```
%tensorflow_version 1.x
```

얼굴 인식

copy from : https://github.com/santhalakshminarayana/face-recognition/blob/master/Face_Recognition.ipynb

아티클 : https://medium.com/analytics-vidhya/face-recognition-with-vgg-face-in-keras-96e6bc1951d5

데이터 준비

```
%cd /content
```

```
/content
```

```
!rm -rf __MACOSX
!rm -rf face_data.zip
!rm -rf face_data
```

https://github.com/eunahlee-viola/WISET-D_Offline/blob/master/Day3/Test_images.zip

아래 쎌에 들어있던 이미지 데이터 주소
https://github.com/dhrim/wiset_2020_06/raw/master/material/deep_learning/face_data.zip
https://github.com/eunahlee-viola/WISET-D_Offline/blob/master/Day3/face_data.zip

```
#!wget https://github.com/eunahlee-viola/WISET-D_Offline/blob/master/Day3/face_data.zip
```

```
--2020-06-24 01:26:38--  https://github.com/eunahlee-viola/WISET-D_Offline/blob/master/Day3/
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: ‘face_data.zip.1’

face_data.zip.1         [ <=>                ]  72.68K   206KB/s    in 0.4s

2020-06-24 01:26:40 (206 KB/s) - ‘face_data.zip.1’ saved [74427]
```

이유는 모르겠으나, github에서 가져온 zip 파일이 70kb..

Github에서 zip 파일 불러오는게 실패인듯

이후로 아무것도 안됨.. ㅜ.ㅜ

```
#
```

```
Archive:  face_data.zip
  End-of-central-directory signature not found.  Either this file is not
  a zipfile, or it constitutes one disk of a multi-part archive.  In the
  latter case the central directory and zipfile comment will be found on
  the last disk(s) of this archive.
unzip:  cannot find zipfile directory in one of face_data.zip or
        face_data.zip.zip, and cannot find face_data.zip.ZIP, period.
```

데이터 준비

아래와 같이 데이터를 준비한다.

```
face_data/
  Images/
    jobs_1.jpg
    jobs_2.jpg
    ...
    faker_1.jpg
    faker_2.jpg
    ...
    iu_1.jpg
    iu_2.jpg
  Test_images/
    jobs_11.jpg
    jobs_12.jpg
    ...
    faker_11.jpg
    faker_12.jpg
    ...
    iu_11.jpg
    iu_12.jpg
```

```
from google.colab import drive #EunAh: Google drive에 올리고 가져오는 과정 실행하고 나서 인증 필요
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/
```

```
path='/content/face_data'
```

```
!cp "/content/drive/My Drive/content/face_data.zip" ./ #EunAh: 현재 위치에 가져오는 코드
```

```
!unzip face_data.zip #EunAh: 위에 파일가져오는 코드를 추가해 넣으면서 unzip 코드를 다시 추가했음
```

```
Archive:  face_data.zip
  inflating: Images/faker_1.jpg
  inflating: Images/faker_10.jpg
  inflating: Images/faker_11.jpg
  inflating: Images/faker_15.jpg
  inflating: Images/faker_16.jpg
  inflating: Images/faker_18.jpg
  inflating: Images/faker_2.jpg
  inflating: Images/faker_21.jpg
  inflating: Images/faker_22.jpg
  inflating: Images/faker_23.jpg
  inflating: Images/faker_24.jpg
  inflating: Images/faker_3.jpg
 extracting: Images/faker_7.jpg
  inflating: Images/faker_8.jpg
  inflating: Images/faker_9.jpg
  inflating: Images/iu_10.jpg
  inflating: Images/iu_11.jpg
  inflating: Images/iu_12.jpg
  inflating: Images/iu_13.jpg
  inflating: Images/iu_14.jpg
  inflating: Images/iu_15.jpg
  inflating: Images/iu_16.jpg
  inflating: Images/iu_17.jpg
```

```
!mkdir face_data #EunAH: 압축 마다 방식이 달라 경로가 달라졌음. 이를 바로잡기 위해 파일 경로를 바로
!mv Test_images face_data/
!mv Images face_data/
```

```
  inflating: Images/iu_6.jpg
```

```
path='/content/face_data'
```

```
  inflating: Images/jobs_14.jpeg
```

```
%cd $path
```

```
/content/face_data
```

```
  inflating: Images/jobs_26.jpg
```

```
import os
import glob
```

```
  inflating: Images/jobs_7.jpg
```

```
# Get Image names stored in "Images" folder
image_path_names=[]
person_names=set()
for file_name in glob.glob(path+'/Images/*_[1-9]*.jpg'):
  image_path_names.append(file_name)
  person_names.add(image_path_names[-1].split('/')[-1].split('_')[0])
```

```
  inflating: Images/lea_5.jpg
```

```
len(image_path_names)
```

```
48
```

```
  inflating: Test_images/faker_28.jpg
```

```
person_names
```

```
{'faker', 'iu', 'jobs', 'lea'}
```

There are total 60 images containing 10 images per person.

얼굴부분 crop

얼굴 부분 탐지하여 crop한다.

dlib의 face detector를 사용

```
# Download Dlib CNN face detector
! wget http://dlib.net/files/mmod_human_face_detector.dat.bz2
```

```
--2020-06-24 03:40:11--  http://dlib.net/files/mmod_human_face_detector.dat.bz2
Resolving dlib.net (dlib.net)... 107.180.26.78
Connecting to dlib.net (dlib.net)|107.180.26.78|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 694709 (678K)
Saving to: ‘mmod_human_face_detector.dat.bz2’

mmod_human_face_det 100%[===================>] 678.43K   522KB/s    in 1.3s

2020-06-24 03:40:13 (522 KB/s) - ‘mmod_human_face_detector.dat.bz2’ saved [694709/694709]
```

```
!bzip2 -dk mmod_human_face_detector.dat.bz2
```

```
%rm mmod_human_face_detector.dat.bz2
```

```
import cv2
import matplotlib.pyplot as plt
import dlib
```

DLIB FaceDetector 로딩

Images 폴더 파일들 Crop

Images 폴더의 파일들을 읽어 Images_crop에 생성

```
# Load CNN face detector into dlib
dnnFaceDetector=dlib.cnn_face_detection_model_v1("mmod_human_face_detector.dat")
```

```
os.mkdir(path+'/Images_crop/')
```

```
# For each person create a separate folder
for person in person_names:
```

```
os.mkdir(path+'/Images_crop/'+person+'/')
```

```
# Detect face, crop detected face and save them in corresponding person folder
for file_name in image_path_names:
  img=cv2.imread(file_name)
  gray=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
  rects=dnnFaceDetector(gray,1)
  left,top,right,bottom=0,0,0,0
  for (i,rect) in enumerate(rects):
    left=rect.rect.left() #x1
    top=rect.rect.top() #y1
    right=rect.rect.right() #x2
    bottom=rect.rect.bottom() #y2
  if right==0: continue
  width=right-left
  height=bottom-top
  img_crop=img[top:top+height,left:left+width]
  img_path=path+'/Images_crop/'+file_name.split('/')[-1].split('_')[0]+'/'+file_name.split('/')[-1]
  cv2.imwrite(img_path,img_crop)
```

```
from IPython.display import Image
display(Image('Images/iu_10.jpg'))
display(Image('Images_crop/iu/iu_10.jpg'))
```

```
# Get Image names for testing
test_image_path_names=[]
for file_name in glob.glob(path+'/Test_images/*_*.jpg'):
  test_image_path_names.append(file_name)
```

```
len(test_image_path_names)
```

