SphereFace: Deep Hypersphere Embedding for Face Recognition

Weiyang Liu[1] Yandong Wen[2] Zhiding Yu[2] Ming Li[3] Bhiksha Raj[2] Le Song[1]
[1]Georgia Institute of Technology [2]Carnegie Mellon University [3]Sun Yat-Sen University
wyliu@gatech.edu, {yandongw,yzhiding}@andrew.cmu.edu, lsong@cc.gatech.edu

Abstract

This paper addresses deep face recognition (FR) problem under open-set protocol, where ideal face features are expected to have smaller maximal intra-class distance than minimal inter-class distance under a suitably chosen metric space. However, few existing algorithms can effectively achieve this criterion. To this end, we propose the angular softmax (A-Softmax) loss that enables convolutional neural networks (CNNs) to learn angularly discriminative features. Geometrically, A-Softmax loss can be viewed as imposing discriminative constraints on a hypersphere manifold, which intrinsically matches the prior that faces also lie on a manifold. Moreover, the size of angular margin can be quantitatively adjusted by a parameter m. We further derive specific m to approximate the ideal feature criterion. Extensive analysis and experiments on Labeled Face in the Wild (LFW), Youtube Faces (YTF) and MegaFace Challenge show the superiority of A-Softmax loss in FR tasks. The code has also been made publicly available[1].

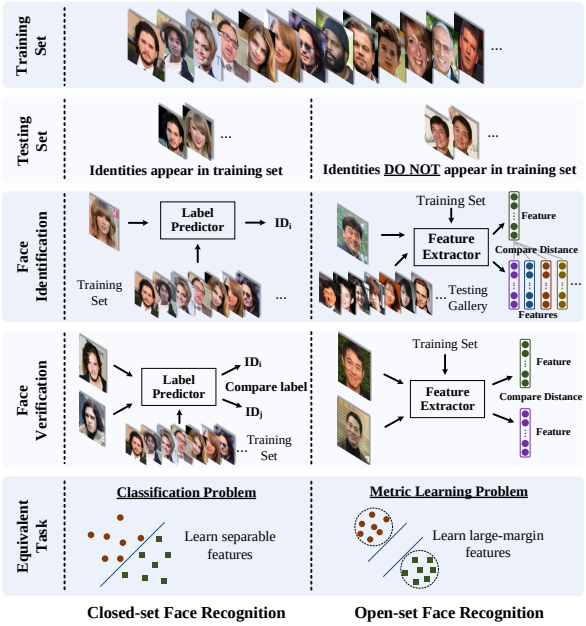


Figure 1: Comparison of open-set and closed-set face recognition.

1. Introduction

Recent years have witnessed the great success of convolutional neural networks (CNNs) in face recognition (FR). Owing to advanced network architectures [13, 23, 29, 4] and discriminative learning approaches [25, 22, 34], deep CNNs have boosted the FR performance to an unprecedent level. Typically, face recognition can be categorized as face identification and face verification [8, 11]. The former classifies a face to a specific identity, while the latter determines whether a pair of faces belongs to the same identity.

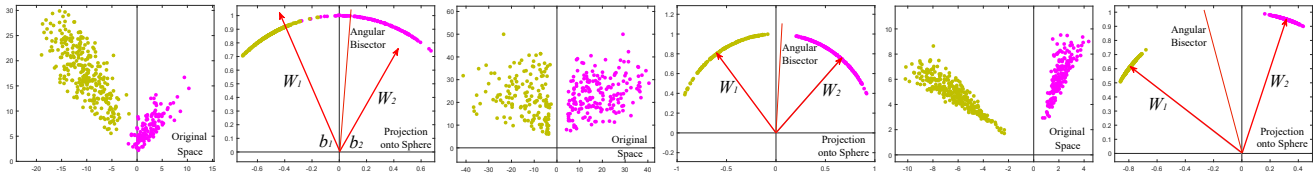
In terms of testing protocol, face recognition can be evaluated under closed-set or open-set settings, as illustrated in Fig. 1. For closed-set protocol, all testing identities are predefined in training set. It is natural to classify testing face images to the given identities. In this scenario, face verification is equivalent to performing identification for a pair of faces respectively (see left side of Fig. 1). Therefore, closed-set FR can be well addressed as a classification problem, where features are expected to be separable. For open-set protocol, the testing identities are usually disjoint from the training set, which makes FR more challenging yet close to practice. Since it is impossible to classify faces to known identities in training set, we need to map faces to a discriminative feature space. In this scenario, face identification can be viewed as performing face verification between the probe face and every identity in the gallery (see right side of Fig. 1). Open-set FR is essentially a metric learning problem, where the key is to learn discriminative large-margin features.

Desired features for open-set FR are expected to satisfy the criterion that the maximal intra-class distance is smaller than the minimal inter-class distance under a certain metric space. This criterion is necessary if we want to achieve perfect accuracy using nearest neighbor. However, learning features with this criterion is generally difficult because of the intrinsically large intra-class variation and high inter-class similarity [21] that faces exhibit.

Few CNN-based approaches are able to effectively formulate the aforementioned criterion in loss functions. Pi-

[1]See the code at https://github.com/wy1iu/sphereface.

Difference between open and closed sets.
This is important.



(a) Original Softmax Loss (b) Original Softmax Loss (c) Modified Softmax Loss (d) Modified Softmax Loss (e) A-Softmax Loss (f) A-Softmax Loss

Figure 2: Comparison among softmax loss, modified softmax loss and A-Softmax loss. In this toy experiment, we construct a CNN to learn 2-D features on a subset of the CASIA face dataset. In specific, we set the output dimension of FC1 layer as 2 and visualize the learned features. Yellow dots represent the first class face features, while purple dots represent the second class face features. One can see that features learned by the original softmax loss can not be classified simply via angles, while modified softmax loss can. Our A-Softmax loss can further increase the angular margin of learned features.

Oh i see i see. It is more discriminative.

oneering work [30, 26] learn face features via the softmax loss[2], but softmax loss only learns separable features that are not discriminative enough. To address this, some methods combine softmax loss with contrastive loss [25, 28] or center loss [34] to enhance the discrimination power of features. [22] adopts triplet loss to supervise the embedding learning, leading to state-of-the-art face recognition results. However, center loss only explicitly encourages intra-class compactness. Both contrastive loss [3] and triplet loss [22] can not constrain on each individual sample, and thus require carefully designed pair/triplet mining procedure, which is both time-consuming and performance-sensitive.

triplet loss needs to be carefully designed.

It seems to be a widely recognized choice to impose Euclidean margin to learned features, but a question arises: *Is Euclidean margin always suitable for learning discriminative face features?* To answer this question, we first look into how Euclidean margin based losses are applied to FR.

Most recent approaches [25, 28, 34] combine Euclidean margin based losses with softmax loss to construct a joint supervision. However, as can be observed from Fig. 2, the features learned by softmax loss have intrinsic angular distribution (also verified by [34]). In some sense, Euclidean margin based losses are incompatible with softmax loss, so it is not well motivated to combine these two type of losses.

In this paper, we propose to incorporate angular margin instead. We start with a binary-class case to analyze the softmax loss. The decision boundary in softmax loss is $(\boldsymbol{W}_1-\boldsymbol{W}_2)\boldsymbol{x}+b_1-b_2=0$, where $\boldsymbol{W}_i$ and b_i are weights and bias[3] in softmax loss, respectively. If we define $\boldsymbol{x}$ as a feature vector and constrain $\|\boldsymbol{W}_1\|=\|\boldsymbol{W}_2\|=1$ and $b_1=b_2=0$, the decision boundary becomes $\|\boldsymbol{x}\|(\cos(\theta_1)-\cos(\theta_2))=0$, where θ_i is the angle between $\boldsymbol{W}_i$ and $\boldsymbol{x}$. The new decision boundary only depends on θ_1 and θ_2. Modified softmax loss is able to directly optimize angles, enabling CNNs to learn angularly distributed features (Fig. 2).

