Code wins arguments

Menu

Face Recognition with FaceNet in Keras

September 3, 2018 / Machine Learning

Support me on Patreon

BECOME A PATRON

Haven't you subscribe my channel yet?

Follow me on Twitter

Follow @serengil

Google announced FaceNet as its deep learning based face recognition model. It was built on the Inception model. We have been familiar with Inception in kaggle imagenet competitions. Basically, the idea to recognize face lies behind representing two images as smaller dimension vectors and decide identity based on similarity just like in Oxford's VGG-Face.

Katy Perry wears a funeral face net

Objective

Face recognition is a combination of CNN, Autoencoders and Transfer Learning studies. I strongly recommend you to read How Face Recognition Works post to understand what a face recognition pipeline is.

Transfer learning

We will apply transfer learning to have outcomes of previous researches. David Sandberg shared pre-trained weights after 30 hours training with GPU. However, that work was on raw TensorFlow. **Your friendly neighborhood blogger** converted the pre-trained weights into Keras format. I put the weights in Google Drive because it exceeds the upload size of GitHub. You can find pre-trained weights here. Also, FaceNet has a very complex model structure. You can find the model structure here in json format.

We can create FaceNet mode as illustrated below.

```
from keras.models import model_from_json

#facenet model structure: https://github.com/serengil/tensorflow-101/bl
model = model_from_json(open("facenet_model.json", "r").read())

#pre-trained weights https://drive.google.com/file/d/1971Xk5RwedbudGgTI
model.load_weights('facenet_weights.h5')

model.summary()
```

Some people informed me about that python 3.6 users need to downgrade their python version to 3.5 to import model structure json. If you are a 3.6 user and insist to use this version you can alternatively load the model as demonstrated below.

```
# get the following file in the same directoryhttps
# https://github.com/serengil/tensorflow-101/blob/master/model/inceptio
from inception_resnet_v1 import *
model = InceptionResNetV1()
```

FaceNet model expects 160×160 RGB images whereas it produces 128-dimensional representations. Auto-encoded representations called embeddings in the research paper. Additionally, researchers put an extra l2 normalization layer at the end of the network. Remember what l2 normalization is.

$l2 = \sqrt{(\sum x_i^2)}$ while (i=0 to n) for n-dimensional vector

They also constrained 128-dimensional output embedding to live on the 128-dimensional hyperspace. This means that element wise should be applied to output and l2 normalized form pair.

```
def l2_normalize(x):
return x / np.sqrt(np.sum(np.multiply(x, x)))
```

512 dimensional FaceNet model

The original Facenet study creates 128 dimensional vectors. The model mentioned above creates 128 dimensions as well. Here, David Sandberg published an extended version of Facenet and it creates 512 dimensions. He got 99.60% accuracy on LFW data set! Here, you can find how to build Facenet 512 model.

```
#!pip install deepface
model = DeepFace.build_model("Facenet512")
```

Finding similarity

Researchers also mentioned that they used euclidean distance instead of cosine similarity to find similarity between two vectors. Euclidean distance basically finds distance of two vectors on an euclidean space.

```
def findEuclideanDistance(source_representation, test_representation):
euclidean_distance = source_representation - test_representation
euclidean_distance = np.sum(np.multiply(euclidean_distance, euclidean_d
euclidean_distance = np.sqrt(euclidean_distance)
return euclidean_distance
```

Finally, we can find the distance between two different images via FaceNet.

```
img1_representation = l2_normalize(model.predict(preprocess_image('img1
img2_representation = l2_normalize(model.predict(preprocess_image('img2

euclidean_distance = findEuclideanDistance(img1_representation, img2_re
```

Distance should be small for images of same person whereas distance should be large for pictures of different people. Setting the threshold to 0.20 in the research paper but I got successful results when it is set to 0.35.

If you wonder how to determine the threshold value for this face recognition model, then this blog post explains it deeply.

```
threshold = 0.35
if euclidean_distance < threshold:
print("verified... they are same person")
else:
print("unverified! they are not same person!")
```

Still, we can check cosine similarity between two vectors. In this case, I got the most successful results when I set the threshold to 0.07. Notice that l2 normalization skipped for this metric.

```
def findCosineSimilarity(source_representation, test_representation):
a = np.matmul(np.transpose(source_representation), test_representation
b = np.sum(np.multiply(source_representation, source_representation))
c = np.sum(np.multiply(test_representation, test_representation))
return 1 - (a / (np.sqrt(b) * np.sqrt(c)))

img1_representation = model.predict(preprocess_image('img1.jpg'))[0,:]
img2_representation = model.predict(preprocess_image('img2.jpg'))[0,:]

cosine_similarity = findCosineSimilarity(img1_representation, img2_rep
print("cosine similarity: ",cosine_similarity)
threshold = 0.07
if cosine_similarity <threshold:
print("verified... they are same person")
else:
print("unverified! they are not same person!")
```

Testing

Well, we designed the model. The important thing how successful designed model is. I test the FaceNet with same instances in VGG-Face testing.