25

Test_Image 파일들 crop

Test_Images의 파일들을 읽어 Test_Images_crop에 생성

```
os.mkdir(path+'/Test_Images_crop/')
```

```
# Create Separate folder for each person in "Test_Images_crop" folder
for person in person_names:
  os.mkdir(path+'/Test_Images_crop/'+person+'/')
```

```
# Detect face,crop face and save in corresponding folder
for file_name in test_image_path_names:
  img=cv2.imread(file_name)
  gray=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
  rects=dnnFaceDetector(gray,1)
  left,top,right,bottom=0,0,0,0
  for (i,rect) in enumerate(rects):
    left=rect.rect.left() #x1
    top=rect.rect.top() #y1
    right=rect.rect.right() #x2
    bottom=rect.rect.bottom() #y2
  if right==0: continue
  width=right-left
  height=bottom-top
  img_crop=img[top:top+height,left:left+width]
  img_path=path+'/Test_Images_crop/'+file_name.split('/')[-1].split('_')[0]+'/'+file_name.split('/
  cv2.imwrite(img_path,img_crop)
```

```
Here images are sorted to corresponding test and train folders of same person
Directory structure :
|Images /
|  |-- (60 images)
|Images_crop /
|  |--faker
|     |--(images)
|  |--iu /
|     |--(images)
|  |--jobs /
|     |--(imgaes)
|Test_Images /
|  |-- .. / (18 images)
|Test_Images_crop /
|  |--faker
|     |--(images)
|  |--iu /
|     |--(images)
|  |--jobs /
|     |--(imgaes)
|mmod_human_face_detector.dat
```

VGG Face Model 다운로드

```
! pip install gdown
```

```
Requirement already satisfied: gdown in /usr/local/lib/python3.6/dist-packages (3.6.4)
Requirement already satisfied: tqdm in /usr/local/lib/python3.6/dist-packages (from gdown) (4
Requirement already satisfied: requests in /usr/local/lib/python3.6/dist-packages (from gdown
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from gdown) (1
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/pyth
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist-packages (
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-packages (from
```

```
#Download pre-trained vgg-face-model-weights as .h5 file
! gdown https://drive.google.com/uc?id=1CPSeum3HpopfomUEK1gybeulVoeJT_Eo
```

```
Permission denied: https://drive.google.com/uc?id=1CPSeum3HpopfomUEK1gybeulVoeJT_Eo
Maybe you need to change permission over 'Anyone with the link'?
```

```
# EunAh: 위 Weight 가져오는데 permission denied 되어서 다운로드 받아 구글드라이브 링크를 추가
! gdown https://drive.google.com/file/d/1bgCVdhhwn_pH2MFw18iFcdven3Xo_7Ed/view?usp=sharing
```

```
/usr/local/lib/python2.7/dist-packages/gdown/parse_url.py:31: UserWarning: You specified Goo
  .format(url='https://drive.google.com/uc?id={}'.format(file_id))
Downloading...
From: https://drive.google.com/file/d/1bgCVdhhwn_pH2MFw18iFcdven3Xo_7Ed/view?usp=sharing
To: /content/face_data/view?usp=sharing
69.9kB [00:00, 605kB/s]
```

```
# EunAh: 그래도 안돼서 dropbox 링크로 추가
! gdown https://www.dropbox.com/s/favuqa1zp09dfz7/vgg_face_weights.h5?dl=0
```

```
Downloading...
From: https://www.dropbox.com/s/favuqa1zp09dfz7/vgg_face_weights.h5?dl=0
To: /content/face_data/vgg_face_weights.h5?dl=0
377kB [00:00, 1.75MB/s]
```

파일이름이 달라진것 같음 끝에 이상한 "?dl=0" 이 붙었음. 어떻게 떼지? 이것 때문에 아래에서 불러오지 못하고 에러 발생하는듯

```
!mv vgg_face_weights.h5?dl=0 vgg_face_weights.ht #EunAh: 리눅스 파일이름 변경해주는 명령어
```

```
!mv vgg_face_weights.ht vgg_face_weights.h5 #EunAh: 오타 때문에 다시 실행 ㅜ.ㅜ
```

```
%ls -al
```

```
total 1108
drwxr-xr-x 6 root root   4096 Jun 24 03:42 ./
drwxr-xr-x 1 root root   4096 Jun 24 03:39 ../
drwxr-xr-x 2 root root   4096 Jun 24 03:39 Images/
drwxr-xr-x 6 root root   4096 Jun 24 03:41 Images_crop/
-rw-r--r-- 1 root root 729940 Oct  8  2016 mmod_human_face_detector.dat
drwxr-xr-x 2 root root   4096 Jun 24 03:39 Test_images/
drwxr-xr-x 6 root root   4096 Jun 24 03:41 Test_Images_crop/
-rw-r--r-- 1 root root 376528 Jun 24 03:41 vgg_face_weights.h5
```

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential,Model
from tensorflow.keras.layers import ZeroPadding2D,Convolution2D,MaxPooling2D
from tensorflow.keras.layers import Dense,Dropout,Softmax,Flatten,Activation,BatchNormalization
from tensorflow.keras.preprocessing.image import load_img,img_to_array
from tensorflow.keras.applications.imagenet_utils import preprocess_input
import tensorflow.keras.backend as K
```

```
#Define VGG_FACE_MODEL architecture
model = Sequential()
model.add(ZeroPadding2D((1,1),input_shape=(224,224, 3)))
model.add(Convolution2D(64, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(128, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))
model.add(Convolution2D(4096, (7, 7), activation='relu'))
model.add(Dropout(0.5))
model.add(Convolution2D(4096, (1, 1), activation='relu'))
```

```
model.add(Convolution2D(4096, (1, 1), activation='relu'))
model.add(Dropout(0.5))
model.add(Convolution2D(2622, (1, 1)))
model.add(Flatten())
model.add(Activation('softmax'))
```

```
# Load VGG Face model weights
model.load_weights('vgg_face_weights.h5')
```

```
---------------------------------------------------------------------------
OSError                                   Traceback (most recent call last)
<ipython-input-192-aab7e4770c63> in <module>()
      1 # Load VGG Face model weights
----> 2 model.load_weights('vgg_face_weights.h5')

                                 3 frames
/usr/local/lib/python3.6/dist-packages/h5py/_hl/files.py in make_fid(name, mode, userblock_s
    171         if swmr and swmr_support:
    172             flags |= h5f.ACC_SWMR_READ
--> 173         fid = h5f.open(name, flags, fapl=fapl)
    174     elif mode == 'r+':
    175         fid = h5f.open(name, h5f.ACC_RDWR, fapl=fapl)

h5py/_objects.pyx in h5py._objects.with_phil.wrapper()

h5py/_objects.pyx in h5py._objects.with_phil.wrapper()

h5py/h5f.pyx in h5py.h5f.open()

OSError: Unable to open file (file signature not found)

