FaceNet: A Unified Embedding for Face Recognition and Clustering

Ankur Haritosh
Computer Science Department
Jaypee Institute of
Information Technology
Noida, India
ankurharitosh@gmail.com

Prakhar Saxena
Computer Science Department
Jaypee Institute of
Information Technology
Noida, India
prakharsaxena303@gmail.com

Sonu Gupta
Computer Science Department
Jaypee Institute of
Information Technology
Noida, India
sonug3189@gmail.com

***Abstract*—In this paper we present a system, called FaceNet, that directly learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity. Once this space has been produced, tasks such as face recognition, verification and clustering can be easily implemented using standard techniques with FaceNet embeddings as feature vectors. Their method uses a deep convolutional network trained to directly optimize the embedding itself, rather than an intermediate bottleneck layer as in previous deep learning approaches. To train, they have use triplets of roughly aligned matching / non-matching face patches generated using a novel online triplet mining method. The benefit of our approach is much greater representational efficiency: they have achieved state-of-the-art face recognition performance using only 128-bytes per face. On the widely used Labeled Faces in the Wild (LFW) dataset, their system achieves a new record accuracy of 99.63%. On YouTube Faces DB, it achieves 95.12%.**

***Index Terms*—Face Recognition, LFW Dataset, Triplet Loss**

I. Introduction

Facial recognition technology was in experimental stages in the 1960s. However, today, with the advancement of science and technology the security technology is automatic and supported by sophisticated computer power. Half a century ago it would have been difficult to imagine that the experiment would be so omnipresent and significant from todays security point of view. Presently, it is used by security professionals as well as the government to protect sensitive information. In fact, the technology also has consumer benefit as it can help to password-protect a device.It is easy to say that the technology behind facial recognition is just getting started and still has miles to go before it reaches its peak.

Today, software giants like Microsoft are using facial-recognition software as a way to authenticate people in Windows 10 while Apple is reportedly finding ways for its users to share photos with tagged friends automatically.On the other hand, social networking users have already had a brush with the technology when they were able to tag friends on social networking platforms like Facebook and Google. Recent years have seen a widespread predominance of facial recognition at airports and other public areas.

Security technologies are on an inevitable rise, and facial recognition systems seem to be an integral part of the technology. However, there are various challenges that the world faces before we can be sure about the technology. As per a BBC report on London riots in 2011, UK police could only identify one person from 4,000 images taken at the time.Besides this, there have been questions on privacy and data transparency while using the security technology.However, among these concerns one thing is understandable the facial recognition feature enhances the security level, which is highly critical in confidential matters

In this paper, we present a unified system for face verification (is this the same person), recognition (who is this person) and clustering (find common people among these faces).

II. Advantages

Previous methods are based on learning a Euclidean embedding per image using a deep convolutional network. The network is trained such that the squared L2 distances in the embedding space directly correspond to face similarity: 1neck layer as a representation used to generalize recognition beyond the set of identities used in training. The downsides of this approach are its indirectness and its inefficiency: one has to hope that the bottleneck representation generalizes well to new faces; and by using a bottleneck layer the representation size per face is usually very large (1000s of dimensions).

In contrast to these approaches, FaceNet directly trains its output to be a compact 128-D embedding using a triplet-based loss function based on LMNN [1]. Our triplets consist of two matching face thumbnails and a non-matching face thumbnail and the loss aims to separate the positive pair from the negative by a distance margin. The thumbnails are tight crops of the face area, no 2D or 3D alignment, other than scale and

Fig. 1. Example images of 5 individuals in the LFW dataset.

translation is performed.

Another strength of our model is that it only requires minimal alignment (tight crop around the face area), for example, performs a complex 3D alignment.

III. Methodology

In this paper, we explore two different deep network architectures that have been recently used to great success in the computer vision community. Both are deep convolutional networks. The first architecture is based on the Zeiler&Fergus [2] model which consists of multiple interleaved layers of convolutions, non-linear activations, local response normalizations, and max pooling layers. We additionally add several 11d convolution layers.

The second architecture is based on the Inception model of Szegedy et al. which was recently used as the winning approach for ImageNet 2014 [3]. These networks use mixed layers that run several different convolutional and pooling layers in parallel and concatenate their responses. We have found that these models can reduce the number of parameters by up to 20 times and have the potential to reduce the number of FLOPS required for comparable performance.

