Deep Learning Face Representations with Different Loss Functions for Face Recognition

Hongsheng Li
Dept. Electronic Engineering
Chinese University of Hong Kong

Agenda

- **Preview of several evaluation datasets of face recognition**
- **Preview of several different metrics**
- **Some Euclidean metric based losses**
- **Some Cosine metric based losses**

Agenda

- **Preview of several evaluation datasets of face recognition**
- **Preview of several different metrics**
- **Some Euclidean metric based losses**
- **Some Cosine metric based losses**

LFW Face Verification Protocol

- Labeled faces in the wild (LFW) dataset is a widely used face verification (1:1) protocol, which contains 6,000 face pairs. In all 6,000 pairs, match and mismatch pairs each account for half.

Match Pairs

Benjamin Netanyahu

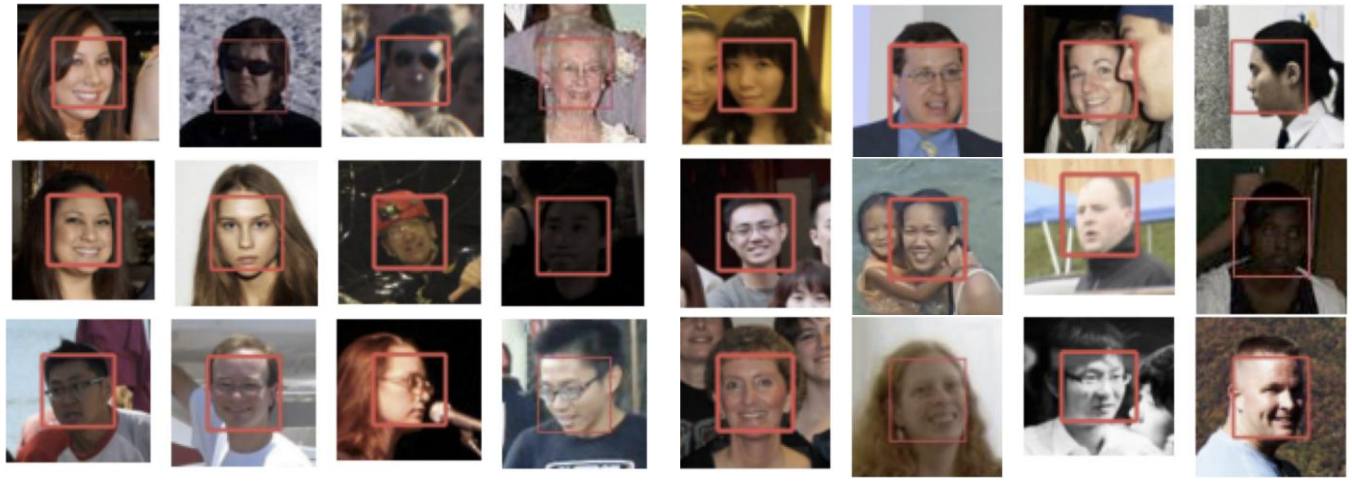
Mismatch Pairs

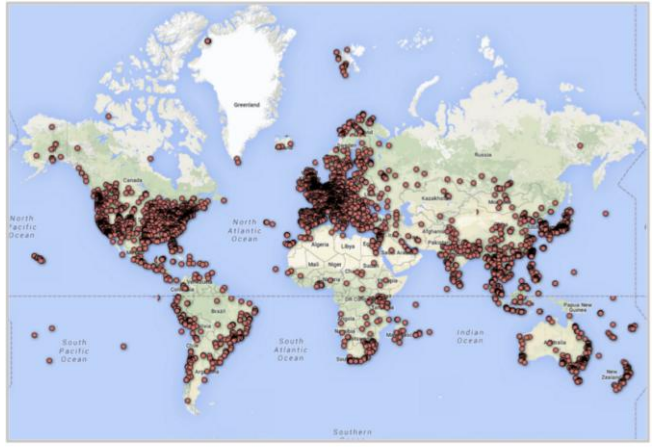
Barbara Felt Miller Leticia Dolera

Huang, Gary B., et al. "Labeled faces in the wild: A database forstudying face recognition in unconstrained environments." *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*. 2008.

MegaFace 1M Face Identification Protocol

- The MegaFace identification dataset includes 1M images of 690K different individuals (from Flickr) as the gallery set and 100K photos of 530 unique individuals from FaceScrub as the probe set.





GPS locations

Radom sample of MegaFace Photos with provided detections in red

Kemelmacher-Shlizerman, Ira, et al. "The megaface benchmark: 1 million faces for recognition at scale." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

Ng, Hong-Wei, and Stefan Winkler. "A data-driven approach to cleaning large face datasets." 2014 IEEE International Conference on Image Processing (ICIP). IEEE, 2014.

Evaluation Criterion

- 1:1 face verification:
  - Accuracy
- 1:N face identification
  - Accuracy with large number of distractors
  - True Accept Rate (TAR) @ False Accept Rate (VAR) (e.g., 99.06% @ 10-2)

Agenda

- Preview of several evaluation datasets of face recognition
- **Preview of several different metrics for training**
- Some Euclidean metric based losses
- Some Cosine metric based losses

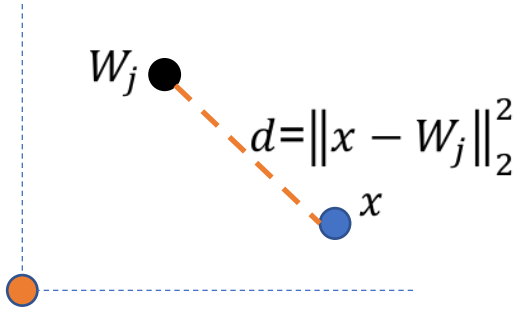
Preview of several different metrics for training

- Loss functions utilize metrics to evaluate the distance between outputs and target classes (or samples). Three different types of metrics are widely used in losses in learning deep face representation.

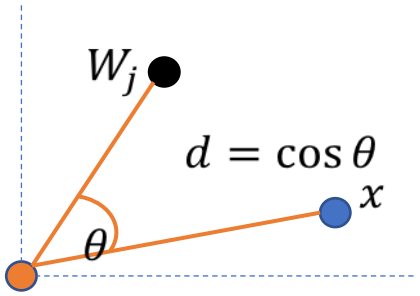
For a metric, the following conditions are satisfied:

$$d(x, y) \geq 0$$
$$d(x, y) = 0 \Leftrightarrow x = y$$
$$d(x, y) = d(y, x)$$
$$d(x, z) \leq d(x, y) + d(y, z)$$

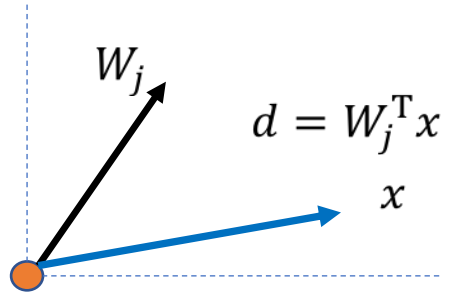
Euclidean Metric



Cosine Metric



Inner Product



Agenda

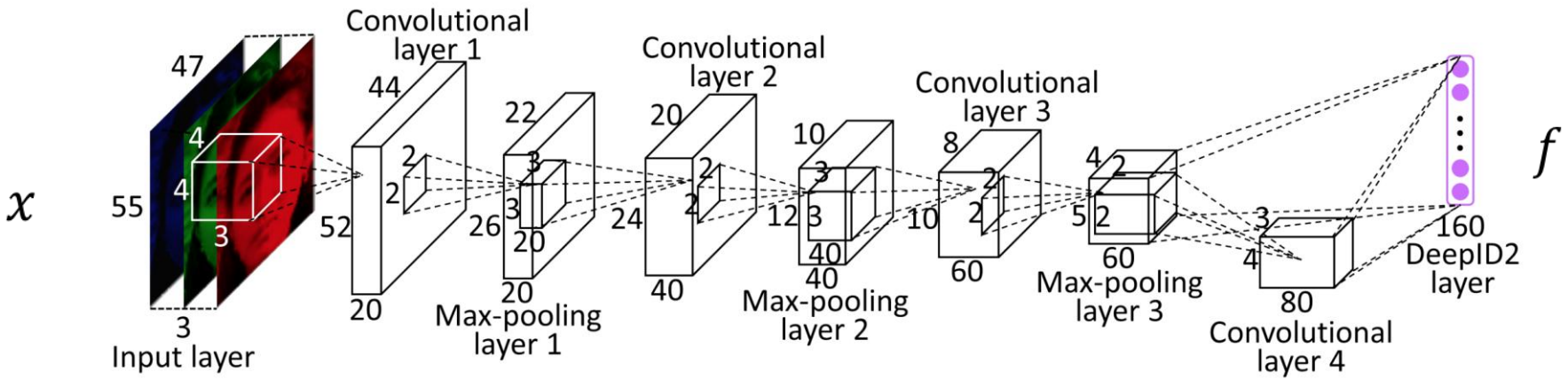
- Preview of several evaluation datasets of face recognition
- Preview of several different metrics
- **Some Euclidean metric based losses**
- Some Cosine metric based losses

Some Euclidean metric based losses

- Deep learning face representation by joint identification-verification (NIPS 2014)
- FaceNet: A unified embedding for face recognition and clustering (CVPR 2015)
- A discriminative feature learning approach for deep face recognition（ECCV 2016）

DeepID2

- The key challenge of face recognition is to develop effective feature representations for reducing intra-personal variations while enlarging inter-personal differences. DeepID2 uses both face identification and verification signals as loss functions.