This is ok because W_1 and W_2 are 'unit' vectors. Essentially boils down to dot product: =W1\cdot x - W2\cdot x = ||W1|| ||x|| cos(theta1) - || W2|| ||x|| cos(theta2) = ||x||(cos(theta1) - cos(theta2)).

Compared to original softmax loss, the features learned by modified softmax loss are angularly distributed, but not necessarily more discriminative. To the end, we generalize the modified softmax loss to angular softmax (A-Softmax) loss. Specifically, we introduce an integer m ($m\geq 1$) to quantitatively control the decision boundary. In binary-class case, the decision boundaries for class 1 and class 2 become $\|\boldsymbol{x}\|(\cos(m\theta_1)-\cos(\theta_2))=0$ and $\|\boldsymbol{x}\|(\cos(\theta_1)-\cos(m\theta_2))=0$, respectively. m quantitatively controls the size of angular margin. Furthermore, A-Softmax loss can be easily generalized to multiple classes, similar to softmax loss. By optimizing A-Softmax loss, the decision regions become more separated, simultaneously enlarging the inter-class margin and compressing the intra-class angular distribution.

These are equations. We have inequalities > or < 0 which indicate belonging or not-belonging respectively (presumably)

A-Softmax loss has clear geometric interpretation. Supervised by A-Softmax loss, the learned features construct a discriminative angular distance metric that is equivalent to geodesic distance on a hypersphere manifold. A-Softmax loss can be interpreted as constraining learned features to be discriminative on a hypersphere manifold, which intrinsically matches the prior that face images lie on a manifold [14, 5, 31]. The close connection between A-Softmax loss and hypersphere manifolds makes the learned features more effective for face recognition. For this reason, we term the learned features as *SphereFace*.

Moreover, A-Softmax loss can quantitatively adjust the angular margin via a parameter m, enabling us to do quantitative analysis. In the light of this, we derive lower bounds for the parameter m to approximate the desired open-set FR criterion that the maximal intra-class distance should be smaller than the minimal inter-class distance.

Our major contributions can be summarized as follows:

(1) We propose A-Softmax loss for CNNs to learn discriminative face features with clear and novel geometric interpretation. The learned features discriminatively span on a hypersphere manifold, which intrinsically matches the prior that faces also lie on a manifold.

(2) We derive lower bounds for m such that A-Softmax loss can approximate the learning task that minimal inter-class distance is larger than maximal intra-class distance.

(3) We are the very first to show the effectiveness of angular margin in FR. Trained on publicly available CASIA dataset [37], *SphereFace* achieves competitive results on several benchmarks, including Labeled Face in the Wild (LFW), Youtube Faces (YTF) and MegaFace Challenge 1.

[2]Following [16], we define the softmax loss as the combination of the last fully connected layer, softmax function and cross-entropy loss.

[3]If not specified, the weights and biases in the paper are corresponding to the fully connected layer in the softmax loss.

2. Related Work

Metric learning. Metric learning aims to learn a similarity (distance) function. Traditional metric learning [36, 33, 12, 38] usually learns a matrix $\boldsymbol{A}$ for a distance metric $\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_{\boldsymbol{A}}=\sqrt{(\boldsymbol{x}_1-\boldsymbol{x}_2)^T\boldsymbol{A}(\boldsymbol{x}_1-\boldsymbol{x}_2)}$ upon the given features $\boldsymbol{x}_1, \boldsymbol{x}_2$. Recently, prevailing deep metric learning [7, 17, 24, 30, 25, 22, 34] usually uses neural networks to automatically learn discriminative features $\boldsymbol{x}_1, \boldsymbol{x}_2$ followed by a simple distance metric such as Euclidean distance $\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2$. Most widely used loss functions for deep metric learning are contrastive loss [1, 3] and triplet loss [32, 22, 6], and both impose Euclidean margin to features.

Deep face recognition. Deep face recognition is arguably one of the most active research area in the past few years. [30, 26] address the open-set FR using CNNs supervised by softmax loss, which essentially treats open-set FR as a multi-class classification problem. [25] combines contrastive loss and softmax loss to jointly supervise the CNN training, greatly boosting the performance. [22] uses triplet loss to learn a unified face embedding. Training on nearly 200 million face images, they achieve current state-of-the-art FR accuracy. Inspired by linear discriminant analysis, [34] proposes center loss for CNNs and also obtains promising performance. In general, current well-performing CNNs [28, 15] for FR are mostly built on either contrastive loss or triplet loss. One could notice that state-of-the-art FR methods usually adopt ideas (e.g. contrastive loss, triplet loss) from metric learning, showing open-set FR could be well addressed by discriminative metric learning.

L-Softmax loss [16] also implicitly involves the concept of angles. As a regularization method, it shows great improvement on closed-set classification problems. Differently, A-Softmax loss is developed to learn discriminative face embedding. The explicit connections to hypersphere manifold makes our learned features particularly suitable for open-set FR problem, as verified by our experiments. In addition, the angular margin in A-Softmax loss is explicitly imposed and can be quantitatively controlled (e.g. lower bounds to approximate desired feature criterion), while [16] can only be analyzed qualitatively.

3. Deep Hypersphere Embedding

3.1. Revisiting the Softmax Loss

We revisit the softmax loss by looking into the decision criteria of softmax loss. In binary-class case, the posterior probabilities obtained by softmax loss are

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

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

where $\boldsymbol{x}$ is the learned feature vector. $\boldsymbol{W}_i$ and b_i are weights and bias of last fully connected layer corresponding to class i, respectively. The predicted label will be assigned to class 1 if $p_1>p_2$ and class 2 if $p_1<p_2$. By comparing p_1 and p_2, it is clear that $\boldsymbol{W}_1^T\boldsymbol{x}+b_1$ and $\boldsymbol{W}_2^T\boldsymbol{x}+b_2$ determine the classification result. The decision boundary is $(\boldsymbol{W}_1-\boldsymbol{W}_2)\boldsymbol{x}+b_1-b_2=0$. We then rewrite $\boldsymbol{W}_i^T\boldsymbol{x}+b_i$ as $\|\boldsymbol{W}_i^T\|\|\boldsymbol{x}\|\cos(\theta_i)+b_i$ where θ_i is the angle between $\boldsymbol{W}_i$ and $\boldsymbol{x}$. Notice that if we normalize the weights and zero the biases ($\|\boldsymbol{W}_i\|=1$, $b_i=0$), the posterior probabilities become $p_1=\|\boldsymbol{x}\|\cos(\theta_1)$ and $p_2=\|\boldsymbol{x}\|\cos(\theta_2)$. Note that p_1 and p_2 share the same $\boldsymbol{x}$, the final result only depends on the angles θ_1 and θ_2. The decision boundary also becomes $\cos(\theta_1)-\cos(\theta_2)=0$ (i.e. angular bisector of vector $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$). Although the above analysis is built on binary-calss case, it is trivial to generalize the analysis to multi-class case. During training, the modified softmax loss ($\|\boldsymbol{W}_i\|=1, b_i=0$) encourages features from the i-th class to have smaller angle θ_i (larger cosine distance) than others, which makes angles between $\boldsymbol{W}_i$ and features a reliable metric for classification.

