Article

Double Additive Margin Softmax Loss for Face Recognition

Shengwei Zhou , Caikou Chen , Guojiang Han and Xielian Hou

College of Information Engineering, Yangzhou University, Yangzhou 225127, China; zhoushengwei326@163.com (S.Z.); hanguojiang0821@163.com (G.H.); xielhou@sina.com (X.H.)

* Correspondence: yzcck@126.com; Tel.: +86-139-5278-0010

Received: 30 October 2019; Accepted: 17 December 2019; Published: 19 December 2019

Abstract: Learning large-margin face features whose intra-class variance is small and inter-class diversity is one of important challenges in feature learning applying Deep Convolutional Neural Networks (DCNNs) for face recognition. Recently, an appealing line of research is to incorporate an angular margin in the original softmax loss functions for obtaining discriminative deep features during the training of DCNNs. In this paper we propose a novel loss function, termed as double additive margin Softmax loss (DAM-Softmax). The presented loss has a clearer geometrical explanation and can obtain highly discriminative features for face recognition. Extensive experimental evaluation of several recent state-of-the-art softmax loss functions are conducted on the relevant face recognition benchmarks, CASIA-Webface, LFW, CALFW, CPLFW, and CFP-FP. We show that the proposed loss function consistently outperforms the state-of-the-art.

Keywords: Softmax; angular margin; ResNet; face recognition

1. Introduction

Face recognition problems are ubiquitous in the computer vision domain. In the past few years, Deep Convolutional Neural Networks (DCNNs) have set the community of face recognition (FR) on fire [1]. Due to effectively layered end-to-end learning network frameworks and carefully deep feature extracting techniques from local to global, which are the most important ingredients for their success, DCNNs has immensely improved the state of the art in real-world face recognition scenarios. Numerous layered network architectures for face recognition tasks such as AlexNet [2], VGG [3], InceptionNet [4], ResNet [5], and DenseNet [6], have been proposed. Among them, the most representative one is AlexNet, originally proposed by Krizhevsky et al. AlexNet has become a pioneered architecture developed for image classification and was the winner of the ImageNet Large Scale Challenge in 2012.

It is well known that the effective feature representation for face images plays an important role in FR. Over the recent years, a hot research trend towards DCNNs has been devoted to learning with more discriminative deep features. Intuitively, the learnt deep features for FR are desired if the maximal within-class variance is less than the minimal between-class variance. However, learning deep features satisfying this condition is generally not easy owing to the inherently large intra-class variation and high inter-class similarity [7] in many FR applications. Despite the softmax function with the cross-entropy loss (called softmax loss) is popularly used in the training of a DCNN, recent studies [8,9] made it clear that the current softmax loss is insufficient to encourage the desired deep features meeting the above condition. To boost the discriminative ability of DCNNs and inspired by the previous idea, the center loss [10], pairwise loss [11] and triplet loss [12] were proposed. They unanimously proposed the enhancement of discrimination power of deep features by minimizing within-class variance and maximizing between-class variance in the Euclidean space of features. While these methods is

superior to the traditional softmax loss over classification performance, they usually suffer from some drawbacks. The center loss only explicitly enhanced the intra-class compactness while disregarding the inter-class separability. For the pairwise loss and triplet loss, They require the careful mining of pairs or triplets of samples, which is highly time-consuming.

Due to the fact that few existing softmax losses can effectively achieve the discriminative condition that the maximal within-class variance is less than the minimal between-class variance under the conventional Euclidean metric space, more recently, approaches have been proposed to address this problem by transforming the original Euclidean space of features to an corresponding angular space [10,13–15]. Specifically, both Large-Margin Softmax Loss [13] and A-Softmax Loss [14] are an angular softmax loss that enables DCNNs to learn angular deep features by imposing an angular margin constraint for larger inter-class variance. Compared to the Euclidean margin suggested by [2,16], the learnt angular features is more discriminative with the angular margin because the angular metric with cosine similarity is intrinsically more suitable to the softmax loss. During training of A-Softmax Loss, the original Softmax loss must be combined to ensure the convergence. To overcome the optimisation problem of A-Softmax Loss, the Additive Margin Softmax loss (AM-Softmax) [15] is proposed. The loss integrates a angular margin to the softmax loss in an additive manner. Its implementation and optimisation are much easier than A-Softmax Loss since A-Softmax Loss integrates the angular margin in a multiplicative way. AM-Softmax is easily reproducible and achieves state-of-the-art performance.