To this end we employ the triplet loss that directly reflects what we want to achieve in face verification, recognition and clustering. Namely, we strive for an embedding f (x), from an image x into a feature space R d , such that the squared distance between all faces, independent of imaging conditions, of the same identity is small, whereas the squared distance be- tween a pair of face images from different identities is large.

$$TripletLoss = \sum_{i}^{N}[||f(x_i^a) - f(x_i^p)||^2 - ||f(x_i^a) - f(x_i^n)||^2 + \alpha] \tag{1}$$

IV. Results

A. Performance on LFW dataset

Their model is evaluated in two modes: 1. Fixed center crop of the LFW provided thumbnail. 2. A proprietary face

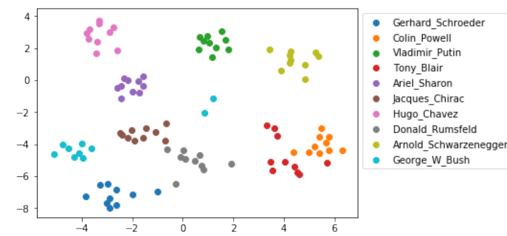


Fig. 2. Visualization of LFW Dataset

detector (similar to Picasa [3]) is run on the provided LFW thumbnails. If it fails to align the face (this happens for two images), the LFW alignment is used.

We achieve a classification accuracy of 98.87%0.15 when using the fixed center crop described in (1) and the record breaking 99.63%0.09 standard error of the mean when using the extra face alignment (2).

B. Our Performance on a small subset of LFW dataset

TABLE I
Results on a small subset of LFW

SVM	*KNN*
0.98	0.96

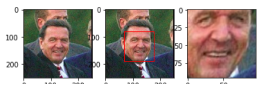
V. Comparison

TABLE II
Comparison on LFW dataset

Models	*Accuracy*
Tom-vs-Pete Classifiers [4]	93.10±1.3%
Blessing of Dimensionality [5]	93.18±1.07%
Practical Transfer Learning Algorithm [6]	96.33±1.08%
Hybrid Deep Learning [7]	91.75±0.48%
Part-Based One-vs-One Features [8]	93.13±0.40%
Learning Discriminant Face Descriptor [9]	84.02±0.44%

Face alignment

```python
import cv2
import matplotlib.pyplot as plt
import matplotlib.patches as patches

from align import AlignDlib

%matplotlib inline

def load_image(path):
    img = cv2.imread(path, 1)
    # OpenCV loads images with color channels
    # in BGR order. So we need to reverse them
    return img[...,::-1]

# Initialize the OpenFace face alignment utility
alignment = AlignDlib('/home/ankur248/PycharmProjects/OpenFace/face_recognition/models/shape_predictor_68_face_

# Load an image of Jacques Chirac
jc_orig = load_image(metadata[2].image_path())

# Detect face and return bounding box
bb = alignment.getLargestFaceBoundingBox(jc_orig)

# Transform image using specified face landmark indices and crop image to 96x96
jc_aligned = alignment.align(96, jc_orig, bb, landmarkIndices=AlignDlib.OUTER_EYES_AND_NOSE)

# Show original image
plt.subplot(131)
plt.imshow(jc_orig)

# Show original image with bounding box
plt.subplot(132)
plt.imshow(jc_orig)
plt.gca().add_patch(patches.Rectangle((bb.left(), bb.top()), bb.width(), bb.height(), fill=False, color='red'))

# Show aligned image
plt.subplot(133)
plt.imshow(jc_aligned);
```



Face recognition

Using KNN and SVM on 128-d embeddings

For training these classifiers we use 50% of the dataset, for evaluation the other 50%.

