]:
```
predictions = []

for batch in tqdm(chunker(submission.img_pair.values)):
    X1 = [x.split("-")[0] for x in batch]
    X1_FN = np.array([read_img_fn(test_path + x) for x in X1])
    X1_VGG = np.array([read_img_vgg(test_path + x) for x in X1])

    X2 = [x.split("-")[1] for x in batch]
    X2_FN = np.array([read_img_fn(test_path + x) for x in X2])
    X2_VGG = np.array([read_img_vgg(test_path + x) for x in X2])

    pred = model.predict([X1_FN, X2_FN, X1_VGG, X2_VGG]).ravel().tolist()

    predictions += pred

submission['is_related'] = predictions

submission.to_csv("face_vgg.csv", index=False)
```

```
166it [04:10,  1.58s/it]
```

After trainig the model with suffucuent number of epochs, it gave 0.890 private score and 0.878 public score on the test dataset.

2nd Model

This model uses only VGG16 architecture with resnet model for face embeddings

Input image size is 197,197 for this model.

In [0]:
```
def read_img(path):
    """function to read image from path and convert to target size 197 x 19
7"""
    img = image.load_img(path, target_size=(197, 197))
    img = np.array(img).astype(np.float)
    return preprocess_input(img, version=2)
```

In [0]:
```
def gen(list_tuples, person_to_images_map, batch_size=16):
    """a genereator function used to generte batches of images and labels"""
    ppl = list(person_to_images_map.keys())
    while True:
        batch_tuples = sample(list_tuples, batch_size // 2)
        labels = [1] * len(batch_tuples)
        while len(batch_tuples) < batch_size:
            p1 = choice(ppl)
            p2 = choice(ppl)

            if p1 != p2 and (p1, p2) not in list_tuples and (p2, p1) not in li
st_tuples:
                batch_tuples.append((p1, p2))
                labels.append(0)

        for x in batch_tuples:
            if not len(person_to_images_map[x[0]]):
                print(x[0])

        X1 = [choice(person_to_images_map[x[0]]) for x in batch_tuples]
        X1 = np.array([read_img(x) for x in X1])

        X2 = [choice(person_to_images_map[x[1]]) for x in batch_tuples]
        X2 = np.array([read_img(x) for x in X2])

        yield [X1, X2], labels
```

In [0]:
```
valx=gen(val, val_person_to_images_map, batch_size=100)
```

In [0]:
```
for i in valx:
    valx=i
    break
```

Model Architecture

In [0]:
```
input_1 = Input(shape=(197, 197, 3)) #input image 1
input_2 = Input(shape=(197, 197, 3)) #input image 2

#using bottleneck features of vggface model with trainable layers.
vgg_model = VGGFace(model='resnet50', include_top=False)

for x in vgg_model.layers[:-3]:
     x.trainable = True

x1 = vgg_model(input_1)
x2 = vgg_model(input_2)

concat1 = Concatenate(axis=-1)([GlobalMaxPool2D()(x1), GlobalAvgPool2D()(x1)])
concat2 = Concatenate(axis=-1)([GlobalMaxPool2D()(x2), GlobalAvgPool2D()(x2)])

subtract1 = Subtract()([concat1, concat2])   #creating new layer by subtractin
g x1 & x2
sqare3 = Multiply()([subtract1, subtract1])   #creating new layer by squaring
 x3

x1_ = Multiply()([concat1, concat1])  #creating new layer by squaring x1
x2_ = Multiply()([concat2, concat2])  #creating new layer by squaring x2
x4 = Subtract()([x1_, x2_])
x = Concatenate(axis=-1)([x4, sqare3])  #finally concatenating all the above l
ayers

x = Dense(100, activation="relu")(x)
x = Dropout(0.01)(x)
out = Dense(1, activation="sigmoid")(x)

model = Model([input_1, input_2], out)  #defining model
```

In [0]:
```
model.compile(loss="binary_crossentropy", metrics=[auc], optimizer=Adam(0.0000
1))

model.summary()
```

```
Model: "model_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            (None, 197, 197, 3)  0
__________________________________________________________________________________________________
input_2 (InputLayer)            (None, 197, 197, 3)  0
__________________________________________________________________________________________________
vggface_resnet50 (Model)        multiple             23561152    input_1[0]
[0]
                                                                 input_2[0]
[0]
__________________________________________________________________________________________________
global_max_pooling2d_1 (GlobalM (None, 2048)         0           vggface_resn
et50[1][0]
__________________________________________________________________________________________________
global_average_pooling2d_1 (Glo (None, 2048)         0           vggface_resn
et50[1][0]
__________________________________________________________________________________________________
global_max_pooling2d_2 (GlobalM (None, 2048)         0           vggface_resn
et50[2][0]
__________________________________________________________________________________________________
global_average_pooling2d_2 (Glo (None, 2048)         0           vggface_resn
et50[2][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, 4096)         0           global_max_p
ooling2d_1[0][0]
                                                                 global_avera
ge_pooling2d_1[0][0]
__________________________________________________________________________________________________
concatenate_2 (Concatenate)     (None, 4096)         0           global_max_p
ooling2d_2[0][0]
                                                                 global_avera
ge_pooling2d_2[0][0]
__________________________________________________________________________________________________
multiply_2 (Multiply)           (None, 4096)         0           concatenate_
1[0][0]
                                                                 concatenate_
1[0][0]
__________________________________________________________________________________________________
multiply_3 (Multiply)           (None, 4096)         0           concatenate_
2[0][0]
                                                                 concatenate_
2[0][0]
__________________________________________________________________________________________________
```

```
________________________
subtract_1 (Subtract)           (None, 4096)         0           concatenate_
1[0][0]
                                                                 concatenate_
2[0][0]
____________________________________________________________________________
________________________
subtract_2 (Subtract)           (None, 4096)         0           multiply_2
[0][0]
                                                                 multiply_3
[0][0]
____________________________________________________________________________
________________________
multiply_1 (Multiply)           (None, 4096)         0           subtract_1
[0][0]
                                                                 subtract_1
[0][0]
____________________________________________________________________________
________________________
concatenate_3 (Concatenate)     (None, 8192)         0           subtract_2
[0][0]
                                                                 multiply_1
[0][0]
____________________________________________________________________________
________________________
dense_1 (Dense)                 (None, 100)          819300      concatenate_
3[0][0]
____________________________________________________________________________
________________________
dropout_1 (Dropout)             (None, 100)          0           dense_1[0]
[0]
____________________________________________________________________________
________________________
dense_2 (Dense)                 (None, 1)            101         dropout_1[0]
[0]
============================================================================
=====================
Total params: 24,380,553
Trainable params: 24,327,433
Non-trainable params: 53,120
____________________________________________________________________________
________________________
```