To give a formal expression for the modified softmax loss, we first define the input feature $\boldsymbol{x}_i$ and its label y_i. The original softmax loss can be written as

$$L=\frac{1}{N}\sum_i L_i=\frac{1}{N}\sum_i-\log\Big(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\Big) \tag{3}$$

where f_j denotes the j-th element ($j\in[1,K]$, K is the class number) of the class score vector $\boldsymbol{f}$, and N is the number of training samples. In CNNs, $\boldsymbol{f}$ is usually the output of a fully connected layer $\boldsymbol{W}$, so $f_j=\boldsymbol{W}_j^T\boldsymbol{x}_i+b_j$ and $f_{y_i}=\boldsymbol{W}_{y_i}^T\boldsymbol{x}_i+b_{y_i}$ where $\boldsymbol{x}_i$, $\boldsymbol{W}_j, \boldsymbol{W}_{y_i}$ are the i-th training sample, the j-th and y_i-th column of $\boldsymbol{W}$ respectively. We further reformulate L_i in Eq. (3) as

$$\begin{aligned}L_i&=-\log\Big(\frac{e^{\boldsymbol{W}_{y_i}^T\boldsymbol{x}_i+b_{y_i}}}{\sum_j e^{\boldsymbol{W}_j^T\boldsymbol{x}_i+b_j}}\Big)\\&=-\log\Big(\frac{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\cos(\theta_{y_i,i})+b_{y_i}}}{\sum_j e^{\|\boldsymbol{W}_j\|\|\boldsymbol{x}_i\|\cos(\theta_{j,i})+b_j}}\Big)\end{aligned} \tag{4}$$

in which $\theta_{j,i}(0\le\theta_{j,i}\le\pi)$ is the angle between vector $\boldsymbol{W}_j$ and $\boldsymbol{x}_i$. As analyzed above, we first normalize $\|\boldsymbol{W}_j\|=1, \forall j$ in each iteration and zero the biases. Then we have the modified softmax loss:

$$L_{\text{modified}}=\frac{1}{N}\sum_i-\log\Big(\frac{e^{\|\boldsymbol{x}_i\|\cos(\theta_{y_i,i})}}{\sum_j e^{\|\boldsymbol{x}_i\|\cos(\theta_{j,i})}}\Big) \tag{5}$$

Although we can learn features with angular boundary with the modified softmax loss, these features are still not necessarily discriminative. Since we use angles as the distance metric, it is natural to incorporate angular margin to learned features in order to enhance the discrimination power. To this end, we propose a novel way to combine angular margin.

3.2. Introducing Angular Margin to Softmax Loss

Instead of designing a new type of loss function and constructing a weighted combination with softmax loss (similar

A softmax is more restrictive here.

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

Table 1: Comparison of decision boundaries in binary case. Note that, θ_i is the angle between $\boldsymbol{W}_i$ and $\boldsymbol{x}$.

to contrastive loss) , we propose a more natural way to learn angular margin. From the previous analysis of softmax loss, we learn that decision boundaries can greatly affect the feature distribution, so our basic idea is to manipulate decision boundaries to produce angular margin. We first give a motivating binary-class example to explain how our idea works.

Assume a learned feature $\boldsymbol{x}$ from class 1 is given and θ_i is the angle between $\boldsymbol{x}$ and $\boldsymbol{W}_i$, it is known that the modified softmax loss requires $\cos(\theta_1)>\cos(\theta_2)$ to correctly classify $\boldsymbol{x}$. But what if we instead require $\cos(m\theta_1)>\cos(\theta_2)$ where $m\geq 2$ is a integer in order to correctly classify $\boldsymbol{x}$? It is essentially making the decision more stringent than previous, because we require a lower bound[4] of $\cos(\theta_1)$ to be larger than $\cos(\theta_2)$. The decision boundary for class 1 is $\cos(m\theta_1)=\cos(\theta_2)$. Similarly, if we require $\cos(m\theta_2)>\cos(\theta_1)$ to correctly classify features from class 2, the decision boundary for class 2 is $\cos(m\theta_2)=\cos(\theta_1)$. Suppose all training samples are correctly classified, such decision boundaries will produce an angular margin of $\frac{m-1}{m+1}\theta_2^1$ where θ_2^1 is the angle between $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$. From angular perspective, correctly classifying $\boldsymbol{x}$ from identity 1 requires $\theta_1<\frac{\theta_2}{m}$, while correctly classifying $\boldsymbol{x}$ from identity 2 requires $\theta_2<\frac{\theta_1}{m}$. Both are more difficult than original $\theta_1<\theta_2$ and $\theta_2<\theta_1$, respectively. By directly formulating this idea into the modified softmax loss Eq. (5), we have

$$L_{\text{ang}}=\frac{1}{N}\sum_i-\log\Big(\frac{e^{\|\boldsymbol{x}_i\|\cos(m\theta_{y_i,i})}}{e^{\|\boldsymbol{x}_i\|\cos(m\theta_{y_i,i})}+\sum_{j\neq y_i}e^{\|\boldsymbol{x}_i\|\cos(\theta_{j,i})}}\Big) \tag{6}$$

Why?

where $\theta_{y_i,i}$ has to be in the range of $[0,\frac{\pi}{m}]$. In order to get rid of this restriction and make it optimizable in CNNs, we expand the definition range of $\cos(\theta_{y_i,i})$ by generalizing it to a monotonically decreasing angle function $\psi(\theta_{y_i,i})$ which should be equal to $\cos(\theta_{y_i,i})$ in $[0,\frac{\pi}{m}]$. Therefore, our proposed A-Softmax loss is formulated as:

$$L_{\text{ang}}=\frac{1}{N}\sum_i-\log\Big(\frac{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})}}{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})}+\sum_{j\neq y_i}e^{\|\boldsymbol{x}_i\|\cos(\theta_{j,i})}}\Big) \tag{7}$$

in which we define $\psi(\theta_{y_i,i})=(-1)^k\cos(m\theta_{y_i,i})-2k$, $\theta_{y_i,i}\in[\frac{k\pi}{m},\frac{(k+1)\pi}{m}]$ and $k\in[0,m-1]$. $m\geq 1$ is an integer that controls the size of angular margin. When $m=1$, it becomes the modified softmax loss.

The justification of A-Softmax loss can also be made from decision boundary perspective. A-Softmax loss adopts different decision boundary for different class (each boundary is more stringent than the original), thus producing angular margin. The comparison of decision boundaries is given in Table 1. From original softmax loss to modified softmax loss, it is from optimizing inner product to optimizing angles. From modified softmax loss to A-Softmax loss, it makes the decision boundary more stringent and separated. The angular margin increases with larger m and be zero if $m=1$.

[4]The inequality $\cos(\theta_1)>\cos(m\theta_1)$ holds while $\theta_1\in[0,\frac{\pi}{m}]$, $m\geq 2$.

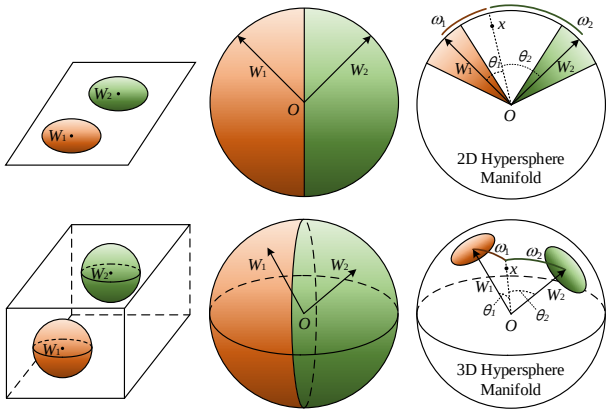


Figure 3: Geometry Interpretation of Euclidean margin loss (e.g. contrastive loss, triplet loss, center loss, etc.), modified softmax loss and A-Softmax loss. The first row is 2D feature constraint, and the second row is 3D feature constraint. The orange region indicates the discriminative constraint for class 1, while the green region is for class 2.

Supervised by A-Softmax loss, CNNs learn face features with geometrically interpretable angular margin. Because A-Softmax loss requires $\boldsymbol{W}_i=1, b_i=0$, it makes the prediction only depends on angles between the sample $\boldsymbol{x}$ and $\boldsymbol{W}_i$. So $\boldsymbol{x}$ can be classified to the identity with smallest angle. The parameter m is added for the purpose of learning an angular margin between different identities.

To facilitate gradient computation and back propagation, we replace $\cos(\theta_{j,i})$ and $\cos(m\theta_{y_i,i})$ with the expressions only containing $\boldsymbol{W}$ and $\boldsymbol{x}_i$, which is easily done by definition of cosine and multi-angle formula (also the reason why we need m to be an integer). Without θ, we can compute derivative with respect to $\boldsymbol{x}$ and $\boldsymbol{W}$, similar to softmax loss.