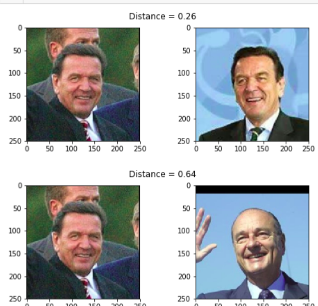
```python
from sklearn.preprocessing import LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import LinearSVC

targets = np.array([m.name for m in metadata])

encoder = LabelEncoder()
encoder.fit(targets)

# Numerical encoding of identities
y = encoder.transform(targets)

train_idx = np.arange(metadata.shape[0]) % 2 != 0
test_idx = np.arange(metadata.shape[0]) % 2 == 0

# 50 train examples of 10 identities (5 examples each)
X_train = embedded[train_idx]
# 50 test examples of 10 identities (5 examples each)
X_test = embedded[test_idx]

y_train = y[train_idx]
y_test = y[test_idx]

knn = KNeighborsClassifier(n_neighbors=1, metric='euclidean')
svc = LinearSVC()

knn.fit(X_train, y_train)
svc.fit(X_train, y_train)

acc_knn = accuracy_score(y_test, knn.predict(X_test))
acc_svc = accuracy_score(y_test, svc.predict(X_test))

print(f'KNN accuracy = {acc_knn}, SVM accuracy = {acc_svc}')
```

```
KNN accuracy = 0.96, SVM accuracy = 0.94
```

Fig. 5. Training

Triplet Loss based on L2 distance for positive and negative anchors for input image

```python
def distance(emb1, emb2):
    return np.sum(np.square(emb1 - emb2))

def show_pair(idx1, idx2):
    plt.figure(figsize=(8,3))
    plt.suptitle(f'Distance = {distance(embedded[idx1], embedded[idx2]):.2f}')
    plt.subplot(121)
    plt.imshow(load_image(metadata[idx1].image_path()))
    plt.subplot(122)
    plt.imshow(load_image(metadata[idx2].image_path()));

show_pair(2, 3)
show_pair(2, 12)
```



Testing

```python
import warnings
# Suppress LabelEncoder warning
warnings.filterwarnings('ignore')

example_idx = 29

example_image = load_image(metadata[test_idx][example_idx].image_path())
example_prediction = svc.predict([embedded[test_idx][example_idx]])
example_identity = encoder.inverse_transform(example_prediction)[0]

plt.imshow(example_image)
plt.title(f'Recognized as {example_identity}');
```

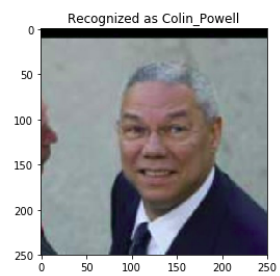


Fig. 6. Testing

VI. Conclusion

They provide a method to directly learn an embedding into an Euclidean space for face verification. This sets it apart from other methods who use the CNN bottleneck layer, or require additional post-processing such as concatenation of multiple models and PCA, as well as SVM classification. They also utilize the ground braking Triplet loss, which calculates loss using seperation between anchor to positive and negative images. Hence, they have achieved a record breaking accuracy of 99.63%.

VII. References

[1] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In NIPS. MIT Press, 2006. 2, 3

[2] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. CoRR, abs/1311.2901, 2013. 2, 3, 4, 6

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. CoRR, abs/1409.4842, 2014. 2, 3, 4, 5, 6, 10

[4] Berg, Thomas Belhumeur, Peter. (2012). Tom-vs-Pete Classifiers and Identity-Preserving Alignment for Face Verification. BMVC 2012 - Electronic Proceedings of the British Machine Vision Conference 2012. 10.5244/C.26.129.

[5] Chen, Dong Cao, Xudong Wen, Fang Sun, Jian. (2013). Blessing of Dimensionality: High-Dimensional Feature and Its Efficient Compression for Face Verification. Proceedings / CVPR, IEEE Computer Society Conference on

Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 3025-3032. 10.1109/CVPR.2013.389.

[6] Cao, Xudong Wipf, David Wen, Fang Duan, Genquan Sun, Jian. (2013). A Practical Transfer Learning Algorithm for Face Verification. Proceedings of the IEEE International Conference on Computer Vision. 3208-3215. 10.1109/ICCV.2013.398.

[7] Sun, Yi Wang, Xiaogang Tang, Xiaoou. (2013). Hybrid Deep Learning for Face Verification. IEEE Transactions on Pattern Analysis and Machine Intelligence. 38. 1489-1496. 10.1109/ICCV.2013.188.

[8] Berg, Thomas Belhumeur, Peter. (2013). POOF: Part-Based One-vs.-One Features for Fine-Grained Categorization, Face Verification, and Attribute Estimation. Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 955-962. 10.1109/CVPR.2013.128.

[9] Learning Discriminant Face Descriptor for Face Recognition, 2014 ————— Lei, Zhen Li, Stan. (2012). Learning Discriminant Face Descriptor for Face Recognition. 7725. 748-759. 10.1007/978-3-642-37444-9$_5$8.