Training the model and saving it with name vgg_only.h5

In [0]:
```
import datetime
from keras.callbacks import TensorBoard,EarlyStopping

# Clear any logs from previous runs
!rm -rf ./logs/

log_dir="logs"
tensorboard_callback = TensorBoard(log_dir=log_dir, histogram_freq=1)

es = EarlyStopping(monitor='val_auc', mode='max', verbose=1, patience=10)

checkpoint = ModelCheckpoint('"vgg_only.h5', monitor='val_auc', verbose=1, sav
e_best_only=True, mode='max')

reduce_on_plateau = ReduceLROnPlateau(monitor="val_auc", mode="max", factor=0.
1, patience=20, verbose=1)

callbacks_list = [tensorboard_callback, checkpoint, reduce_on_plateau, es]

history1 = model.fit_generator(gen(train, train_person_to_images_map, batch_si
ze=16), use_multiprocessing=True,
                    validation_data=(valx[0],valx[1]), epochs=50, verbose=1,
                    workers = 4,callbacks=callbacks_list, steps_per_epoch=200)
```

```
Epoch 1/50
200/200 [==============================] - 60s 300ms/step - loss: 3.5311 - au
c: 0.5708 - val_loss: 3.1825 - val_auc: 0.5991

Epoch 00001: val_auc improved from -inf to 0.59913, saving model to "vgg_onl
y.h5
Epoch 2/50
200/200 [==============================] - 44s 222ms/step - loss: 1.6618 - au
c: 0.6150 - val_loss: 1.3822 - val_auc: 0.6296

Epoch 00002: val_auc improved from 0.59913 to 0.62957, saving model to "vgg_o
nly.h5
Epoch 3/50
200/200 [==============================] - 45s 223ms/step - loss: 0.9340 - au
c: 0.6387 - val_loss: 0.6914 - val_auc: 0.6483

Epoch 00003: val_auc improved from 0.62957 to 0.64826, saving model to "vgg_o
nly.h5
Epoch 4/50
200/200 [==============================] - 44s 222ms/step - loss: 0.7325 - au
c: 0.6548 - val_loss: 0.6619 - val_auc: 0.6613

Epoch 00004: val_auc improved from 0.64826 to 0.66133, saving model to "vgg_o
nly.h5
Epoch 5/50
200/200 [==============================] - 44s 222ms/step - loss: 0.6323 - au
c: 0.6680 - val_loss: 0.6052 - val_auc: 0.6746

Epoch 00005: val_auc improved from 0.66133 to 0.67460, saving model to "vgg_o
nly.h5
Epoch 6/50
200/200 [==============================] - 44s 222ms/step - loss: 0.5942 - au
c: 0.6802 - val_loss: 0.5889 - val_auc: 0.6856

Epoch 00006: val_auc improved from 0.67460 to 0.68557, saving model to "vgg_o
nly.h5
Epoch 7/50
200/200 [==============================] - 44s 222ms/step - loss: 0.5363 - au
c: 0.6919 - val_loss: 0.5807 - val_auc: 0.6987

Epoch 00007: val_auc improved from 0.68557 to 0.69871, saving model to "vgg_o
nly.h5
Epoch 8/50
200/200 [==============================] - 44s 222ms/step - loss: 0.5145 - au
c: 0.7047 - val_loss: 0.5376 - val_auc: 0.7109

Epoch 00008: val_auc improved from 0.69871 to 0.71092, saving model to "vgg_o
nly.h5
Epoch 9/50
200/200 [==============================] - 44s 222ms/step - loss: 0.4980 - au
c: 0.7167 - val_loss: 0.5854 - val_auc: 0.7222

Epoch 00009: val_auc improved from 0.71092 to 0.72223, saving model to "vgg_o
nly.h5
Epoch 10/50
200/200 [==============================] - 44s 222ms/step - loss: 0.4886 - au
c: 0.7276 - val_loss: 0.5005 - val_auc: 0.7326
```

```
Epoch 00010: val_auc improved from 0.72223 to 0.73265, saving model to "vgg_o
nly.h5
Epoch 11/50
200/200 [==============================] - 44s 222ms/step - loss: 0.4481 - au
c: 0.7379 - val_loss: 0.5123 - val_auc: 0.7435

Epoch 00011: val_auc improved from 0.73265 to 0.74349, saving model to "vgg_o
nly.h5
Epoch 12/50
200/200 [==============================] - 44s 222ms/step - loss: 0.4598 - au
c: 0.7481 - val_loss: 0.5368 - val_auc: 0.7526

Epoch 00012: val_auc improved from 0.74349 to 0.75262, saving model to "vgg_o
nly.h5
Epoch 13/50
200/200 [==============================] - 44s 222ms/step - loss: 0.4479 - au
c: 0.7567 - val_loss: 0.5289 - val_auc: 0.7607

Epoch 00013: val_auc improved from 0.75262 to 0.76069, saving model to "vgg_o
nly.h5
Epoch 14/50
200/200 [==============================] - 45s 223ms/step - loss: 0.4314 - au
c: 0.7650 - val_loss: 0.5673 - val_auc: 0.7688

Epoch 00014: val_auc improved from 0.76069 to 0.76877, saving model to "vgg_o
nly.h5
Epoch 15/50
200/200 [==============================] - 44s 222ms/step - loss: 0.4457 - au
c: 0.7718 - val_loss: 0.5515 - val_auc: 0.7749

Epoch 00015: val_auc improved from 0.76877 to 0.77491, saving model to "vgg_o
nly.h5
Epoch 16/50
200/200 [==============================] - 44s 222ms/step - loss: 0.4313 - au
c: 0.7780 - val_loss: 0.5429 - val_auc: 0.7809

Epoch 00016: val_auc improved from 0.77491 to 0.78095, saving model to "vgg_o
nly.h5
Epoch 17/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3994 - au
c: 0.7843 - val_loss: 0.5619 - val_auc: 0.7876

Epoch 00017: val_auc improved from 0.78095 to 0.78760, saving model to "vgg_o
nly.h5
Epoch 18/50
200/200 [==============================] - 45s 223ms/step - loss: 0.3983 - au
c: 0.7908 - val_loss: 0.5045 - val_auc: 0.7937

Epoch 00018: val_auc improved from 0.78760 to 0.79366, saving model to "vgg_o
nly.h5
Epoch 19/50
200/200 [==============================] - 44s 222ms/step - loss: 0.4168 - au
c: 0.7960 - val_loss: 0.4965 - val_auc: 0.7984

Epoch 00019: val_auc improved from 0.79366 to 0.79837, saving model to "vgg_o
nly.h5
```

```
Epoch 20/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3777 - au
c: 0.8012 - val_loss: 0.4905 - val_auc: 0.8039

Epoch 00020: val_auc improved from 0.79837 to 0.80386, saving model to "vgg_o
nly.h5
Epoch 21/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3890 - au
c: 0.8061 - val_loss: 0.5671 - val_auc: 0.8084

Epoch 00021: val_auc improved from 0.80386 to 0.80843, saving model to "vgg_o
nly.h5
Epoch 22/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3920 - au
c: 0.8105 - val_loss: 0.5580 - val_auc: 0.8125

Epoch 00022: val_auc improved from 0.80843 to 0.81251, saving model to "vgg_o
nly.h5
Epoch 23/50
200/200 [==============================] - 44s 221ms/step - loss: 0.3653 - au
c: 0.8148 - val_loss: 0.5536 - val_auc: 0.8170

Epoch 00023: val_auc improved from 0.81251 to 0.81698, saving model to "vgg_o
nly.h5
Epoch 24/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3684 - au
c: 0.8191 - val_loss: 0.5719 - val_auc: 0.8211

Epoch 00024: val_auc improved from 0.81698 to 0.82110, saving model to "vgg_o
nly.h5
Epoch 25/50
200/200 [==============================] - 45s 223ms/step - loss: 0.3581 - au
c: 0.8230 - val_loss: 0.5409 - val_auc: 0.8250

Epoch 00025: val_auc improved from 0.82110 to 0.82501, saving model to "vgg_o
nly.h5
Epoch 26/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3623 - au
c: 0.8268 - val_loss: 0.5189 - val_auc: 0.8286

Epoch 00026: val_auc improved from 0.82501 to 0.82859, saving model to "vgg_o
nly.h5
Epoch 27/50
200/200 [==============================] - 45s 223ms/step - loss: 0.3622 - au
c: 0.8304 - val_loss: 0.5507 - val_auc: 0.8319

Epoch 00027: val_auc improved from 0.82859 to 0.83194, saving model to "vgg_o
nly.h5
Epoch 28/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3431 - au
c: 0.8336 - val_loss: 0.5408 - val_auc: 0.8354

Epoch 00028: val_auc improved from 0.83194 to 0.83536, saving model to "vgg_o
nly.h5
Epoch 29/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3611 - au
c: 0.8367 - val_loss: 0.4970 - val_auc: 0.8382
```

```
Epoch 00029: val_auc improved from 0.83536 to 0.83820, saving model to "vgg_o
nly.h5
Epoch 30/50
200/200 [==============================] - 44s 221ms/step - loss: 0.3586 - au
c: 0.8395 - val_loss: 0.5148 - val_auc: 0.8409

Epoch 00030: val_auc improved from 0.83820 to 0.84093, saving model to "vgg_o
nly.h5
Epoch 31/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3239 - au
c: 0.8424 - val_loss: 0.5444 - val_auc: 0.8440

Epoch 00031: val_auc improved from 0.84093 to 0.84403, saving model to "vgg_o
nly.h5
Epoch 32/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3502 - au
c: 0.8453 - val_loss: 0.5959 - val_auc: 0.8465

Epoch 00032: val_auc improved from 0.84403 to 0.84654, saving model to "vgg_o
nly.h5
Epoch 33/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3350 - au
c: 0.8478 - val_loss: 0.5900 - val_auc: 0.8491

Epoch 00033: val_auc improved from 0.84654 to 0.84908, saving model to "vgg_o
nly.h5
Epoch 34/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3375 - au
c: 0.8503 - val_loss: 0.4962 - val_auc: 0.8514

Epoch 00034: val_auc improved from 0.84908 to 0.85141, saving model to "vgg_o
nly.h5
Epoch 35/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3293 - au
c: 0.8526 - val_loss: 0.5521 - val_auc: 0.8538

Epoch 00035: val_auc improved from 0.85141 to 0.85380, saving model to "vgg_o
nly.h5
Epoch 36/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3147 - au
c: 0.8549 - val_loss: 0.5810 - val_auc: 0.8562

Epoch 00036: val_auc improved from 0.85380 to 0.85621, saving model to "vgg_o
nly.h5
Epoch 37/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3024 - au
c: 0.8574 - val_loss: 0.5691 - val_auc: 0.8586

Epoch 00037: val_auc improved from 0.85621 to 0.85864, saving model to "vgg_o
nly.h5
Epoch 38/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3164 - au
c: 0.8598 - val_loss: 0.5357 - val_auc: 0.8609

Epoch 00038: val_auc improved from 0.85864 to 0.86085, saving model to "vgg_o
nly.h5
```

```
Epoch 39/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3074 - au
c: 0.8619 - val_loss: 0.5387 - val_auc: 0.8630

Epoch 00039: val_auc improved from 0.86085 to 0.86298, saving model to "vgg_o
nly.h5
Epoch 40/50
200/200 [==============================] - 44s 222ms/step - loss: 0.2745 - au
c: 0.8642 - val_loss: 0.5681 - val_auc: 0.8654

Epoch 00040: val_auc improved from 0.86298 to 0.86544, saving model to "vgg_o
nly.h5
Epoch 41/50
200/200 [==============================] - 44s 222ms/step - loss: 0.3149 - au
c: 0.8664 - val_loss: 0.5405 - val_auc: 0.8673

Epoch 00041: val_auc improved from 0.86544 to 0.86733, saving model to "vgg_o
nly.h5
Epoch 42/50
200/200 [==============================] - 44s 222ms/step - loss: 0.2984 - au
c: 0.8684 - val_loss: 0.4787 - val_auc: 0.8693

Epoch 00042: val_auc improved from 0.86733 to 0.86931, saving model to "vgg_o
nly.h5
Epoch 43/50
200/200 [==============================] - 44s 222ms/step - loss: 0.2929 - au
c: 0.8703 - val_loss: 0.4861 - val_auc: 0.8712

Epoch 00043: val_auc improved from 0.86931 to 0.87123, saving model to "vgg_o
nly.h5
Epoch 44/50
200/200 [==============================] - 44s 222ms/step - loss: 0.2942 - au
c: 0.8722 - val_loss: 0.5558 - val_auc: 0.8731

Epoch 00044: val_auc improved from 0.87123 to 0.87311, saving model to "vgg_o
nly.h5
Epoch 45/50
200/200 [==============================] - 45s 223ms/step - loss: 0.2799 - au
c: 0.8740 - val_loss: 0.5090 - val_auc: 0.8750

Epoch 00045: val_auc improved from 0.87311 to 0.87496, saving model to "vgg_o
nly.h5
Epoch 46/50
200/200 [==============================] - 44s 222ms/step - loss: 0.2869 - au
c: 0.8758 - val_loss: 0.5045 - val_auc: 0.8767

Epoch 00046: val_auc improved from 0.87496 to 0.87667, saving model to "vgg_o
nly.h5
Epoch 47/50
200/200 [==============================] - 44s 222ms/step - loss: 0.2835 - au
c: 0.8775 - val_loss: 0.5352 - val_auc: 0.8783

Epoch 00047: val_auc improved from 0.87667 to 0.87833, saving model to "vgg_o
nly.h5
Epoch 48/50
200/200 [==============================] - 44s 222ms/step - loss: 0.2741 - au
c: 0.8792 - val_loss: 0.5426 - val_auc: 0.8800
```

```
Epoch 00048: val_auc improved from 0.87833 to 0.88003, saving model to "vgg_o
nly.h5
Epoch 49/50
200/200 [==============================] - 44s 222ms/step - loss: 0.2749 - au
c: 0.8809 - val_loss: 0.5680 - val_auc: 0.8816

Epoch 00049: val_auc improved from 0.88003 to 0.88165, saving model to "vgg_o
nly.h5
Epoch 50/50
200/200 [==============================] - 44s 222ms/step - loss: 0.2655 - au
c: 0.8825 - val_loss: 0.5804 - val_auc: 0.8833

Epoch 00050: val_auc improved from 0.88165 to 0.88330, saving model to "vgg_o
nly.h5
```