Here x is the input face image and f is the extracted DeepID2 vector (feature vector). Loss function utilize f to compute the cost.

Sun, Yi, Xiaogang Wang, and Xiaoou Tang. "Deep learning face representation from predicting 10,000 classes." CVPR. 2014.
Sun, Yi, et al. "Deep learning face representation by joint identification-verification." NIPS. 2014.

DeepID2

- DeepID2 feature vectors are learned under two supervisory signals.
- The identification loss classifies each face image into one of n different identities. Identification is achieved by an n-way softmax layer, which outputs a probability distribution over the n classes. Then the distribution is inputted to cross-entropy loss.
- The verification loss encourages features from faces of the same identity to be similar. The verification signal directly regularizes features and can effectively reduce the intra-personal variations.

Softmax classifier

Identification Loss:

$$\text{Ident}(f, t, \theta_{id}) = -\sum_{i=1}^{n} -p_i \log \hat{p}_i = -\log \hat{p}_t \qquad \hat{p}_t = \frac{e^{W_t^T f_i + b_t}}{\sum_{j=1}^{N} e^{W_j^T f_i + b_j}}$$

Verification Loss:

$$\text{Verif}(f_i, f_j, y_{ij}, \theta_{ve}) = \begin{cases} \frac{1}{2} \|f_i - f_j\|_2^2 & \text{if } y_{ij} = 1 \quad \text{same person} \\ \frac{1}{2} \max\left(0, m - \|f_i - f_j\|_2\right)^2 & \text{if } y_{ij} = -1 \quad \text{different person} \end{cases}$$

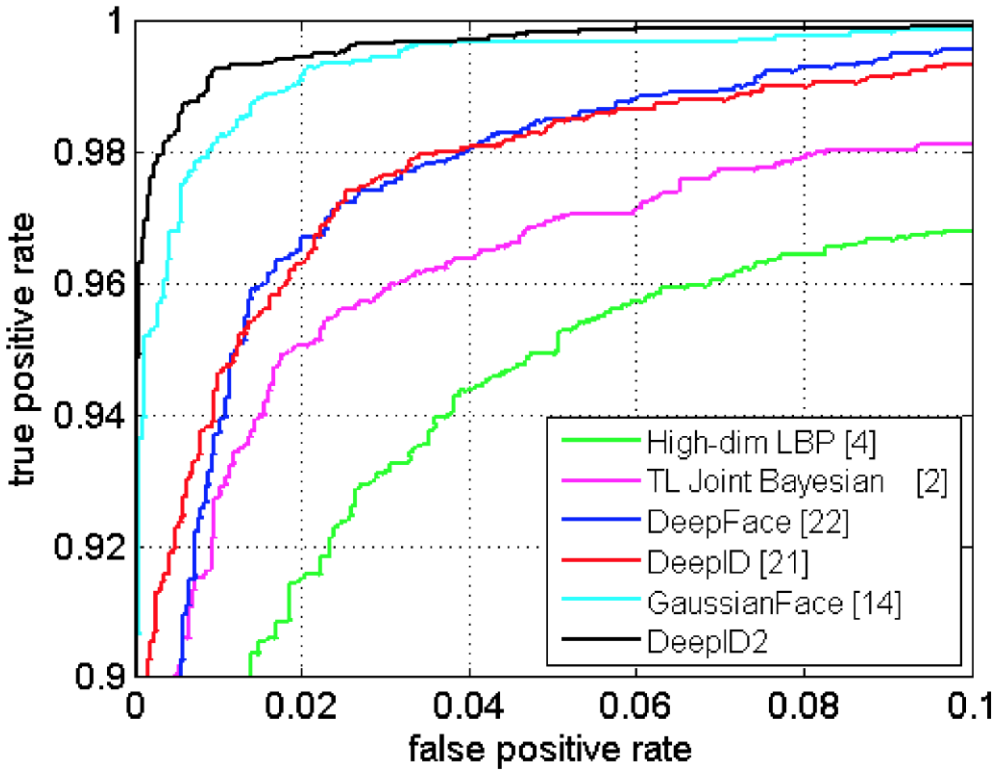
Sun, Yi, Xiaogang Wang, and Xiaoou Tang. "Deep learning face representation from predicting 10,000 classes." CVPR. 2014.
Sun, Yi, et al. "Deep learning face representation by joint identification-verification." NIPS. 2014.

DeepID2

- Some results of DeepID2

method	accuracy (%)
high-dim LBP [4]	95.17 ± 1.13
TL Joint Bayesian [2]	96.33 ± 1.08
DeepFace [22]	97.35 ± 0.25
DeepID [21]	97.45 ± 0.26
GaussianFace [14]	98.52 ± 0.66
DeepID2	99.15 ± 0.13

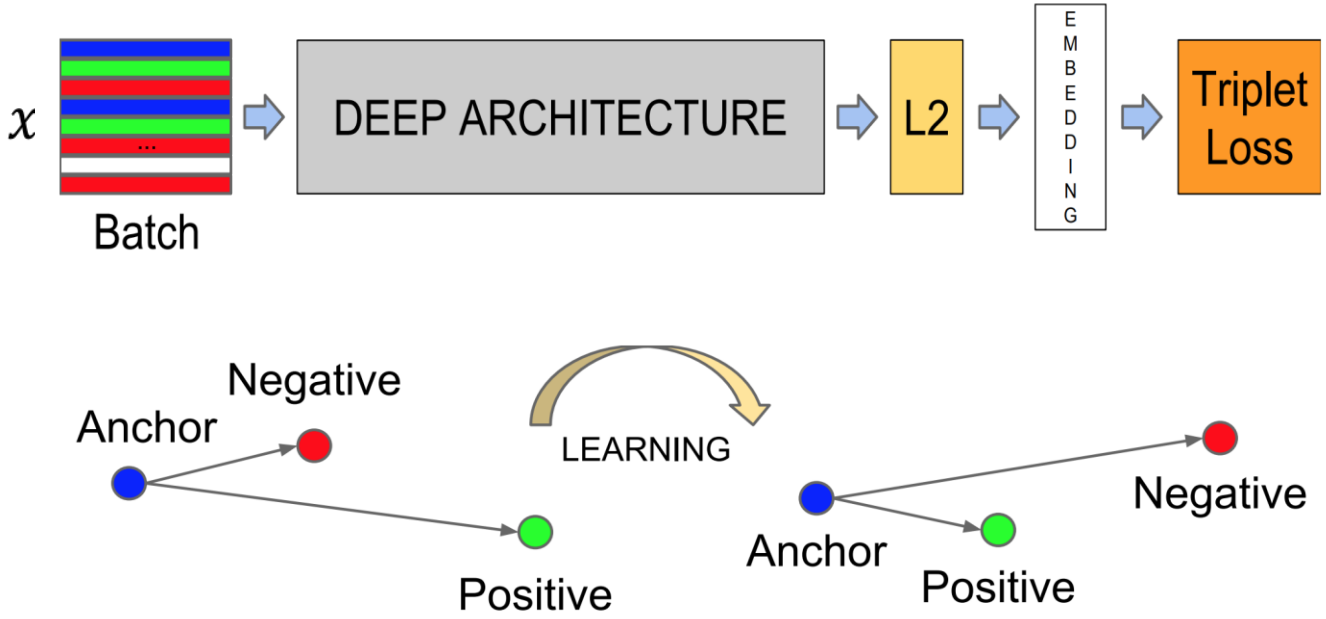
Accuracy comparison with the previous best results on LFW at that time.



Sun, Yi, Xiaogang Wang, and Xiaoou Tang. "Deep learning face representation from predicting 10,000 classes." CVPR. 2014.
Sun, Yi, et al. "Deep learning face representation by joint identification-verification." NIPS. 2014.

Triplet Loss (Google FaceNet)

- The network consists of a batch input layer and a deep CNN followed by L2 normalization, which results in the face embedding.
- 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.