SEARCH STACK OVERFLOW
```

```
model.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
zero_padding2d_39 (ZeroPaddi (None, 226, 226, 3)       0
_________________________________________________________________
conv2d_48 (Conv2D)           (None, 224, 224, 64)      1792
_________________________________________________________________
zero_padding2d_40 (ZeroPaddi (None, 226, 226, 64)      0
_________________________________________________________________
conv2d_49 (Conv2D)           (None, 224, 224, 64)      36928
_________________________________________________________________
max_pooling2d_15 (MaxPooling (None, 112, 112, 64)      0
_________________________________________________________________
zero_padding2d_41 (ZeroPaddi (None, 114, 114, 64)      0
_________________________________________________________________
conv2d_50 (Conv2D)           (None, 112, 112, 128)     73856
_________________________________________________________________
zero_padding2d_42 (ZeroPaddi (None, 114, 114, 128)     0
_________________________________________________________________
conv2d_51 (Conv2D)           (None, 112, 112, 128)     147584
_________________________________________________________________
max_pooling2d_16 (MaxPooling (None, 56, 56, 128)       0
_________________________________________________________________
zero_padding2d_43 (ZeroPaddi (None, 58, 58, 128)       0
_________________________________________________________________
conv2d_52 (Conv2D)           (None, 56, 56, 256)       295168
_________________________________________________________________
zero_padding2d_44 (ZeroPaddi (None, 58, 58, 256)       0
_________________________________________________________________
conv2d_53 (Conv2D)           (None, 56, 56, 256)       590080
_________________________________________________________________
zero_padding2d_45 (ZeroPaddi (None, 58, 58, 256)       0
_________________________________________________________________
conv2d_54 (Conv2D)           (None, 56, 56, 256)       590080
_________________________________________________________________
max_pooling2d_17 (MaxPooling (None, 28, 28, 256)       0
_________________________________________________________________
zero_padding2d_46 (ZeroPaddi (None, 30, 30, 256)       0
_________________________________________________________________
conv2d_55 (Conv2D)           (None, 28, 28, 512)       1180160
_________________________________________________________________
zero_padding2d_47 (ZeroPaddi (None, 30, 30, 512)       0
_________________________________________________________________
conv2d_56 (Conv2D)           (None, 28, 28, 512)       2359808
_________________________________________________________________
zero_padding2d_48 (ZeroPaddi (None, 30, 30, 512)       0
_________________________________________________________________
conv2d_57 (Conv2D)           (None, 28, 28, 512)       2359808
_________________________________________________________________
max_pooling2d_18 (MaxPooling (None, 14, 14, 512)       0
_________________________________________________________________
zero_padding2d_49 (ZeroPaddi (None, 16, 16, 512)       0
_________________________________________________________________
conv2d_58 (Conv2D)           (None, 14, 14, 512)       2359808
_________________________________________________________________
zero_padding2d_50 (ZeroPaddi (None, 16, 16, 512)       0
```

```
conv2d_59 (Conv2D)           (None, 14, 14, 512)       2359808
zero_padding2d_51 (ZeroPaddi (None, 16, 16, 512)       0
conv2d_60 (Conv2D)           (None, 14, 14, 512)       2359808
```

```
# Remove Last Softmax layer and get model upto last flatten layer with outputs 2622 units
vgg_face=Model(inputs=model.layers[0].input,outputs=model.layers[-2].output)
```

```
conv2d_61 (Conv2D)           (None, 1, 1, 4096)        102764544
```

학습 모델 준비

```
#Prepare Training Data
x_train=[]
y_train=[]
person_folders=os.listdir(path+'/Images_crop/')
person_rep=dict()
for i,person in enumerate(person_folders):
  person_rep[i]=person
  image_names=os.listdir('Images_crop/'+person+'/')
  for image_name in image_names:
    img=load_img(path+'/Images_crop/'+person+'/'+image_name,target_size=(224,224))
    img=img_to_array(img)
    img=np.expand_dims(img,axis=0)
    img=preprocess_input(img)
    img_encode=vgg_face(img)
    x_train.append(np.squeeze(K.eval(img_encode)).tolist())
    y_train.append(i)
```

```
person_rep
```

```
{0: 'lea', 1: 'faker', 2: 'iu', 3: 'jobs'}
```

```
x_train=np.array(x_train)
y_train=np.array(y_train)
```

```
#Prepare Test Data
x_test=[]
y_test=[]
person_folders=os.listdir(path+'/Test_Images_crop/')
for i,person in enumerate(person_folders):
  image_names=os.listdir('Test_Images_crop/'+person+'/')
  for image_name in image_names:
    img=load_img(path+'/Test_Images_crop/'+person+'/'+image_name,target_size=(224,224))
    img=img_to_array(img)
    img=np.expand_dims(img,axis=0)
    img=preprocess_input(img)
    img_encode=vgg_face(img)
    x_test.append(np.squeeze(K.eval(img_encode)).tolist())
    y_test.append(i)
```

```
x_test=np.array(x_test)
y_test=np.array(y_test)
```

```
# Save test and train data for later use
np.save('train_data',x_train)
np.save('train_labels',y_train)
np.save('test_data',x_test)
np.save('test_labels',y_test)
```

```
# Load saved data
x_train=np.load('train_data.npy')
y_train=np.load('train_labels.npy')
x_test=np.load('test_data.npy')
y_test=np.load('test_labels.npy')
```

모델 정의

```
# Softmax regressor to classify images based on encoding
classifier_model=Sequential()
classifier_model.add(Dense(units=100,input_dim=x_train.shape[1],kernel_initializer='glorot_uniform
classifier_model.add(BatchNormalization())
classifier_model.add(Activation('tanh'))
classifier_model.add(Dropout(0.3))
classifier_model.add(Dense(units=10,kernel_initializer='glorot_uniform'))
classifier_model.add(BatchNormalization())
classifier_model.add(Activation('tanh'))
classifier_model.add(Dropout(0.2))
classifier_model.add(Dense(units=6,kernel_initializer='he_uniform'))
classifier_model.add(Activation('softmax'))
classifier_model.compile(loss=tf.keras.losses.SparseCategoricalCrossentropy(),optimizer='nadam',met
```

모델 학습

```
classifier_model.fit(x_train,y_train,epochs=100,validation_data=(x_test,y_test))
```

```
Train on 47 samples, validate on 25 samples
Epoch 1/100
47/47 [==============================] - 1s 29ms/sample - loss: 1.7646 - acc: 0.2979 - val_l
Epoch 2/100
47/47 [==============================] - 0s 681us/sample - loss: 1.3303 - acc: 0.5532 - val_
Epoch 3/100
47/47 [==============================] - 0s 528us/sample - loss: 0.9666 - acc: 0.7872 - val_
Epoch 4/100
47/47 [==============================] - 0s 593us/sample - loss: 0.9826 - acc: 0.7234 - val_
Epoch 5/100
47/47 [==============================] - 0s 487us/sample - loss: 0.9499 - acc: 0.7660 - val_
Epoch 6/100
47/47 [==============================] - 0s 505us/sample - loss: 0.7605 - acc: 0.8298 - val_
Epoch 7/100
47/47 [==============================] - 0s 518us/sample - loss: 0.6907 - acc: 0.8723 - val_
Epoch 8/100
47/47 [==============================] - 0s 552us/sample - loss: 0.6962 - acc: 0.9362 - val_
Epoch 9/100
47/47 [==============================] - 0s 630us/sample - loss: 0.6291 - acc: 0.9149 - val_
Epoch 10/100
47/47 [==============================] - 0s 533us/sample - loss: 0.6298 - acc: 0.8723 - val_
Epoch 11/100
47/47 [==============================] - 0s 499us/sample - loss: 0.5661 - acc: 1.0000 - val_
Epoch 12/100
47/47 [==============================] - 0s 494us/sample - loss: 0.5096 - acc: 0.9787 - val_
Epoch 13/100
47/47 [==============================] - 0s 548us/sample - loss: 0.5517 - acc: 0.9787 - val_
Epoch 14/100
47/47 [==============================] - 0s 686us/sample - loss: 0.5160 - acc: 0.9787 - val_
Epoch 15/100
47/47 [==============================] - 0s 652us/sample - loss: 0.5499 - acc: 0.9787 - val_
Epoch 16/100
47/47 [==============================] - 0s 646us/sample - loss: 0.5113 - acc: 1.0000 - val_
Epoch 17/100
47/47 [==============================] - 0s 534us/sample - loss: 0.4991 - acc: 1.0000 - val_
Epoch 18/100
47/47 [==============================] - 0s 617us/sample - loss: 0.4549 - acc: 0.9787 - val_
Epoch 19/100
47/47 [==============================] - 0s 630us/sample - loss: 0.5050 - acc: 1.0000 - val_
Epoch 20/100
47/47 [==============================] - 0s 530us/sample - loss: 0.5368 - acc: 1.0000 - val_
Epoch 21/100
47/47 [==============================] - 0s 538us/sample - loss: 0.5030 - acc: 1.0000 - val_
Epoch 22/100
47/47 [==============================] - 0s 504us/sample - loss: 0.4863 - acc: 0.9362 - val_
Epoch 23/100
47/47 [==============================] - 0s 475us/sample - loss: 0.3972 - acc: 1.0000 - val_
Epoch 24/100
47/47 [==============================] - 0s 520us/sample - loss: 0.3923 - acc: 1.0000 - val_
Epoch 25/100
47/47 [==============================] - 0s 467us/sample - loss: 0.4359 - acc: 1.0000 - val_
Epoch 26/100
47/47 [==============================] - 0s 540us/sample - loss: 0.4419 - acc: 1.0000 - val_
Epoch 27/100
47/47 [==============================] - 0s 474us/sample - loss: 0.4821 - acc: 0.9362 - val_
Epoch 28/100
47/47 [==============================] - 0s 564us/sample - loss: 0.4401 - acc: 0.9787 - val_
Epoch 29/100
```

