



## Article

# GicoFace: A Deep Face Recognition Model Based on Global-Information Loss Function <sup>†</sup>

Xin Wei , Wei Du, Xiaoping Hu, Jie Huang and Weidong Min \* 

School of Software, Nanchang University, 235 East Nanjing Road, Nanchang 330047, China; xinwei@ncu.edu.cn (X.W.); duwei@email.ncu.edu.cn (W.D.); xiaopinghu@email.ncu.edu.cn (X.H.); jiehuang@email.ncu.edu.cn (J.H.)

\* Correspondence: minweidong@ncu.edu.cn

<sup>†</sup> This paper is an extended version of our paper published in 2019 IEEE International Conference on Image Processing (ICIP) under the title “Gicoface: Global Information-Based Cosine Optimal Loss for Deep Face Recognition”.

**Abstract:** As CNNs have a strong capacity to learn discriminative facial features, CNNs have greatly promoted the development of face recognition, where the loss function plays a key role in this process. Nonetheless, most of the existing loss functions do not simultaneously apply weight normalization, apply feature normalization and follow the two goals of enhancing the discriminative capacity (optimizing intra-class/inter-class variance). In addition, they are updated by only considering the feedback information of each mini-batch, but ignore the information from the entire training set. This paper presents a new loss function called Gico loss. The deep model trained with Gico loss in this paper is then called GicoFace. Gico loss satisfies the four aforementioned key points, and is calculated with the global information extracted from the entire training set. The experiments are carried out on five benchmark datasets including LFW, SLLFW, YTF, MegaFace and FaceScrub. Experimental results confirm the efficacy of the proposed method and show the state-of-the-art performance of the method.



**Citation:** Wei, X.; Du, W.; Hu, X.; Huang, J.; Min, W. GicoFace: A Deep Face Recognition Model Based on Global-Information Loss Function. *Electronics* **2021**, *10*, 2387. <https://doi.org/10.3390/electronics10192387>

Academic Editor: Donghyeon Cho

Received: 30 August 2021

Accepted: 18 September 2021

Published: 29 September 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 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 (<https://creativecommons.org/licenses/by/4.0/>).

**Keywords:** face recognition; loss function; convolutional neural networks; intra-class variance; inter-class variance; discriminative capacity; deep learning

## 1. Introduction

CNNs have greatly promoted the development of face recognition, where the loss function plays a key role in training the CNNs. Among a large number of loss functions, cross entropy loss is the most widely used one in deep learning-based classification, but it is not the best choice in face recognition as it only aims at learning separable features instead of discriminative features [1]. Most of the face recognition tasks are open-set tasks that require the features to have strong discriminative capacity. To enhance the discriminative capacity of the learned features, two targets ought to be thought of: (1) minimizing intra-class variance, and (2) maximizing inter-class variance. Over the past decade, many different loss functions [1–12] have been proposed for learning highly discriminative features for face recognition. These loss functions can be broadly grouped into two categories—the Euclidean distance-based loss functions [1–5] and the cosine similarity-based loss functions [6–12], where the vast majority of these loss functions are derived from cross entropy loss by modifying cross entropy loss with additional constraints or adding a penalty to it. However, only a few of them explicitly follow the aforementioned two targets.

Typical Euclidean distance-based losses include Center loss [1], Marginal loss [2] and Range loss [3]. All of them add another penalty to implement the joint supervision with cross entropy loss. Specifically, Center loss adds a penalty to softmax via computing and limiting the distances between the within-class samples and the corresponding class

center, but it does not significantly optimize the inter-class margin. Marginal loss specifies a threshold value and considers all possible combinations of the sample pairs in a mini-batch, forcing the sample pairs from the different classes to have a margin larger than the threshold and the sample pairs from the same classes to have a margin smaller than the threshold. However, it is not reasonable to use only one threshold to limit the intra-class and inter-class distance simultaneously. Range loss calculates the distances between the samples within each class, and chooses two sample pairs that have the largest distances as the intra-class constraint; at the same time, Range loss calculates the distance between the class centers, and forces the class center pair that has the smallest distance to have a larger margin than the designated threshold. This method can effectively optimize the positions of the hard samples in the feature space, but ignores the optimization of other samples, so it is unable to learn the optimal feature space. From the relevant experimental results of the methods above [1–5], the performance of face recognition benefits from both two targets of improving discriminative capacity can be found.