In [0]:

```python
predictions = []

for batch in tqdm(chunker(submission.img_pair.values)):
    X1 = [x.split("-")[0] for x in batch]
    X1 = np.array([read_img(test_path + x) for x in X1])

    X2 = [x.split("-")[1] for x in batch]
    X2 = np.array([read_img(test_path + x) for x in X2])

    pred = model.predict([X1, X2]).ravel().tolist()
    predictions += pred

submission['is_related'] = predictions

submission.to_csv("vgg_only0.csv", index=False)
```

```
166it [02:06,  1.05it/s]
```

Visualizing metric using tensorboard

In [0]:

```python
%load_ext tensorboard
```

```
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
```

In [0]:

```python
%tensorboard --logdir logs
```

```
Reusing TensorBoard on port 6006 (pid 647), started 1:16:45 ago. (Use '!kill
647' to kill it.)
```

After trainig the model with suffucuent number of epochs, it gave 0.855 private score and 0.839 public score on the test dataset.

Model 3

This model also uses only VGG16 architecture with resnet model for face embedding but different layers at the end.

In [0]:
```
def read_img(path):
    """function to read image and convert it into target size of 224 x 224."""
    img = cv2.imread(path)
    img = np.array(img).astype(np.float)
    return preprocess_input(img, version=2)


def gen(list_tuples, person_to_images_map, batch_size=16):
    """generator funtion will generate images in the right format while traini
ng the model """
    ppl = list(person_to_images_map.keys())
    while True:
        batch_tuples = sample(list_tuples, batch_size // 2)
        labels = [1] * len(batch_tuples)
        while len(batch_tuples) < batch_size:
            p1 = choice(ppl)
            p2 = choice(ppl)

            if p1 != p2 and (p1, p2) not in list_tuples and (p2, p1) not in li
st_tuples:
                batch_tuples.append((p1, p2))
                labels.append(0)

        for x in batch_tuples:
            if not len(person_to_images_map[x[0]]):
                print(x[0])

        X1 = [choice(person_to_images_map[x[0]]) for x in batch_tuples]
        X1 = np.array([read_img(x) for x in X1])

        X2 = [choice(person_to_images_map[x[1]]) for x in batch_tuples]
        X2 = np.array([read_img(x) for x in X2])

        yield [X1, X2], labels
```