```
47/47 [==============================] - 0s 525us/sample - loss: 0.3814 - acc: 1.0000 - val_
Epoch 30/100
47/47 [==============================] - 0s 497us/sample - loss: 0.3921 - acc: 1.0000 - val_
Epoch 31/100
47/47 [==============================] - 0s 508us/sample - loss: 0.4058 - acc: 1.0000 - val_
Epoch 32/100
47/47 [==============================] - 0s 494us/sample - loss: 0.3627 - acc: 1.0000 - val_
Epoch 33/100
47/47 [==============================] - 0s 539us/sample - loss: 0.3309 - acc: 1.0000 - val_
Epoch 34/100
47/47 [==============================] - 0s 453us/sample - loss: 0.3696 - acc: 1.0000 - val_
Epoch 35/100
47/47 [==============================] - 0s 481us/sample - loss: 0.3316 - acc: 1.0000 - val_
Epoch 36/100
47/47 [==============================] - 0s 707us/sample - loss: 0.3674 - acc: 1.0000 - val_
Epoch 37/100
47/47 [==============================] - 0s 502us/sample - loss: 0.3840 - acc: 0.9574 - val_
Epoch 38/100
47/47 [==============================] - 0s 458us/sample - loss: 0.3857 - acc: 1.0000 - val_
Epoch 39/100
47/47 [==============================] - 0s 547us/sample - loss: 0.3897 - acc: 1.0000 - val_
Epoch 40/100
47/47 [==============================] - 0s 671us/sample - loss: 0.3686 - acc: 0.9787 - val_
Epoch 41/100
47/47 [==============================] - 0s 537us/sample - loss: 0.3565 - acc: 1.0000 - val_
Epoch 42/100
47/47 [==============================] - 0s 476us/sample - loss: 0.3427 - acc: 1.0000 - val_
Epoch 43/100
47/47 [==============================] - 0s 712us/sample - loss: 0.3143 - acc: 1.0000 - val_
Epoch 44/100
47/47 [==============================] - 0s 512us/sample - loss: 0.3429 - acc: 0.9787 - val_
Epoch 45/100
47/47 [==============================] - 0s 622us/sample - loss: 0.3203 - acc: 1.0000 - val_
Epoch 46/100
47/47 [==============================] - 0s 599us/sample - loss: 0.3285 - acc: 1.0000 - val_
Epoch 47/100
47/47 [==============================] - 0s 482us/sample - loss: 0.3150 - acc: 1.0000 - val_
Epoch 48/100
47/47 [==============================] - 0s 480us/sample - loss: 0.3339 - acc: 1.0000 - val_
Epoch 49/100
47/47 [==============================] - 0s 476us/sample - loss: 0.3066 - acc: 1.0000 - val_
Epoch 50/100
47/47 [==============================] - 0s 536us/sample - loss: 0.3200 - acc: 1.0000 - val_
Epoch 51/100
47/47 [==============================] - 0s 533us/sample - loss: 0.2843 - acc: 1.0000 - val_
Epoch 52/100
47/47 [==============================] - 0s 503us/sample - loss: 0.2824 - acc: 1.0000 - val_
Epoch 53/100
47/47 [==============================] - 0s 568us/sample - loss: 0.3580 - acc: 1.0000 - val_
Epoch 54/100
47/47 [==============================] - 0s 565us/sample - loss: 0.3045 - acc: 0.9787 - val_
Epoch 55/100
47/47 [==============================] - 0s 461us/sample - loss: 0.2745 - acc: 1.0000 - val_
Epoch 56/100
47/47 [==============================] - 0s 463us/sample - loss: 0.2534 - acc: 1.0000 - val_
Epoch 57/100
47/47 [==============================] - 0s 466us/sample - loss: 0.2471 - acc: 1.0000 - val_
Epoch 58/100
```

```
47/47 [==============================] - 0s 571us/sample - loss: 0.2335 - acc: 1.0000 - val_
Epoch 59/100
47/47 [==============================] - 0s 459us/sample - loss: 0.3003 - acc: 0.9787 - val_
Epoch 60/100
47/47 [==============================] - 0s 513us/sample - loss: 0.2651 - acc: 1.0000 - val_
Epoch 61/100
47/47 [==============================] - 0s 531us/sample - loss: 0.2444 - acc: 1.0000 - val_
Epoch 62/100
47/47 [==============================] - 0s 499us/sample - loss: 0.2352 - acc: 1.0000 - val_
Epoch 63/100
47/47 [==============================] - 0s 556us/sample - loss: 0.2421 - acc: 1.0000 - val_
Epoch 64/100
47/47 [==============================] - 0s 584us/sample - loss: 0.2713 - acc: 1.0000 - val_
Epoch 65/100
47/47 [==============================] - 0s 499us/sample - loss: 0.2485 - acc: 1.0000 - val_
Epoch 66/100
47/47 [==============================] - 0s 472us/sample - loss: 0.2725 - acc: 1.0000 - val_
Epoch 67/100
47/47 [==============================] - 0s 539us/sample - loss: 0.2188 - acc: 1.0000 - val_
Epoch 68/100
47/47 [==============================] - 0s 563us/sample - loss: 0.2426 - acc: 1.0000 - val_
Epoch 69/100
47/47 [==============================] - 0s 534us/sample - loss: 0.2191 - acc: 1.0000 - val_
Epoch 70/100
47/47 [==============================] - 0s 777us/sample - loss: 0.2376 - acc: 1.0000 - val_
Epoch 71/100
47/47 [==============================] - 0s 544us/sample - loss: 0.2238 - acc: 1.0000 - val_
Epoch 72/100
47/47 [==============================] - 0s 536us/sample - loss: 0.2642 - acc: 1.0000 - val_
Epoch 73/100
47/47 [==============================] - 0s 584us/sample - loss: 0.2605 - acc: 1.0000 - val_
Epoch 74/100
47/47 [==============================] - 0s 534us/sample - loss: 0.2402 - acc: 1.0000 - val_
Epoch 75/100
47/47 [==============================] - 0s 562us/sample - loss: 0.1749 - acc: 1.0000 - val_
Epoch 76/100
47/47 [==============================] - 0s 551us/sample - loss: 0.2064 - acc: 1.0000 - val_
Epoch 77/100
47/47 [==============================] - 0s 678us/sample - loss: 0.1910 - acc: 1.0000 - val_
Epoch 78/100
47/47 [==============================] - 0s 551us/sample - loss: 0.2010 - acc: 1.0000 - val_
Epoch 79/100
47/47 [==============================] - 0s 519us/sample - loss: 0.2384 - acc: 1.0000 - val_
Epoch 80/100
47/47 [==============================] - 0s 475us/sample - loss: 0.2427 - acc: 1.0000 - val_
Epoch 81/100
47/47 [==============================] - 0s 481us/sample - loss: 0.2168 - acc: 1.0000 - val_
Epoch 82/100
47/47 [==============================] - 0s 628us/sample - loss: 0.2027 - acc: 1.0000 - val_
Epoch 83/100
47/47 [==============================] - 0s 504us/sample - loss: 0.2264 - acc: 1.0000 - val_
Epoch 84/100
47/47 [==============================] - 0s 501us/sample - loss: 0.1830 - acc: 1.0000 - val_
Epoch 85/100
47/47 [==============================] - 0s 599us/sample - loss: 0.2059 - acc: 1.0000 - val_
Epoch 86/100
47/47 [==============================] - 0s 558us/sample - loss: 0.2197 - acc: 1.0000 - val_
Epoch 87/100
```