Motivated by AM-Softmax loss, This paper propose a new additive angular margin Loss, namely double additive margin softmax loss (DAM-Softmax). The idea behind the proposed loss is to impose an additive margin m to both the intra-class angular variation and inter-class angular variation simultaneously to enhance the intra-class compactness and inter-class discrepancy of the learned features. Compared to AM-Softmax loss, our loss has a stronger geometrical significance and will lead to obtain more discriminative features. Experimental results on some relevant face recognition benchmarks show that the proposed loss achieves better classification performance than the current state-of-the-art losses.

The rest of this paper is organized as follows. In Section 2, We will briefly introduce the related works such as the original softmax Loss, L-Softmax Loss, A-Softmax Loss and AM-Softmax Loss. Then we discuss the proposed loss, Double Additive Margin Softmax Loss in detail in Section 3. Finally, extensive experiments are presented in Section 4.

2. Preliminaries

In order to clearly understand the proposed DAM-Softmax loss, we will briefly review the classical softmax loss and AM-softmax loss. The classical softmax loss is formulated by

$$L_1 = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{\mathbf{w}_{y_i}^\top \mathbf{f}_i}}{\sum_{c=1}^C e^{\mathbf{w}_c^\top \mathbf{f}_i}} \quad (1)$$

where $\mathbf{w}_c$ ($c = 1, \dots, C$, C is the number of classes) denotes the weight vector of the last fully connected classifier layer, $\mathbf{f}_i$ is the learned deep feature input vector of the last fully connected classifier layer corresponding to the original input $\mathbf{x}_i$ with the label y_i , and N is the number of training samples in a minibatch. The inner product, $\mathbf{w}_c^\top \mathbf{f}_i$, between $\mathbf{w}_c$ and $\mathbf{f}_i$ can be also factorized into $\|\mathbf{w}_c\| \|\mathbf{f}_i\| \cos(\theta_c)$ where θ_c is the angle between $\mathbf{w}_c$ and $\mathbf{f}_i$, the loss can thus be rewritten as

$$L_2 = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{\|\mathbf{w}_{y_i}\| \|\mathbf{f}_i\| \cos(\theta_{y_i})}}{\sum_{c=1}^C e^{\|\mathbf{w}_c\| \|\mathbf{f}_i\| \cos(\theta_c)}} \quad (2)$$

The A-Softmax is a new loss function derived from the classical Softmax loss which proposed to impose a constraint to make $\|\mathbf{w}_c\| = 1$ and generalize the modified softmax loss to angular softmax (A-Softmax) loss by replacing $\|\mathbf{f}_i\| \cos(\theta_{y_i})$ with $\|\mathbf{f}_i\| \psi(\theta_{y_i})$,

$$L_3 = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{\|\mathbf{f}_i\| \psi(\theta_{y_i})}}{e^{\|\mathbf{f}_i\| \psi(\theta_{y_i})} + \sum_{c=1, c \neq y_i}^C e^{\|\mathbf{f}_c\| \cos(\theta_c)}} \quad (3)$$

where the authors proposed to define $\psi(\theta)$ as $\psi(\theta) = (-1)^k \cos(m\theta) - 2k$, $\theta \in [\frac{k\pi}{m}, \frac{(k+1)\pi}{m}]$ and $k \in [0, m-1]$ for removing the restriction which θ must be in the range of $[0, \frac{\pi}{m}]$.