It succeeded when I tested the model for really different Angelina Jolie images.

True positive results for Angelina Jolie

Similarly, FaceNet succeeded when tested for different photos of Jennifer Aniston .

True positive results for Jennifer Aniston

We can process true negative cases successfully.

Angelina Jolie and Jennifer Aniston are true negative

Real Time Implementation

We can run Google Facenet model in real time as well. The following video applies facenet to find the vector representations of both images in the database and captured one. OpenCV handles face detection here. Euclidean distance checks the distance between two images.

I removed l2 normalization step here because it produces unstable results in real time. That's why, threshold value changed. My experiments show that face images have a euclidean distance less than 21 are same if l2 normalization disabled. My experiments also show that l2 normalization disabled euclidean distance is more stable than cosine distance for Facenet model.

You can find the source code for this real time implementation in GitHub.

Face alignment

Modern face recognition pipelines consist of 4 stages: detect, align, represent and classify / verify. We've skipped the face detection and face alignment steps not to make this post so complex. However, it is really important for face recognition tasks.

Face detection can be done with many solutions such as OpenCV, Dlib or MTCNN. OpenCV offers haar cascade, single shot multibox detector (SSD). Dlib offers Histogram of Oriented Gradients (HOG) and Max-Margin Object Detection (MMOD). Finally, MTCNN is a popular solution in the open source community as well. Herein, SSD, MMOD and MTCNN are modern deep learning based approaches whereas haar cascade and HoG are legacy methods. Besides, SSD is the fastest one. You can monitor the detection performance of those methods in the following video.

Here, you can watch how to use different face detectors in Python.

Moreover, Google declared that face alignment increases its face recognition model **FaceNet** from 98.87% to 99.63%. This is almost 1% accuracy improvement which means a lot for engineering studies. Here, you can find a detailed tutorial for face alignment in Python within OpenCV.

Face alignment

You can find out the math behind alignment more on the following video:

Besides, face detectors detect faces in a rectangle area. So, detected faces come with some noise such as background color. We can find 68 different landmarks of a face with dlib. In this way, we can get rid of any noise of a facial image.

In addition, MediaPipe can find 468 landmarks. Please see its real time implementation in the following video. Recommended tutorials: Deep Face Detection with MediaPipe, Zoom Style Virtual Background Setup with MediaPipe.

Here, retinaface is the cutting-edge face detection technology. It can even detect faces in the crowd and it finds facial landmarks including eye coordinates. That's why, its alignment score is very high.

Conclusion

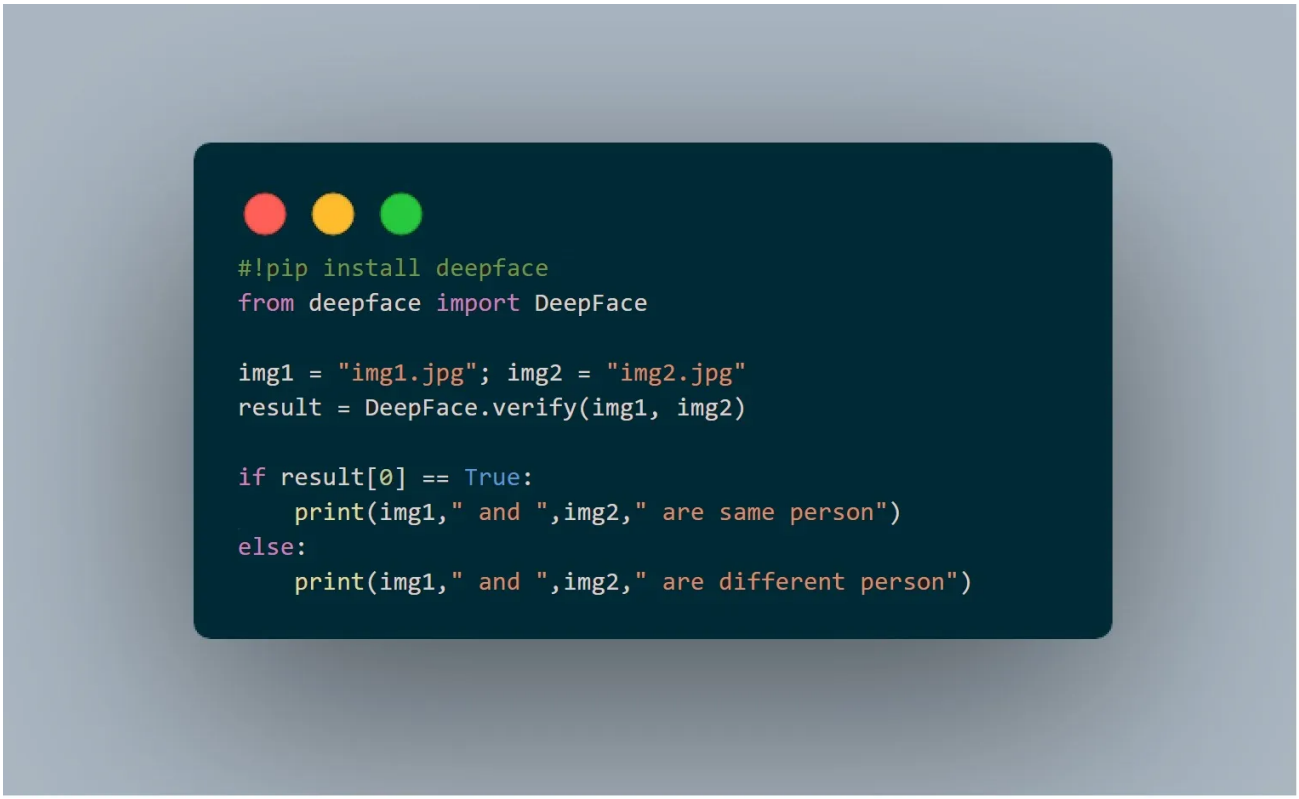
So, we've implemented Google's face recognition model on-premise in this post. We have combined representations with autoencoders, transfer learning and vector similarity concepts to build FaceNet. Original paper includes face alignment steps but we skipped them in this post. Instead of including alignment, I fed already aligned images as inputs. Moreover, FaceNet has a much more complex model structure than VGG-Face. Still, VGG-Face produces more successful results than FaceNet based on experiments. This might cause to produce slower results in real time. Finally, I pushed the code of this post into GitHub.