In [0]:
```
valx=gen(val, val_person_to_images_map, batch_size=100)
```

In [0]:
```
for i in valx:
    valx=i
    break
```

In [0]:
```
input_1 = Input(shape=(224, 224, 3))    #input image 1
input_2 = Input(shape=(224, 224, 3))    #input image 2

base_model = VGGFace(model='resnet50', include_top=False)

#using bottleneck features of vggface model with trainable layers.
for layer in base_model.layers[:-3]:
    layer.trainable = True

x1 = base_model(input_1)
x2 = base_model(input_2)


merged_add = Add()([x1, x2])    #adding both images
merged_sub = Subtract()([x1,x2])#subtracting both images

#Sending above to layers to convolution layers
merged_add = Conv2D(100 , [1,1] )(merged_add)
merged_sub = Conv2D(100 , [1,1] )(merged_sub)

#finally concatenating all the layers
merged = Concatenate(axis=-1)([merged_add, merged_sub])

merged = Flatten()(merged) #flattening the layer which is to be submitted to t
he dense layers

merged = Dense(200, activation="relu")(merged)
merged = Dropout(0.3)(merged)
merged = Dense(100, activation="relu")(merged)
merged = Dropout(0.3)(merged)
merged = Dense(25, activation="relu")(merged)
merged = Dropout(0.3)(merged)
out = Dense(1, activation="sigmoid")(merged)

model = Model([input_1, input_2], out)

model.compile(loss="binary_crossentropy", metrics= [auc], optimizer=Adam(0.000
01))

model.summary()
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:4271: The name tf.nn.avg_pool is deprecated. Please use
tf.nn.avg_pool2d instead.

Downloading data from https://github.com/rcmalli/keras-vggface/releases/downl
oad/v2.0/rcmalli_vggface_tf_notop_resnet50.h5
94699520/94694792 [==============================] - 2s 0us/step
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimize
rs.py:793: The name tf.train.Optimizer is deprecated. Please use tf.compat.v
1.train.Optimizer instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:3657: The name tf.log is deprecated. Please use tf.mat
h.log instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_cor
e/python/ops/nn_impl.py:183: where (from tensorflow.python.ops.array_ops) is
deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_cor
e/python/ops/metrics_impl.py:808: div (from tensorflow.python.ops.math_ops) i
s deprecated and will be removed in a future version.
Instructions for updating:
Deprecated in favor of operator or tf.math.divide.
Model: "model_1"
_____________________________________________________________________________
_____________________
Layer (type)                     Output Shape         Param #     Connected to
=============================================================================
=====================
input_1 (InputLayer)             (None, 224, 224, 3)  0
_____________________________________________________________________________
_____________________
input_2 (InputLayer)             (None, 224, 224, 3)  0
_____________________________________________________________________________
_____________________
vggface_resnet50 (Model)         multiple             23561152    input_1[0]
[0]
                                                                  input_2[0]
[0]
_____________________________________________________________________________
_____________________
add_17 (Add)                     (None, 1, 1, 2048)   0           vggface_resn
et50[1][0]
                                                                  vggface_resn
et50[2][0]
_____________________________________________________________________________
_____________________
subtract_1 (Subtract)            (None, 1, 1, 2048)   0           vggface_resn
et50[1][0]
                                                                  vggface_resn
et50[2][0]
_____________________________________________________________________________
_____________________
conv2d_1 (Conv2D)                (None, 1, 1, 100)    204900      add_17[0][0]
_____________________________________________________________________________
```

```
____________________
conv2d_2 (Conv2D)              (None, 1, 1, 100)   204900      subtract_1
[0][0]
__________________________________________________________________________
____________________
concatenate_1 (Concatenate)    (None, 1, 1, 200)   0           conv2d_1[0]
[0]
                                                               conv2d_2[0]
[0]
__________________________________________________________________________
____________________
flatten_1 (Flatten)            (None, 200)         0           concatenate_
1[0][0]
__________________________________________________________________________
____________________
dense_1 (Dense)                (None, 200)         40200       flatten_1[0]
[0]
__________________________________________________________________________
____________________
dropout_1 (Dropout)            (None, 200)         0           dense_1[0]
[0]
__________________________________________________________________________
____________________
dense_2 (Dense)                (None, 100)         20100       dropout_1[0]
[0]
__________________________________________________________________________
____________________
dropout_2 (Dropout)            (None, 100)         0           dense_2[0]
[0]
__________________________________________________________________________
____________________
dense_3 (Dense)                (None, 25)          2525        dropout_2[0]
[0]
__________________________________________________________________________
____________________
dropout_3 (Dropout)            (None, 25)          0           dense_3[0]
[0]
__________________________________________________________________________
____________________
dense_4 (Dense)                (None, 1)           26          dropout_3[0]
[0]
==========================================================================
====================
Total params: 24,033,803
Trainable params: 23,980,683
Non-trainable params: 53,120
__________________________________________________________________________
____________________
```

In [0]:

```
K.clear_session()
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:107: The name tf.reset_default_graph is deprecated. Ple
ase use tf.compat.v1.reset_default_graph instead.
```