In the AM-Softmax loss, the authors suggested to introduce an additive margin to its decision boundary by defining $\psi(\theta)$ as $\cos(\theta) - m$. In addition, both the deep feature vector $\mathbf{f}_i$ and weight vectors $\mathbf{w}_c$ are normalized during the implementation. Thus, The AM-Softmax loss is given by

$$\begin{aligned} L_4 &= -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cdot (\cos(\theta_{y_i}) - m)}}{e^{s \cdot (\cos(\theta_{y_i}) - m)} + \sum_{c=1, c \neq y_i}^C e^{s \cdot \cos(\theta_c)}} \\ &= -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cdot (\mathbf{w}_{y_i}^\top \mathbf{f}_i - m)}}{e^{s \cdot (\mathbf{w}_{y_i}^\top \mathbf{f}_i - m)} + \sum_{c=1, c \neq y_i}^C e^{s \cdot \mathbf{w}_c^\top \mathbf{f}_i}} \end{aligned} \quad (4)$$

where s is a hyper-parameter for scaling the cosine values.

In order to simultaneously enlarge the between-class angular margin and compress the within-class angular variation, It is clear to learn that both the A-Softmax loss and AM-Softmax loss share a common idea to generalize the original softmax loss to angular softmax loss by introducing an integer m to quantitatively control the decision boundary. specifically, In binary class case, a learned feature $\mathbf{f}$ from class 1 is given and θ_i is the angle between $\mathbf{f}$ and $\mathbf{w}_i$. A-softmax loss requires $\cos(m\theta_1) > \cos(\theta_2)$ to correctly classify $\mathbf{f}$. AM-softmax loss instead proposes $\cos(\theta_1) - m > \cos(\theta_2)$ to correctly classify $\mathbf{f}$. Both of them explicitly enforce the intra-class compactness to achieve more discriminative deep features by imposing an intra-class angular margin in the multiplicative manner and in the additive manner, respectively. Compared with the A-Softmax loss, AM-Softmax loss is simpler which is simpler and reaches better performance. In addition, It is much easier to implement because the computation of the gradient for back-propagation is no longer required.

3. Double Additive Margin Softmax Loss

One can obviously learn that AM-Softmax loss can obtain better performance by incorporating a single additive margin to its intra-class angular variation. Inspired by that, we propose to impose an additive margin to both the intra-class angular variation and inter-class angular distribution simultaneously to enhance the intra-class compactness and inter-class discrepancy. To give a formal formulation for the idea, we first define a function $g(\theta) = \cos(\theta)$. The Equation (4) can be rewritten as

$$L_5 = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cdot \psi(\theta_{y_i})}}{e^{s \cdot \psi(\theta_{y_i})} + \sum_{c=1, c \neq y_i}^C e^{s \cdot g(\theta_c)}} \quad (5)$$

where $\psi(\theta_{y_i}) = \cos(\theta_{y_i}) - m$ and $g(\theta_c) = \cos(\theta_c)$.

As analyzed above, we impose an additive margin m to both the intra-class and inter-class angular variation angular distribution simultaneously. Then we have the formulations:

$$\begin{aligned} \psi(\theta_{y_i}) &= \cos(\theta_{y_i}) - m \\ g(\theta_c) &= \cos(\theta_c) + m \end{aligned} \quad (6)$$

Compared to AM-Softmax loss, our formulation is also simple while explicitly encourages intra-class compactness and inter-class separability simultaneously, we thus term the loss as Double Additive Margin Softmax loss (DAM). Finally, the proposed loss function can be formulated by

$$\begin{aligned}
 L_6 &= -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cdot (\cos(\theta_{y_i}) - m)}}{e^{s \cdot (\cos(\theta_{y_i}) - m)} + \sum_{c=1, c \neq y_i}^C e^{s \cdot (\cos(\theta_c) + m)}} \\
 &= -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cdot (\mathbf{w}_{y_i}^T \mathbf{f}_i - m)}}{e^{s \cdot (\mathbf{w}_{y_i}^T \mathbf{f}_i - m)} + \sum_{c=1, c \neq y_i}^C e^{s \cdot (\mathbf{w}_c^T \mathbf{f}_i + m)}}
 \end{aligned} \tag{7}$$