Typical cosine similarity-based losses include L-Softmax loss [8], A-Softmax loss [9] and AM-Softmax loss [10]. L-Softmax transforms the measurement from Euclidean distance to cosine similarity by reformulating the output of the softmax layer from  $W \cdot f$  to  $|W| \cdot |f| \cdot \cos \theta$ . In addition, L-Softmax enlarges the angular margins between different identities by adding multiplicative angular constraints to  $\cos \theta$ . Nevertheless, L-Softmax does not apply L2 weight and feature normalization. Therefore, the difference between samples is determined by the angle and size of the feature vectors, which is inconsistent with the effort to optimize the feature space only by angle. Based on L-Softmax loss, A-Softmax applies L2 weight normalization, so  $W \cdot f$  can be further reformulated to  $|f| \cdot \cos \theta$ , which simplifies the training target. With L2 weight normalization, A-Softmax helps CNNs to learn features with geometrically interpretable angular margin. The experiments in [9] show that the performance can be enhanced by L2 weight normalization, although the improvement is very limited. However, A-Softmax still keeps the multiplicative angular constraints, the multiplicative angular constraints are difficult to control and it is difficult to explain their geometrical meaning.

AM-Softmax uses the additive angular constraints instead of the multiplicative angular constraints, that is, it replaces  $\cos(m\theta)$  with  $\cos \theta - m$ . AM-Softmax also applies feature normalization and makes  $|W| \cdot |f| = s$ , where  $s = 30$  is introduced as the global scaling factor. Hence, the training target  $|W| \cdot |f| \cdot \cos \theta$  is again simplified to  $s \cdot \cos \theta$ . In addition, feature normalization brings benefits such as higher recognition accuracy, better mathematical interpretation and better geometrical interpretation. These benefits are disclosed in [13–16].

The properties of the best-performing and the most recent losses are summarized in Table 1, from which we can see that loss functions such as Center loss, Range loss, Contrastive loss, Marginal loss and Triplet loss do not apply weight and feature normalization, and loss functions such as A-Softmax loss, AM-Softmax loss, L-Softmax loss and ArcFace do not explicitly follow the two targets of improving discriminative capacity. However, according to the previous description, it can be seen that these four properties contribute to the improvement of recognition performance to varying degrees. This paper presents a new loss function, which is called Global Information-based Cosine Optimal loss (i.e., Gico loss), and the deep model trained with Gico loss is named GicoFace accordingly. An overview of the proposed training framework is shown in Figure 1. Table 1 shows the properties of Gico loss, where it can be seen that Gico loss satisfies all four aforementioned properties. To break through the hardware constraints and make Gico loss possible, Gico loss is calculated with the global distribution information from the entire training set, which is different from all other loss functions. The main contribution of this paper lies in the following aspects:

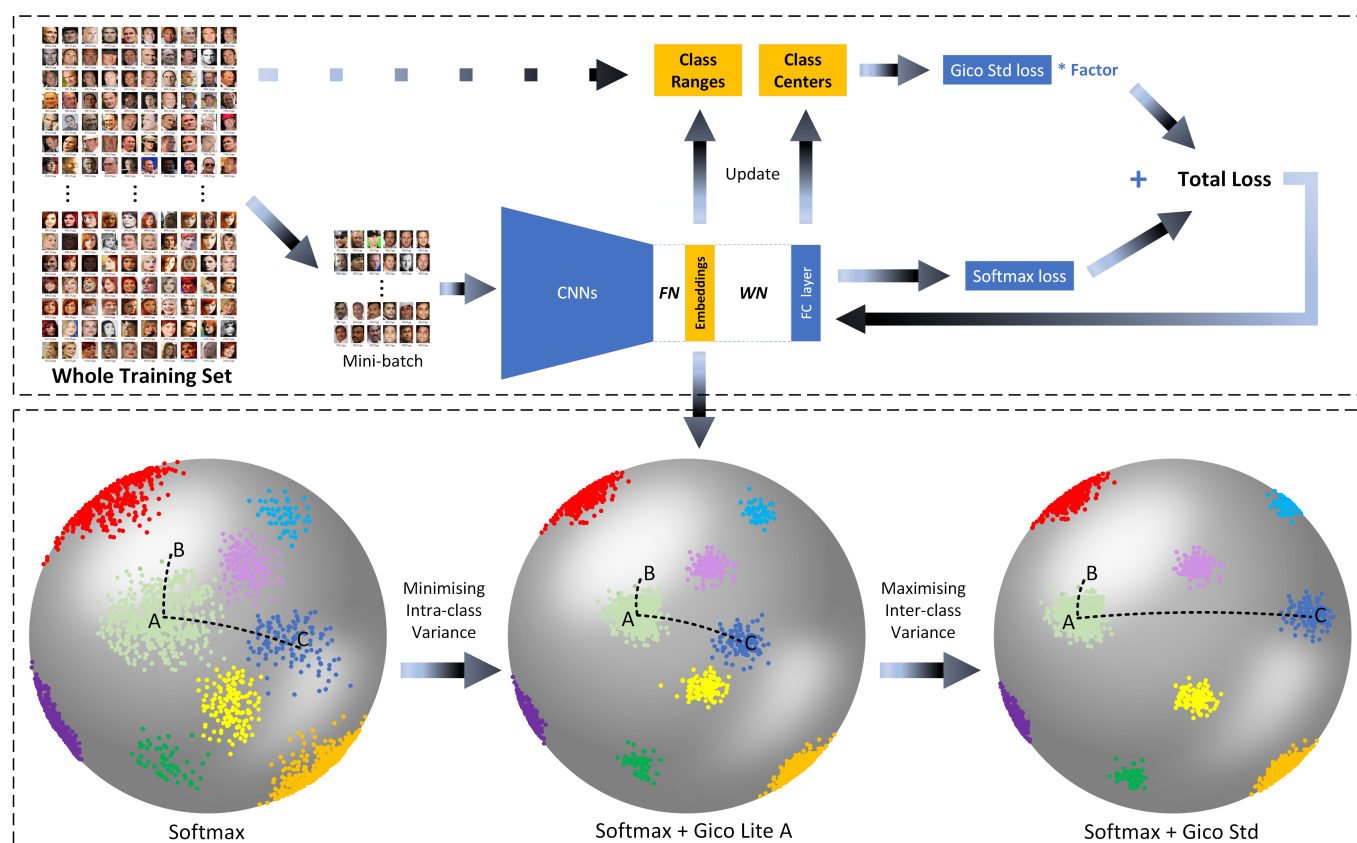
1. We propose a novel loss function to enhance the discriminative capacity of the deep features. To the best of our knowledge, it is the first loss that simultaneously satisfies all the first four properties in Table 1 and also the first attempt to use global information as the feedback information;

2. We propose and implement three different versions of Gico loss and analyze their performance variation on multiple datasets;
3. To break through the hardware constraints and make Gico loss possible, we propose an algorithm to learn the cosine similarity between the class center and the class edge;
4. We conduct extensive experiments on multiple public benchmark datasets including LFW [17], SLLFW [18], YTF [19], MegaFace [20] and FaceScrub [21] datasets. Experimental results presented in Section 3 confirm the efficacy of the proposed method and show the state-of-the-art performance of the method.

**Table 1.** The properties of different losses in deep face recognition.

	Optimize Intra-Class Variance	Optimize Inter-Class Variance	WN	FN	Feedback Source
Contrastive loss [5]	Yes	Yes	No	No	mini-batch
Triplet loss [4]	Yes	Yes	No	No	mini-batch
Center loss [1]	Yes	No	No	No	mini-batch
Marginal loss [2]	Yes	Yes	No	No	mini-batch
Range loss [3]	Yes	Yes	No	No	mini-batch
Fair loss [7]	No	Yes	Yes	Yes	mini-batch
SFace loss [12]	Yes	Yes	Yes	Yes	mini-batch
CVM loss [11]	Yes	Yes	No	No	mini-batch
L-Softmax loss [8]	No	Yes	No	No	mini-batch
A-Softmax loss [9]	No	Yes	Yes	No	mini-batch
AM-Softmax loss [10]	No	Yes	Yes	Yes	mini-batch
ArcFace [6]	No	Yes	Yes	Yes	mini-batch
<b>Gico loss</b>	Yes	Yes	Yes	Yes	<b>global info</b>

Note: WN: weight normalization. FN: feature normalization.



**Figure 1.** An overview of the proposed training framework. FN and WN represent *feature normalization* and *weight normalization*, respectively. FC layer is the abbreviation of fully connected layer. A and C are the class centers of the corresponding classes. AB represents the class range and AC represents the distance between two class centers.