3.3. Hypersphere Interpretation of A-Softmax Loss

A-Softmax loss has stronger requirements for a correct classification when $m\geq 2$, which generates an angular classification margin between learned features of different classes. A-Softmax loss not only imposes discriminative power to the learned features via angular margin, but also renders nice and novel hypersphere interpretation. As shown in Fig. 3, A-Softmax loss is equivalent to learning features that are discriminative on a hypersphere manifold, while Euclidean margin losses learn features in Euclidean space.

To simplify, We take the binary case to analyze the hypersphere interpretation. Considering a sample $\boldsymbol{x}$ from class 1 and two column weights $\boldsymbol{W}_1,\boldsymbol{W}_2$, the classification rule for

This equivalently part makes it easier to think of.

A-Softmax loss is $\cos(m\theta_1) > \cos(\theta_2)$, equivalently $m\theta_1 < \theta_2$. Notice that θ_1, θ_2 are equal to their corresponding arc length ω_1, ω_2[5] on unit hypersphere $\{v_j, \forall j | \sum_j v_j^2 = 1, v \geq 0\}$. Because $\|\boldsymbol{W}\|_1 = \|\boldsymbol{W}\|_2 = 1$, the decision replies on the arc length ω_1 and ω_2. The decision boundary is equivalent to $m\omega_1 = \omega_2$, and the constrained region for correctly classifying $\boldsymbol{x}$ to class 1 is $m\omega_1 < \omega_2$. Geometrically speaking, this is a hypercircle-like region lying on a hypersphere manifold. For example, it is a circle-like region on the unit sphere in 3D case, as illustrated in Fig. 3. Note that larger m leads to smaller hypercircle-like region for each class, which is an explicit discriminative constraint on a manifold. For better understanding, Fig. 3 provides 2D and 3D visualizations. One can see that A-Softmax loss imposes arc length constraint on a unit circle in 2D case and circle-like region constraint on a unit sphere in 3D case. Our analysis shows that optimizing angles with A-Softmax loss essentially makes the learned features more discriminative on a hypersphere.

3.4. Properties of A-Softmax Loss

Property 1. *A-Softmax loss defines a large angular margin learning task with adjustable difficulty. With larger m, the angular margin becomes larger, the constrained region on the manifold becomes smaller, and the corresponding learning task also becomes more difficult.*

We know that the larger m is, the larger angular margin A-Softmax loss constrains. There exists a minimal m that constrains the maximal intra-class angular distance to be smaller than the minimal inter-class angular distance, which can also be observed in our experiments.

Definition 1 (minimal m for desired feature distribution)**.** *m_{min} is the minimal value such that while $m > m_{\text{min}}$, A-Softmax loss defines a learning task where the maximal intra-class angular feature distance is constrained to be smaller than the minimal inter-class angular feature distance.*

Property 2 (lower bound of m_{min} in binary-class case)**.** *In binary-class case, we have $m_{\text{min}} \geq 2 + \sqrt{3}$.*

Proof. We consider the space spaned by $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$. Because $m \geq 2$, it is easy to obtain the maximal angle that class 1 spans is $\frac{\theta_{12}}{m-1} + \frac{\theta_{12}}{m+1}$ where θ_{12} is the angle between $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$. To require the maximal intra-class feature angular distance smaller than the minimal inter-class feature angular distance, we need to constrain

$$\underbrace{\frac{\theta_{12}}{m-1} + \frac{\theta_{12}}{m+1}}_{\text{max intra-class angle}} \leq \underbrace{\frac{(m-1)\theta_{12}}{m+1}}_{\text{min inter-class angle}}, \quad \theta_{12} \leq \frac{m-1}{m}\pi \tag{8}$$

$$\underbrace{\frac{2\pi - \theta_{12}}{m+1} + \frac{\theta_{12}}{m+1}}_{\text{max intra-class angle}} \leq \underbrace{\frac{(m-1)\theta_{12}}{m+1}}_{\text{min inter-class angle}}, \quad \theta_{12} > \frac{m-1}{m}\pi \tag{9}$$

[5] ω_i is the shortest arc length (geodesic distance) between $\boldsymbol{W}_i$ and the projected point of sample $\boldsymbol{x}$ on the unit hypersphere, while the corresponding θ_i is the angle between $\boldsymbol{W}_i$ and $\boldsymbol{x}$.

After solving these two inequalities, we could have $m_{\text{min}} \geq 2 + \sqrt{3}$, which is a lower bound for binary case. □

Property 3 (lower bound of m_{min} in multi-class case)**.** *Under the assumption that $\boldsymbol{W}_i, \forall i$ are uniformly spaced in the Euclidean space, we have $m_{\text{min}} \geq 3$.*

Proof. We consider the 2D k-class ($k \geq 3$) scenario for the lower bound. Because $\boldsymbol{W}_i, \forall i$ are uniformly spaced in the 2D Euclidean space, we have $\theta_i^{i+1} = \frac{2\pi}{k}$ where θ_i^{i+1} is the angle between $\boldsymbol{W}_i$ and $\boldsymbol{W}_{i+1}$. Since $\boldsymbol{W}_i, \forall i$ are symmetric, we only need to analyze one of them. For the i-th class ($\boldsymbol{W}_i$), We need to constrain

$$\underbrace{\frac{\theta_i^{i+1}}{m+1} + \frac{\theta_{i-1}^{i}}{m+1}}_{\text{max intra-class angle}} \leq \underbrace{\min\left\{\frac{(m-1)\theta_i^{i+1}}{m+1}, \frac{(m-1)\theta_{i-1}^{i}}{m+1}\right\}}_{\text{min inter-class angle}} \tag{10}$$

After solving this inequality, we obtain $m_{\text{min}} \geq 3$, which is a lower bound for multi-class case. □

Based on this, we use $m = 4$ to approximate the desired feature distribution criteria. Since the lower bounds are not necessarily tight, giving a tighter lower bound and a upper bound under certain conditions is also possible, which we leave to the future work. Experiments also show that larger m consistently works better and $m = 4$ will usually suffice.

3.5. Discussions

Why angular margin. First and most importantly, angular margin directly links to discriminativeness on a manifold, which intrinsically matches the prior that faces also lie on a manifold. Second, incorporating angular margin to softmax loss is actually a more natural choice. As Fig. 2 shows, features learned by the original softmax loss have an intrinsic angular distribution. So directly combining Euclidean margin constraints with softmax loss is not reasonable.

Comparison with existing losses. In deep FR task, the most popular and well-performing loss functions include contrastive loss, triplet loss and center loss. First, they only impose Euclidean margin to the learned features (w/o normalization), while ours instead directly considers angular margin which is naturally motivated. Second, both contrastive loss and triplet loss suffer from data expansion when constituting the pairs/triplets from the training set, while ours requires no sample mining and imposes discriminative constraints to the entire mini-batches (compared to contrastive and triplet loss that only affect a few representative pairs/triplets).

e.g. triplet loss is O(N^2).

4. Experiments (more in Appendix)

4.1. Experimental Settings

Preprocessing. We only use standard preprocessing. The face landmarks in all images are detected by MTCNN [39]. The cropped faces are obtained by similarity transformation. Each pixel ($[0, 255]$) in RGB images is normalized by subtracting 127.5 and then being divided by 128.