Training the model and saving it with name vgg_only1.h5

In [0]:
```
from keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau,
TensorBoard

# Clear any logs from previous runs
!rm -rf ./logs/

log_dir="logs"
tensorboard_callback = TensorBoard(log_dir=log_dir, histogram_freq=1)

#es = tf.keras.callbacks.EarlyStopping(monitor='val_auc', mode='max', verbose=
1, patience=10)

checkpoint = ModelCheckpoint('"vgg_only1.h5', monitor='val_auc', verbose=1, sa
ve_best_only=True, mode='max')

reduce_on_plateau = ReduceLROnPlateau(monitor="val_auc", mode="max", factor=0.
1, patience=20, verbose=1)

callbacks_list = [tensorboard_callback, checkpoint, reduce_on_plateau]

history2 = model.fit_generator(gen(train, train_person_to_images_map, batch_si
ze=16), use_multiprocessing=True,
                    validation_data=(valx[0],valx[1]), epochs=50, verbose=1,
                    workers = 4,callbacks=callbacks_list, steps_per_epoch=200)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:1033: The name tf.assign_add is deprecated. Please use
tf.compat.v1.assign_add instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/
tensorflow_backend.py:1020: The name tf.assign is deprecated. Please use tf.c
ompat.v1.assign instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1120: The name tf.summary.histogram is deprecated. Please use tf.compat.
v1.summary.histogram instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1122: The name tf.summary.merge_all is deprecated. Please use tf.compat.
v1.summary.merge_all instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1125: The name tf.summary.FileWriter is deprecated. Please use tf.compa
t.v1.summary.FileWriter instead.

Epoch 1/50
200/200 [==============================] - 293s 1s/step - loss: 1.2582 - auc:
0.4915 - val_loss: 0.7500 - val_auc: 0.5194
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1265: The name tf.Summary is deprecated. Please use tf.compat.v1.Summary
instead.


Epoch 00001: val_auc improved from -inf to 0.51939, saving model to "vgg_only
1.h5
Epoch 2/50
200/200 [==============================] - 146s 729ms/step - loss: 0.9281 - a
uc: 0.5298 - val_loss: 0.7202 - val_auc: 0.5451

Epoch 00002: val_auc improved from 0.51939 to 0.54506, saving model to "vgg_o
nly1.h5
Epoch 3/50
200/200 [==============================] - 96s 478ms/step - loss: 0.8232 - au
c: 0.5525 - val_loss: 0.6789 - val_auc: 0.5576

Epoch 00003: val_auc improved from 0.54506 to 0.55761, saving model to "vgg_o
nly1.h5
Epoch 4/50
200/200 [==============================] - 70s 350ms/step - loss: 0.7468 - au
c: 0.5646 - val_loss: 0.6538 - val_auc: 0.5703

Epoch 00004: val_auc improved from 0.55761 to 0.57030, saving model to "vgg_o
nly1.h5
Epoch 5/50
200/200 [==============================] - 53s 266ms/step - loss: 0.7161 - au
c: 0.5744 - val_loss: 0.6215 - val_auc: 0.5793

Epoch 00005: val_auc improved from 0.57030 to 0.57933, saving model to "vgg_o
nly1.h5
Epoch 6/50
200/200 [==============================] - 51s 254ms/step - loss: 0.6900 - au
c: 0.5834 - val_loss: 0.6115 - val_auc: 0.5881
```

```
Epoch 00006: val_auc improved from 0.57933 to 0.58815, saving model to "vgg_o
nly1.h5
Epoch 7/50
200/200 [==============================] - 51s 253ms/step - loss: 0.6561 - au
c: 0.5939 - val_loss: 0.6251 - val_auc: 0.5991

Epoch 00007: val_auc improved from 0.58815 to 0.59908, saving model to "vgg_o
nly1.h5
Epoch 8/50
200/200 [==============================] - 51s 254ms/step - loss: 0.6525 - au
c: 0.6026 - val_loss: 0.6304 - val_auc: 0.6077

Epoch 00008: val_auc improved from 0.59908 to 0.60770, saving model to "vgg_o
nly1.h5
Epoch 9/50
200/200 [==============================] - 51s 253ms/step - loss: 0.6347 - au
c: 0.6117 - val_loss: 0.6230 - val_auc: 0.6163

Epoch 00009: val_auc improved from 0.60770 to 0.61635, saving model to "vgg_o
nly1.h5
Epoch 10/50
200/200 [==============================] - 50s 252ms/step - loss: 0.6402 - au
c: 0.6190 - val_loss: 0.5996 - val_auc: 0.6220

Epoch 00010: val_auc improved from 0.61635 to 0.62201, saving model to "vgg_o
nly1.h5
Epoch 11/50
200/200 [==============================] - 50s 251ms/step - loss: 0.6314 - au
c: 0.6250 - val_loss: 0.6140 - val_auc: 0.6280

Epoch 00011: val_auc improved from 0.62201 to 0.62798, saving model to "vgg_o
nly1.h5
Epoch 12/50
200/200 [==============================] - 50s 252ms/step - loss: 0.6297 - au
c: 0.6306 - val_loss: 0.6006 - val_auc: 0.6330

Epoch 00012: val_auc improved from 0.62798 to 0.63299, saving model to "vgg_o
nly1.h5
Epoch 13/50
200/200 [==============================] - 50s 251ms/step - loss: 0.6173 - au
c: 0.6352 - val_loss: 0.5958 - val_auc: 0.6380

Epoch 00013: val_auc improved from 0.63299 to 0.63803, saving model to "vgg_o
nly1.h5
Epoch 14/50
200/200 [==============================] - 53s 265ms/step - loss: 0.6035 - au
c: 0.6410 - val_loss: 0.6412 - val_auc: 0.6435

Epoch 00014: val_auc improved from 0.63803 to 0.64349, saving model to "vgg_o
nly1.h5
Epoch 15/50
200/200 [==============================] - 50s 250ms/step - loss: 0.6150 - au
c: 0.6457 - val_loss: 0.5865 - val_auc: 0.6475

Epoch 00015: val_auc improved from 0.64349 to 0.64745, saving model to "vgg_o
nly1.h5
```

```
Epoch 16/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5986 - au
c: 0.6496 - val_loss: 0.5953 - val_auc: 0.6522

Epoch 00016: val_auc improved from 0.64745 to 0.65222, saving model to "vgg_o
nly1.h5
Epoch 17/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5980 - au
c: 0.6543 - val_loss: 0.5977 - val_auc: 0.6565

Epoch 00017: val_auc improved from 0.65222 to 0.65654, saving model to "vgg_o
nly1.h5
Epoch 18/50
200/200 [==============================] - 49s 247ms/step - loss: 0.5868 - au
c: 0.6589 - val_loss: 0.6000 - val_auc: 0.6613

Epoch 00018: val_auc improved from 0.65654 to 0.66128, saving model to "vgg_o
nly1.h5
Epoch 19/50
200/200 [==============================] - 49s 247ms/step - loss: 0.5781 - au
c: 0.6637 - val_loss: 0.6220 - val_auc: 0.6658

Epoch 00019: val_auc improved from 0.66128 to 0.66576, saving model to "vgg_o
nly1.h5
Epoch 20/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5741 - au
c: 0.6677 - val_loss: 0.6134 - val_auc: 0.6701

Epoch 00020: val_auc improved from 0.66576 to 0.67013, saving model to "vgg_o
nly1.h5
Epoch 21/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5995 - au
c: 0.6715 - val_loss: 0.6388 - val_auc: 0.6727

Epoch 00021: val_auc improved from 0.67013 to 0.67266, saving model to "vgg_o
nly1.h5
Epoch 22/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5874 - au
c: 0.6739 - val_loss: 0.6101 - val_auc: 0.6753

Epoch 00022: val_auc improved from 0.67266 to 0.67531, saving model to "vgg_o
nly1.h5
Epoch 23/50
200/200 [==============================] - 49s 247ms/step - loss: 0.5698 - au
c: 0.6767 - val_loss: 0.5828 - val_auc: 0.6786

Epoch 00023: val_auc improved from 0.67531 to 0.67859, saving model to "vgg_o
nly1.h5
Epoch 24/50
200/200 [==============================] - 49s 247ms/step - loss: 0.5782 - au
c: 0.6802 - val_loss: 0.5820 - val_auc: 0.6814

Epoch 00024: val_auc improved from 0.67859 to 0.68143, saving model to "vgg_o
nly1.h5
Epoch 25/50
200/200 [==============================] - 49s 246ms/step - loss: 0.5680 - au
c: 0.6829 - val_loss: 0.5774 - val_auc: 0.6844
```

```
Epoch 00025: val_auc improved from 0.68143 to 0.68440, saving model to "vgg_o
nly1.h5
Epoch 26/50
200/200 [==============================] - 49s 247ms/step - loss: 0.5458 - au
c: 0.6862 - val_loss: 0.5623 - val_auc: 0.6882

Epoch 00026: val_auc improved from 0.68440 to 0.68820, saving model to "vgg_o
nly1.h5
Epoch 27/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5420 - au
c: 0.6899 - val_loss: 0.5453 - val_auc: 0.6919

Epoch 00027: val_auc improved from 0.68820 to 0.69192, saving model to "vgg_o
nly1.h5
Epoch 28/50
200/200 [==============================] - 49s 247ms/step - loss: 0.5245 - au
c: 0.6940 - val_loss: 0.5059 - val_auc: 0.6962

Epoch 00028: val_auc improved from 0.69192 to 0.69623, saving model to "vgg_o
nly1.h5
Epoch 29/50
200/200 [==============================] - 49s 247ms/step - loss: 0.5429 - au
c: 0.6980 - val_loss: 0.5357 - val_auc: 0.6997

Epoch 00029: val_auc improved from 0.69623 to 0.69970, saving model to "vgg_o
nly1.h5
Epoch 30/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5226 - au
c: 0.7015 - val_loss: 0.6157 - val_auc: 0.7035

Epoch 00030: val_auc improved from 0.69970 to 0.70345, saving model to "vgg_o
nly1.h5
Epoch 31/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5157 - au
c: 0.7055 - val_loss: 0.5970 - val_auc: 0.7073

Epoch 00031: val_auc improved from 0.70345 to 0.70725, saving model to "vgg_o
nly1.h5
Epoch 32/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5263 - au
c: 0.7088 - val_loss: 0.5130 - val_auc: 0.7105

Epoch 00032: val_auc improved from 0.70725 to 0.71053, saving model to "vgg_o
nly1.h5
Epoch 33/50
200/200 [==============================] - 49s 247ms/step - loss: 0.5185 - au
c: 0.7122 - val_loss: 0.5075 - val_auc: 0.7139

Epoch 00033: val_auc improved from 0.71053 to 0.71388, saving model to "vgg_o
nly1.h5
Epoch 34/50
200/200 [==============================] - 49s 246ms/step - loss: 0.5109 - au
c: 0.7154 - val_loss: 0.5817 - val_auc: 0.7171

Epoch 00034: val_auc improved from 0.71388 to 0.71708, saving model to "vgg_o
nly1.h5
```

```
Epoch 35/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4883 - au
c: 0.7188 - val_loss: 0.5484 - val_auc: 0.7207

Epoch 00035: val_auc improved from 0.71708 to 0.72067, saving model to "vgg_o
nly1.h5
Epoch 36/50
200/200 [==============================] - 49s 246ms/step - loss: 0.5096 - au
c: 0.7223 - val_loss: 0.6009 - val_auc: 0.7236

Epoch 00036: val_auc improved from 0.72067 to 0.72361, saving model to "vgg_o
nly1.h5
Epoch 37/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4874 - au
c: 0.7252 - val_loss: 0.5351 - val_auc: 0.7270

Epoch 00037: val_auc improved from 0.72361 to 0.72700, saving model to "vgg_o
nly1.h5
Epoch 38/50
200/200 [==============================] - 50s 248ms/step - loss: 0.5091 - au
c: 0.7283 - val_loss: 0.4967 - val_auc: 0.7297

Epoch 00038: val_auc improved from 0.72700 to 0.72969, saving model to "vgg_o
nly1.h5
Epoch 39/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4964 - au
c: 0.7312 - val_loss: 0.5872 - val_auc: 0.7325

Epoch 00039: val_auc improved from 0.72969 to 0.73255, saving model to "vgg_o
nly1.h5
Epoch 40/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4744 - au
c: 0.7341 - val_loss: 0.5648 - val_auc: 0.7355

Epoch 00040: val_auc improved from 0.73255 to 0.73553, saving model to "vgg_o
nly1.h5
Epoch 41/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4707 - au
c: 0.7370 - val_loss: 0.5738 - val_auc: 0.7385

Epoch 00041: val_auc improved from 0.73553 to 0.73852, saving model to "vgg_o
nly1.h5
Epoch 42/50
200/200 [==============================] - 50s 248ms/step - loss: 0.4704 - au
c: 0.7400 - val_loss: 0.5242 - val_auc: 0.7414

Epoch 00042: val_auc improved from 0.73852 to 0.74140, saving model to "vgg_o
nly1.h5
Epoch 43/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4616 - au
c: 0.7428 - val_loss: 0.5606 - val_auc: 0.7442

Epoch 00043: val_auc improved from 0.74140 to 0.74422, saving model to "vgg_o
nly1.h5
Epoch 44/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4783 - au
c: 0.7456 - val_loss: 0.6073 - val_auc: 0.7467
```

```
Epoch 00044: val_auc improved from 0.74422 to 0.74673, saving model to "vgg_o
nly1.h5
Epoch 45/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4781 - au
c: 0.7478 - val_loss: 0.5958 - val_auc: 0.7490

Epoch 00045: val_auc improved from 0.74673 to 0.74904, saving model to "vgg_o
nly1.h5
Epoch 46/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4857 - au
c: 0.7501 - val_loss: 0.5394 - val_auc: 0.7512

Epoch 00046: val_auc improved from 0.74904 to 0.75116, saving model to "vgg_o
nly1.h5
Epoch 47/50
200/200 [==============================] - 50s 248ms/step - loss: 0.4550 - au
c: 0.7525 - val_loss: 0.6085 - val_auc: 0.7537

Epoch 00047: val_auc improved from 0.75116 to 0.75370, saving model to "vgg_o
nly1.h5
Epoch 48/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4464 - au
c: 0.7551 - val_loss: 0.5402 - val_auc: 0.7564

Epoch 00048: val_auc improved from 0.75370 to 0.75640, saving model to "vgg_o
nly1.h5
Epoch 49/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4496 - au
c: 0.7575 - val_loss: 0.5379 - val_auc: 0.7588

Epoch 00049: val_auc improved from 0.75640 to 0.75884, saving model to "vgg_o
nly1.h5
Epoch 50/50
200/200 [==============================] - 49s 247ms/step - loss: 0.4481 - au
c: 0.7600 - val_loss: 0.5895 - val_auc: 0.7613

Epoch 00050: val_auc improved from 0.75884 to 0.76130, saving model to "vgg_o
nly1.h5
```