Please note that an earlier version of this paper [22] was presented at the International Conference on Image Processing. Compared with the earlier version, this journal paper adds about 50% new content: (1) experiments on MegaFace and FaceScrub datasets to further verify the effectiveness of the proposed methods; (2) more detailed description on related works; (3) more discussion on the proposed methods to answer some key scientific questions; (4) more details about the complete algorithm are given.

## 2. From Cross Entropy Loss to Gico Loss

To better understand the proposed loss, firstly we give a brief review of related works including cross entropy loss, Center loss and some variants of cross entropy loss based on cosine similarity. Then we focus on the proposed Gico loss and give a detailed analysis.

### 2.1. Cross Entropy Loss and Center Loss

Cross entropy loss is the most commonly used loss function in deep learning, which can be formulated as:

$$\mathcal{L}_S = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{W_{y_i}^T f_i + b_{y_i}}}{\sum_{j=1}^P e^{W_j^T f_i + b_j}}, \quad (1)$$

where  $W_j \in R^d$  is the  $j$ th column of the weight matrix  $W$  in the final fully connected layer,  $f_i \in R^d$  is the feature vector of the  $i$ th sample belonging to the  $y_i$ th class,  $b_j$  is the bias term of the  $j$ th class,  $P$  is the number of classes in the entire training set and  $N$  denotes batch size. A summary of notation declarations of this paper is shown in Table 2. From Equation (1), it can be seen that cross entropy loss is essentially calculating the cross-entropy between the predicted label and the true label, indicating that cross entropy loss focuses only on optimizing the correctness of the classification results on the training set. In other words, cross entropy loss aims at separating the training samples of different classes instead of learning highly discriminative features and enlarging the margin between those overlapped or non-overlapped neighbor classes. Cross entropy loss is appropriate for closed-set tasks, where all the testing classes are predefined in the training set, as with most cases in object recognition and behavior recognition. Nevertheless, in face recognition, it is almost impossible to collect all the faces that may appear in the test stage, so most real applications of face recognition are open-set tasks. Open-set tasks require the learned features to have strong discriminative capacity so as to classify the unseen sample correctly. To improve the discriminative capacity of the features, Center loss is proposed by Wen et al. [1]. Center loss can minimize the intra-class distance, which is formulated as follows:

$$\mathcal{L}_C = \frac{1}{2} \sum_{i=1}^N \|f_i - c_{y_i}\|_2^2, \quad (2)$$

where  $c_{y_i}$  denotes the class center of the  $y_i$ th class. Center loss is the sum of all the distances between each sample and its class center. Center loss is used in conjunction with cross entropy loss:

$$\mathcal{L} = \mathcal{L}_S + \lambda \mathcal{L}_C, \quad (3)$$

where  $\lambda$  is a hyper-parameter for adjusting the impact of these two losses. Center loss optimizes only the intra-class variance and it does not apply weight and feature normalization.

### 2.2. Variants of Cross Entropy Loss Based on Cosine Similarity

L-Softmax loss, A-Softmax loss, AM-Softmax loss and ArcFace loss are variants of cross entropy loss based on cosine similarity. They have all been proposed in the past three years. All of them are derived from the original cross entropy loss in Equation (1), replacing the distance measurement from Euclidean distance to cosine similarity. In the cosine space, the similarity between two vectors is only up to the angle between them if feature normalization and weight normalization are applied. This makes the training process more focused on distinguishing different types of samples by optimizing the angle

between the vectors, without having to consider the complex multi-dimensional spatial structure in the Euclidean space. The aforementioned variants transform the FC layer formulation from  $W_{y_i}^T f_i + b_{y_i}$  to  $\|W_{y_i}\| \|f_i\| \cos \theta_{y_i}$  by setting the bias  $b_{y_i}$  to 0, where  $\theta_{y_i}$  is the angle between  $W_{y_i}$  and  $f_i$ . However, they have different choices for weight and feature normalization, and use different ways to add marginal constraints.