Schroff, Florian, Dmitry Kalenichenko, and James Philbin. "Facenet: A unified embedding for face recognition and clustering." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

Triplet Loss (Google FaceNet)

- The embedding is represented by $f(x)$. It embeds an image x into a d-dimensional Euclidean space. Here we want to ensure that an image x_i^a (anchor) of a specific person is closer to all other images x_i^p (positive) of the same person than it is to any image x_i^n (negative) of any other person. The loss that is being minimized is then

$$L = \sum_i^N \max(0, \|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha)$$

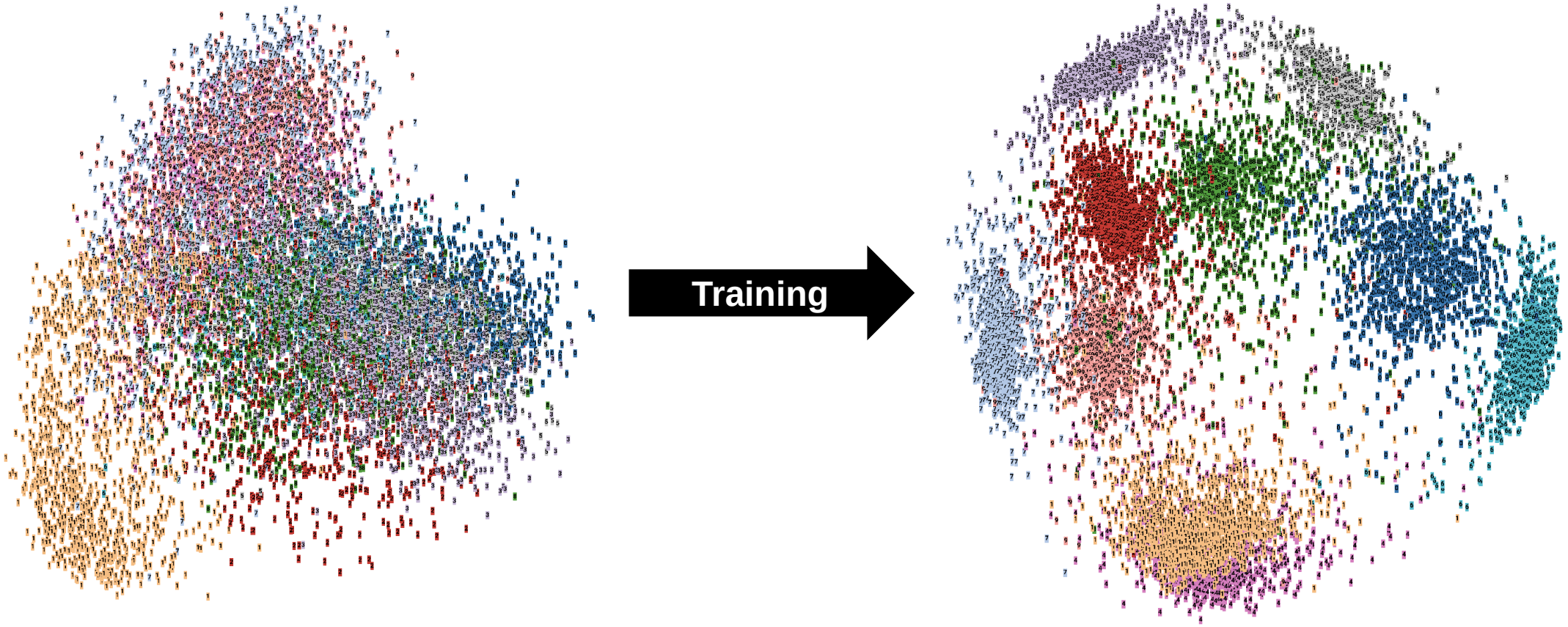
- In order to ensure fast convergence, the following formulation helps to select x_i^n such that

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

Schroff, Florian, Dmitry Kalenichenko, and James Philbin. "FaceNet: A unified embedding for face recognition and clustering." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

Triplet Loss (Google FaceNet)

- MNIST feature distribution of training with triplet loss



Schroff, Florian, Dmitry Kalenichenko, and James Philbin. "Facenet: A unified embedding for face recognition and clustering." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

Center Loss

- As training approaches, DeepID2 and Triplet Loss respectively construct loss functions for image pairs and triplet. However, compared to the image samples, the number of training pairs or triplets dramatically grows. It inevitably results in slow convergence and instability.
- Center loss also uses identification loss (softmax cross-entropy loss) as one of training supervisory signals:

$$\mathcal{L}_S = -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T \boldsymbol{x}_i + b_j}}$$

- Besides softmax loss, an auxiliary loss item is added to gather features in their corresponding centers:

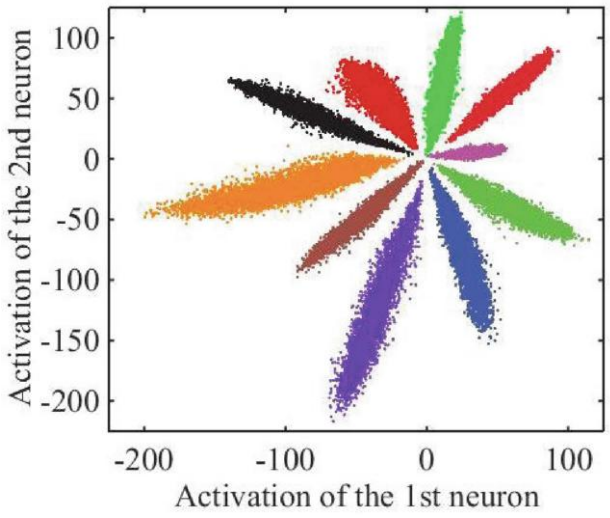
$$\mathcal{L} = \mathcal{L}_S + \lambda \mathcal{L}_C$$

$$= -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T \boldsymbol{x}_i + b_j}} + \frac{\lambda}{2} \sum_{i=1}^{m} \| \boldsymbol{x}_i - \boldsymbol{c}_{y_i} \|_2^2$$

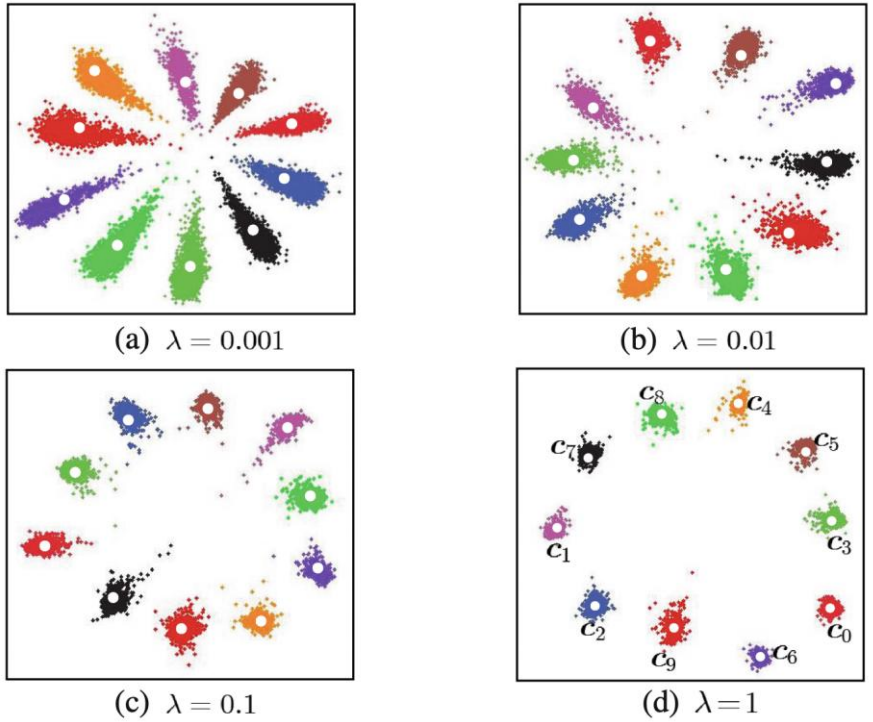
Wen, Yandong, et al. "A discriminative feature learning approach for deep face recognition." European conference on computer vision. Springer, Cham, 2016.

Center Loss

- Feature visualization of softmax and center loss



Softmax Loss only



Softmax combined with Center Loss

Wen, Yandong, et al. "A discriminative feature learning approach for deep face recognition." European conference on computer vision. Springer, Cham, 2016.

Center Loss

- Some results:

Method	Images	Networks	Acc. on LFW	Acc. on YTF
DeepFace [34]	4M	3	97.35 %	91.4 %
DeepID-2+ [32]	-	1	98.70 %	-
DeepID-2+ [32]	-	25	99.47 %	93.2 %
FaceNet [27]	200M	1	99.63 %	95.1 %
Deep FR [25]	2.6M	1	98.95 %	97.3 %
Baidu [21]	1.3M	1	99.13 %	-
model A	0.7M	**1**	97.37 %	91.1 %
model B	0.7M	**1**	99.10 %	93.8 %
model C (Proposed)	**0.7M**	**1**	**99.28 %**	**94.9 %**

Verification performance of different methods on LFW and YTF datasets

Barebones_FR - cnn	Small	59.363 %
NTechLAB - facenx_small	Small	58.218 %
3DiVi Company - tdvm6	Small	33.705 %
Model A-	Small	41.863 %
Model B-	Small	57.175 %
Model C- (Proposed)	**Small**	**65.234 %**

Identification rates of different methods on MegaFace with 1M distractors

Wen, Yandong, et al. "A discriminative feature learning approach for deep face recognition." European conference on computer vision. Springer, Cham, 2016.