Layer	4-layer CNN	10-layer CNN	20-layer CNN	36-layer CNN	64-layer CNN
Conv1.x	[3×3, 64]×1, S2	[3×3, 64]×1, S2	[3×3, 64]×1, S2 $\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times1$	[3×3, 64]×1, S2 $\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	[3×3, 64]×1, S2 $\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$
Conv2.x	[3×3, 128]×1, S2	[3×3, 128]×1, S2 $\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times1$	[3×3, 128]×1, S2 $\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	[3×3, 128]×1, S2 $\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	[3×3, 128]×1, S2 $\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times8$
Conv3.x	[3×3, 256]×1, S2	[3×3, 256]×1, S2 $\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	[3×3, 256]×1, S2 $\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times4$	[3×3, 256]×1, S2 $\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times8$	[3×3, 256]×1, S2 $\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times16$
Conv4.x	[3×3, 512]×1, S2	[3×3, 512]×1, S2	[3×3, 512]×1, S2 $\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times1$	[3×3, 512]×1, S2 $\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	[3×3, 512]×1, S2 $\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$
FC1	512	512	512	512	512

Table 2: Our CNN architectures with different convolutional layers. Conv1.x, Conv2.x and Conv3.x denote convolution units that may contain multiple convolution layers and residual units are shown in double-column brackets. E.g., [3×3, 64]×4 denotes 4 cascaded convolution layers with 64 filters of size 3×3, and S2 denotes stride 2. FC1 is the fully connected layer.

CNNs Setup. Caffe [10] is used to implement A-Softmax loss and CNNs. The general framework to train and extract *SphereFace* features is shown in Fig. 4. We use residual units [4] in our CNN architecture. For fairness, all compared methods use the same CNN architecture (including residual units) as *SphereFace*. CNNs with different depths (4, 10, 20, 36, 64) are used to better evaluate our method. The specific settings for diffferent CNNs we used are given in Table 2. According to the analysis in Section 3.4, we usually set m as 4 in A-Softmax loss unless specified. These models are trained with batch size of 128 on four GPUs. The learning rate begins with 0.1 and is divided by 10 at the 16K, 24K iterations. The training is finished at 28K iterations.

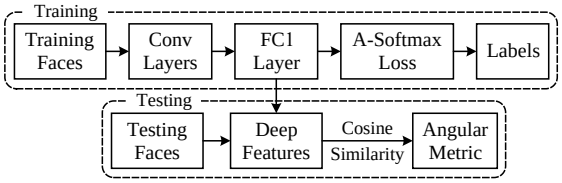


Figure 4: Training and Extracting *SphereFace* features.

Training Data. We use publicly available web-collected training dataset CASIA-WebFace [37] (after excluding the images of identities appearing in testing sets) to train our CNN models. CASIA-WebFace has 494,414 face images belonging to 10,575 different individuals. These face images are horizontally flipped for data augmentation. Notice that the scale of our training data (0.49M) is relatively small, especially compared to other private datasets used in DeepFace [30] (4M), VGGFace [20] (2M) and FaceNet [22] (200M).

Testing. We extract the deep features (*SphereFace*) from the output of the FC1 layer. For all experiments, the final representation of a testing face is obtained by concatenating its original face features and its horizontally flipped features. The score (metric) is computed by the cosine distance of two features. The nearest neighbor classifier and thresholding are used for face identification and verification, respectively.

4.2. Exploratory Experiments

Effect of m. To show that larger m leads to larger angular margin (i.e. more discriminative feature distribution on manifold), we perform a toy example with different m. We train A-Softmax loss with 6 individuals that have the most samples in CASIA-WebFace. We set the output feature dimension (FC1) as 3 and visualize the training samples in Fig. 5. One can observe that larger m leads to more discriminative distribution on the sphere and also larger angular margin, as expected. We also use class 1 (blue) and class 2 (dark green) to construct positive and negative pairs to evaluate the angle distribution of features from the same class and different classes. The angle distribution of positive and negative pairs (the second row of Fig. 5) quantitatively shows the angular margin becomes larger while m increases and every class also becomes more distinct with each other.

Besides visual comparison, we also perform face recognition on LFW and YTF to evaluate the effect of m. For fair comparison, we use 64-layer CNN (Table 2) for all losses. Results are given in Table 3. One can observe that while m becomes larger, the accuracy of A-Softmax loss also becomes better, which shows that larger angular margin can bring stronger discrimination power.

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**

Table 3: Accuracy(%) comparison of different m (A-Softmax loss) and original softmax loss on LFW and YTF dataset.

Effect of CNN architectures. We train A-Softmax loss ($m=4$) and original softmax loss with different number of convolution layers. Specific CNN architectures can be found in Table 2. From Fig. 6, one can observe that A-Softmax loss consistently outperforms CNNs with softmax loss (1.54%∼1.91%), indicating that A-Softmax loss is more suitable for open-set FR. Besides, the difficult learning task

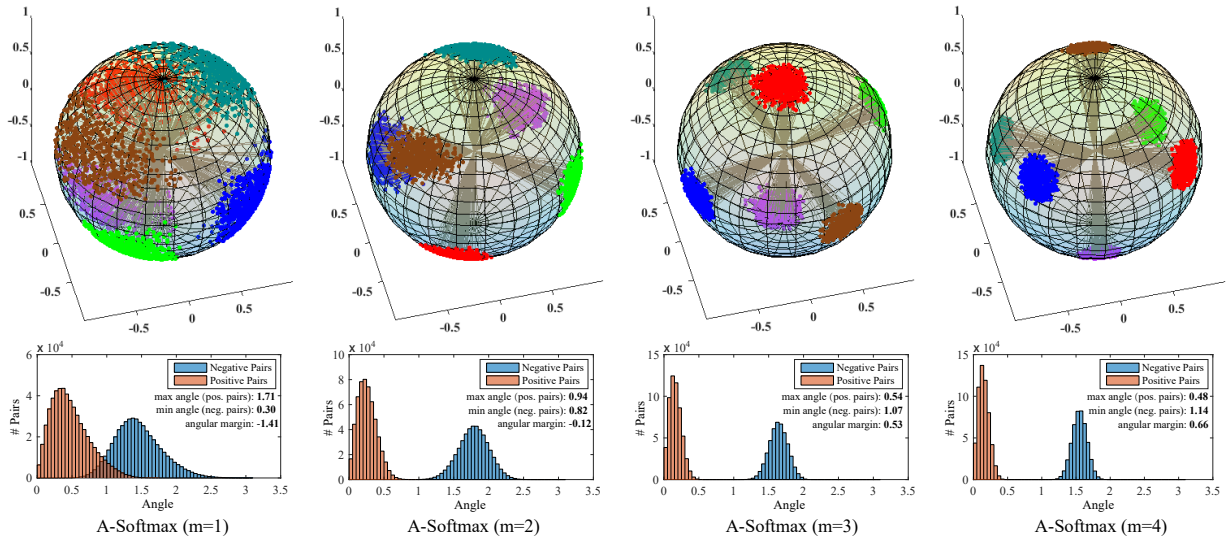


Figure 5: Visualization of features learned with different m. The first row shows the 3D features projected on the unit sphere. The projected points are the intersection points of the feature vectors and the unit sphere. The second row shows the angle distribution of both positive pairs and negative pairs (we choose class 1 and class 2 from the subset to construct positive and negative pairs). Orange area indicates positive pairs while blue indicates negative pairs. All angles are represented in radian. Note that, this visualization experiment uses a 6-class subset of the CASIA-WebFace dataset.

defined by A-Softmax loss makes full use of the superior learning capability of deeper architectures. A-Softmax loss greatly improve the verification accuracy from 98.20% to 99.42% on LFW, and from 93.4% to 95.0% on YTF. On the contrary, the improvement of deeper standard CNNs is unsatisfactory and also easily get saturated (from 96.60% to 97.75% on LFW, from 91.1% to 93.1% on YTF).

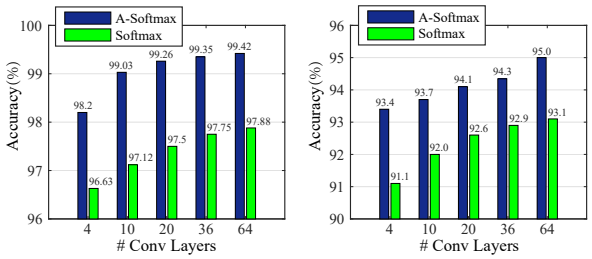


Figure 6: Accuracy (%) on LFW and YTF with different number of convolutional layers. Left side is for LFW, while right side is for YTF.