3.1. Geometric Interpretation

Our double additive margin has a more explicit geometric interpretation on the hypersphere manifold. To simplify the geometric interpretation, we project the features onto two dimensional space and discuss the binary classification case on the hypersphere manifold where there are only $\mathbf{w}_1$ and $\mathbf{w}_2$, and $\|\mathbf{w}_1\| = \|\mathbf{w}_2\| = 1$. Thus, the classification performance depends totally on the angles θ_1 between $\mathbf{f}$ and $\mathbf{w}_1$, and θ_2 between $\mathbf{f}$ and $\mathbf{w}_2$.

Classification Stringency. In order to correctly classify the learned feature vector $\mathbf{f}$ as class 1, the traditional softmax loss requires $\theta_1 < \theta_2$ which easily has $\theta_1 - \theta_2 < 0$, the AM-Softmax loss requires $\cos(\theta_1) - m < \cos(\theta_2)$ which can be rewritten as $\cos(\theta_1) - \cos(\theta_2) < m$, while our DAM-Softmax loss needs $\cos(\theta_1) - m < \cos(\theta_2) + m$ which can be reformulated into $\cos(\theta_1) - \cos(\theta_2) < 2m$. It is obvious to see that the DAM-Softmax loss is more stringent than the original softmax loss and AM-Softmax loss for satisfying the corresponding classification criteria.

Classification Boundary. In Figure 1, we draw a schematic diagram to show the classification boundary of the classical softmax loss, AM-Softmax loss and the proposed DAM-Softmax loss. The classification boundary of the traditional softmax loss is denoted as the vector $\mathbf{p}_0$. In this case, we have $\mathbf{w}_1^T \mathbf{p}_0 = \mathbf{w}_2^T \mathbf{p}_0$ at the decision boundary $\mathbf{p}_0$ ($\mathbf{w}_1 \in \text{class1}, \mathbf{w}_2 \in \text{class2}$). For the AM-Softmax, the boundary becomes a marginal region instead of a single vector. At the new boundary $\mathbf{p}_1$ for class 1, one has $\mathbf{w}_1^T \mathbf{p}_1 - m = \mathbf{w}_2^T \mathbf{p}_1$, which gives $m = (\mathbf{w}_1^T - \mathbf{w}_2^T) \mathbf{p}_1 = \cos(\theta_{\mathbf{w}_1, \mathbf{p}_1}) - \cos(\theta_{\mathbf{w}_2, \mathbf{p}_1})$. If we further assume that all the classes have the same intra-class variance and the boundary for class 2 is at $\mathbf{p}_2$, we can get $\cos(\theta_{\mathbf{w}_2, \mathbf{p}_1}) = \cos(\theta_{\mathbf{w}_1, \mathbf{p}_2})$. Thus, $m = \cos(\theta_{\mathbf{w}_1, \mathbf{p}_1}) - \cos(\theta_{\mathbf{w}_1, \mathbf{p}_2})$, which is the difference of the cosine scores for class 1 between the two sides of the margin region. For our DAM-Softmax loss, the boundary becomes a wider marginal region than the one of AM-Softmax loss. At the new boundary $\mathbf{p}_3$ for class 1, one has $\mathbf{w}_3^T \mathbf{p}_3 - m = \mathbf{w}_4^T \mathbf{p}_3 + m$ ($\mathbf{w}_3 \in \text{class1}, \mathbf{w}_4 \in \text{class2}$), which gives $2m = (\mathbf{w}_3^T - \mathbf{w}_4^T) \mathbf{p}_3 = \cos(\theta_{\mathbf{w}_3, \mathbf{p}_3}) - \cos(\theta_{\mathbf{w}_4, \mathbf{p}_3})$. If we further assume that all the classes have the same intra-class variance and the boundary for class 2 is at $\mathbf{p}_4$, we can get $\cos(\theta_{\mathbf{w}_4, \mathbf{p}_3}) = \cos(\theta_{\mathbf{w}_3, \mathbf{p}_4})$. Thus, $2m = \cos(\theta_{\mathbf{w}_3, \mathbf{p}_3}) - \cos(\theta_{\mathbf{w}_3, \mathbf{p}_4})$, which is the difference of the cosine scores for class 1 between the two sides of the margin region. Obviously, the DAM-Softmax loss leads to a larger classification margin between class 1 and class 2.