Agenda

- Preview of several evaluation datasets of face recognition
- Preview of several different metrics
- Some Euclidean metric based losses
- **Some Cosine metric based losses**

Some Cosine metric based losses

- Large-Margin Softmax Loss for Convolutional Neural Networks (ICML 2016)
- SphereFace: Deep Hypersphere Embedding for Face Recognition（CVPR 2017）
- NormFace: L2 Hypersphere Embedding for Face Verification (ACM-MM 2017)
- CosFace: Large Margin Cosine Loss for Deep Face Recognition (CVPR 2018)
- ArcFace: Additive Angular Margin Loss for Deep Face Recognition (CVPR 2019)
- AdaCos: Adaptively Scaling Cosine Logit for Learning Deep Face Representation (CVPR 2019)

L-Softmax

- L-softmax casts a novel view on generalizing the original softmax loss.
- It denote the i-th input feature x_i having the label y_i. Then the original softmax loss can be written as:

$$L = \frac{1}{N}\sum_i L_i = \frac{1}{N}\sum_i -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right)$$

where

$$L_i = -\log\left(\frac{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\cos(\theta_{y_i})}}{\sum_j e^{\|\boldsymbol{W}_j\|\|\boldsymbol{x}_i\|\cos(\theta_j)}}\right)$$

- L-softmax adds a margin hyperparameter in the following formulation:

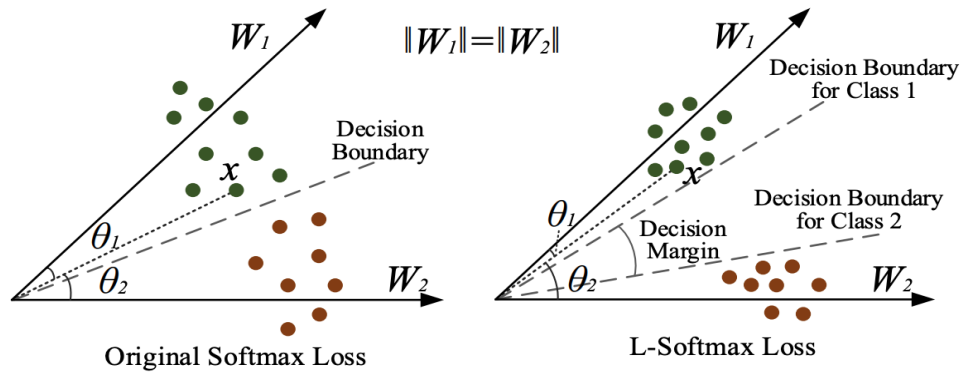
$$L_i = -\log\left(\frac{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\psi(\theta_{y_i})}}{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\psi(\theta_{y_i})} + \sum_{j\neq y_i} e^{\|\boldsymbol{W}_j\|\|\boldsymbol{x}_i\|\cos(\theta_j)}}\right)$$

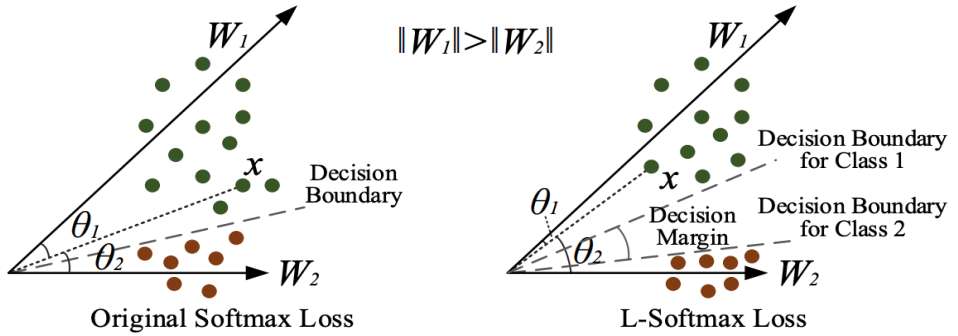
$$\psi(\theta) = \begin{cases} \cos(m\theta), & 0 \le \theta \le \dfrac{\pi}{m} \\ \mathcal{D}(\theta), & \dfrac{\pi}{m} < \theta \le \pi \end{cases}$$

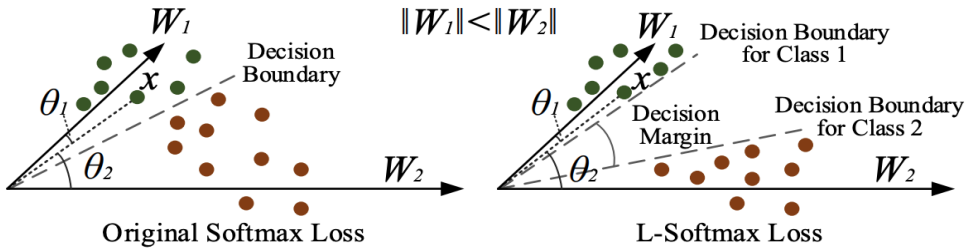
where m is a integer that is closely related to the classification margin. With larger m, the classification margin becomes larger and the learning objective also becomes harder.

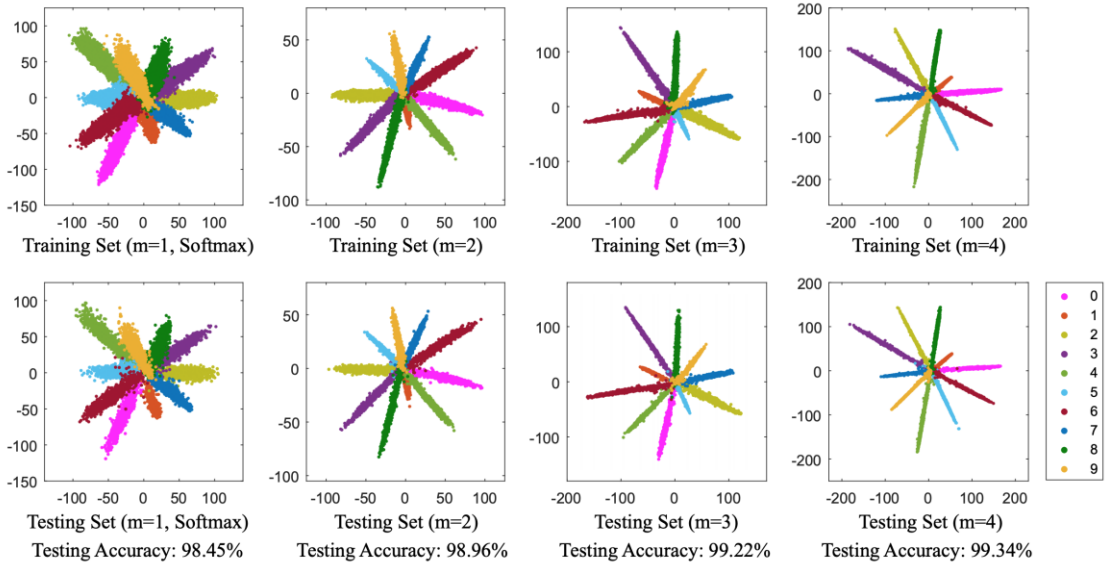
Liu, Weiyang, et al. "Large-Margin Softmax Loss for Convolutional Neural Networks." ICML. Vol. 2. No. 3. 2016.

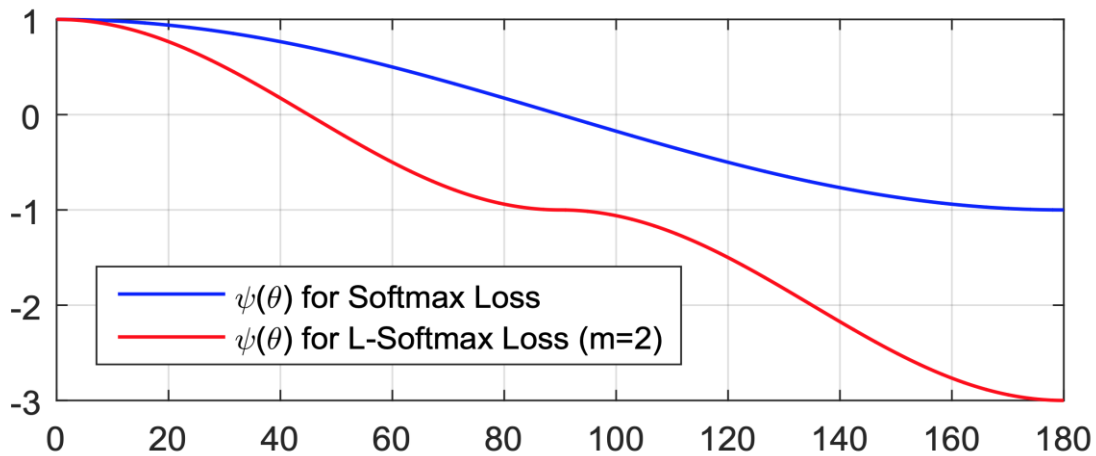
L-Softmax











ψ(θ) for softmax loss and L-Softmax loss.

