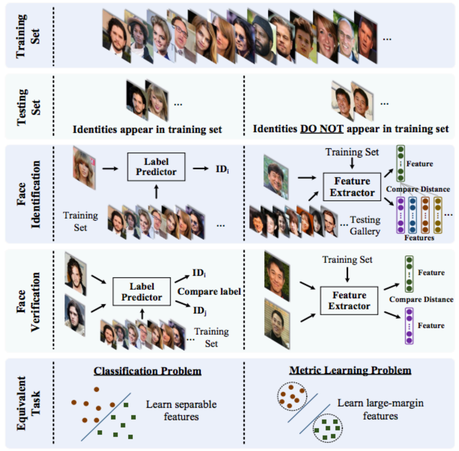
SphereFace: Deep Hypersphere Embedding for Face Recognition

Weiyang Liu[1], Yandong Wen[2], Zhiding Yu[2], Ming Li[2,3], Bhiksha Raj[2], Le Song[1]

1. *Georgia Institute of Technology* 2. *Carnegie Mellon University* 3. *Sun Yet-Sen University*

CVPR July 21-26 2017 HONOLULU

What are the keys to open-set face recognition?



Open-set face recognition

- Face identities do not appear simultaneously in training set and testing set. Knowledge need to be transferred to testing set.
- Open-set face recognition requires more generalization power than close-set face recognition. Overfitting degrades performance
- Essentially, it can be viewed as a deep metric learning problem.

Prevailing methods for deep face recognition

- Deep-ID 2 (CUHK)

Combine the softmax loss and the contrastive loss to learn discriminative face representation.

- FaceNet (Google)

Use Triplet loss to supervise the network learning, but require very large amount of data. (200 million face images)

- Things in common

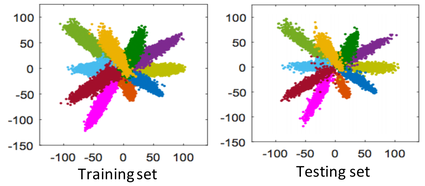
They all explicitly treat the open-set face recognition problem as metric learning problem, since contrastive loss and triplet loss are both originally used in metric learning.

- Drawbacks

Deep-ID network combines the softmax loss and contrastive loss, but they produces different feature distribution. So it may not be a natural choice. For FaceNet, it requires large amount of data. It is computationally expensive

Softmax loss learns angularly distributed features

- Softmax loss can naturally learn angularly distributed features, so it will not be naturally motivated to incorporate any Euclidean losses.



A motivating binary classification example

Softmax computes the probability for two classes as

$$p_1 = \frac{\exp(\boldsymbol{W}_1^T \boldsymbol{x} + b_1)}{\exp(\boldsymbol{W}_1^T \boldsymbol{x} + b_1) + \exp(\boldsymbol{W}_2^T \boldsymbol{x} + b_2)} \quad (1)$$

$$p_2 = \frac{\exp(\boldsymbol{W}_2^T \boldsymbol{x} + b_2)}{\exp(\boldsymbol{W}_1^T \boldsymbol{x} + b_1) + \exp(\boldsymbol{W}_2^T \boldsymbol{x} + b_2)} \quad (2)$$

The decision boundary produced by softmax loss is

$$(\boldsymbol{W}_1 - \boldsymbol{W}_2)\boldsymbol{x} + b_1 - b_2 = 0$$

To achieve angular decision boundary, the weights for the final FC layer is in fact useless. So we will first normalize the weights and zero out the biases. To further introduce angular margin, we propose to make the classification more difficult.

Loss Function	Decision Boundary
Softmax Loss	$(\boldsymbol{W}_1 - \boldsymbol{W}_2)\boldsymbol{x} + b_1 - b_2 = 0$
Modified Softmax Loss	$\|\boldsymbol{x}\|(\cos\theta_1 - \cos\theta_2) = 0$
A-Softmax Loss	$\|\boldsymbol{x}\|(\cos m\theta_1 - \cos\theta_2) = 0$ for class 1 $\|\boldsymbol{x}\|(\cos\theta_1 - \cos m\theta_2) = 0$ for class 2

SphereFace Algorithm

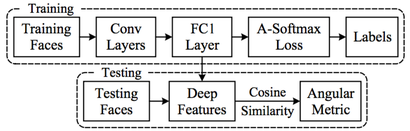
SphereFace uses the angular softmax (A-Softmax) loss defined as

$$L_{\text{ang}} = \frac{1}{N}\sum_i -\log\left(\frac{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})}}{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})} + \sum_{j\neq y_i} e^{\|\boldsymbol{x}_i\|\cos(\theta_{j,i})}}\right)$$

where $\psi(\theta_{y_i,i}) = (-1)^k \cos(m\theta_{y_i,i}) - 2k$ and $\theta_{y_i,i} \in [\frac{k\pi}{m}, \frac{(k+1)\pi}{m}]$.