Visualizing metric using tensorboard

In [0]:
```
%load_ext tensorboard
```

In [0]:
```
%tensorboard --logdir logs
```

```
Reusing TensorBoard on port 6006 (pid 647), started 0:03:05 ago. (Use '!kill
647' to kill it.)
```

Predicting the probability on test data.

In [0]:
```
predictions = []

for batch in tqdm(chunker(submission.img_pair.values)):
    X1 = [x.split("-")[0] for x in batch]
    X1 = np.array([read_img(test_path + x) for x in X1])

    X2 = [x.split("-")[1] for x in batch]
    X2 = np.array([read_img(test_path + x) for x in X2])

    pred = model.predict([X1, X2]).ravel().tolist()
    predictions += pred

submission['is_related'] = predictions

submission.to_csv("vgg_only1.csv", index=False)
```

```
166it [33:29,  9.03s/it]
```

After trainig the model with suffucuent number of epochs, it gave 0.843 private score and 0.846 public score on the test dataset.

Model 4

In this model we will be using pretrained weights of the vgg base model to get the face embeddings.

In [0]:
```
input_1 = Input(shape=(224, 224, 3))    #image 1 input
input_2 = Input(shape=(224, 224, 3))    #image 2 input

#using bottleneck features of vggface model with trainable layers.
base_model = VGGFace(model='resnet50', include_top=False)

for x in base_model.layers: #Using Pretrained weights
    x.trainable = False

x1 = base_model(input_1)
x2 = base_model(input_2)

x = Concatenate()([x1, x2])
x = Flatten()(x)
x = Dense(512, activation="relu",kernel_regularizer=regularizers.l2(0.01))(x)
x = Dropout(0.5)(x)
x = Dense(256, activation="relu",kernel_regularizer=regularizers.l2(0.01))(x)
x = Dropout(0.5)(x)
x = Dense(25, activation="relu",kernel_regularizer=regularizers.l2(0.01))(x)
x = Dropout(0.5)(x)
out = Dense(1, activation="sigmoid")(x)

model = Model([input_1, input_2], out)

model.compile(loss="binary_crossentropy", metrics=[auc], optimizer=Adam(lr=1e-
4))

model.summary()
```

```
Model: "model_1"
____________________________________________________________________________
______________________
Layer (type)                    Output Shape          Param #     Connected to
============================================================================
=====================
input_1 (InputLayer)            (None, 224, 224, 3)  0
____________________________________________________________________________
______________________
input_2 (InputLayer)            (None, 224, 224, 3)  0
____________________________________________________________________________
______________________
vggface_resnet50 (Model)        multiple             23561152    input_1[0]
[0]
                                                                 input_2[0]
[0]
____________________________________________________________________________
______________________
concatenate_1 (Concatenate)     (None, 1, 1, 4096)   0           vggface_resn
et50[1][0]
                                                                 vggface_resn
et50[2][0]
____________________________________________________________________________
______________________
flatten_1 (Flatten)             (None, 4096)         0           concatenate_
1[0][0]
____________________________________________________________________________
______________________
dense_1 (Dense)                 (None, 512)          2097664     flatten_1[0]
[0]
____________________________________________________________________________
______________________
dropout_1 (Dropout)             (None, 512)          0           dense_1[0]
[0]
____________________________________________________________________________
______________________
dense_2 (Dense)                 (None, 256)          131328      dropout_1[0]
[0]
____________________________________________________________________________
______________________
dropout_2 (Dropout)             (None, 256)          0           dense_2[0]
[0]
____________________________________________________________________________
______________________
dense_3 (Dense)                 (None, 25)           6425        dropout_2[0]
[0]
____________________________________________________________________________
______________________
dropout_3 (Dropout)             (None, 25)           0           dense_3[0]
[0]
____________________________________________________________________________
______________________
dense_4 (Dense)                 (None, 1)            26          dropout_3[0]
[0]
============================================================================
=====================
Total params: 25,796,595
```

```
Trainable params: 2,235,443
Non-trainable params: 23,561,152
```