Katy Perry with her Face Net

Python Library

Herein, deepface is a lightweight face recognition framework for Python. It currently supports the most common face recognition models including VGG-Face, Facenet and OpenFace, DeepID.

It handles model building, loading pre-trained weights, finding vector embedding of faces and applying similarity metrics to recognize faces in the background. You can verify faces with a just few lines of code. It is available on PyPI. You should run the command "**pip install deepface**" to have it. Its code is also open-sourced in GitHub. GitHub repo has a detailed documentation for developers. BTW, you can support this project by starring the repo.



deepface

Here, you can watch the how to video for deepface.

Besides, you can run deepface in real time with your webcam as well.

Large scale face recognition

Large scale face recognition requires to apply face verification several times. However, we can store the representations of ones in our database. In this way, we just need to find the representation of target image. Finding distances between representations can be handled very fast. So, we can find an identity in a large scale data set in just seconds. Deepface offers an out-of-the-box function to handle large scale face recognition as well.

Notice that face recognition has O(n) time complexity and this might be problematic for millions or billions level data. Herein, approximate nearest neighbor (a-nn) algorithm reduces time complexity dramatically. Spotify Annoy, Facebook Faiss and NMSLIB are amazing a-nn libraries. Besides, Elasticsearch wraps an NMSLIB and it comes with highly scalability. You should run deepface within those a-nn libraries if you have really large scale data base.

On the other hand, a-nn algorithm does not guarantee to find the closest one always. We can still apply k-nn algorithm here. Map reduce technology of big data systems might satisfy the both speed and confidence here. mongoDB, Cassandra and Hadoop are the most popular solutions for no-sql databases. Besides, if you have a powerful database such as Oracle Exadata, then RDBMS and regular sql might satisfy your concerns as well.

Tech Stack Recommendations

Face recognition is mainly based on representing facial images as vectors. Herein, storing the vector representations is a key factor for building robust facial recognition systems. I summarize the tech stack recommendations in the following video.

Ensemble method

We've mentioned just a single face recognition model. On the other hand, there are several state-of-the-art models: VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace and DeepID. Even though all of those models perform well, there is no absolute better model. Still, we can apply an ensemble method to build a grandmaster model. In this approach, we will feed the predictions of those models to a boosting model. Accuracy metrics including precision, recall and f1 score increase dramatically in ensemble method whereas running time lasts longer.

The Best Single Model

There are a few state-of-the art face recognition models: VGG-Face, FaceNet, OpenFace and DeepFace. Some are designed by tech giant companies such as Google and Facebook whereas some are designed by the top universities in the world such as University of Oxford and Carnegie Mellon University. We will have a discussion about the best single face recognition model in this video.

#autoencoder, #face recognition, #google, #transfer learning

« Previous / Next »

86 Comments

soulspirit

September 11, 2018 at 10:43 am

Hi,
I used the same code with this example. I used mac. And the error occurs when read json file.

Is there any restriction about the keras version or tensorflow version? Or can you teach me how to convert pb file to h5 and json file?

Appreciate for your help.

Reply

Sefik Serengil
September 11, 2018 at 10:48 am

It might be because of the end of lines. Windows OS puts CR RF whereas Mac expects just LF. Could you try to change end of line characters in the file?