Liu, Weiyang, et al. "Large-Margin Softmax Loss for Convolutional Neural Networks." ICML. Vol. 2. No. 3. 2016.

L-Softmax

Some results of L-Softmax

Method	Outside Data	Accuracy
FaceNet (Schroff et al., 2015)	200M*	**99.65**
Deep FR (Parkhi et al., 2015)	2.6M	98.95
DeepID2+ (Sun et al., 2015)	300K*	98.70
(Yi et al., 2014)	WebFace	97.73
(Ding & Tao, 2015)	WebFace	98.43
Softmax	WebFace	96.53
Softmax + Contrastive	WebFace	97.31
L-Softmax (m=2)	WebFace	97.81
L-Softmax (m=3)	WebFace	98.27
L-Softmax (m=4)	WebFace	**98.71**

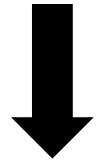
Verification performance (%) on LFW dataset.

Liu, Weiyang, et al. "Large-Margin Softmax Loss for Convolutional Neural Networks." ICML. Vol. 2. No. 3. 2016.

A-Softmax (SphereFace)

- Compared with L-Softmax, A-Softmax normalized all class weights so that maps them into a hypersphere. This can make cosine metric more nature.

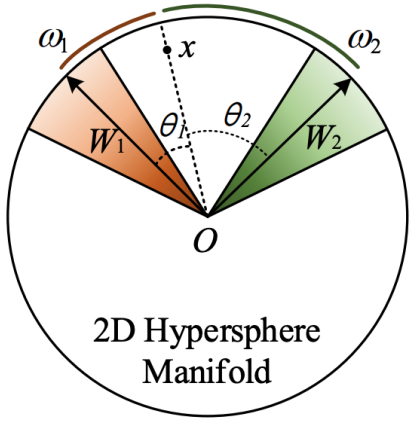
$$L_i = -\log\left(\frac{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\psi(\theta_{y_i})}}{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\psi(\theta_{y_i})} + \sum_{j\neq y_i} e^{\|\boldsymbol{W}_j\|\|\boldsymbol{x}_i\|\cos(\theta_j)}}\right)$$

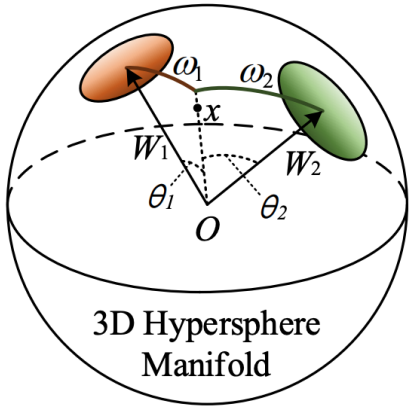


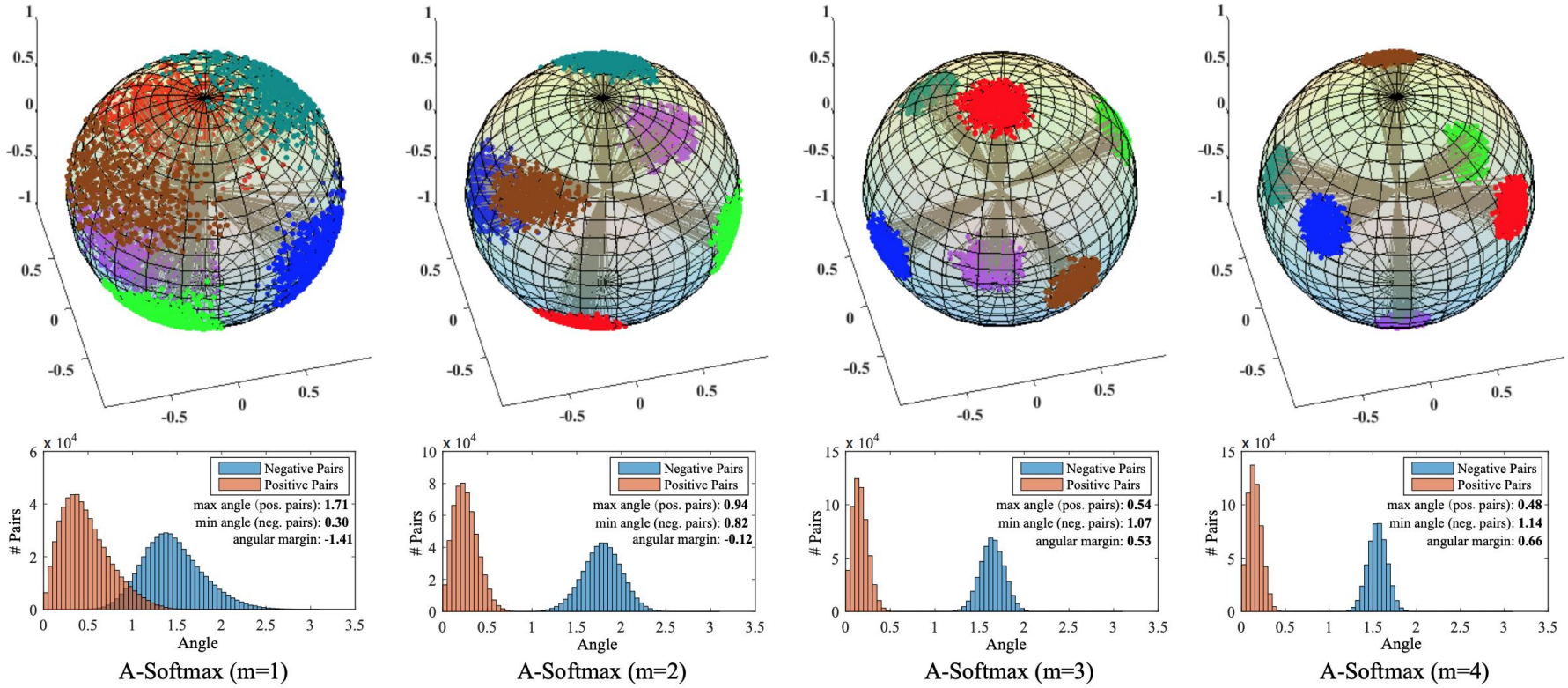
$$L_i = -\log\left(\frac{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})}}{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})} + \sum_{j\neq y_i} e^{\|\boldsymbol{x}_i\|\cos(\theta_{j,i})}}\right)$$

Liu, Weiyang, et al. "Sphereface: Deep hypersphere embedding for face recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

A-Softmax (SphereFace)







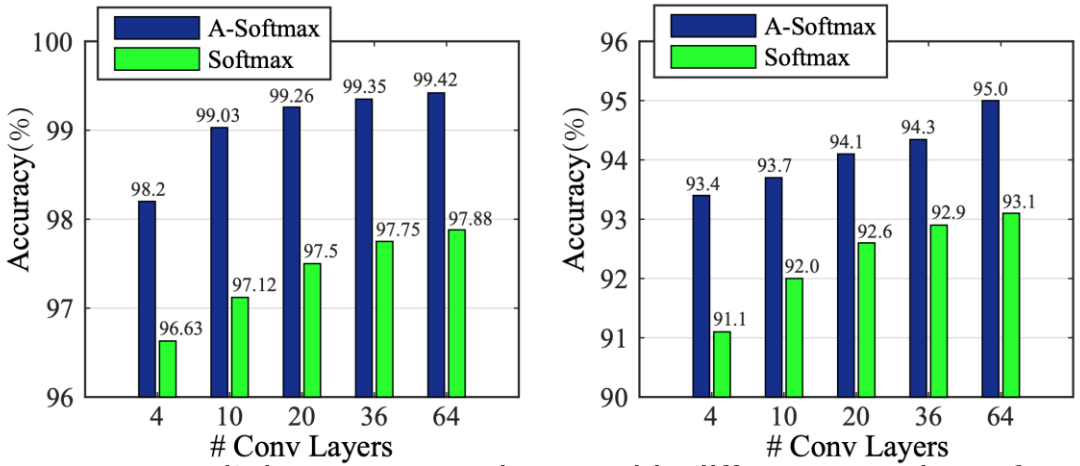
- Visualization of features learned with different m by using a 6-class subset of the CASIA-WebFace dataset.

Liu, Weiyang, et al. "Sphereface: Deep hypersphere embedding for face recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

A-Softmax (SphereFace)

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**

Performance (%) on MegaFace challenge.



Accuracy (%) on LFW and YTF with different number of convolutional layers. Left side is for LFW, while right side is for YTF.

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**

Accuracy (%) on LFW and YTF dataset.

Liu, Weiyang, et al. "Sphereface: Deep hypersphere embedding for face recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

NormFace

- NormFace normalized both features and class weights in softmax cross-entropy loss so that all logits in softmax become cosine metirc.