Figure 1. Geometric difference.

3.2. Feature Distribution Visualization on MNIST Dataset

In order to better study and verify the effectiveness of the proposed DAM-Softmax loss function, we conducted an experiment on the MNIST dataset [17] to visualize the learned feature distributions. We chose the 7-layer CNN models with the original Softmax loss, AM-Softmax loss and DAM-Softmax loss for training and required to output two-dimensional deep features for visualization. After the 2-dimensional features were obtained, we then made the normalization to them and plotted them on a circle in the two dimensional space.

The visualization from Figure 2 can well demonstrate that our DAM-Softmax outperforms AM-Softmax [15] when the hyperparameters s and m is 30 and 0.4, respectively. Compared to AM-Softmax [15], the DAM-Softmax loss can lead to the larger inter-class margin and smaller intra-class variance property to the features without tuning too many hyper-parameters.

Figure 2. Features visualization in MNIST dataset: Softmax (left), AM-Softmax (middle) and DAM-Softmax (right). Specifically, we set the feature (output of the network) dimension to 2 and then draw the classification result as a scatter plot.

3.3. Algorithm

The proposed DAM-Softmax loss is extremely easy to implement in the popular deep learning frameworks, e.g., Pytorch [18] and Tensorflow [19]. The algorithm for DAM-Softmax loss is given as follow.

Algorithm 1: The steps of the DAM-Softmax Loss

Input: Feature Scale s , Margin Parameter m in Equation (7), Randomly initialized weights $\mathbf{w}$, Input images $\mathbf{f}$, Batch size N

1. Normalize the input image $\mathbf{f}$ ($\hat{\mathbf{f}} = \frac{\mathbf{f}}{\|\mathbf{f}\|}$), and make the new $\mathbf{f} = \hat{\mathbf{f}}$
2. Normalize the weight $\mathbf{w}$ ($\hat{\mathbf{w}} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$), and make the new $\mathbf{w} = \hat{\mathbf{w}}$
3. According to the Equation (7), introducing the variable substitutions (the new $\mathbf{f}_i = \hat{\mathbf{f}}_i$ and the new $\mathbf{w}_{y_i} = \hat{\mathbf{w}}_{y_i}$) and get the $\cos(\theta_{y_i}) = \mathbf{w}_{y_i}^\top \mathbf{f}_i$
4. According to the Equation (7), introducing the variable substitutions (the new $\mathbf{f}_c = \hat{\mathbf{f}}_c$ and the new $\mathbf{w}_c = \hat{\mathbf{w}}_c$) and get $\cos(\theta_c) = \mathbf{w}_c^\top \mathbf{f}_c$
5. Calculate " $\cos(\theta_{y_i}) - m$ ", and get " $s \cdot (\cos(\theta_{y_i}) - m)$ " in the Equation (6) and the Equation (7)
6. Calculate " $\cos(\theta_c) + m$ ", and get " $s \cdot (\cos(\theta_c) + m)$ " in the Equation (6) and the Equation (7)
7. Construct loss function: $L = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cdot (\mathbf{w}_{y_i}^\top \mathbf{f}_i - m)}}{e^{s \cdot (\mathbf{w}_{y_i}^\top \mathbf{f}_i - m)} + \sum_{j \neq y_i} e^{s \cdot (\mathbf{w}_c^\top \mathbf{f}_c + m)}}$

Output: Loss function L

4. Experiment

In this section, we first introduce the experiment settings. Then, we will discuss the effect of the hyperparameters. Finally, we will evaluate the performance of our loss function with several existing state-of-the-art loss functions on the benchmark datasets.