Reply

---

soulspirit
September 12, 2018 at 1:41 am

Sorry, I didn't put the error:

File "C:\Users\IS96273\Desktop\inception_resnet_v1.py", line 26, in scaling
SystemError: unknown opcode

Reply

Sefik Serengil
September 12, 2018 at 4:18 am

Oh, copy that. That is my desktop location. You must change it as your local configuration.

BTW, inception_resnet_v1.py term is not mentioned in that post. Where did you find it?

Reply

Diego Ruiz
September 15, 2018 at 8:36 am

The error "C: \ Users \ IS96273 \ Desktop \ inception_resnet_v1.py" occurs once you read the json, I guess it is because the json saved the path of your desktop, could you indicate how to save the json? json to work on other machines? Thank you

Sefik Serengil

September 17, 2018 at 5:56 am

Before what version of tensorflow and keras you are working? My testing are 1.9.0 for tensorflow, and 2.2.0 for keras, and 3.5.5 for python. Can you try to run code on these version of frameworks? I found same error in this link https://github.com/keras-team/keras/issues/7297 . They mentioned that problem is resolved when switching the version.

Diego Ruiz

September 17, 2018 at 7:11 pm

With that version of python it worked, thanks

Diego Ruiz

September 15, 2018 at 8:35 am

El error “C:\Users\IS96273\Desktop\inception_resnet_v1.py” se produce una vez que lee el json, supongo que es por culpa de que al guardar el json se guardo la ruta de tu escritorio, ¿Podrias indicar como guardar el json para que funcione en otras maquinas? Gracias

Reply

Diego Ruiz

September 15, 2018 at 8:35 am

The error “C: \ Users \ IS96273 \ Desktop \ inception_resnet_v1.py” occurs once you read the json, I guess it is because the json saved the path of your desktop, could you indicate how to save the json? json to work on other machines? Thank you

Reply

Tantael

October 14, 2018 at 7:18 pm

Tested this code and sadly it is not working.

Reply

Sefik Serengil

October 14, 2018 at 7:20 pm

Code is working, I pushed it after testing. What is the error message?

Version comparability might cause the error. My python, tensorflow and keras versions are mentioned below. Please update your environment based on these versions.

(tensorflow) C:\>python –version

Python 3.5.5 :: Anaconda, Inc.

(tensorflow) C:\>python

Python 3.5.5 |Anaconda, Inc.| (default, Apr 7 2018, 04:52:34) [MSC v.1900 64 bit (AMD64)] on win32

Type “help”, “copyright”, “credits” or “license” for more information.

>>> import tensorflow as tf

>>> tf.__version__

‘1.9.0’

>>> import keras

Using TensorFlow backend.

>>> keras.__version__

‘2.2.0’

Reply

Rodger Wilson

March 11, 2019 at 6:25 am

I am getting the error

Traceback (most recent call last):
File “test.py”, line 21, in
img1_representation = l2_normalize(model.predict(preprocess_image(‘image1.png’))[0, :])
NameError: name ‘preprocess_image’ is not defined

I am guessing I need some import but I can’t figure it out.

I think it would be nice if there was a github with the full code and the images and the weights.

Reply

Sefik Serengil
March 11, 2019 at 6:27 am

Hello, actually it was already mentioned in the post. You can find the full code here: https://github.com/serengil/tensorflow-101/blob/master/python/facenet.ipynb

Reply

Anuroop Behera
March 24, 2019 at 11:22 am

Hello, great article, but when i implemented it i get pretty terrible results. can you give me an idea of what possible reasons could be?

Reply

Sefik Serengil
March 24, 2019 at 11:39 am

I just used cropped faces in my tests. Opencv's face detection module helped me a lot. You did it in a similar way?

Reply

Swarup Ghosh
July 11, 2019 at 11:08 pm

The image is being passed through function preprocess_input (keras.applications.imagenet_utils.preprocess_input) which uses default mode='caffe' instead of 'tf'. Not sure why the caffe preprocessing is being used.
Assuming that the keras weights are a port the davidsandberg's FaceNet implementation (which was trained on Tensorflow, right?)

Reply

Sefik Serengil
July 12, 2019 at 5:10 am

This might cause your environment versions. Could you please try this with TF 1.9.0 and Keras 2.2.0? BTW, when you import tensorflow, it says that it is using tensorflow backend by default?

Reply

Swarup Ghosh
July 12, 2019 at 11:46 am

Yes that's true when using keras, tensorflow backend is the sane default but the method keras.applications.imagenet_utils.preprocess_input still uses caffe mode for preprocessing. Pretty sure about this cause I got it confirmed through a GitHub issue relating to the same. (https://github.com/keras-team/keras/issues/10279)

Also, in this case the operational difference would be that:
– mode: 'caffe'
will convert the images from RGB to BGR, then will zero-center each color channel with respect to the ImageNet dataset, without scaling.
– mode: 'tf'
will scale pixels between -1 and 1, sample wise

as mentioned in official documentation.

Is it intended that a pre-trained FaceNet model (originally trained using Tensorflow) have inputs pre-processed like that?