The parameter m is to control the margin size. Note that the weights are normalized to 1 in each iteration. Larger m gives larger margin.

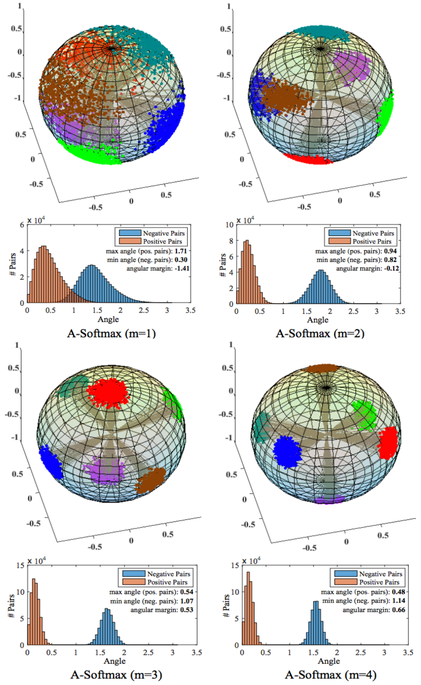
The learning and inference pipeline of SphereFace:



It is the same as traditional CNN framework, so it is extremely simple and compatible with any advanced network architecture such as VGG, GoogLeNet, ResNet, etc. But with the proposed angular softmax loss, the learned features will be much more discriminative.

SphereFace feature visualization

- The SphereFace features are very discriminative in the angular space.



Exploratory study

Code is available!

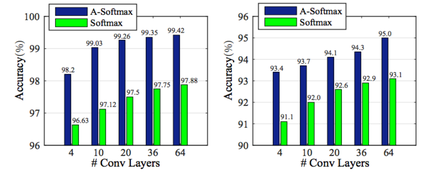
- How m affects the performance

The verification accuracy on LFW dataset

Dataset	Original	m=1	m=2	m=3	m=4
LFW	97.88	97.90	98.40	99.25	**99.42**
YTF	93.1	93.2	93.8	94.4	**95.0**

- How the number of conv layers affects the performance

The verification accuracy on LFW dataset



Experiments on LFW and YTF dataset

- Among all the methods trained on the publicly available WebFace dataset, we achieve the current best performance and significantly outperforms the #2 performance.

Method	Models	Data	LFW	YTF
DeepFace [30]	3	4M*	97.35	91.4
FaceNet [22]	1	200M*	**99.65**	95.1
Deep FR [20]	1	2.6M	98.95	**97.3**
DeepID2+ [27]	1	300K*	98.70	N/A
DeepID2+ [27]	25	300K*	99.47	93.2
Baidu [15]	1	1.3M*	99.13	N/A
Center Face [34]	1	0.7M*	99.28	94.9
Yi et al. [37]	1	WebFace	97.73	92.2
Ding et al. [2]	1	WebFace	98.43	N/A
Liu et al. [16]	1	WebFace	98.71	N/A
Softmax Loss	1	WebFace	97.88	93.1
Softmax+Contrastive [26]	1	WebFace	98.78	93.5
Triplet Loss [22]	1	WebFace	98.70	93.4
L-Softmax Loss [16]	1	WebFace	99.10	94.0
Softmax+Center Loss [34]	1	WebFace	99.05	94.4
SphereFace	1	WebFace	**99.42**	**95.0**

Experiments on Megaface Challenge

- SphereFace achieves the state-of-the-art performance using only publicly available small-scale dataset, while the other commercial algorithms use private and large-scale datasets.

Method	protocol	Rank1 Acc.	Ver.
NTechLAB - facenx large	Large	73.300	85.081
Vocord - DeepVo1	Large	**75.127**	67.318
Deepsense - Large	Large	74.799	**87.764**
Shanghai Tech	Large	74.049	86.369
Google - FaceNet v8	Large	70.496	86.473
Beijing FaceAll_Norm_1600	Large	64.804	67.118
Beijing FaceAll_1600	Large	63.977	63.960
Deepsense - Small	Small	**70.983**	**82.851**
SIAT_MMLAB	Small	65.233	76.720
Barebones FR - cnn	Small	59.363	59.036
NTechLAB - facenx_small	Small	58.218	66.366
3DiVi Company - tdvm6	Small	33.705	36.927
Softmax Loss	Small	54.855	65.925
Softmax+Contrastive Loss [26]	Small	65.219	78.865
Triplet Loss [22]	Small	64.797	78.322
L-Softmax Loss [16]	Small	67.128	80.423
Softmax+Center Loss [34]	Small	65.494	80.146
SphereFace (single model)	Small	**72.729**	**85.561**
SphereFace (3-patch ensemble)	Small	**75.766**	**89.142**