**Table 2.** Notation Declaration.

Notations	Interpretations
$d$	the number of dimensionality
$W$	the weight matrix in the final fully connected layer
$W_j$	the $j$ th column of $W$
$y_i$	the label of $i$ th sample
$f$	feature vector
$f_i$	the feature vector of the $i$ th sample belonging to $y_i$ th class
$b_j$	the bias term of the $j$ th class
$P$	the number of classes in the entire training set
$N$	batch size
$c_{y_i}$	the class center of the $y_i$ th class
$\lambda$	a hyper-parameter in the center loss
$W_{y_i}$	the weight matrix of the $y_i$ th class
$\theta_{y_i}$	the angle between $W_{y_i}$ and $f_i$
$m$	inter-class constraint
$c_j$	the center of class $j$
$e_j$	the farthest sample of class $j$ from the class center
$R(j)$	the cosine range of class $j$
$\beta$	the shrink rate for adjusting the shrink speed of the learned class range
$A$	set $A$
$\sum_{Top}(A, K)$	the sum of the $K$ largest elements in $A$

Equations (4) and (5) show the formulation of the L-Softmax loss and the A-Softmax loss, respectively:

$$\mathcal{L}_L = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{\|W_{y_i}\| \|f_i\| \psi(\theta_{y_i})}}{e^{\|W_{y_i}\| \|f_i\| \psi(\theta_{y_i})} + \sum_{j=1, j \neq y_i}^P e^{\|W_j\| \|f_i\| \psi(\theta_j)}} \quad (4)$$

$$\mathcal{L}_A = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{\|f_i\| \psi(\theta_{y_i})}}{e^{\|f_i\| \psi(\theta_{y_i})} + \sum_{j=1, j \neq y_i}^P e^{\|f_i\| \psi(\theta_j)}}, \quad (5)$$

where  $\psi(\theta_{y_i}) = (-1)^k \cos(m\theta_{y_i}) - 2k, \theta_{y_i} \in (\frac{k\pi}{m}, \frac{(k+1)\pi}{m})$ ,  $k \in (0, m-1)$ ,  $m \geq 1$  is the angular margin. With greater  $m$ , the between-class margin becomes larger and the learning objective also becomes harder. In L-Softmax loss and A-Softmax loss,  $m$  is used as a multiplier on the angle, so we say that L-Softmax loss and A-Softmax loss apply the multiplicative angular margin. Different from L-Softmax loss, weight normalization is introduced in A-Softmax loss, which sets  $\|W_{y_i}\| = 1$  by L2 normalization, which makes all class centers to lie on the hypersphere.

On the basis of L-Softmax loss and A-Softmax loss, AM-Softmax loss further adopts feature normalization and uses the additive cosine margin to replace the multiplicative angular margin. Feature normalization makes the samples of all classes to lie on the hypersphere, while the additive cosine margin forces the different classes to be separated from the cosine similarity level. AM-Softmax loss is formulated as follows:

$$\mathcal{L}_{AM} = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s(\cos(\theta_{y_i})-m)}}{e^{s(\cos(\theta_{y_i})-m)} + \sum_{j=1, j \neq y_i}^P e^{s(\cos(\theta_j))}}, \quad (6)$$



where  $\|f_i\|$  is fixed by L2 normalization and is re-scaled to  $s$ . So  $\|f_i\|$  is replaced with  $s$  in Equation (6). After AM-Softmax loss, ArcFace loss again replaces  $\cos(\theta_{y_i}) - m$  with  $\cos(\theta_{y_i} + m)$ , which enables  $m$  to clearly represent the meaning of angle geometrically. Therefore Arcface loss is computed as follows:

$$\mathcal{L}_{arc} = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s(\cos(\theta_{y_i} + m))}}{e^{s(\cos(\theta_{y_i} + m))} + \sum_{j=1, j \neq y_i}^P e^{s \cos(\theta_j)}}. \quad (7)$$

### 2.3. The Proposed Gico Loss

After reviewing the recent loss functions used in deep face recognition, we present a new loss function, namely Gico loss (Global Information-based Cosine Optimal loss). Gico loss utilizes the global information from the entire training set and integrates the advantages of the existing losses. Firstly, L2 weight normalization is applied by fixing  $b_j = 0$  and  $\|W_j\| = 1$ . Secondly, we apply L2 normalization on the feature vector  $f_i$  and re-scale  $\|f_i\|$  to  $s$ . Similar to Center loss, Gico loss is also used in conjunction with another loss function. Here, the cross entropy loss is adopted like the Center loss, we choose AM-Softmax loss, as AM-Softmax loss shows slightly better performance than cross entropy loss. The total loss is formulated as follows:

$$\mathcal{L} = \mathcal{L}_{AM} + \lambda \mathcal{L}_G. \quad (8)$$

In designing the Gico loss, two sub-tasks are considered: minimizing the intra-class variance and maximizing the inter-class variance. To cope with these two sub-tasks, two “lite” versions of Gico loss are designed, respectively. Finally, we construct a standard version of Gico loss, which is the combination of these two lite versions. To minimize the intra-class variance, we propose a “lite” version of Gico loss (Gico Lite A), which is formulated as below:

