论文：FaceNet:A Unified Embedding for Face Recognition and Clustering

---CVPR2015

一、 创新点

1、 提出一个 Facenet 系统，系统学习从面部图像到紧凑的欧几里德空间的映射，其中距离直接对应于面部相似度度量。

2、 基于使用深度卷积网络学习每幅图像的欧几里德原理，嵌入空间中的平方发，距离对应人脸相似度，同一个人距离短，不同的人距离长。

二、 算法框架

1、模型结构

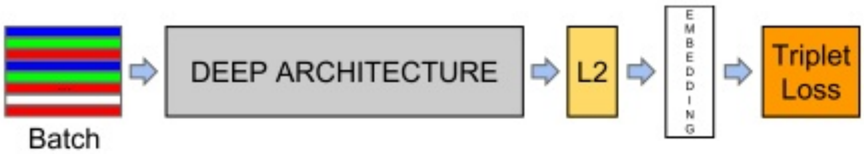

Figure 2. **Model structure.** Our network consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training.

FaceNet 直接使用基于 triplets 的 LMNN（最大边界近邻分类）的 loss 函数训练神经网络，网络直接输出为 128 维度的向量空间。我们选取的 triplets（三联子）包含两个匹配脸部缩略图和一个非匹配的脸部缩略图，loss 函数目标是通过距离边界区分正负类。

2、三联子（triplets）loss

Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

模型的目的是将人脸图像 X embedding 入 d 维度的欧几里得空间 $f(x)\epsilon R^d$，在空间向量内，Triplet Loss 最小化该图像 x_i^a(anchor)和该个体的其他图像 x_i^p(positive)的距离近,与其他个体的图像 x_n^i(negative)远。

$$\|x_i^a - x_i^p\|_2^2 + \alpha < \|x_i^a - x_i^n\|_2^2, \ \forall \left(x_i^a, x_i^p, x_i^n\right) \in \mathcal{T} \ . \quad (1)$$

The loss that is being minimized is then $L =$

$$\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+ \ . \quad (2)$$

其中，α为 positive/negtive 的边界，T 为训练所有可能三胞胎的集合。

3、triplets 筛选

文章采用在线生成 triplets 的方法。选择了大样本的 mini-batch（1800 样本/batch）来增加每个 batch 的样本数量。每个 mini-batch 中，我们对单个个体选择 40 张人脸图片作为正样本，随机筛选其它人脸图片作为负样本。负样本选择不当也可能导致训练过早进入局部最小。为了避免，采用如下公式来帮助筛选负样本：

$$\|f(x_i^a) - f(x_i^p)\|_2^2 < \|f(x_i^a) - f(x_i^n)\|_2^2 \ . \qquad (3)$$

4、深度卷积网络

采用 adagrad 优化器，使用随机梯度下降法(SGD)训练 CNN 模型，学习率设定为 0.05。在 cpu 集群上训练了 1000-2000 小时。边界值α设定为 0.2。总共实验了两类模型如下图所示：

layer	size-in	size-out	kernel	param	FLPS
conv1	220×220×3	110×110×64	7×7×3, 2	9K	115M
pool1	110×110×64	55×55×64	3×3×64, 2	0	
rnorm1	55×55×64	55×55×64		0	
conv2a	55×55×64	55×55×64	1×1×64, 1	4K	13M
conv2	55×55×64	55×55×192	3×3×64, 1	111K	335M
rnorm2	55×55×192	55×55×192		0	
pool2	55×55×192	28×28×192	3×3×192, 2	0	
conv3a	28×28×192	28×28×192	1×1×192, 1	37K	29M
conv3	28×28×192	28×28×384	3×3×192, 1	664K	521M
pool3	28×28×384	14×14×384	3×3×384, 2	0	
conv4a	14×14×384	14×14×384	1×1×384, 1	148K	29M
conv4	14×14×384	14×14×256	3×3×384, 1	885K	173M
conv5a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv5	14×14×256	14×14×256	3×3×256, 1	590K	116M
conv6a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv6	14×14×256	14×14×256	3×3×256, 1	590K	116M
pool4	14×14×256	7×7×256	3×3×256, 2	0	
concat	7×7×256	7×7×256		0	
fc1	7×7×256	1×32×128	maxout p=2	103M	103M
fc2	1×32×128	1×32×128	maxout p=2	34M	34M
fc7128	1×32×128	1×1×128		524K	0.5M
L2	1×1×128	1×1×128		0	
total				140M	1.6B