4.1. Implementation Settings

4.1.1. Datasets

Training Datasets. The CASIA-WebFace [7] dataset used for training consists of 49,4414 color face images from 10,575 classes.

Test Datasets. LFW dataset [17] contains 13,233 web-collected images from 5749 different identities, with large variations in pose, expression and illuminations. CFP dataset [18] consists of 500 subjects, each with 10 frontal and 4 profile images. The evaluation protocol includes frontal-frontal (FF) and frontal-profile (FP) face verification, each having 10 folders with 350 same-person pairs and 350 different-person pairs. In our experiments, we employ to test the performance the most challenging subsets CFP-FP which contains images of celebrities in frontal and profile views, CPLFW [20] and CALFW [21] which have higher pose and age variations with same identities from LFW dataset. Specific details of the three datasets above are shown in Table 1 and some example images from CFP-FP, CPLFW and CALFW are given in Figure 3–5, respectively.

Table 1. Face datasets for training and testing.

Datasets	Identity	Image
CPLFW [20]	5749	11625
CALFW [21]	5749	12174
CFP-FP [18]	500	7000

Figure 3. Samples of the same people in the CFP-FP.

Figure 4. Samples of different people in the CALFW.

Figure 5. Samples of different people in the CPLFW.

Data Preprocessing. We adopt the data preprocessing method used in [14,22] to detect faces and facial landmarks in images and align them. Then, we crop the aligned face images and resize them to 112×112 , and proceed to perform the normalization for the cropped face images by subtracting 128 and dividing 128.

Dataset Overlap Removal. To develop open-set evaluations, we use the overlap checking code provided by F. Wang [22] to get rid of the overlapped subjects between the training dataset of CASIA-WebFace and the testing dataset of LFW.

4.2. Network Architecture and Parameter Settings

For the fair comparison, the CNN architecture used in all experiments of this paper is the ResNet-face18 model specially designed for the training of face recognition, which is a modified ResNet [5]. The model has an improved residual block of BN-Conv-BN-PReLU-Conv-BN structure in which the kernel size and stride in the first convolutional layer is 3×3 and 1 instead of the original 7×7 and 2, and the stride in the second convolutional layer is set to be 2 instead of 1. In addition, PReLU [23] is used to replace the original ReLu. All implementations in the paper are conducted by Pytorch [18]. We set the batch size to be 256 and the weight decay parameter 5×10^{-4} . The initial learning rate is set as 10^{-1} . We set the learning decay rate to be 0.05 which means that the learning rate will be reduced by 5% when the loss value increases. In addition, the total epoch is set as 110. The SGD [9] is used in the optimization process of ResNet-face18.

4.3. Effect of Hyperparameter m

According the discussion in Section 3, our proposed DAM-Softmax loss has two hyperparameters which are the scale s and the margin m . More importantly, the two hyperparameters plays an key role for the performance of the proposed loss. Several recent works [15,24] have already discussed the scale s , we thus follow [15,24] to directly set it to 30 and will no longer discuss it in this paper. In this case, we can focus on the other hyperparameter, margin m . We train the ResNet-face18 model with the DAM-Softmax loss on CASIA-Webface dataset to conduct experiments to seek the best angular margin. For comparison, we train the same network model with the AM-Softmax loss on the same dataset. In Tables 2 and 3, we list the performance of AM-Softmax loss and the proposed DAM-Softmax loss. under the variation of m from 0.1 to 0.25. As illustrated in Tables 2 and 3, we can see that the recognition rate increases gradually from $m = 0.25$ to 0.3 and arrive at the saturated at $m = 0.4$, the performance then begins to drop from 0.4 to 0.5. For DAM-Softmax loss, the classification accuracy improves significantly from $m = 0.1$ to 0.17 and reaches the best at $m = 0.18$, from 0.18 to 0.25, the performance turns to decrease. Therefore, we fix the margins of AM-Softmax loss and DAM-Softmax loss as 0.4 and 0.18, respectively. The experiments for both AM-Softmax loss and DAM-Softmax loss can result in

excellent performance without observing any difficulty in convergence. The proposed loss get to the best verification accuracy on CASIA-Webface training dataset.