4.3. Experiments on LFW and YTF

LFW dataset [9] includes 13,233 face images from 5749 different identities, and YTF dataset [35] includes 3,424 videos from 1,595 different individuals. Both datasets contains faces with large variations in pose, expression and illuminations. We follow the unrestricted with labeled outside data protocol [8] on both datasets. The performance of *SphereFace* are evaluated on 6,000 face pairs from LFW and 5,000 video pairs from YTF. The results are given in Table 4. For contrastive loss and center loss, we follow the FR convention to form a weighted combination with softmax loss. The weights are selected via cross validation on training set. For L-Softmax [16], we also use $m=4$. All the compared loss functions share the same 64-layer CNN architecture.

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**

Table 4: Accuracy (%) on LFW and YTF dataset. * denotes the outside data is private (not publicly available). For fair comparison, all loss functions (including ours) we implemented use 64-layer CNN architecture in Table 2.

Most of the existing face verification systems achieve high performance with huge training data or model ensemble. While using single model trained on publicly available dataset (CAISA-WebFace, relatively small and having noisy labels), *SphereFace* achieves 99.42% and 95.0% accuracies on LFW and YTF datasets. It is the current best performance trained on WebFace and considerably better than the other models trained on the same dataset. Compared with models trained on high-quality private datasets, *SphereFace* is still very competitive, outperforming most of the existing results in Table 4. One should notice that our single model performance is only worse than Google FaceNet which is trained with more than 200 million data.

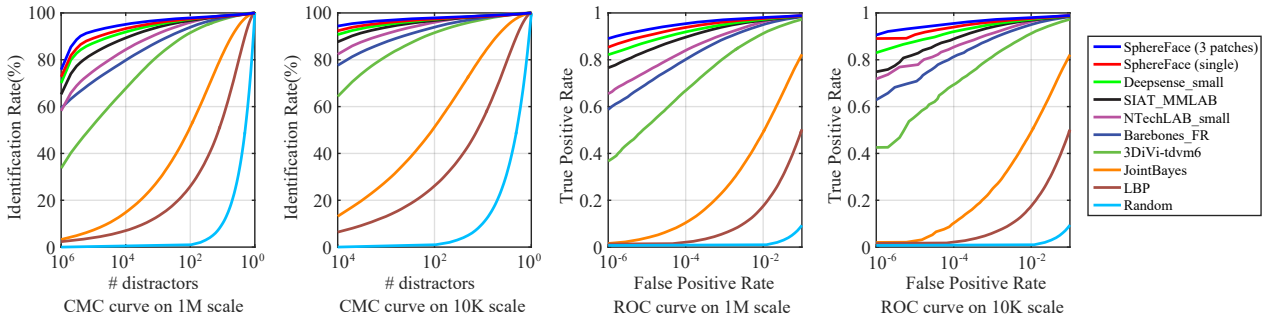


Figure 7: CMC and ROC curves of different methods under the small training set protocol.

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**

Table 5: Performance (%) on MegaFace challenge. "Rank-1 Acc." indicates rank-1 identification accuracy with 1M distractors, and "Ver." indicates verification TAR for 10^{-6} FAR. TAR and FAR denote True Accept Rate and False Accept Rate respectively. For fair comparison, all loss functions (including ours) we implemented use the same deep CNN architecture.

For fair comparison, we also implement the softmax loss, contrastive loss, center loss, triplet loss, L-Softmax loss [16] and train them with the same 64-layer CNN architecture as A-Softmax loss. As can be observed in Table 4, *SphereFace* consistently outperforms the features learned by all these compared losses, showing its superiority in FR tasks.

4.4. Experiments on MegaFace Challenge

MegaFace dataset [18] is a recently released testing benchmark with very challenging task to evaluate the performance of face recognition methods at the million scale of distractors. MegaFace dataset contains a gallery set and a probe set. The gallery set contains more than 1 million images from 690K different individuals. The probe set consists of two existing datasets: Facescrub [19] and FGNet. MegaFace has several testing scenarios including identification, verification and pose invariance under two protocols (large or small training set). The training set is viewed as small if it is less than 0.5M. We evaluate *SphereFace* under the small training set protocol. We adopt two testing protocols: face identification and verification. The results are given in Fig. 7 and Tabel 5. Note that we use simple 3-patch feature concatenation ensemble as the final performance of *SphereFace*.

Fig. 7 and Tabel 5 show that *SphereFace* (3 patches ensemble) beats the second best result by a large margins (4.8% for rank-1 identification rate and 6.3% for verification rate) on MegaFace benchmark under the small training dataset protocol. Compared to the models trained on large dataset (500 million for Google and 18 million for NTechLAB), our method still performs better (0.64% for id. rate and 1.4% for veri. rate). Moreover, in contrast to their sophisticated network design, we only employ typical CNN architecture supervised by A-Softamx to achieve such excellent performance. For single model *SphereFace*, the accuracy of face identification and verification are still 72.73% and 85.56% respectively, which already outperforms most state-of-the-art methods. For better evaluation, we also implement the softmax loss, contrastive loss, center loss, triplet loss and L-Softmax loss [16]. Compared to these loss functions trained with the same CNN architecture and dataset, *SphereFace* also shows significant and consistent improvements. These results convincingly demonstrate that the proposed *SphereFace* is well designed for open-set face recognition. One can also see that learning features with large inter-class angular margin can significantly improve the open-set FR performance.

5. Concluding Remarks

This paper presents a novel deep hypersphere embedding approach for face recognition. In specific, we propose the angular softmax loss for CNNs to learn discriminative face features (*SphereFace*) with angular margin. A-Softmax loss renders nice geometric interpretation by constraining learned features to be discriminative on a hypersphere manifold, which intrinsically matches the prior that faces also lie on a non-linear manifold. This connection makes A-Softmax very effective for learning face representation. Competitive results on several popular face benchmarks demonstrate the superiority and great potentials of our approach. We also believe A-Softmax loss could also benefit some other tasks like object recognition, person re-identification, etc.

References

[1] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005. 3

[2] C. Ding and D. Tao. Robust face recognition via multimodal deep face representation. *IEEE TMM*, 17(11):2049–2058, 2015. 7

[3] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006. 2, 3

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 6

[5] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using laplacianfaces. *TPAMI*, 27(3):328–340, 2005. 2

[6] E. Hoffer and N. Ailon. Deep metric learning using triplet network. *arXiv preprint:1412.6622*, 2014. 3

[7] J. Hu, J. Lu, and Y.-P. Tan. Discriminative deep metric learning for face verification in the wild. In *CVPR*, 2014. 3

[8] G. B. Huang and E. Learned-Miller. Labeled faces in the wild: Updates and new reporting procedures. *Dept. Comput. Sci., Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep*, pages 14–003, 2014. 1, 7

[9] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report, 2007. 7

[10] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint:1408.5093*, 2014. 6

[11] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *CVPR*, 2016. 1

[12] M. Köstinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof. Large scale metric learning from equivalence constraints. In *CVPR*, 2012. 3

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[14] K.-C. Lee, J. Ho, M.-H. Yang, and D. Kriegman. Video-based face recognition using probabilistic appearance manifolds. In *CVPR*, 2003. 2

[15] J. Liu, Y. Deng, and C. Huang. Targeting ultimate accuracy: Face recognition via deep embedding. *arXiv preprint:1506.07310*, 2015. 3, 7

[16] W. Liu, Y. Wen, Z. Yu, and M. Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, 2016. 2, 3, 7, 8, 10, 11, 12

[17] J. Lu, G. Wang, W. Deng, P. Moulin, and J. Zhou. Multi-manifold deep metric learning for image set classification. In *CVPR*, 2015. 3

[18] D. Miller, E. Brossard, S. Seitz, and I. Kemelmacher-Shlizerman. Megaface: A million faces for recognition at scale. *arXiv preprint:1505.02108*, 2015. 8

[19] H.-W. Ng and S. Winkler. A data-driven approach to cleaning large face datasets. In *ICIP*, 2014. 8