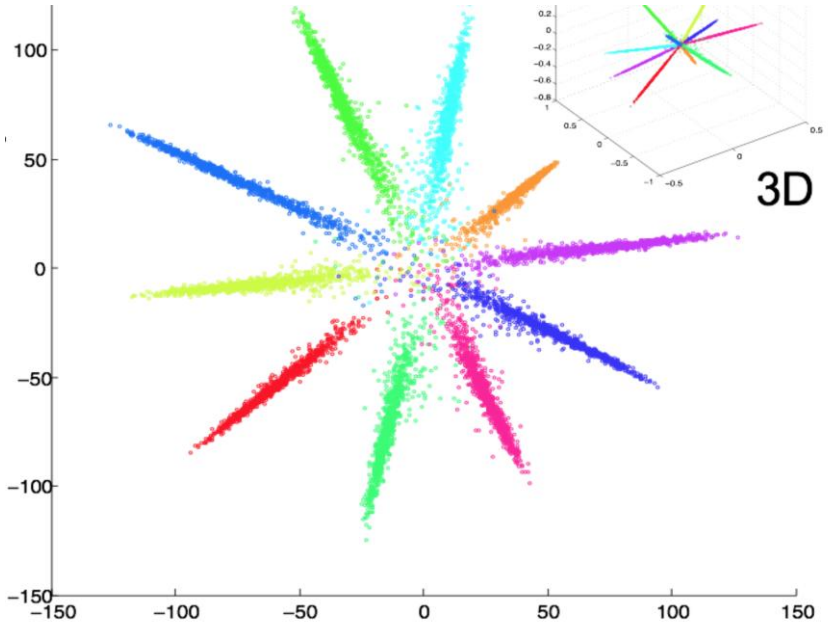
$$f_{i,j} = s \cdot \cos\theta_{i,j} \qquad P_{i,j} = \frac{e^{f_{i,j}}}{\sum_{k=1}^{C} e^{f_{i,k}}}$$

$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{N}\sum_{i=1}^{N}\log P_{i,y_i} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{f_{i,y_i}}}{\sum_{k=1}^{C} e^{f_{i,k}}}$$

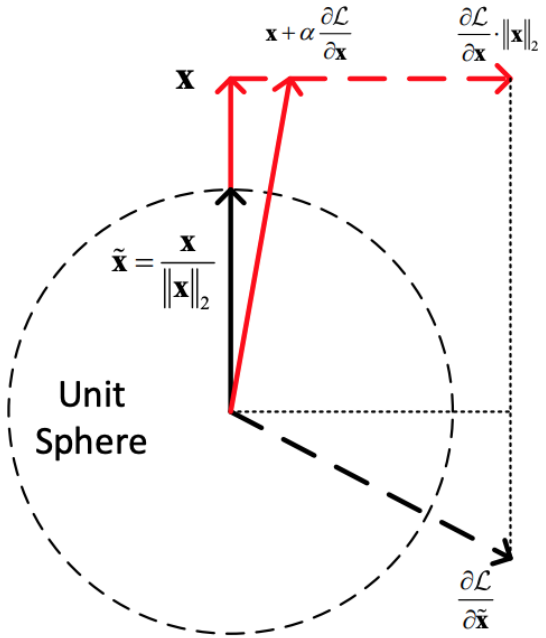
where hyper parameter s is the scaling parameter that enlarges the range of cosine logits

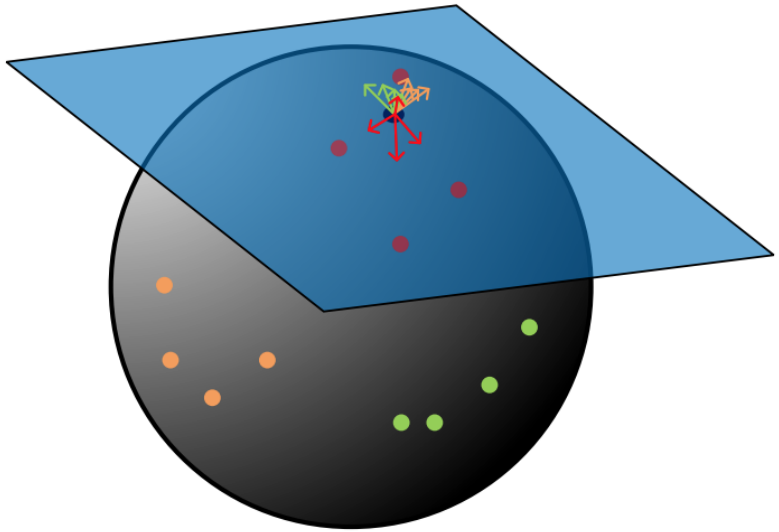
Wang, Feng, et al. "NormFace: l 2 hypersphere embedding for face verification." Proceedings of the 25th ACM international conference on Multimedia. ACM, 2017.

NormFace



MNIST 2-D feature visualization.





The normalization operation and its gradient in 2-dimensional space.

Wang, Feng, et al. "Normface: l 2 hypersphere embedding for face verification." Proceedings of the 25th ACM international conference on Multimedia. ACM, 2017.

NormFace

loss function	Normalization	Accuracy
softmax	No	98.28%
softmax + dropout	No	98.35%
softmax + center[36]	No	99.03%
softmax	feature only	98.72%
softmax	weight only	98.95%
softmax	Yes	99.16% ± 0.025%
softmax + center	Yes	99.17% ± 0.017%
C-contrasitve	Yes	99.15% ± 0.017%
C-triplet	Yes	99.11% ± 0.008%
C-triplet + center	Yes	99.13% ± 0.017%
softmax + C-contrastive	Yes	99.19% ± 0.008%

LFW resutls. Here normalization indicate w/o NormFace operation.

loss function	Normalization	Accuracy
softmax + center[36]	No	93.74%
softmax	Yes	94.24%
softmax + HIK-SVM	Yes	94.56%
C-triplet + center	Yes	94.3%
C-triplet + center + HIK-SVM	Yes	94.58%
softmax + C-contrastive	Yes	94.34%
softmax + C-contrastive + HIK-SVM	Yes	94.72%

YTF resutls. Here normalization indicate w/o NormFace operation.

Wang, Feng, et al. "Normface: l 2 hypersphere embedding for face verification." Proceedings of the 25th ACM international conference on Multimedia. ACM, 2017.

CosFace & ArcFace

- Both CosFace and ArcFace add a margin hyperparameter in NormFace. Their difference is the location where margins are added. For CosFace, margin is added on cosine metrics:

$$L_{lmc} = \frac{1}{N}\sum_{i} -\log \frac{e^{s(\cos(\theta_{y_i,i})-m)}}{e^{s(\cos(\theta_{y_i,i})-m)} + \sum_{j\neq y_i} e^{s\cos(\theta_{j,i})}}$$

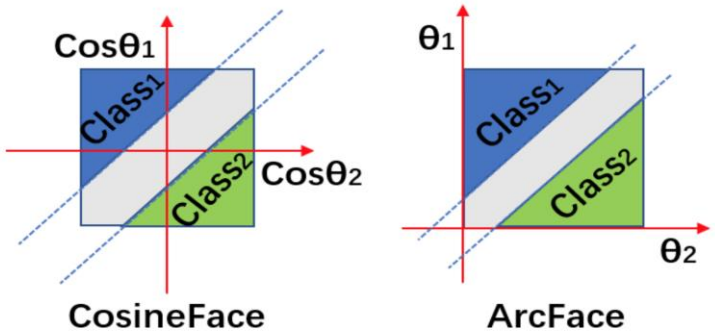
- For ArcFace, margin is added on angular:

$$L_3 = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))} + \sum_{j=1,j\neq y_i}^{n} e^{s\cos\theta_j}}$$

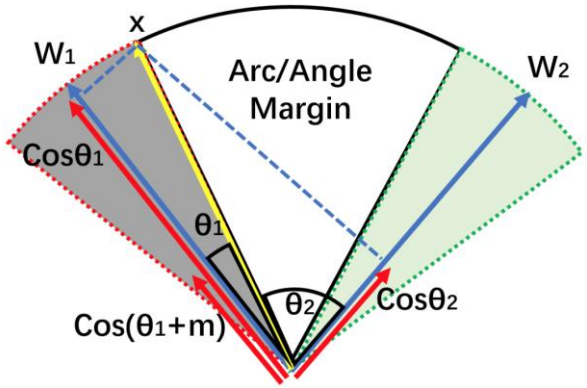
- Although their formulations are different, their main ideas are same.

Wang, Hao, et al. "Cosface: Large margin cosine loss for deep face recognition." Proceedings of the IEEE CVPR. 2018.
Deng, Jiankang, et al. "Arcface: Additive angular margin loss for deep face recognition." Proceedings of the IEEE CVPR. 2019.