Table 2. Experimental results of different values of m with AM-Softmax.

Parameter m	0.25	0.3	0.35	0.4	0.45	0.5
Accuracy Rate	97.21%	97.34%	97.45%	97.68%	97.57%	97.59%

Table 3. Experimental results of different values of m (110 epoch) with DAM-Softmax.

Parameter m	0.1	0.13	0.15	0.17	0.18	0.2	0.22	0.25
Accuracy Rate	97.68%	97.74%	97.82%	97.94%	97.97%	97.89%	97.81%	97.65%

4.4. Comparison with State of the Art Loss Functions on LFW Dataset

In this part, we evaluate the performance of the proposed DAM-Softmax loss and the state-of-the-art loss functions. Following the previous experimental setting, we train a ResNet-Face18 model under the guidance of the original softmax, L-Softmax, A-Softmax, AM-Softmax and DAM-Softmax on the training dataset of CAISAWebFace. The experimental results on the test dataset of LFW are shown in Table 4.

Table 4. Some results of comparative testing experiment.

Model	Accuracy Rate
Softmax (resnet-face18, 110 epoch)	97.08%
L-Softmax (resnet-face18, 110 epoch) [10]	97.33%
A-Softmax (resnet-face18, 110 epoch) [14]	97.52%
AM-Softmax (resnet-face18, 110 epoch) [15]	97.68%
DAM-Softmax (resnet-face18, 47 epoch)	97.83%
DAM-Softmax (resnet-face18, 110 epoch)	97.97%

From Figure 6, It can be seen that the verification accuracy of DAM-Softmax loss is over 80% after one epoch while AM-Softmax loss requires 20 epoches to achieve the similar accuracy. In the 40th epoch, DAM-Softmax loss reaches the best performance which is still superior to the one of AM-Softmax loss. Figure 7 reports the training loss under the variation of epoch. When epoch = 75, the training loss of the original Softmax loss approaches stabilization with the value of about 13 by using softmax, the AM-softmax's training loss reach stabilization at around 10 when epoch is 55, while DAM-softmax get to stabilization at the 40th epoch and has a lower training loss. Therefore, This can demonstrate that the proposed loss has a faster convergence speed than AM-Softmax loss. As can be seen in Table 4, our proposed DAM-Softmax loss consistently arrives at competitive results compared to the other losses, which demonstrates the effectiveness of DAM-Softmax loss.

Figure 6. Accuracy rate of the test phase.

Figure 7. Training loss v.s Epoch.

4.5. Comparison with State of the Art Loss Functions on CFP-FP, CPLFW and CALFW Datasets

In order to further verify the effectiveness and robustness of DAM-Softmax, we compare the performance of the proposed losse with related baseline methods, e.g., the original softmax, L-Softmax, A-Softmax, AM-Softmax and DAM-Softmax on three datasets which have large-pose, large-age and different-angle. The experimental results are listed in the Table 5. The details of CFP-FP, CPLFW and CALFW datasets are listed in the Table 1.

Table 5. Verification results of different datasets based on different algorithms.

Method	CALFW	CPLFW	CFP-FP
Softmax	88.21%	77.54%	89.54%
AM-Softmax	89.72%	80.21%	92.12%
DAM-softmax	90.17%	82.08%	93.26%

As seen in Table 5, The proposed DAM-Softmax loss obtains the best performance. From Table 5, we can see that DAM-Softmax works much better on three datasets than AM-Softmax loss. Thus, we further demonstrate that our DAM-Softmax loss has stronger robustness.

5. Conclusions and Future Work

In this paper, we present a novel Double Additive Angular Margin Loss function for face recognition. specifically, we propose to simultaneously impose a angular margin to the intra-class and inter-class variation on the hypersphere manifold, which can effectively enhance the discriminative

power of learned deep features. Competitive performance on several popular face benchmarks verify the superiority and robustness of our approach.