[20] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *BMVC*, 2015. 6, 7

[21] A. Ross and A. K. Jain. Multimodal biometrics: An overview. In *Signal Processing Conference, 2004 12th European*, pages 1221–1224. IEEE, 2004. 1

[22] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 1, 2, 3, 6, 7, 8

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint:1409.1556*, 2014. 1

[24] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016. 3

[25] Y. Sun, Y. Chen, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. In *NIPS*, 2014. 1, 2, 3

[26] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *CVPR*, 2014. 2, 3, 7, 8

[27] Y. Sun, X. Wang, and X. Tang. Deeply learned face representations are sparse, selective, and robust. In *CVPR*, 2015. 7

[28] Y. Sun, X. Wang, and X. Tang. Sparsifying neural network connections for face recognition. In *CVPR*, 2016. 2, 3

[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 1

[30] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, 2014. 2, 3, 6, 7

[31] A. Talwalkar, S. Kumar, and H. Rowley. Large-scale manifold learning. In *CVPR*, 2008. 2

[32] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, 2014. 3

[33] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009. 3

[34] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016. 1, 2, 3, 7, 8

[35] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *CVPR*, 2011. 7

[36] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. *NIPS*, 2003. 3

[37] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *arXiv preprint:1411.7923*, 2014. 2, 6, 7

[38] Y. Ying and P. Li. Distance metric learning with eigenvalue optimization. *JMLR*, 13(Jan):1–26, 2012. 3

[39] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multi-task cascaded convolutional networks. *arXiv preprint:1604.02878*, 2016. 5

Appendix

A. The intuition of removing the last ReLU

Standard CNNs usually connect ReLU to the bottom of FC1, so the learned features will only distribute in the non-negative range $[0, +\infty)$, which limits the feasible learning space (angle) for the CNNs. To address this shortcoming, both SphereFace and [16] first propose to remove the ReLU nonlinearity that is connected to the bottom of FC1 in SphereFace networks. Intuitively, removing the ReLU can greatly benefit the feature learning, since it provides larger feasible learning space (from angular perspective).

Visualization on MNIST. Fig. 8 shows the 2-D visualization of feature distributions in MNIST with and without the last ReLU. One can observe with ReLU the 2-D feature could only distribute in the first quadrant. Without the last ReLU, the learned feature distribution is much more reasonable.

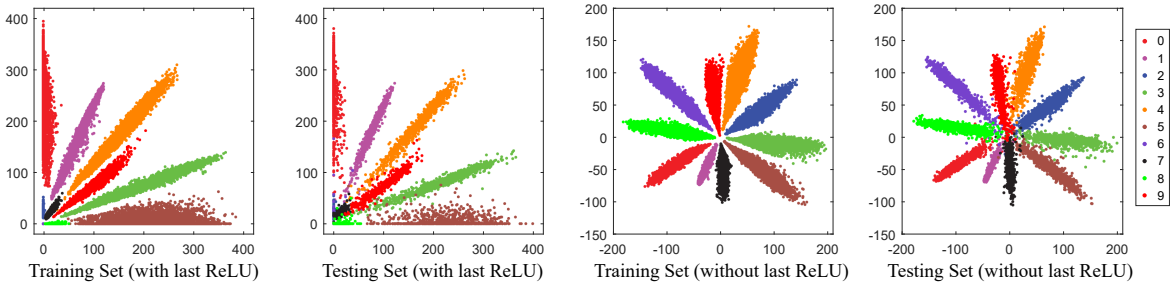


Figure 8: 2-D visualization before and after removing the last ReLU.

B. Normalizing the weights could reduce the prior caused by the training data imbalance

We have emphasized in the main paper that normalizing the weights can give better geometric interpretation. Besides this, we also justify why we want to normalize the weights from a different perspective. We find that normalizing the weights can implicitly reduce the prior brought by the training data imbalance issue (e.g., the long-tail distribution of the training data). In other words, we argue that normalizing the weights can partially address the training data imbalance problem.

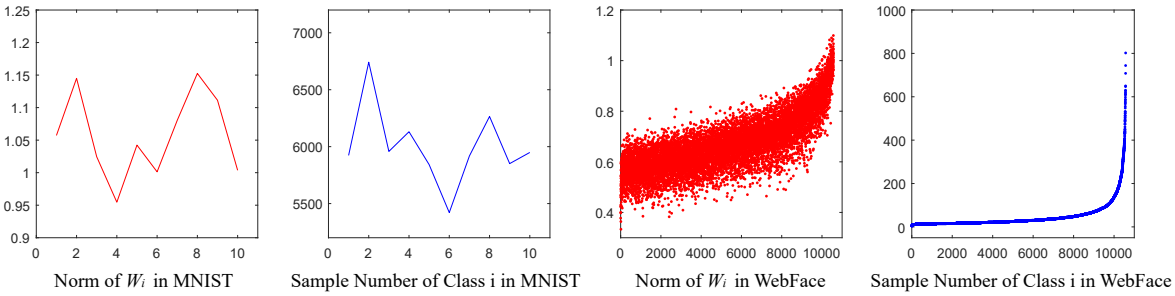


Figure 9: Norm of $\boldsymbol{W}_i$ and sample number of class i in MNIST dataset and CASIA-WebFace dataset.

We have an empirical study on the relation between the sample number of each class and the 2-norm of the weights corresponding to the same class (the i-th column of $\boldsymbol{W}$ is associated to the i-th class). By computing the norm of $\boldsymbol{W}_i$ and sample number of class i with respect to each class (see Fig. 9), we find that the larger sample number a class has, the larger the associated norm of weights tends to be. We argue that the norm of weights $\boldsymbol{W}_i$ with respect to class i is largely determined by its sample distribution and sample number. Therefore, norm of weights $\boldsymbol{W}_i, \forall i$ can be viewed as a learned prior hidden in training datasets. Eliminating such prior is often beneficial to face verification. This is because face verification requires to test on a dataset whose idenities can not appear in training datasets, so the prior from training dataset should not be transferred to the testing. This prior may even be harmful to face verification performance. To eliminate such prior, we normalize the norm of weights of FC2[6].

[6]FC2 refers to the fully connected layer in the softmax loss (or A-Softmax loss).

C. Empirical experiment of zeroing out the biases

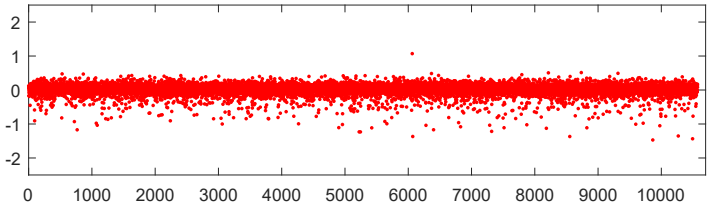


Figure 10: Biases of last fully connected layer learned in CASIA-WebFace dataset.

Standard CNNs usually preserve the bias term in the fully connected layers, but these bias terms make it difficult to analyze the proposed A-Softmax loss. This is because SphereFace aims to optimize the angle and produce the angular margin. With bias of FC2, the angular geometry interpretation becomes much more difficult to analyze. To facilitate the analysis, we zero out the bias of FC2 following [16]. By setting the bias of FC2 to zero, the A-Softmax loss has clear geometry interpretation and therefore becomes much easier to analyze. We show all the biases of FC2 from a CASIA-pretrained model in Fig. 10. One can observe that the most of the biases are near zero, indicating these biases are not necessarily useful for face verification.