```
47/47 [==============================] - 0s 490us/sample - loss: 0.1748 - acc: 1.0000 - val_
Epoch 88/100
47/47 [==============================] - 0s 474us/sample - loss: 0.1811 - acc: 1.0000 - val_
Epoch 89/100
47/47 [==============================] - 0s 456us/sample - loss: 0.2135 - acc: 1.0000 - val_
Epoch 90/100
47/47 [==============================] - 0s 547us/sample - loss: 0.1825 - acc: 1.0000 - val_
Epoch 91/100
47/47 [==============================] - 0s 488us/sample - loss: 0.1816 - acc: 1.0000 - val_
Epoch 92/100
47/47 [==============================] - 0s 481us/sample - loss: 0.1651 - acc: 1.0000 - val_
Epoch 93/100
47/47 [==============================] - 0s 541us/sample - loss: 0.1671 - acc: 1.0000 - val_
```

```
# Save model for later use
tf.keras.models.save_model(classifier_model,'face_classifier_model.h5')
```

```
47/47 [==============================] - 0s 567us/sample - loss: 0.1535 - acc: 1.0000 - val_
```

```
# Load saved model
classifier_model=tf.keras.models.load_model('face_classifier_model.h5')
```

```
WARNING:tensorflow:From /tensorflow-1.15.2/python3.6/tensorflow_core/python/ops/init_ops.py:
Instructions for updating:
Call initializer instance with the dtype argument instead of passing it to the constructor
WARNING:tensorflow:From /tensorflow-1.15.2/python3.6/tensorflow_core/python/ops/init_ops.py:
Instructions for updating:
Call initializer instance with the dtype argument instead of passing it to the constructor
WARNING:tensorflow:From /tensorflow-1.15.2/python3.6/tensorflow_core/python/ops/init_ops.py:
Instructions for updating:
Call initializer instance with the dtype argument instead of passing it to the constructor
WARNING:tensorflow:From /tensorflow-1.15.2/python3.6/tensorflow_core/python/ops/init_ops.py:
Instructions for updating:
Call initializer instance with the dtype argument instead of passing it to the constructor
```

```
# Path to folder which contains images to be tested and predicted
test_images_path=path+'/Test_Images/'
```

```
dnnFaceDetector=dlib.cnn_face_detection_model_v1("mmod_human_face_detector.dat")
```

얼굴 인식 실행

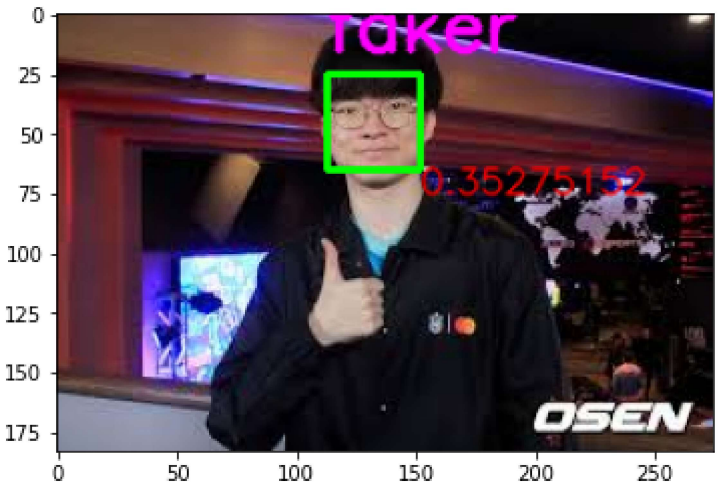
```
def plot(img):
  plt.figure(figsize=(8,4))
  plt.imshow(img[:,:,::-1])
  plt.show()
```

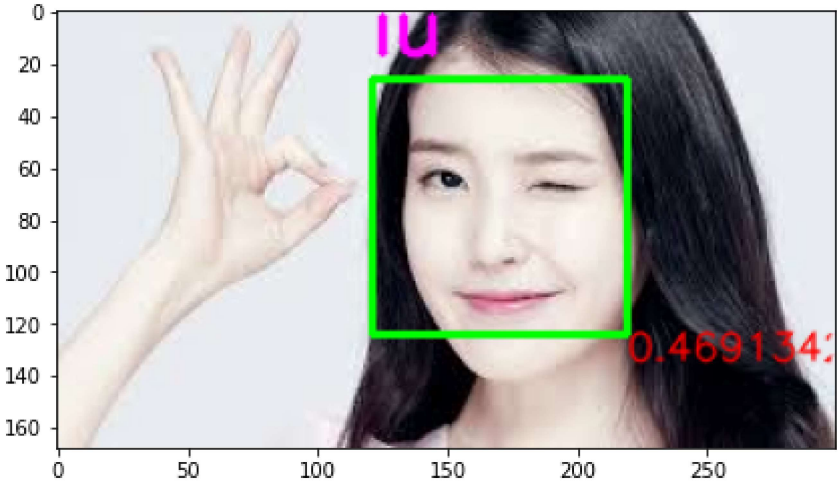
```
# Label names for class numbers
# person_rep={0: 'iu', 1: 'jobs', 2: 'faker'}
person_rep={0: 'lea', 1: 'faker', 2: 'iu', 3: 'jobs'} # person 추가
```

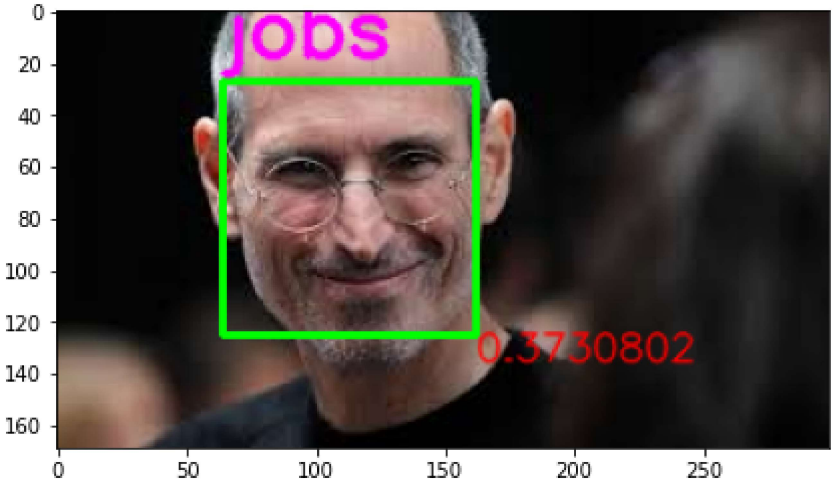
```
os.mkdir(path+'/Predictions')
```

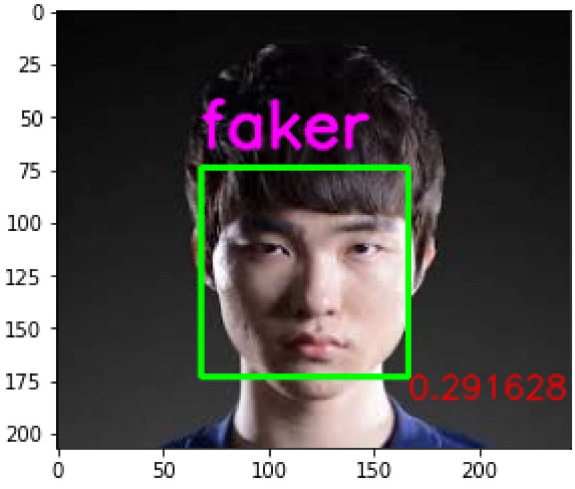
```
os.mkdir(path+'/Predictions')
```