$$\mathcal{L}_{GA} = \frac{P}{\sum_{j=1}^P \frac{R(j)+1}{2}} \quad (9)$$

$$R(j) = \cos(c_j, e_j),$$

where  $c_j$  is the center of class  $j$ ,  $e_j$  represents the farthest sample of class  $j$  from the class center,  $R(j)$  represents the cosine range of class  $j$ , namely the cosine similarity between the class center and the edge of class  $j$ , and  $P$  is the number of the classes in the entire training set. During the training, the deep features change after each mini-batch, which also leads to the change of  $c_j$  and  $e_j$ . To make  $c_j$  and  $e_j$  as accurate as possible, ideally,  $c_j$  and  $e_j$  should be calculated by traversing the entire training set and updated after each mini-batch. Nevertheless, this is totally unfeasible in terms of the power of the existing hardware. The reason lies in two constraints: the computing power and the memory size of GPU, TPU or other similar processing units. If the computing power constraint can be ignored, the deep neural network could take the entire training set as the source of feedback information; if the memory size constraint can be ignored, the deep neural network would input the entire training set into the memory and get rid of the size limitation of a mini-batch. Perhaps just because of the above two constraints, there is no loss that uses the entire dataset as the source of feedback information to optimize the CNNs in face recognition.

In this paper, the first constraint is broken through by two approximation solutions. From Equation (6), it can be seen that the key optimisation object of the AM-Softmax loss is to minimize  $\theta_{y_i}$  and maximize  $\theta_j$ , where  $\theta_{y_i}$  represents the angle between  $f_i$  and  $W_{y_i}$ .  $\theta_j$  represents the angle between  $f_i$  and  $W_j$ , where  $j \neq y_i$ . In other words, AM-Softmax loss is aimed at decreasing the distances between  $W_j$  and the sample features in the  $j$ th class ( $j = 1, 2, \dots, P$ ). As the training goes on,  $W_j$  is updated automatically to the center of class  $j$  ( $j = 1, 2, \dots, P$ ), as this leads to the minimum distance sum between  $W_j$  and the sample features in the  $j$ th class. Therefore, we can simply use  $W_j$  as the substitution of  $c_j$  without any extra computing power. For  $e_j$  and  $R(j)$ , we propose a learning algorithm to

recursively update the range of each class. In the beginning,  $R(j)$  is set to 1 initially, then we update  $R(j)$  using the following iterations:

$$R(j)^{t+1} = R(j)^t + \sum_{i=1}^N \phi(y_i, j) \cdot \Delta R_i, j = 1, 2, \dots, P. \quad (10)$$

$$\Delta R_i = \begin{cases} \cos(W_{y_i}, f_i) - R(y_i)^t, & R(y_i)^t > \cos(W_{y_i}, f_i) \\ \beta \cdot (\cos(W_{y_i}, f_i) - R(y_i)^t), & R(y_i)^t \leq \cos(W_{y_i}, f_i), \end{cases} \quad (11)$$

where  $\beta$  is the shrink rate for adjusting the shrink speed of the learned class range.  $\phi(y_i, j) = 0$  when  $y_i \neq j$ , otherwise  $\phi(y_i, j) = 1$ . The learning algorithm takes two cases into consideration and performs two operations accordingly: (a) Replace the class range directly with the cosine similarity between the input sample and its corresponding class center, if their cosine similarity is smaller than the recorded class range; (b) Let the class range shrink by scaling their cosine similarities with  $\beta$ , if the cosine similarity between the input sample and its corresponding class center is larger than the recorded class range. Operation (a) keeps the learned class range up to date. Nevertheless, as the training goes on, the real class range will become smaller and smaller, so operation (b) is performed to help the learned class range shrink to its real value.