Table 1. **NN1.** This table show the structure of our Zeiler&Fergus [22] based model with 1×1 convolutions inspired by [9]. The input and output sizes are described in $rows \times cols \times \#filters$. The kernel is specified as $rows \times cols, stride$ and the maxout [6] pooling size as $p = 2$.

type	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj (p)	params	FLOPS
conv1 (7×7×3, 2)	112×112×64	1							9K	119M
max pool + norm	56×56×64	0						m 3×3, 2		
inception (2)	56×56×192	2		64	192				115K	360M
norm + max pool	28×28×192	0						m 3×3, 2		
inception (3a)	28×28×256	2	64	96	128	16	32	m, 32p	164K	128M
inception (3b)	28×28×320	2	64	96	128	32	64	L_2, 64p	228K	179M
inception (3c)	14×14×640	2	0	128	256,2	32	64,2	m 3×3,2	398K	108M
inception (4a)	14×14×640	2	256	96	192	32	64	L_2, 128p	545K	107M
inception (4b)	14×14×640	2	224	112	224	32	64	L_2, 128p	595K	117M
inception (4c)	14×14×640	2	192	128	256	32	64	L_2, 128p	654K	128M
inception (4d)	14×14×640	2	160	144	288	32	64	L_2, 128p	722K	142M
inception (4e)	7×7×1024	2	0	160	256,2	64	128,2	m 3×3,2	717K	56M
inception (5a)	7×7×1024	2	384	192	384	48	128	L_2, 128p	1.6M	78M
inception (5b)	7×7×1024	2	384	192	384	48	128	m, 128p	1.6M	78M
avg pool	1×1×1024	0								
fully conn	1×1×128	1							131K	0.1M
L2 normalization	1×1×128	0								
total									7.5M	1.6B

Table 2. **NN2.** Details of the NN2 Inception incarnation. This model is almost identical to the one described in [16]. The two major differences are the use of L_2 pooling instead of max pooling (m), where specified. The pooling is always 3×3 (aside from the final average pooling) and in parallel to the convolutional modules inside each Inception module. If there is a dimensionality reduction after the pooling it is denoted with p. 1×1, 3×3, and 5×5 pooling are then concatenated to get the final output.

三、 实验结果

由于作者给出 8million 个个体将近 100million-200million 张人脸缩略图，训练量极大，时间太长(作者说在 cpu 集群上需要 1000-2000 小时的训练)，所以就在网上找了别人已经训练好的模型。以下是我自己做的实验：使用 facenet 实现特征点定位以及人脸分类。

实验环境：Ubuntu18.0、python3.5、tensorflow0.14

实验结果：图一为人的五官特征点定位结果，图二为人脸分类结果。

图 1. 五官特征点定位

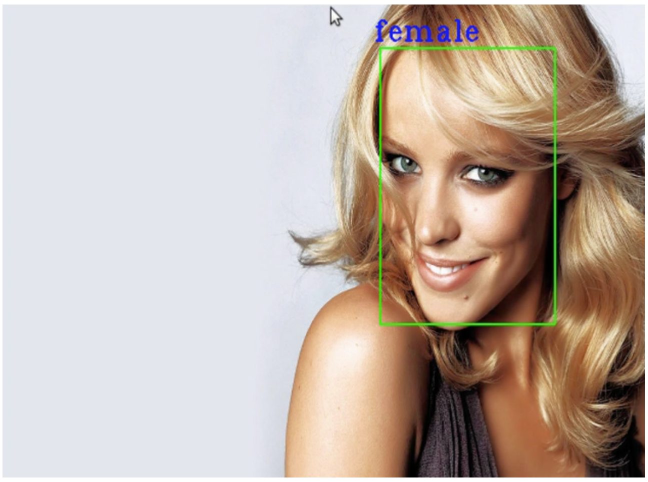

图 2.人脸分类

结果评价：测试了 10 张图片，检测结果还是不错的，但是在中间也出了一些识别的问题，在图 1 中可以发现如果人脸有遮挡是检测不出人脸的并且模型特别大，训练时间过长。

改进点：图片遮挡识别、改进模型结构减少训练时间。

以下是作者的实验结果：

1、深度神经网络结构与 VAL

architecture	VAL
NN1 (Zeiler&Fergus 220×220)	87.9% ± 1.9
NN2 (Inception 224×224)	89.4% ± 1.6
NN3 (Inception 160×160)	88.3% ± 1.7
NN4 (Inception 96×96)	82.0% ± 2.3
NNS1 (mini Inception 165×165)	82.4% ± 2.4
NNS2 (tiny Inception 140×116)	51.9% ± 2.9

Table 3. **Network Architectures.** This table compares the performance of our model architectures on the hold out test set (see section 4.1). Reported is the mean validation rate VAL at 10E-3 false accept rate. Also shown is the standard error of the mean across the five test splits.