```
for img_name in os.listdir(path+'/Test_images/'):
  if img_name=='crop_img.jpg':
    continue
  # Load Image
  img=cv2.imread(path+'/Test_images/'+img_name)
  gray=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

  # 얼굴 탐지
  rects=dnnFaceDetector(gray,1)
  left,top,right,bottom=0,0,0,0
  for (i,rect) in enumerate(rects):
    # Extract Each Face
    left=rect.rect.left() #x1
    top=rect.rect.top() #y1
    right=rect.rect.right() #x2
    bottom=rect.rect.bottom() #y2
    width=right-left
    height=bottom-top
    img_crop=img[top:top+height,left:left+width]
    cv2.imwrite(path+'/Test_images/crop_img.jpg',img_crop)

    # 얼굴 영역 crop
    crop_img=load_img(path+'/Test_images/crop_img.jpg',target_size=(224,224))
    crop_img=img_to_array(crop_img)
    crop_img=np.expand_dims(crop_img,axis=0)
    crop_img=preprocess_input(crop_img)
    img_encode=vgg_face(crop_img)

    # 얼굴 영역 인식
    embed=K.eval(img_encode)
    person=classifier_model.predict(embed)
    name=person_rep[np.argmax(person)]
    os.remove(path+'/Test_images/crop_img.jpg')
    cv2.rectangle(img,(left,top),(right,bottom),(0,255,0), 2)
    img=cv2.putText(img,name,(left,top-10),cv2.FONT_HERSHEY_SIMPLEX,1,(255,0,255),2,cv2.LINE_AA)
    img=cv2.putText(img,str(np.max(person)),(right,bottom+10),cv2.FONT_HERSHEY_SIMPLEX,0.5,(0,0,25
  # Save images with bounding box,name and accuracy
  cv2.imwrite(path+'/Predictions/'+img_name,img)
  plot(img)
```

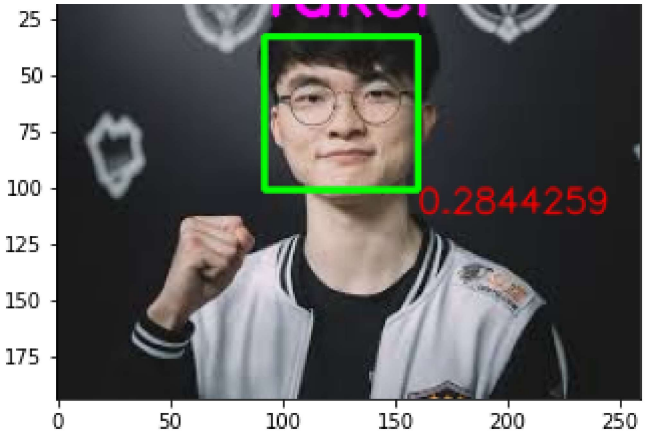

faker
0.35275152
OSEN


iu
0.469134

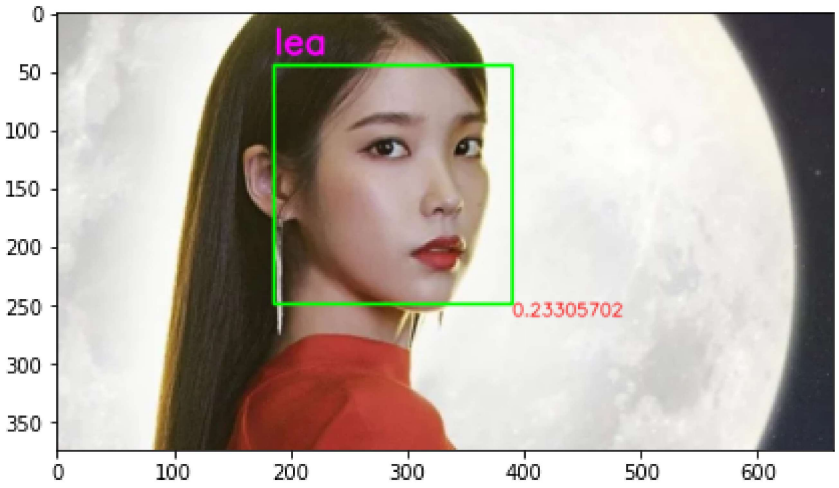

jobs
0.3730802

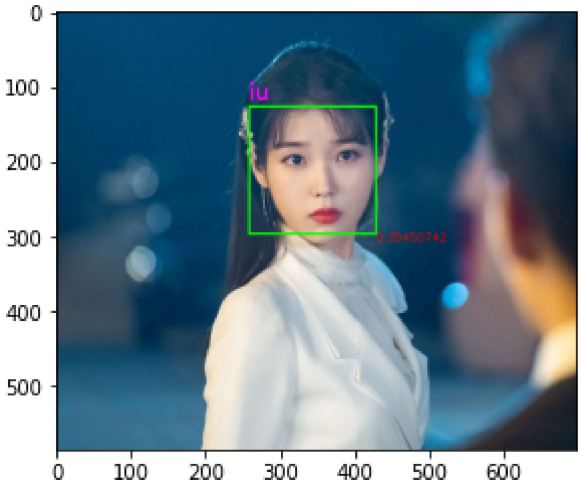

faker
0.291628

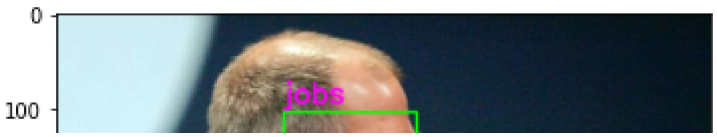

faker

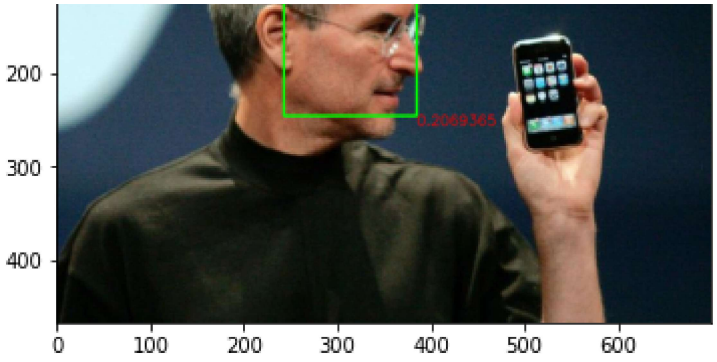

0.2844259

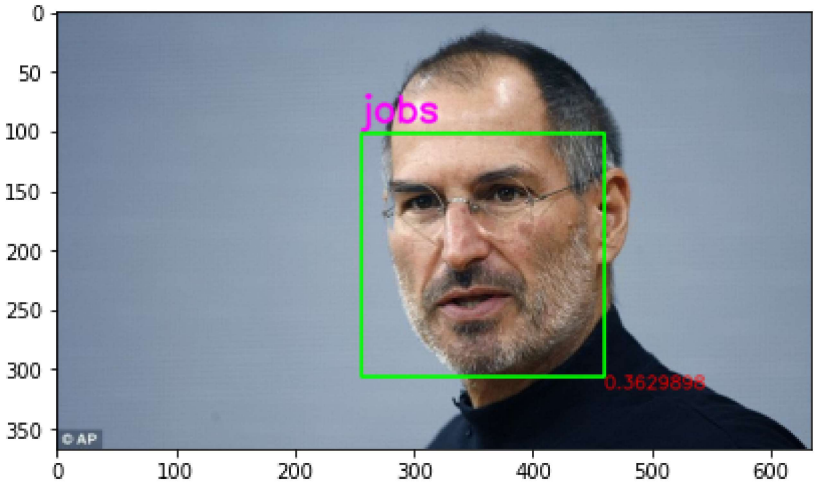

iu
fake
lea
0.2793071
0.24437045
0.22214322

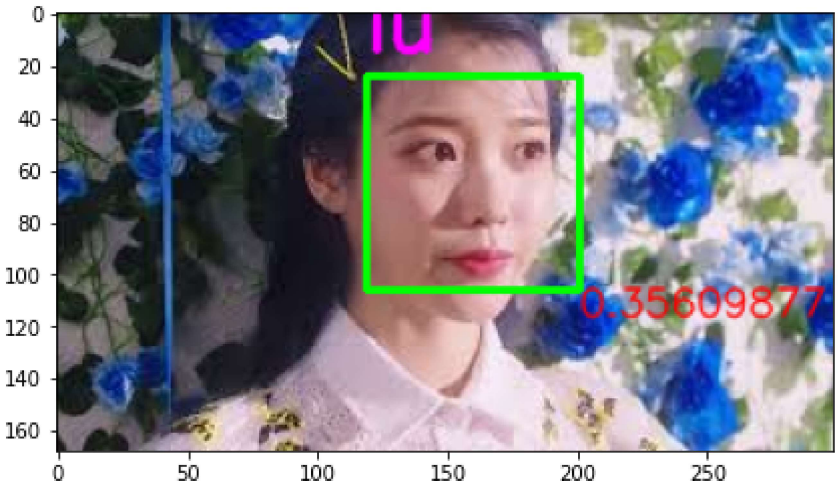