Author Contributions: Conceiving the idea, S.Z., and C.C.; writing original draft, S.Z., and C.C.; writing final original manuscript, S.Z., and C.C.; supervision, C.C.; data curation, S.Z., and G.H.; All authors discussed and revised the results, and have read and approved the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (60875004).

Conflicts of Interest: The authors declare there is no conflicts of interest regarding the publication of this paper.

References

- Schroff, F.; Kalenichenko, D.; Philbin, J. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 815–823.
- Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Nice, France, 2012; pp. 1097–1105.
- Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
- Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
- He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
- Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
- Yi, D.; Lei, Z.; Liao, S.; Li, S.Z. Learning face representation from scratch. *arXiv* **2014**, arXiv:1411.7923.
- Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [[CrossRef](#)]
- Bottou, L. Large-scale machine learning with stochastic gradient descent. In Proceedings of the COMPSTAT'2010, Paris, France, 22–27 August 2010; pp. 177–186.
- Wen, Y.; Zhang, K.; Li, Z.; Qiao, Y. A discriminative feature learning approach for deep face recognition. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 499–515.
- Hadsell, R.; Chopra, S.; LeCun, Y. Dimensionality reduction by learning an invariant mapping. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). IEEE, Santa Barbara, CA, USA, 26–27 October 2006; Volume 2, pp. 1735–1742.
- Cheng, D.; Gong, Y.; Zhou, S.; Wang, J.; Zheng, N. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1335–1344.
- Liu, W.; Wen, Y.; Yu, Z.; Yang, M. Large-margin softmax loss for convolutional neural networks. *arXiv* **2016**, arXiv:1612.02295.
- Liu, W.; Wen, Y.; Yu, Z.; Li, M.; Raj, B.; Song, L. Sphereface: Deep hypersphere embedding for face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 212–220.
- Wang, F.; Cheng, J.; Liu, W.; Liu, H. Additive margin softmax for face verification. *IEEE Signal Process. Lett.* **2018**, *25*, 926–930. [[CrossRef](#)]
- Sun, Y.; Chen, Y.; Wang, X.; Tang, X. Deep learning face representation by joint identification-verification. *arXiv* **2014**, arXiv:1406.4773.
- Huang, G.B.; Learned-Miller, E. *Labeled Faces in the Wild: Updates and New Reporting Procedures*; Dept. Comput. Sci., Univ.: Amherst, MA, USA, 2014; pp. 14–003.

18. Sengupta, S.; Chen, J.C.; Castillo, C.; Patel, V.M.; Chellappa, R.; Jacobs, D.W. Frontal to profile face verification in the wild. In Proceedings of the 2016 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, Lake Placid, NY, USA, 7–9 March 2016; pp. 1–9.
19. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, D.; Greg S.; Davis, A.; Dean, J.; Devin, M. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv* **2016**, arXiv:1603.04467.
20. Kafai, M.; Eshghi, K.; Le, A.; Bhanu, B. A Reference-Based Framework for Pose Invariant Face Recognition. In Proceedings of the IEEE International Conference Workshops on Automatic Face and Gesture Recognition, Ljubljana, Slovenia, 4–8 May 2015.
21. Zheng, T.; Deng, W.; Hu, J. Cross-Age LFW: A Database for Studying Cross-Age Face Recognition in Unconstrained Environments. *arXiv* **2017**, arXiv:1708.08197 .
22. Deng, J.; Guo, J.; Xue, N.; Zafeiriou, S. Arcface: Additive angular margin loss for deep face recognition. *arXiv* **2018**, arXiv:1801.07698.
23. Xu, B.; Wang, N.; Chen, T.; Li, M. Empirical evaluation of rectified activations in convolutional network. *arXiv* **2015**, arXiv:1505.00853.
24. Wang, F.; Xiang, X.; Cheng, J.; Yuille, A.L. Normface: l₂ hypersphere embedding for face verification. In Proceedings of the 25th ACM International Conference on Multimedia, Mountain View, CA, USA, 23–27 October 2017; pp. 1041–1049.

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).