In [0]:
```
valx=gen(val, val_person_to_images_map, batch_size=100)
```

In [0]:
```
for i in valx:
    valx=i
    break
```

Training the model and saving it with name vgg_only2.h5

In [0]:
```
import datetime
from keras.callbacks import TensorBoard,EarlyStopping

# Clear any logs from previous runs
!rm -rf ./logs/

log_dir="logs"
tensorboard_callback = TensorBoard(log_dir=log_dir, histogram_freq=1)

es = EarlyStopping(monitor='val_auc', mode='max', verbose=1, patience=10)

checkpoint = ModelCheckpoint('"vgg_only2.h5', monitor='val_auc', verbose=1, sa
ve_best_only=True, mode='max')

reduce_on_plateau = ReduceLROnPlateau(monitor="val_auc", mode="max", factor=0.
1, patience=20, verbose=1)

callbacks_list = [tensorboard_callback, checkpoint, reduce_on_plateau, es]

history3 = model.fit_generator(gen(train, train_person_to_images_map, batch_si
ze=16), use_multiprocessing=True,
                    validation_data=(valx[0],valx[1]), epochs=50, verbose=1,
                    workers = 4,callbacks=callbacks_list, steps_per_epoch=200)
```

```
Epoch 1/50
200/200 [==============================] - 25s 127ms/step - loss: 13.5424 - a
uc: 0.5174 - val_loss: 12.7669 - val_auc: 0.5266

Epoch 00001: val_auc improved from -inf to 0.52665, saving model to "vgg_only
2.h5
Epoch 2/50
200/200 [==============================] - 20s 102ms/step - loss: 12.3439 - a
uc: 0.5252 - val_loss: 11.7565 - val_auc: 0.5224

Epoch 00002: val_auc did not improve from 0.52665
Epoch 3/50
200/200 [==============================] - 20s 99ms/step - loss: 11.2554 - au
c: 0.5215 - val_loss: 10.6662 - val_auc: 0.5209

Epoch 00003: val_auc did not improve from 0.52665
Epoch 4/50
200/200 [==============================] - 20s 99ms/step - loss: 10.1316 - au
c: 0.5206 - val_loss: 9.5631 - val_auc: 0.5212

Epoch 00004: val_auc did not improve from 0.52665
Epoch 5/50
200/200 [==============================] - 20s 98ms/step - loss: 9.0406 - au
c: 0.5210 - val_loss: 8.4919 - val_auc: 0.5217

Epoch 00005: val_auc did not improve from 0.52665
Epoch 6/50
200/200 [==============================] - 20s 100ms/step - loss: 7.9917 - au
c: 0.5208 - val_loss: 7.4761 - val_auc: 0.5197

Epoch 00006: val_auc did not improve from 0.52665
Epoch 7/50
200/200 [==============================] - 20s 100ms/step - loss: 7.0123 - au
c: 0.5187 - val_loss: 6.5368 - val_auc: 0.5188

Epoch 00007: val_auc did not improve from 0.52665
Epoch 8/50
200/200 [==============================] - 20s 100ms/step - loss: 6.1052 - au
c: 0.5197 - val_loss: 5.6827 - val_auc: 0.5215

Epoch 00008: val_auc did not improve from 0.52665
Epoch 9/50
200/200 [==============================] - 20s 98ms/step - loss: 5.2988 - au
c: 0.5225 - val_loss: 4.9246 - val_auc: 0.5244

Epoch 00009: val_auc did not improve from 0.52665
Epoch 10/50
200/200 [==============================] - 20s 99ms/step - loss: 4.5832 - au
c: 0.5258 - val_loss: 4.2522 - val_auc: 0.5288

Epoch 00010: val_auc improved from 0.52665 to 0.52884, saving model to "vgg_o
nly2.h5
Epoch 11/50
200/200 [==============================] - 20s 100ms/step - loss: 3.9516 - au
c: 0.5312 - val_loss: 3.6646 - val_auc: 0.5358

Epoch 00011: val_auc improved from 0.52884 to 0.53579, saving model to "vgg_o
```

```
nly2.h5
Epoch 12/50
200/200 [==============================] - 20s 99ms/step - loss: 3.4095 - au
c: 0.5402 - val_loss: 3.1754 - val_auc: 0.5448

Epoch 00012: val_auc improved from 0.53579 to 0.54484, saving model to "vgg_o
nly2.h5
Epoch 13/50
200/200 [==============================] - 20s 101ms/step - loss: 2.9597 - au
c: 0.5497 - val_loss: 2.7609 - val_auc: 0.5551

Epoch 00013: val_auc improved from 0.54484 to 0.55506, saving model to "vgg_o
nly2.h5
Epoch 14/50
200/200 [==============================] - 20s 101ms/step - loss: 2.5913 - au
c: 0.5590 - val_loss: 2.4197 - val_auc: 0.5636

Epoch 00014: val_auc improved from 0.55506 to 0.56355, saving model to "vgg_o
nly2.h5
Epoch 15/50
200/200 [==============================] - 20s 100ms/step - loss: 2.2689 - au
c: 0.5679 - val_loss: 2.1475 - val_auc: 0.5723

Epoch 00015: val_auc improved from 0.56355 to 0.57227, saving model to "vgg_o
nly2.h5
Epoch 16/50
200/200 [==============================] - 20s 99ms/step - loss: 2.0113 - au
c: 0.5763 - val_loss: 1.8967 - val_auc: 0.5802

Epoch 00016: val_auc improved from 0.57227 to 0.58016, saving model to "vgg_o
nly2.h5
Epoch 17/50
200/200 [==============================] - 20s 100ms/step - loss: 1.7961 - au
c: 0.5830 - val_loss: 1.6887 - val_auc: 0.5867

Epoch 00017: val_auc improved from 0.58016 to 0.58670, saving model to "vgg_o
nly2.h5
Epoch 18/50
200/200 [==============================] - 20s 101ms/step - loss: 1.6163 - au
c: 0.5896 - val_loss: 1.5327 - val_auc: 0.5935

Epoch 00018: val_auc improved from 0.58670 to 0.59349, saving model to "vgg_o
nly2.h5
Epoch 19/50
200/200 [==============================] - 20s 100ms/step - loss: 1.4702 - au
c: 0.5960 - val_loss: 1.3895 - val_auc: 0.5990

Epoch 00019: val_auc improved from 0.59349 to 0.59903, saving model to "vgg_o
nly2.h5
Epoch 20/50
200/200 [==============================] - 20s 100ms/step - loss: 1.3383 - au
c: 0.6013 - val_loss: 1.2755 - val_auc: 0.6044

Epoch 00020: val_auc improved from 0.59903 to 0.60440, saving model to "vgg_o
nly2.h5
Epoch 21/50
200/200 [==============================] - 20s 99ms/step - loss: 1.2341 - au
```

```
c: 0.6072 - val_loss: 1.1696 - val_auc: 0.6103

Epoch 00021: val_auc improved from 0.60440 to 0.61028, saving model to "vgg_o
nly2.h5
Epoch 22/50
200/200 [==============================] - 20s 100ms/step - loss: 1.1576 - au
c: 0.6126 - val_loss: 1.1153 - val_auc: 0.6155

Epoch 00022: val_auc improved from 0.61028 to 0.61545, saving model to "vgg_o
nly2.h5
Epoch 23/50
200/200 [==============================] - 20s 100ms/step - loss: 1.0903 - au
c: 0.6175 - val_loss: 1.0546 - val_auc: 0.6198

Epoch 00023: val_auc improved from 0.61545 to 0.61981, saving model to "vgg_o
nly2.h5
Epoch 24/50
200/200 [==============================] - 20s 99ms/step - loss: 1.0337 - au
c: 0.6217 - val_loss: 1.0045 - val_auc: 0.6236

Epoch 00024: val_auc improved from 0.61981 to 0.62358, saving model to "vgg_o
nly2.h5
Epoch 25/50
200/200 [==============================] - 20s 101ms/step - loss: 0.9702 - au
c: 0.6254 - val_loss: 0.9616 - val_auc: 0.6277

Epoch 00025: val_auc improved from 0.62358 to 0.62768, saving model to "vgg_o
nly2.h5
Epoch 26/50
200/200 [==============================] - 20s 100ms/step - loss: 0.9460 - au
c: 0.6287 - val_loss: 0.9134 - val_auc: 0.6305

Epoch 00026: val_auc improved from 0.62768 to 0.63050, saving model to "vgg_o
nly2.h5
Epoch 27/50
200/200 [==============================] - 20s 101ms/step - loss: 0.9017 - au
c: 0.6323 - val_loss: 0.8712 - val_auc: 0.6341

Epoch 00027: val_auc improved from 0.63050 to 0.63411, saving model to "vgg_o
nly2.h5
Epoch 28/50
200/200 [==============================] - 20s 102ms/step - loss: 0.8752 - au
c: 0.6355 - val_loss: 0.8328 - val_auc: 0.6372

Epoch 00028: val_auc improved from 0.63411 to 0.63717, saving model to "vgg_o
nly2.h5
Epoch 29/50
200/200 [==============================] - 20s 100ms/step - loss: 0.8410 - au
c: 0.6389 - val_loss: 0.8155 - val_auc: 0.6407

Epoch 00029: val_auc improved from 0.63717 to 0.64073, saving model to "vgg_o
nly2.h5
Epoch 30/50
200/200 [==============================] - 20s 101ms/step - loss: 0.8183 - au
c: 0.6420 - val_loss: 0.8022 - val_auc: 0.6435

Epoch 00030: val_auc improved from 0.64073 to 0.64346, saving model to "vgg_o
```

```
nly2.h5
Epoch 31/50
200/200 [==============================] - 20s 100ms/step - loss: 0.8005 - au
c: 0.6447 - val_loss: 0.7831 - val_auc: 0.6462

Epoch 00031: val_auc improved from 0.64346 to 0.64616, saving model to "vgg_o
nly2.h5
Epoch 32/50
200/200 [==============================] - 20s 100ms/step - loss: 0.7768 - au
c: 0.6473 - val_loss: 0.7823 - val_auc: 0.6491

Epoch 00032: val_auc improved from 0.64616 to 0.64906, saving model to "vgg_o
nly2.h5
Epoch 33/50
200/200 [==============================] - 20s 100ms/step - loss: 0.7633 - au
c: 0.6504 - val_loss: 0.7582 - val_auc: 0.6518

Epoch 00033: val_auc improved from 0.64906 to 0.65176, saving model to "vgg_o
nly2.h5
Epoch 34/50
200/200 [==============================] - 20s 99ms/step - loss: 0.7601 - au
c: 0.6528 - val_loss: 0.7371 - val_auc: 0.6540

Epoch 00034: val_auc improved from 0.65176 to 0.65395, saving model to "vgg_o
nly2.h5
Epoch 35/50
200/200 [==============================] - 20s 101ms/step - loss: 0.7426 - au
c: 0.6551 - val_loss: 0.7394 - val_auc: 0.6561

Epoch 00035: val_auc improved from 0.65395 to 0.65609, saving model to "vgg_o
nly2.h5
Epoch 36/50
200/200 [==============================] - 20s 102ms/step - loss: 0.7296 - au
c: 0.6573 - val_loss: 0.7393 - val_auc: 0.6585

Epoch 00036: val_auc improved from 0.65609 to 0.65854, saving model to "vgg_o
nly2.h5
Epoch 37/50
200/200 [==============================] - 20s 102ms/step - loss: 0.7211 - au
c: 0.6596 - val_loss: 0.7060 - val_auc: 0.6609

Epoch 00037: val_auc improved from 0.65854 to 0.66086, saving model to "vgg_o
nly2.h5
Epoch 38/50
200/200 [==============================] - 20s 100ms/step - loss: 0.7067 - au
c: 0.6619 - val_loss: 0.7066 - val_auc: 0.6633

Epoch 00038: val_auc improved from 0.66086 to 0.66333, saving model to "vgg_o
nly2.h5
Epoch 39/50
200/200 [==============================] - 20s 99ms/step - loss: 0.6900 - au
c: 0.6644 - val_loss: 0.7070 - val_auc: 0.6658

Epoch 00039: val_auc improved from 0.66333 to 0.66585, saving model to "vgg_o
nly2.h5
Epoch 40/50
200/200 [==============================] - 20s 102ms/step - loss: 0.6898 - au
```

```
c: 0.6669 - val_loss: 0.7034 - val_auc: 0.6680

Epoch 00040: val_auc improved from 0.66585 to 0.66804, saving model to "vgg_o
nly2.h5
Epoch 41/50
200/200 [==============================] - 20s 102ms/step - loss: 0.6866 - au
c: 0.6690 - val_loss: 0.6871 - val_auc: 0.6700

Epoch 00041: val_auc improved from 0.66804 to 0.67000, saving model to "vgg_o
nly2.h5
Epoch 42/50
200/200 [==============================] - 20s 100ms/step - loss: 0.6786 - au
c: 0.6712 - val_loss: 0.6937 - val_auc: 0.6724

Epoch 00042: val_auc improved from 0.67000 to 0.67238, saving model to "vgg_o
nly2.h5
Epoch 43/50
200/200 [==============================] - 21s 103ms/step - loss: 0.6748 - au
c: 0.6734 - val_loss: 0.6730 - val_auc: 0.6744

Epoch 00043: val_auc improved from 0.67238 to 0.67439, saving model to "vgg_o
nly2.h5
Epoch 44/50
200/200 [==============================] - 20s 101ms/step - loss: 0.6590 - au
c: 0.6755 - val_loss: 0.6712 - val_auc: 0.6765

Epoch 00044: val_auc improved from 0.67439 to 0.67646, saving model to "vgg_o
nly2.h5
Epoch 45/50
200/200 [==============================] - 20s 100ms/step - loss: 0.6652 - au
c: 0.6774 - val_loss: 0.6874 - val_auc: 0.6785

Epoch 00045: val_auc improved from 0.67646 to 0.67852, saving model to "vgg_o
nly2.h5
Epoch 46/50
200/200 [==============================] - 20s 100ms/step - loss: 0.6673 - au
c: 0.6794 - val_loss: 0.6817 - val_auc: 0.6802

Epoch 00046: val_auc improved from 0.67852 to 0.68025, saving model to "vgg_o
nly2.h5
Epoch 47/50
200/200 [==============================] - 20s 101ms/step - loss: 0.6540 - au
c: 0.6811 - val_loss: 0.6810 - val_auc: 0.6822

Epoch 00047: val_auc improved from 0.68025 to 0.68225, saving model to "vgg_o
nly2.h5
Epoch 48/50
200/200 [==============================] - 20s 101ms/step - loss: 0.6406 - au
c: 0.6834 - val_loss: 0.6529 - val_auc: 0.6846

Epoch 00048: val_auc improved from 0.68225 to 0.68462, saving model to "vgg_o
nly2.h5
Epoch 49/50
200/200 [==============================] - 20s 100ms/step - loss: 0.6536 - au
c: 0.6854 - val_loss: 0.6661 - val_auc: 0.6864

Epoch 00049: val_auc improved from 0.68462 to 0.68636, saving model to "vgg_o
```

```
nly2.h5
Epoch 50/50
200/200 [==============================] - 20s 101ms/step - loss: 0.6419 - au
c: 0.6872 - val_loss: 0.6282 - val_auc: 0.6882

Epoch 00050: val_auc improved from 0.68636 to 0.68821, saving model to "vgg_o
nly2.h5
```