lea
0.23305702

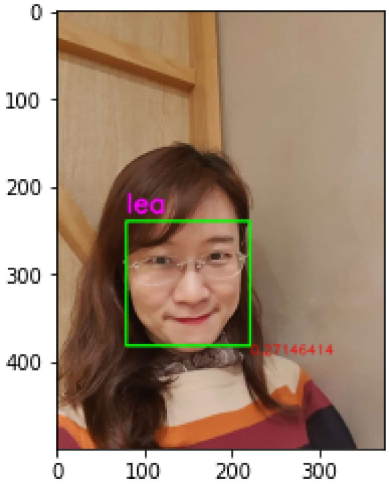

iu

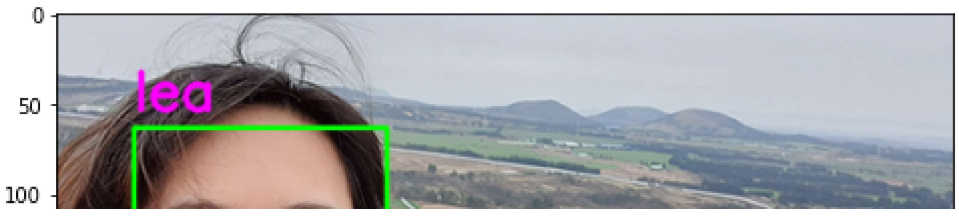

jobs


0.2069365

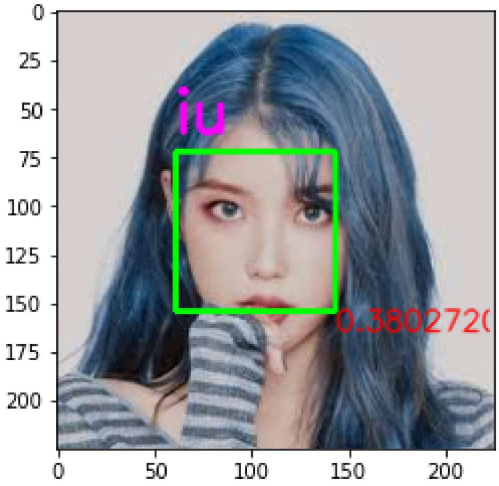

jobs
0.3629898
© AP

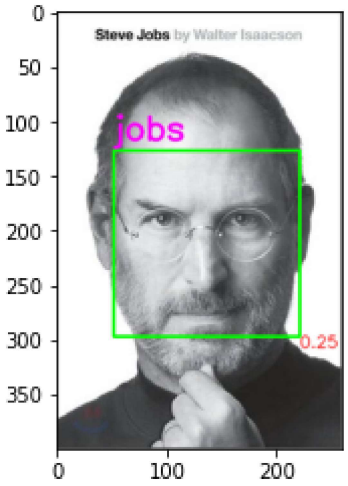

iu
0.35609877

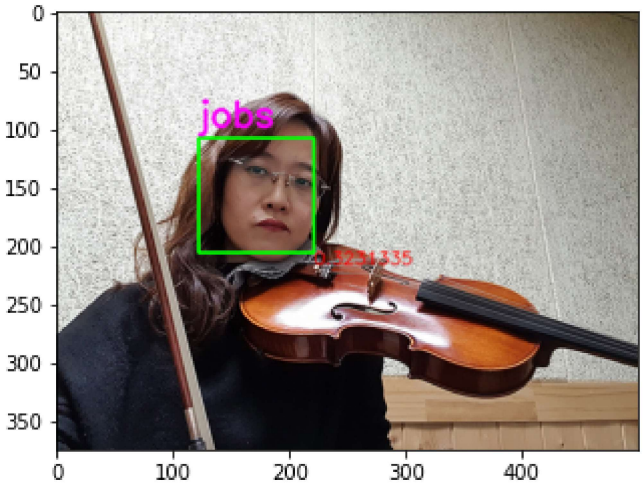

lea

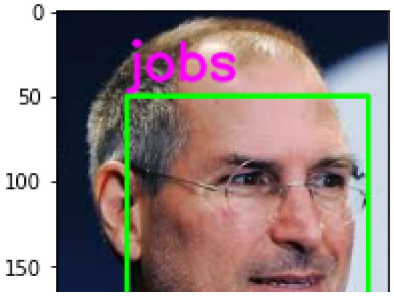

lea

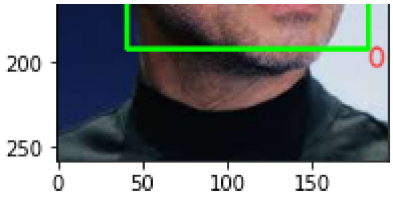

0.31689516

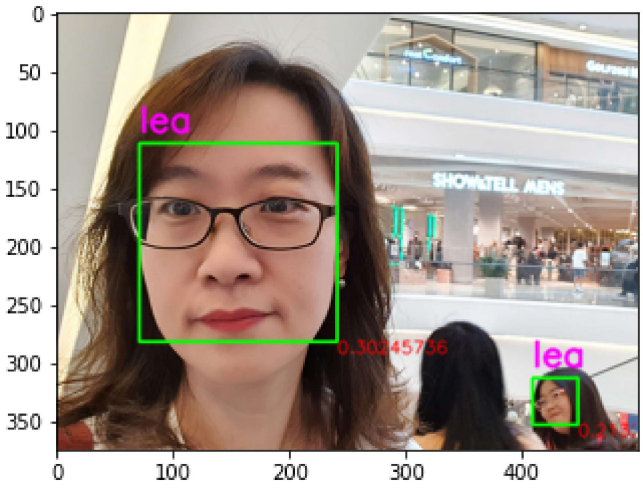

iu

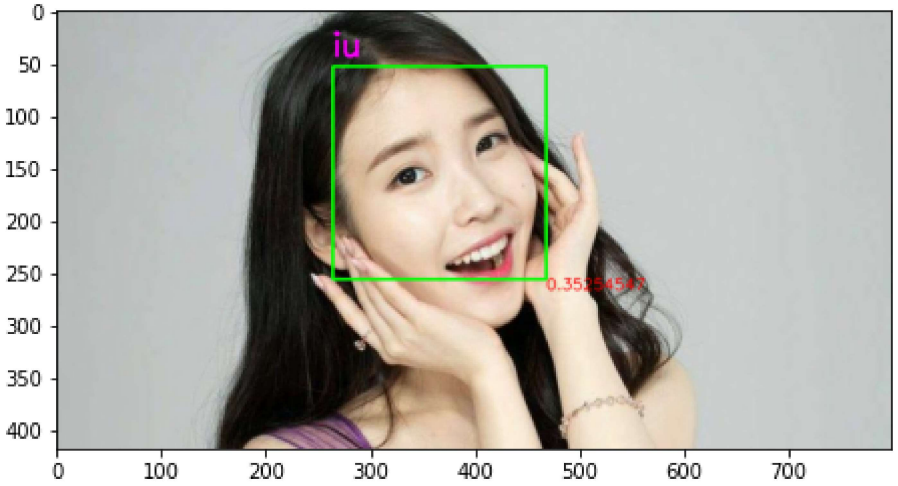

Steve Jobs by Walter Isaacson
jobs
0.25


jobs

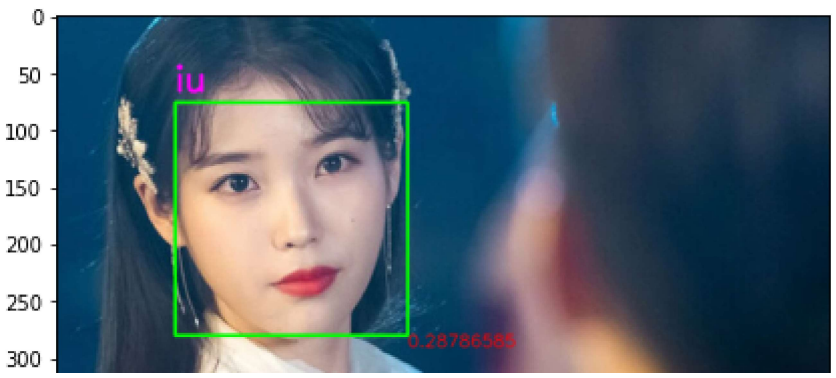

jobs

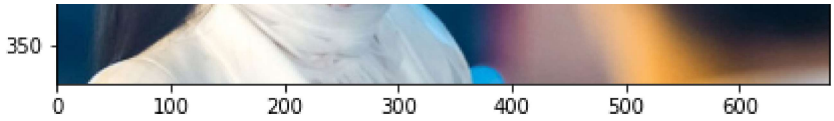

0

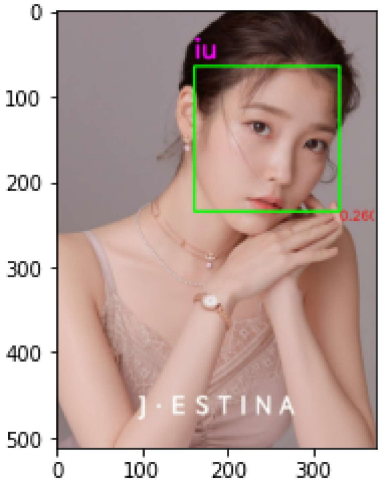

lea
lea

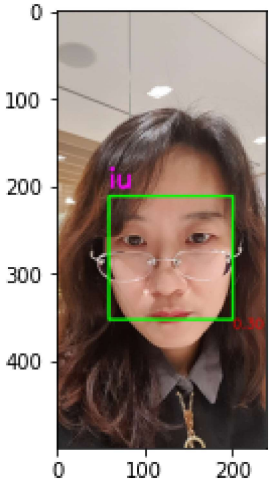

iu

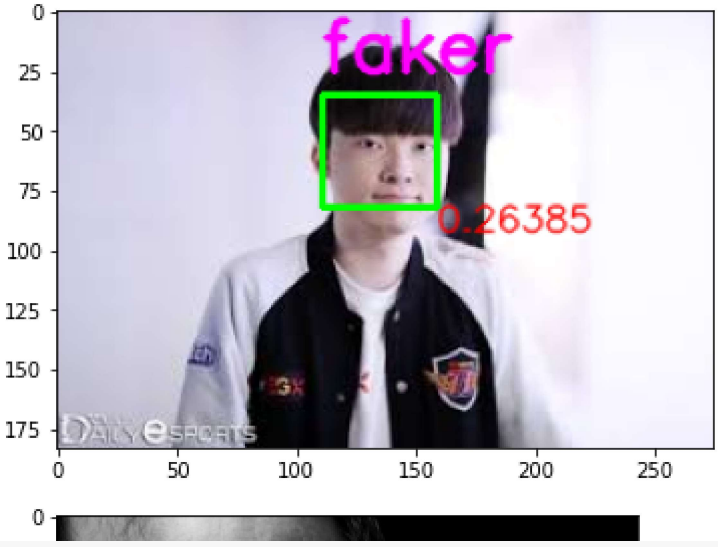

0.26966482

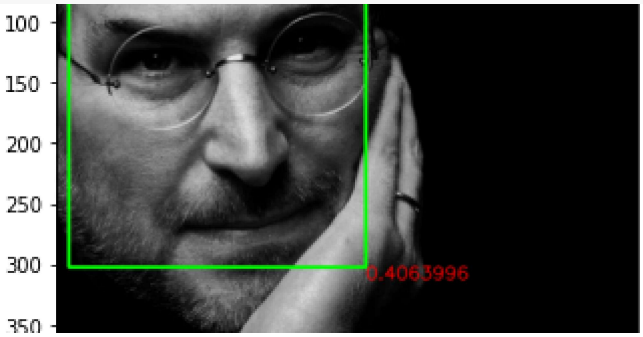

iu

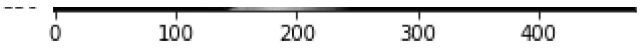



iu
J·ESTINA


iu


faker
0.26385


0.4063996


0
100
200
300
400