Carefully, reviewing all the modes in keras.applications.imagenet_utils.preprocess_input tells me that:
'caffe' mode will produce each pixel value in range -255 to 255 (also performing RGB -> BGR conversion)
'tf' mode will produce each pixel value in range -1 to 1

Reply

Sefik Serengil
July 12, 2019 at 11:48 am

Could you try to by-pass preprocess_input with the following code block?

```
img_pixels = image.img_to_array(detected_face)
img_pixels = np.expand_dims(img_pixels, axis = 0)
img_pixels /= 127.5
img_pixels -= 1
```

preprocess_input basically normalizes input in -1 and +1

Reply

Swarup Ghosh
July 12, 2019 at 2:09 pm

Thanks a lot for sharing.
Actually I did implement it the pretty same way as described by you, working fine!
This confirms that the preprocessing intended is to operationally scale pixel values in the range of -1,+1 (surely, helps!)

Just for your reference you can try this code also,
https://storage.googleapis.com/swgghosh/preprocess_tests.tar.gz

---

kainat sheikh
November 26, 2020 at 12:16 pm

sir i cannot want the preweight for facenet and i think i dnt have specific editor to open this weight plz guide me and i have cpu system hd graphics 520 can i run this code plz tell me?

Reply

Swarup Ghosh
July 11, 2019 at 11:12 pm

More information about preprocess_input might be available here.
https://github.com/keras-team/keras/issues/10279

And, the difference in preprocess_input modes may be found here:
https://github.com/keras-team/keras-applications/blob/master/keras_applications/imagenet_utils.py

Reply

Richard
July 29, 2019 at 11:16 pm

Thank you very much for such a great contribution

Would you be so kind to tell us which version of open cv are you using?

Thanks a lot in advance

Reply

Sefik Serengil
July 30, 2019 at 2:02 pm

Hey, I am using 3.4.4

Reply

Richard
July 30, 2019 at 3:17 pm

Great to know, thank you for that.

I was wondering why is it that I receive the following error:

“Traceback (most recent call last):
File “openface-real-time.py”, line 322, in
faces = face_cascade.detectMultiScale(img, 1.3, 5)
cv2.error: OpenCV(3.4.4) C:\projects\opencv-python\opencv\modules\objdetect\src\cascadedetect.cpp:1698: error: (-215:Assertion failed) !empty() in function ‘cv::CascadeClassifier::detectMultiScale’

[WARN:0] terminating async callback”

As you can see, I’m using the same version of opencv as yours.

Will be very grateful if you could tell what’s wrong with that.

Reply

Richard
July 30, 2019 at 3:27 pm

Solved, I just need to specify better where the XML file was.

Thank you very much

Reply

Sefik Serengil
July 30, 2019 at 6:16 pm

Yeap, it seems that haarcascade xml had not been referenced. You can find it here: https://github.com/serengil/tensorflow-101/blob/master/model/haarcascade_frontalface_default.xml

Reply

William
November 1, 2019 at 8:31 am

“C:\Users\IS96273\Desktop\inception_resnet_v1.py”
SystemError: unknown opcode
I search the path but i didn’t find it 🙁
do you have any idea about this? Thank you.

Reply

Sefik Serengil
November 1, 2019 at 10:37 am

This path exists in my laptop, you should configure it. Inception resnet v1 py exist in github.

Reply

William
November 3, 2019 at 8:26 am

Okay, but could you tell me how to configure the path? is it on the json file? Thanks.

Reply

Sefik Serengil
November 3, 2019 at 8:28 am

inception_resnet_v1.py can be downloaded here: https://github.com/serengil/tensorflow-101/tree/master/model . Put this python file in the same folder of your running code.

Reply

William
November 3, 2019 at 10:35 am

I already try to run the code, however it seems the error occurs when read json file. 😐

File "./tensorflow-101-master/model/facenet-real-time.py", line 40, in
model = model_from_json(open("./facenet_model.json", "r").read())

File "C:\Users\IS96273\Desktop\inception_resnet_v1.py", line 26, in scaling

Sefik Serengil
November 3, 2019 at 10:40 am

1- Deactivate the folowing line
model =
model_from_json(open("C:/Users/IS96273/Desktop/facenet_model.json",
"r").read())

2- Put the folowing file in the same directory
https://github.com/serengil/tensorflow-101/blob/master/model/inception_resnet_v1.py

3- Replace removed line with the following block
from inception_resnet_v1 import *
model = InceptionResNetV1()

This is already mentioned in the blog post
“Some people informed me about that python 3.6 users need to downgrade their python version to 3.5 to import model structure json. If you are a 3.6 user and insist to use this version you can alternatively load the model as demonstrated below.”

William
November 4, 2019 at 4:00 am

It works, thank you ^^
Btw, how many samples to get the better accuracy?

Sefik Serengil
November 4, 2019 at 4:09 am

This is one shot learning. I mean that one sample is enough.