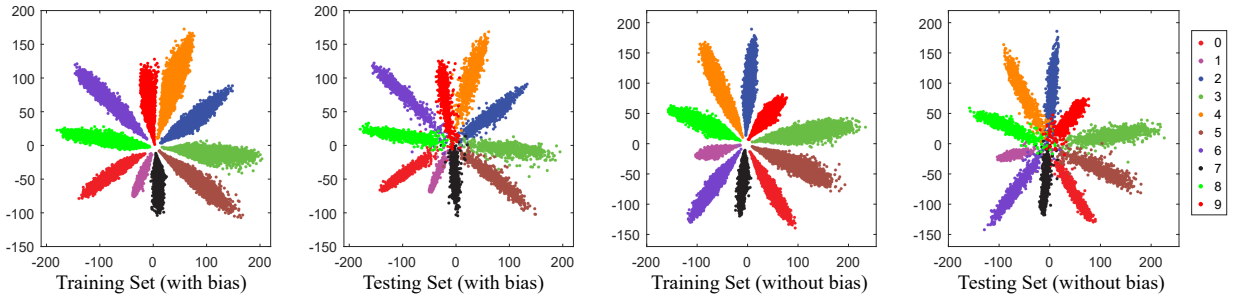


Figure 11: 2-D visualization with and without bias of last fully connected layer in MNIST.

Visualization on MNIST. We visualize the 2-D feature distribution in MNIST dataset with and without bias in Fig. 11. One can observe that zeroing out the bias has no direct influence on the feature distribution. The features learned with and without bias can both make full use of the learning space.

D. 2D visualization of A-Softmax loss on MNIST

We visualize the 2-D feature distribution on MNIST in Fig. 12. It is obvious that with larger m the learned features become much more discriminative due to the larger inter-class angular margin. Most importantly, the learned discriminative features also generalize really well in the testing set.

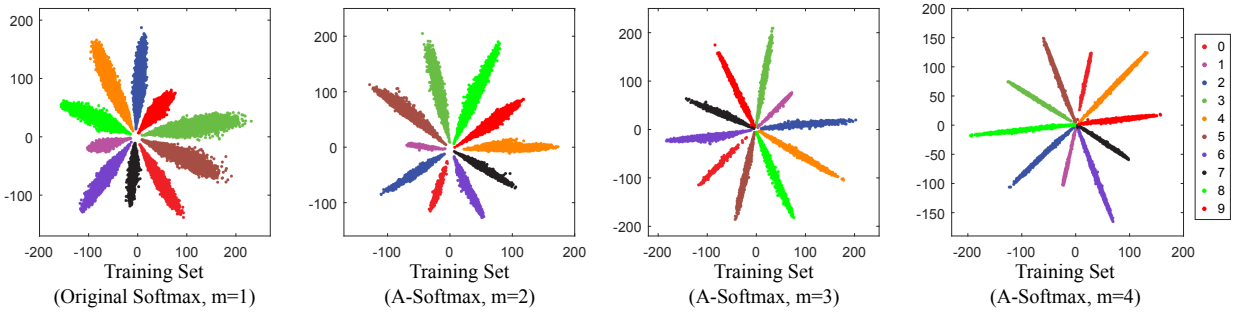


Figure 12: 2-D MNIST visualization of features learned by the softmax loss and the A-Softmax loss ($m = 2, 3, 4$).

E. Angular Fisher score for evaluating the feature discriminativeness and ablation study on our proposed modifications

We first propose an angular Fisher score for evaluating the feature discriminativeness in angular margin feature learning. The angular Fisher score (AFS) is defined by

$$AFS = \frac{S_w}{S_b} \tag{11}$$

where the within-class scatter value is defined as $S_w = \sum_i \sum_{x_j \in X_i} (1 - \cos\langle x_j, m_i \rangle)$ and the between-class scatter value is defined as $S_b = \sum_i n_i (1 - \cos\langle m_i, m \rangle)$. X_i is the i-th class samples, m_i is the mean vector of features from class i, m is the mean vector of the whole dataset, and n_i is the sample number of class i. In general, the lower the fisher value is, the more discriminative the features are.

Next, we perform a comprehensive ablation study on all the proposed modifications: removing last ReLU, removing Biases, normalizing weights and applying A-Softmax loss. The experiments are performed using the 4-layer CNN described in Table 2. The models are trained on CASIA dataset and tested on LFW dataset. The setting is exactly the same as the LFW experiment in the main paper. As shown in Table 6, we could observe that all our modification leads to peformance improvement and our A-Softmax could greatly increase the angular feature discriminativeness.

CNN	Remove Last ReLU	Remove Biases	Normalize Weights	A-Softmax	Accuracy	Angular Fisher Score
A	No	No	No	No	95.13	0.3477
B	Yes	No	No	No	96.37	0.2835
C	Yes	Yes	No	No	96.40	0.2815
D	Yes	Yes	Yes	No	96.63	0.2462
E	Yes	Yes	Yes	Yes (m=2)	97.67	0.2277
F	Yes	Yes	Yes	Yes (m=3)	97.82	0.1791
G	Yes	Yes	Yes	Yes (m=4)	**98.20**	**0.1709**

Table 6: Verification accuracy (%) on LFW dataset.

F. Experiments on MegaFace with different convolutional layers

We also perform the experiment on MegaFace dataset with CNN of different convolutional layers. The results in Table 7 show that the A-Softmax loss could make best use of the network capacity. With more convolutional layers, the A-Softmax loss (i.e., SphereFace) performs better. Most notably, SphereFace with only 4 convolutional layer could peform better than the softmax loss with 64 convolutional layers, which validates the superiority of our A-Softmax loss.

Method	protocol	Rank-1 Id. Acc. with 1M distractors	Ver. TAR for 10^{-6} FAR
Softmax Loss (64 conv layers)	Small	54.855	65.925
SphereFace (4 conv layers)	Small	57.529	68.547
SphereFace (10 conv layers)	Small	65.335	78.069
SphereFace (20 conv layers)	Small	69.623	83.159
SphereFace (36 conv layers)	Small	71.257	84.052
SphereFace (64 conv layers)	Small	**72.729**	**85.561**

Table 7: Performance (%) on MegaFace challenge with different convolutional layers. TAR and FAR denote True Accept Rate and False Accept Rate respectively. For all the SphereFace models, we use $m = 4$. With larger m and proper network optimization, the performance could potentially keep increasing.

G. The annealing optimization strategy for A-Softmax loss

The optimization of the A-Softmax loss is similar to the L-Softmax loss [16]. We use an annealing optimization strategy to train the network with A-Softmax loss. To be simple, the annealing strategy is essentially supervising the newtork from an easy task (i.e., large λ) gradually to a difficult task (i.e., small λ). Specifically, we let $f_{y_i} = \frac{\lambda \|\boldsymbol{x}_i\| \cos(\theta_{y_i}) + \|\boldsymbol{x}_i\| \psi(\theta_{y_i})}{1+\lambda}$ and start the stochastic gradient descent initially with a very large λ (it is equivalent to optimizing the original softmax). Then we gradually reduce λ during training. Ideally λ can be gradually reduced to zero, but in practice, a small value will usually suffice. In most of our face experiments, decaying λ to 5 has already lead to impressive results. Smaller λ could potentially yield a better performance but is also more difficult to train.

H. Details of the 3-patch ensemble strategy in MegaFace challenge

We adopt a common strategy to perform the 3-patch ensemble, as shown in Fig. 13. Although using more patches could keep increasing the performance, but considering the tradeoff between efficiency and accuracy, we use 3-patch simple concatenation ensemble (without the use of PCA). The 3 patches can be selected by cross-validation. The 3 patches we use in the paper are exactly the same as in Fig. 13.

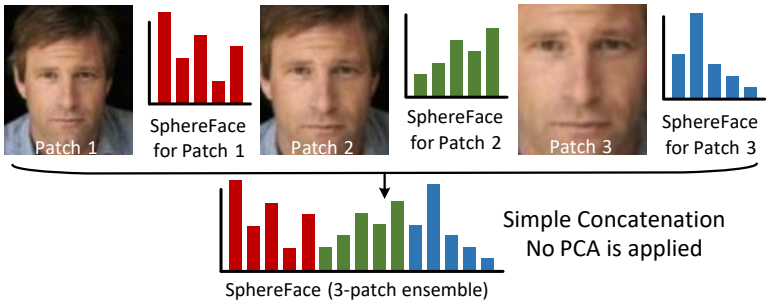


Figure 13: 3-Patch ensembles in SphereFace for MegaFace challenge.