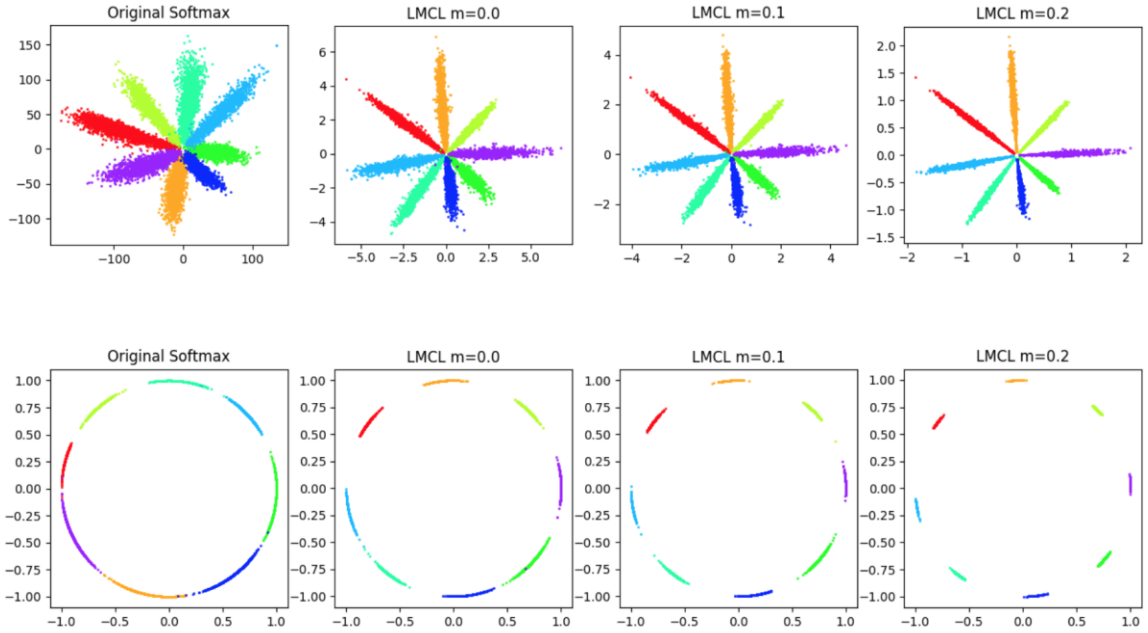
CosFace & ArcFace



Decision margins of CosFace and ArcFace under binary classification case.



Geometrical interpretation of ArcFace.



A toy experiment of CosFace with different margin on 8 identities with 2D features. The first row maps the 2D features onto the Euclidean space, while the second row projects the 2D features onto the angular space. The gap becomes evident as the margin term margin increases

Wang, Hao, et al. "Cosface: Large margin cosine loss for deep face recognition." Proceedings of the IEEE CVPR. 2018.
Deng, Jiankang, et al. "Arcface: Additive angular margin loss for deep face recognition." Proceedings of the IEEE CVPR. 2019.

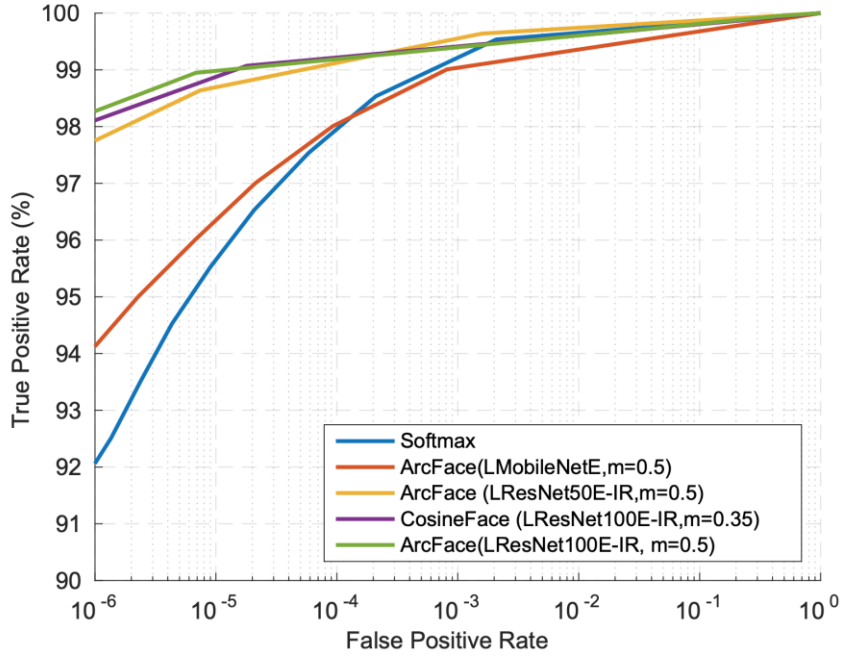
CosFace & ArcFace

Method	LFW	YTF	MF1 Rank1	MF1 Veri.
Softmax Loss [23]	97.88	93.1	54.85	65.92
Softmax+Contrastive [30]	98.78	93.5	65.21	78.86
Triplet Loss [29]	98.70	93.4	64.79	78.32
L-Softmax Loss [24]	99.10	94.0	67.12	80.42
Softmax+Center Loss [42]	99.05	94.4	65.49	80.14
A-Softmax [23]	**99.42**	95.0	72.72	85.56
A-Softmax-NormFea	99.32	95.4	75.42	88.82
LMCL	99.33	**96.1**	**77.11**	**89.88**

Comparison of the proposed CosFace with state-of-the-art loss functions in face recognition community.

Method	Protocol	MF2 Rank1	MF2 Veri.
3DiVi	Large	57.04	66.45
Team 2009	Large	58.93	71.12
NEC	Large	62.12	66.84
GRCCV	Large	75.77	74.84
SphereFace	Large	71.17	84.22
CosFace (Single-patch)	Large	74.11	86.77
CosFace(3-patch ensemble)	Large	**77.06**	**90.30**

CosFace identification and verification evaluation on MegaFace C2.



Methods	Rank1@10^6	VR@FAR10^{-6}
Softmax	78.89	94.95
Softmax-pretrain,Triplet-finetune	80.6	94.65
Softmax-pretrain@VGG2, Triplet-finetune	78.87	95.43
SphereFace(m=4, λ=5)	82.95	97.66
CosineFace(m=0.35)	82.75	98.41
ArcFace(m=0.4)	82.29	98.20
ArcFace(m=0.5)	**83.27**	**98.48**

Identification and verification results of different methods on MegaFace C1.

Wang, Hao, et al. "Cosface: Large margin cosine loss for deep face recognition." Proceedings of the IEEE CVPR. 2018.
Deng, Jiankang, et al. "Arcface: Additive angular margin loss for deep face recognition." Proceedings of the IEEE CVPR. 2019.

AdaCos

- The cosine-based softmax losses and its variants achieve great success in deep learning based face recognition.
- However, hyperparameter settings in these losses have significant influences on the optimization path as well as the final recognition performance.
- Manually tuning those hyperparameters heavily relies on user experience and requires many training tricks.

$$f_{i,j} = s \cdot \cos\left(\theta_{i,j} + \mathbb{1}\{j = y_i\} \cdot m\right)$$

$$P_{i,j} = \frac{e^{f_{i,j}}}{\sum_{k=1}^{C} e^{f_{i,k}}}$$

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N}\sum_{i=1}^{N} \log P_{i,y_i} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{f_{i,y_i}}}{\sum_{k=1}^{C} e^{f_{i,k}}}$$

Here take ArcFace as the example: hyperparameters s and m can influence probability $P_{i,j}$ by directly computing logit $f_{i,j}$

AdaCos

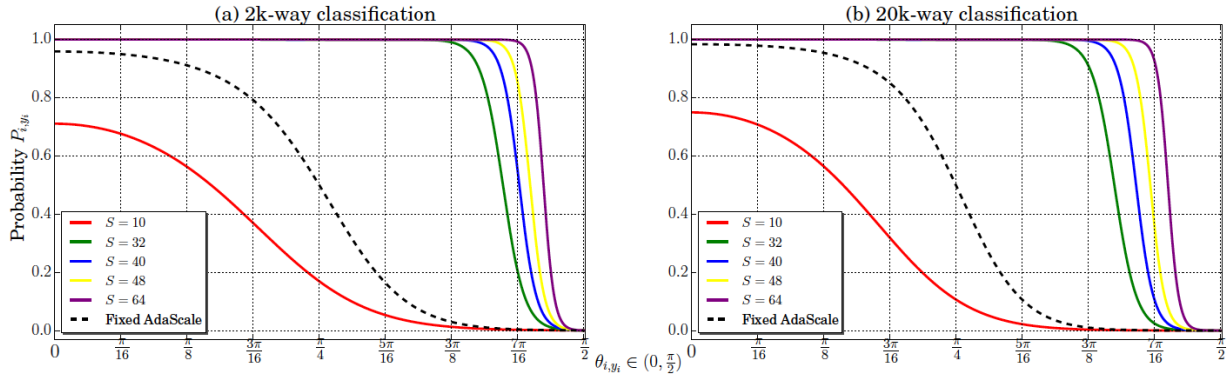


Figure 2. Curves of P_{i,y_i} w.r.t. θ_{i,y_i} when choosing different scale parameters. (Left) $C = 2000$. (Right) $C = 20000$.