William
November 5, 2019 at 4:19 am

Hello, but it give the strange results. I use the cropped image and sometimes the labels are incorrect. Either it labelling unknown or other person. Any suggestion how to improve the results? Thanks.

Reply

Sefik Serengil
November 5, 2019 at 4:20 am

You might try to run VGG-Face. It is more robust model.

Reply

William
November 5, 2019 at 6:03 am

VGG give high accuracy, but can i use the gray scale samples or it should be RGB samples?

Reply

Sefik Serengil
November 5, 2019 at 9:26 am

Both FaceNet and VGG-Face expects RGB

Reply

William
November 5, 2019 at 9:31 am

Okay, Thank you very much sir.

Martin
January 22, 2020 at 2:07 pm

Hello Sefik,

Everything works well but loading 2 employees, it recognizes one only.
The same label on difference faces. Where I'm wrong?
Employee's faces in the database are in jpg format, RGB and 415x415px.

Help could be appreciated.

Thank you for your great work.

————————
Python: 3.6
Tensorflow: 1.15.0
Keras: 2.2.4
————————

Reply

Sefik Serengil
January 22, 2020 at 2:09 pm

Employee photos cropped by just face areas? Could you share your samples?

Reply

Quique
February 28, 2021 at 9:50 am

This is also happening to me. I have prepared a dataset composed by 86 cropped faces and it only recognizes 3 people. The thing is that i am having the same problem as you, i am getting incorrect labels and i do not why. I dont know if it is because the euclidean distance or what.

Reply

Sefik Serengil
February 28, 2021 at 10:45 am

Could you try to run vgg-face within deepface framework for python. It handles crop and alignment stages in the background: https://github.com/serengil/deepface

Reply

Martin
January 22, 2020 at 2:21 pm

Thanks for the quick reply!

My samples here:
https://mega.nz/#F!G1BGCKgR!s46ciTwBqggZIqwkjyggNQ

Update: I'm getting better results cropping more.

Reply

Sefik Serengil
January 22, 2020 at 5:04 pm

You mean that the model recognizes these two faces you've shared as same person? Could you also shared the similarity score including cosine and euclidean?

Reply

shivam
August 11, 2020 at 10:31 am

hello sir i m getting the same issue as you stated.
python 3.7.7
conda environment
tensorflow=2.3.0
keras=2.4.3
i cropped the photos by just face area and it recognise me as different person with 99.9% accuracy and label keeps on switching between unkown and some other person.

also can you help how to include liveness detection in this.
Thank you.

Reply

Sefik Serengil
August 11, 2020 at 1:52 pm

Could you share image pairs you're testing?

Reply

Yasser
March 31, 2020 at 11:05 pm

Hi,
-I am using facenet with SVM classification
-I tried to use your function findEuclideanDistance to get threshold, but the function always returns greater than one even if the person inside the dataset

Reply

Sefik Serengil
April 1, 2020 at 6:01 am

It is not a must that threshold should be less than 1. It is the value of 10 in my case when I use FaceNet and Euclidean distance. This means that face pictures have a distance less than 10 are same person. You should tune the threshold value based on positive and negative examples.

Reply

Yasser
April 1, 2020 at 8:12 am

Thank you for respond
– My dataset contains 10 persons
– I tested 4 persons (3 knowns and one unknown)
Anis: 1.239
Anas: 1.174
Yasser: 1.129

unknown: 1.182

Note: the prediction is correct of the three known persons and the probability of the prediction is high and the probability of the prediction of an unknown person is down

Reply

Rizky
June 20, 2020 at 11:53 am

Hello. i'm using python 3.7 on Raspi 4 and try to run your facenet-real-time then i found this error :

Name Error: name 'model_from_json' is not defined

Then i try put directory into it, after that i found this error :

File "C:\Users\IS96273\Dekstop\inception_resnet_v1.py", line 26, in scaling
SystemError : unknown opcode

Then i try following your python 3.6 instruction but i have error here :

from inception_resnet_v1 import *
ModuleNotFoundError : No module named 'inception_resnet_v1'

What should i do sir ?
——————-
Python : 3.7
TensorFlow : 2.2
Keras : 2.4.2
——————-

Reply

Sefik Serengil
June 20, 2020 at 12:53 pm

Firstly, this is my location: C:\Users\IS96273\Dekstop\inception_resnet_v1.py . That py file must be different directory in your environment.