2、CNN 模型结构对 loss 的影响

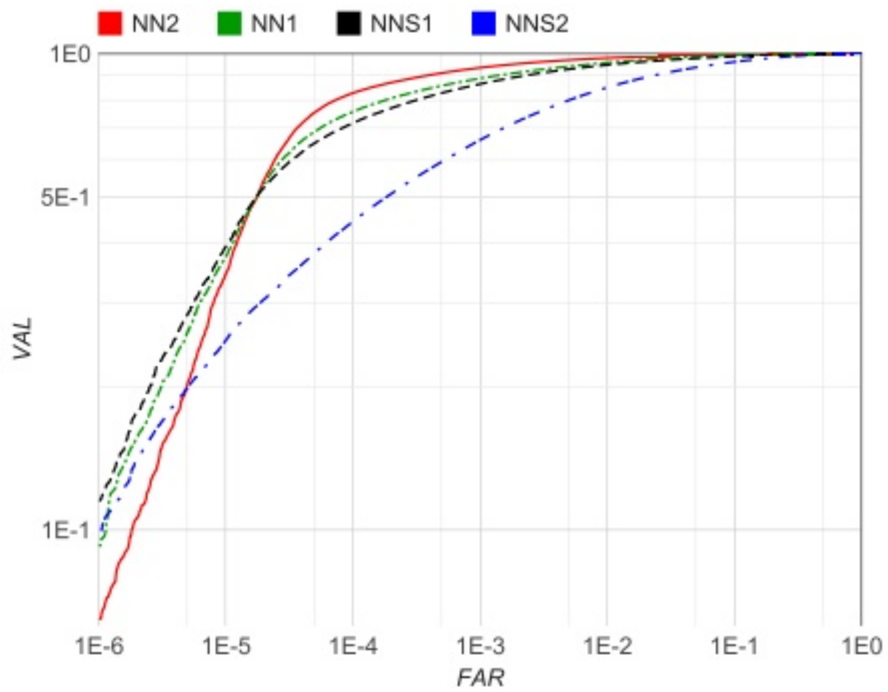

Figure 5. **Network Architectures.** This plot shows the complete ROC for the four different models on our personal photos test set from section 4.2. The sharp drop at 10E-4 FAR can be explained by noise in the groundtruth labels. The models in order of performance are: **NN2:** 224×224 input Inception based model; **NN1:** Zeiler&Fergus based network with 1×1 convolutions; **NNS1:** small Inception style model with only 220M FLOPS; **NNS2:** tiny Inception model with only 20M FLOPS.

3、图像像素对结果影响

jpeg q	val-rate
10	67.3%
20	81.4%
30	83.9%
50	85.5%
70	86.1%
90	86.5%

#pixels	val-rate
1,600	37.8%
6,400	79.5%
14,400	84.5%
25,600	85.7%
65,536	86.4%

Table 4. **Image Quality.** The table on the left shows the effect on the validation rate at 10E-3 precision with varying JPEG quality. The one on the right shows how the image size in pixels effects the validation rate at 10E-3 precision. This experiment was done with NN1 on the first split of our test hold-out dataset.

4、训练数据量对结果的影响

训练数据越多准确率越高

#training images	VAL
2,600,000	76.3%
26,000,000	85.1%
52,000,000	85.1%
260,000,000	86.2%

Table 6. **Training Data Size.** This table compares the performance after 700h of training for a smaller model with 96x96 pixel inputs. The model architecture is similar to NN2, but without the 5x5 convolutions in the Inception modules.

5、评价结果

在 FaceNet 在 LFW 数据集上取得了 99.63%±0.09 的准确率；在 Youtube Faces DB 数据集上获得了 95.12%±0.39 的结果。在个人照片的数据集上，对单个个体进行 embeding 后聚类测试，结果如图所示。

Figure 7. **Face Clustering.** Shown is an exemplar cluster for one user. All these images in the users personal photo collection were clustered together.