Visualizing metric using tensorboard

In [0]:
```
%load_ext tensorboard
```

```
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
```

In [0]:
```
%tensorboard --logdir log_dir
```

Predicting the probability on test data.

In [0]:
```
predictions = []

for batch in tqdm(chunker(submission.img_pair.values)):
    X1 = [x.split("-")[0] for x in batch]
    X1 = np.array([read_img(test_path + x) for x in X1])

    X2 = [x.split("-")[1] for x in batch]
    X2 = np.array([read_img(test_path + x) for x in X2])

    pred = model.predict([X1, X2]).ravel().tolist()
    predictions += pred

submission['is_related'] = predictions

submission.to_csv("vgg_only2.csv", index=False)
```

```
166it [02:40,  1.13it/s]
```

After trainig the model with suffucuent number of epochs, it gave 0.749 private score and 0.744 public score on the test dataset.

Finally we blend all the four diffenent models and took the weighted average.

In [0]:
```
sub1=pd.read_csv('face_vgg.csv')
sub2=pd.read_csv('vgg_only0.csv')
sub3=pd.read_csv('vgg_only1.csv')
sub4=pd.read_csv('vgg_only2.csv')
```

In [0]:
```
submission = pd.read_csv('sample_submission.csv')

submission['is_related'] = (0.4*sub1['is_related'] + 0.2*sub2['is_related'] +
0.2*sub3['is_related'] + 0.2*sub4['is_related'])
submission.to_csv('sub.csv', index=False )
```

Key findings:

> Instead of one single best model we can use simple/weighted average of different stand alone models to increase our scores.
>
> facenet and vggface model works well when training the bottlenesck features instead of using pretrained ones.
>
> Operations like adding features of same image from different base models capture unique more features of face and yields better results.