Secondly, please set your environment tensorflow=1.9.0 and keras=2.2.0

Thirdly, put inception_resnet_v1.py in the same folder with the program you are running. It can be found here: https://github.com/serengil/tensorflow-101/blob/master/model/inception_resnet_v1.py

Reply

Rizky
June 21, 2020 at 10:33 am

Okay thank for the advise 🙂 now it's work on :
Python 3.7 (using python 3.6 method)
Keras : 2.2.0
Tensorflow : 1.14.0

Reply

dinusha
July 10, 2020 at 11:27 am

Dear Rizky,
What do you mean python 3.7 (using pythin 3.6 methode)?

please explain

Thank You

Reply

Rizky
July 10, 2020 at 11:58 am

I'm using Python 3.7, but Mr. Sefiks did not give information about Python 3.7, he only gave information about Python 3.5.5 and Python 3.6. method

So i try using Python 3.6 method on my Python 3.7 and it's work normally

---

dinusha
June 24, 2020 at 11:43 am

is there any full code?
Thank You

Reply

Sefik Serengil
June 24, 2020 at 11:45 am

Please follow the links in the post. It will guide you to the source code.

Reply

---

dinusha
June 26, 2020 at 5:51 am

Dear Sir,

What is the OS and version did you run this? can it be run on VM?

Thank You

Reply

Sefik Serengil
June 26, 2020 at 6:03 am

I run it on my local windows laptop. It can be run anywhere.

Reply

dinusha
June 29, 2020 at 11:49 am

At that time it is very hard to install previous versions. I tried so many times. but unfortunately it could not worked. Could you please instruct how to install python 3.5.5, tensor-flow 1.9 , keras 2.2 on ubuntu ot windows.

Reply

Rizky
June 29, 2020 at 12:02 pm

if you trouble installing the old version, try using Pycharm u can upgrade or downgrade whenever you like, I'm using it on Raspi 4 and it works

Python : 3.7
Keras : 2.2.0
Tensorflow : 1.14

*using Python 3.6 method

Reply

dinusha
June 29, 2020 at 12:14 pm

Thanks Rizky. I will try

Reply

Dinusha
June 29, 2020 at 11:56 am

i tried to learn and setup your FR system. but unfortunately the versions you mentioned in lessons are not in pip. also you have installed Deepface using pip. but you have used 3.5 sir. is that pip3? it is thankful to you if you can provide guide to install pythoon, tensorflow and keras sir.
Thank You

Reply

---

dinusha
June 29, 2020 at 12:21 pm

Thanks Rizky. I will try. If we develop this on pycharm, can we run it without pycharm?

Reply

Asif
July 18, 2020 at 11:54 pm

Hello Sefik, I am having this this error:

ValueError: Negative dimension size caused by subtracting 3 from 1 for 'Conv2d_2a_3x3_7/convolution' (op: 'Conv2D') with input shapes: [?,79,1,32], [3,3,32,32].

The second conv layer from the model returning negative values. Any idea how to fix that?

Reply

Sefik Serengil
July 21, 2020 at 10:57 am

What was your tensorflow and keras versions?

Reply

Selin Duman
December 8, 2020 at 5:04 pm

Hello Sefik, I have an error about this line:
model = model_from_json(open("facenet_model.json", "r").read())

My python version is 3.8. How can I fix this?
By the way, I have to use python 3.8 because I began my project before. I mean, right now I am in the middle of my project.
How can I adapt your code to python version 3.8?

Python : 3.8.3
Keras : 2.4.3

Tensorflow : 2.3.1

Thank You

Reply

Sefik Serengil
December 8, 2020 at 5:07 pm

As mentioned in the post: “Some people informed me about that python 3.6 users need to downgrade their python version to 3.5 to import model structure json. If you are a 3.6 user and insist to use this version you can alternatively load the model as demonstrated below.”

```
# get the following file in the same directoryhttps
# https://github.com/serengil/tensorflow-101/blob/master/model/inception_resnet_v1.py
from inception_resnet_v1 import *
model = InceptionResNetV1()
```

Reply

Krupal Shah
January 10, 2021 at 2:56 pm

Hello Sefik,

Isn’t 220 X 220 input size used in the FaceNet paper ? Do you see any performance difference by using 160 X 160 input size ?

Thanks in advance !

Reply

Sefik Serengil
January 10, 2021 at 4:05 pm

This got >99% accuracy on lfw data set.

Reply

Krupal Shah
January 10, 2021 at 9:16 pm

Dear Sefik,

Thank you very much for your quick response !

This got >99% accuracy – that's super great.
Using 220 X 220 pixels, FaceNet team claim to achieve 99.63% accuracy. I am just curious to know the reason(if any) behind selecting specifically 160 X 160 size and not any other.

Thanking you in advance !

Reply

Sefik Serengil
January 11, 2021 at 1:14 pm

This tutorial wraps the repo: https://github.com/davidsandberg/facenet

Most probably David's data set is equal to this size. He got 99.65% acc on lfw btw. It is greater than the original paper.

Reply

Krupal Shah
January 11, 2021 at 3:31 pm

Thank you for the information and leads, Sefik 🙂
That helps !

Krupal Shah
January 11, 2021 at 3:31 pm

Thank you for the information and leads, Sefik 🙂
That helps !

Reply

Krupal Shah
January 11, 2021 at 3:33 pm

Thank you Sefik !
I shall check it.