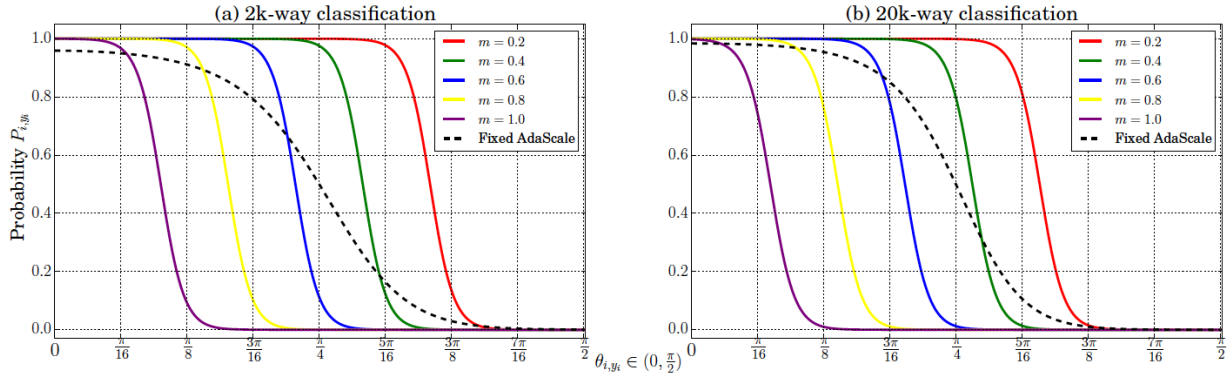


Figure 3. Curves of P_{i,y_i} w.r.t. θ_{i,y_i} when fixing $s = 30$ and choosing different margin parameters. (Left) $C = 2000$. (Right) $C = 20000$.

The effectiveness of hyperparameters s and m can be equal in some degree, hence reserving one of them can simplify the training. Considering s can enlarge the range of cosine metric, AdaCos reserve s as the only hyperparameter.

AdaCos

$$P_{i,y_i} = \frac{e^{f_{i,y_i}}}{e^{f_{i,y_i}} + B_i} = \frac{e^{s \cdot \cos\theta_{i,y_i}}}{e^{s \cdot \cos\theta_{i,y_i}} + B_i}$$

$$B_i = \sum_{k \neq y_i} e^{f_{i,k}} = \sum_{k \neq y_i} e^{s \cdot \cos\theta_{i,k}}$$

$$\frac{\partial^2 P_{i,y_i}(\theta_0)}{\partial {\theta_0}^2} = 0 \qquad s_0 = \frac{\log B_i}{\cos\theta_0}$$

By analyzing the relationship between s and $P_{i,j}$, we found that, given a θ_0 and its expected $P_{i,y_i}(\theta_0)$, a specified s_0 will be computed.

For example, let $P_{i,y_i}(\theta_0)$ have the largest rate of change when $\theta_0 = \frac{\pi}{4}$ is reasonable.

$$\tilde{s}_f = \frac{\log B_i}{\cos\frac{\pi}{4}} = \frac{\log \sum_{k \neq y_i} e^{s \cdot \cos\theta_{i,k}}}{\cos\frac{\pi}{4}}$$

$$\approx \sqrt{2} \cdot \log(C - 1)$$

AdaCos

Since it is easy to figure out the dynamic median θ_{i,y_i} of a mini-batch, we then propose the dynamic AdaCos:

$$B_{\text{avg}}^{(t)} = \frac{1}{N} \sum_{i \subset \mathcal{N}^{(t)}} B_i^{(t)} = \frac{1}{N} \sum_{i \in \mathcal{N}^{(t)}} \sum_{k \neq y_i} e^{\tilde{s}_d^{(t-1)} \cdot \cos \theta_{i,k}}$$

$$\tilde{s}_d^{(t)} = \frac{\log B_{\text{avg}}^{(t)}}{\cos \theta_{\text{med}}^{(t)}}$$

$$\tilde{s}_d^{(t)} = \begin{cases} \sqrt{2} \cdot \log (C-1) & t = 0, \\ \dfrac{\log B_i^{(t)}}{\cos \left(\min(\frac{\pi}{4}, \theta_{\text{med}}^{(t)}) \right)} & t \geq 1, \end{cases}$$

Here each sample/feature have a specified scaling parameter.

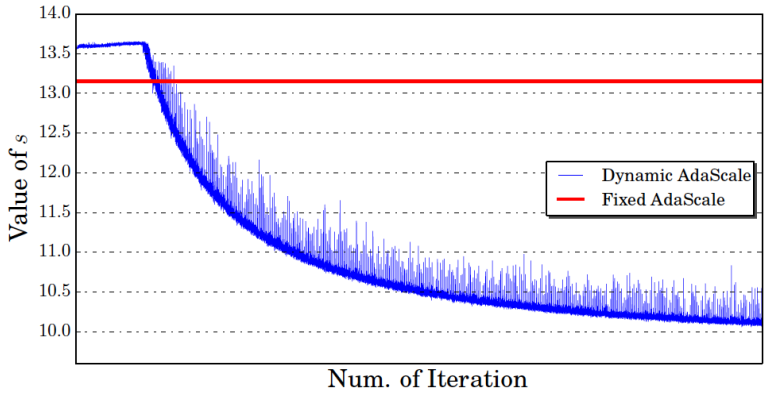


Figure 4. The change of the fixed adaptive scale parameter $\tilde{s}_f$ and dynamic adaptive scale parameter $\tilde{s}_d^{(t)}$ when training on the cleaned WebFace dataset. The dynamic scale parameter $\tilde{s}_d^{(t)}$ gradually and automatically decreases to strengthen training supervisions for feature angles θ_{i,y_i}, which validates our assumption on the adaptive scale parameter in our proposed AdaScale loss. Best viewed in color.

AdaCos

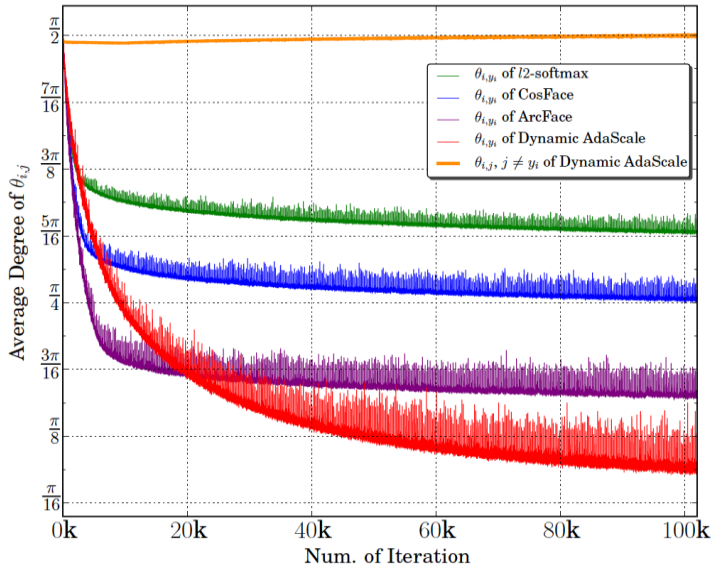


Figure 5. **The change of θ_{i,y_i} when training on the cleaned WebFace dataset.** θ_{i,y_i} represents the angle between the feature vector of i-th sample and the weight vector of its ground truth category y_i. Curves calculated by proposed dynamic AdaScale loss, $l2$-softmax loss [29], CosFace [41] and ArcFace [8] are shown. Best viewed in color.

Method	Num. of Iteration			
	25k	50k	75k	100k
Softmax	70.15	85.33	89.50	93.05
$l2$-softmax [29]	79.08	88.52	93.38	98.22
CosFace [41]	78.17	90.87	98.52	99.37
ArcFace [8]	82.43	92.37	98.78	99.55
Fixed AdaScale	85.10	94.38	99.05	99.63
Dyna. AdaScale	**88.52**	**95.78**	**99.30**	**99.73**

Table 2. Convergence rates of different softmax losses. At the same iterations, training with our proposed dynamic AdaScale loss leads to the best recognition accuracy.

Method	Size of MegaFace Distractor					
	10^1	10^2	10^3	10^4	10^5	10^6
$l2$-softmax	99.73%	99.49%	99.03%	97.85%	95.56%	92.05%
CosFace	99.82%	99.68%	99.46%	98.57%	97.58%	95.50%
ArcFace	99.78%	99.65%	99.48%	98.87%	98.03%	96.88%
Fixed AdaScale	99.85%	99.70%	99.47%	98.80%	97.92%	96.85%
Dynamic AdaScale	**99.88%**	**99.72%**	**99.51%**	**99.02%**	**98.54%**	**97.41%**

Identification and verification results of different methods on MegaFace C1.