Reply

Krupal Shah
January 12, 2021 at 9:12 am

Hello Sefik,

Code ran fine with :
python 3.6
model inception_resnet_v1.py

Weights used :
weights facenet_weights.h5

But, I am not getting expected results :
For Angelina Jolie's pic labeled in your repo as img1 vs img5 euclidean distance (l2 norm) is 0.5534593
img1 vs img7
euclidean distance (l2 norm) is 0.3715043

This results in mismatch with the message "unverified! they are not same person!"

1) Is it because I am not using already aligned images ?(I am using img1.jpg, img5.jpg,… and not 1.jpg 5.jpg,…)

2) How can I apply alignment (make 1.jpg version from img1.jpg) in the same notebook ?

3) Isn't facenet designed to work even without alignment (though acccuracy can degrade by 1%, according to this blog) ?

I tried to put thank you note for my last comment yesterday but the system didn't let it pass saying it's repeated comment : Thank you for guiding me through this journey 🙂

Reply

Sefik Serengil
January 12, 2021 at 11:40 am

I actually fed the detected and aligned facial images to facenet. You can apply detection and alignment with deepface framework: https://github.com/serengil/deepface

```
!pip install deepface
from deepface import DeepFace
img = DeepFace.detectFace('img1.jpg')
```

Reply

Krupal Shah
January 12, 2021 at 3:40 pm

Thank you for clearly putting lines of code. That helps !

As you mentioned, I got faces detected and alignment done. I manually saved aligned images to the local folder.

This time, clearly different people, even from different genders – images img17.jpg Vs. img8.jpg (from your dataset) comes with too less euclidean distance :
euclidean distance (l2 norm): 0.13993745
verified… they are same person

Same happens with the cosine similarity. I tried a lot but couldn't figure out. Any idea why it might be working this way ?

Please let me know in case any other detail is required to guide me.

Reply

Krupal Shah
January 12, 2021 at 5:25 pm

That is because deepface is returning me a numpy array. So, whenever I am trying to save it, I am not able to save an exact image. Those are just total black or similar images.

Do you have any suggested workaround ? How did you save those images ?

Reply

Krupal Shah
January 12, 2021 at 10:10 pm

Hello Sefik and everyone out there,

I could resolve above problem by converting numpy array :
img = DeepFace.detectFace("img.jpg")

#converting numpy array to image was resulting in a black image
#refer https://stackoverflow.com/questions/47290668/image-fromarray-just-produces-black-image for more information
img = (img * 255).astype(np.uint8)
img = Image.fromarray(img, 'RGB')
img.save('img_facealigned.jpg')
display(img)

Krupal Shah
January 12, 2021 at 10:15 pm

Hello Sefik,

Thank you for your help !

1) Code is working fine now but I am not getting expected results yet. I am passing detected-aligned faces from deepface.
Is it because I am using inception_resnet_v1 model but facenet_weights.h5 weights ? Maybe model incompatibility ?

2) I am trying hard but not able to run “ipython notebook” command on python 3.5.

Any leads in either way is much more appreciated 🙂

Reply

Krupal Shah
January 12, 2021 at 10:46 pm

“ipython” problem with python 3.5.5 can be resolved by reinstalling parso, i.e. unstall and install parso again.

I am getting error while loading a trained model :
model = model_from_json(open(“/home/…[path]…/facenet_model.json”, “r”).read())

JSONDecodeError: Expecting value: line 7 column 1 (char 6)

Can you please guide me ?

Reply

Sefik Serengil
January 13, 2021 at 3:12 pm

If you try to load the model structure from json, it causes trouble if you have different tf version than me. You can handle this by building model from scratch.

# get the following file in the same directory: https://github.com/serengil/tensorflow-101/blob/master/model/inception_resnet_v1.py
from inception_resnet_v1 import *
model = InceptionResNetV1()

Reply

Krupal Shah
January 14, 2021 at 7:37 pm

I am using 1.9.0 for tensorflow, 2.2.0 for keras, and 3.5.5 for python as you mentioned on 17th September 2018.

With inception_resnet_v1 I am able to run the code on python 3.6 and 3.5.5 but results are not good. Too bad in fact.

Anyway, thank you for all your help 🙂

Reply

Pingback: DeepFace - Most Popular Deep Face Recognition in 2021 (Guide) | viso.ai

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

ex: jane doe

Email *

ex: janedoe@gmail.com

Website

ex: http://janedoe.wordpress.com

Notify me of follow-up comments by email.

Notify me of new posts by email.

Submit

This site uses Akismet to reduce spam. Learn how your comment data is processed.

Please cite this post if it helps your research. Here is an example of BibTex entry:

```
@misc{sefiks12666,
  author ={Serengil, Sefik Ilkin},
  title = { Face Recognition with FaceNet in Keras },
  howpublished = {
https://sefiks.com/2018/09/03/face-recognition-
with-facenet-in-keras/ },
  year = { 2018 },
  note = "[Online; accessed 2023-01-13]"
}
```