To maximize the inter-class variance, we also propose another "lite" version of Gico loss (Gico Lite B):

$$\mathcal{L}_{G_B} = \frac{\sum_{Top}(A, K)}{K} \quad (12)$$

$$A = \left\{ \frac{\cos(W_a, W_b) + 1}{2} : a, b = 1, 2, 3, \dots, P; a > b \right\},$$

where  $A$  is a set and  $\sum_{Top}(A, K)$  denotes the sum of the  $K$  largest elements in  $A$ . Gico Lite B is aimed at finding  $K$  pairs of nearest class centers in the entire training set and then calculates the sum of their distances. Compared with the non-adjacent class centers, the corresponding classes of the adjacent centers have a high probability to have small margins or overlaps. If all adjacent classes have proper margins, the non-adjacent classes would have larger margins. Therefore, taking all center pairs into account is unnecessary. The most effective way is optimizing the distances of all the adjacent centers, but it is time-consuming to calculate the number of the adjacent center pairs that exist on the hypersphere. Here, a conservative strategy is adopted, namely set the value of  $K$  to  $P$  where  $P$  is the number of classes. As the minimum number of adjacent center pairs is  $P$  which takes place when all the class centers line up in a circle on the hypersphere.

For best performance, we propose the standard version of Gico loss (Gico Std) in the end, which integrates the above two lite versions:

$$\mathcal{L}_{G_{std}} = L_{G_A} \cdot L_{G_B} = \frac{P \cdot \sum_{Top}(A, K)}{K \cdot \sum_{j=1}^P \frac{R(j)+1}{2}}. \quad (13)$$

Algorithm 1 shows the basic learning steps in the CNNs with the finally proposed Gico Std.

---

**Algorithm 1** Learning algorithm in the CNNs with the proposed Gico Std.
 

---

**Input:** Training samples  $\{f_i\}$ , initialized parameters  $\theta_C$  in convolution layers, parameters  $W$  in the final fully connected layer, learning rate  $\mu^t$ , initialized class ranges  $\{R(j) = 1 | j = 1, 2, \dots, P\}$ , class hyperparameters  $\lambda$ , hyperparameters  $\beta$ , and the number of iteration  $t \leftarrow 1$ .

**Output:** The parameters  $\theta_C$  and the total loss  $\mathcal{L}$ .

- 1: **while**  $t < \text{maximum iteration number}$  **do**
  - 2:   Calculate  $\mathcal{L}_{G_A}$  by  $\mathcal{L}_{G_A} = \frac{P}{\sum_{j=1}^P \frac{R(j)+1}{2}}$ , where  $R(j) = \cos(c_j, e_j)$ .
  - 3:   Calculate  $\mathcal{L}_{G_B}$  by  $\mathcal{L}_{G_B} = \frac{\sum_{Top(A,P)} \frac{\cos(W_a, W_b)+1}{2}}{P}$ , where  $A = \{\frac{\cos(W_a, W_b)+1}{2} : a, b = 1, 2, 3, \dots, P; a > b\}$ .
  - 4:   Calculate  $\mathcal{L}_{G_{std}}$  by  $\mathcal{L}_{G_{std}} = \mathcal{L}_{G_A} * \mathcal{L}_{G_B}$ .
  - 5:   Calculate the total loss by  $\mathcal{L} = \mathcal{L}_{AM} + \lambda \mathcal{L}_{G_{std}}$ .
  - 6:   Calculate the backpropagation error  $\frac{\partial \mathcal{L}^t}{\partial f_i^t}$  for each sample  $i$  by  $\frac{\partial \mathcal{L}^t}{\partial f_i^t} = \frac{\partial \mathcal{L}_{AM}^t}{\partial f_i^t} + \lambda \frac{\partial \mathcal{L}_{G_{std}}^t}{\partial f_i^t}$ .
  - 7:   Update  $W$  by  $W^{t+1} = W^t - \mu^t \frac{\partial \mathcal{L}^t}{\partial W^t} = W^t - \mu^t \frac{\partial \mathcal{L}_{AM}^t}{\partial W^t}$ .
  - 8:   Update  $\theta_C$  by  $\theta_C^{t+1} = \theta_C^t - \mu^t \sum_i^N \frac{\partial \mathcal{L}^t}{\partial f_i^t} \frac{\partial f_i^t}{\partial \theta_C^t}$ .
  - 9:   Update  $R(j)^{t+1} = R(j)^t + \sum_{i=1}^N \phi(y_i, j) \cdot \Delta R_i, j = 1, 2, \dots, P$ , where  $\Delta R_i$  is calculated by Equation (11).
  - 10:    $t \leftarrow t + 1$ .
  - 11: **end while**
- 

#### 2.4. Discussion

1. **Why combine  $\mathcal{L}_{G_A}$  and  $\mathcal{L}_{G_B}$  using multiplication instead of simple addition? Does it cause instability?**  
The idea of multiplication is inspired by LDA (Linear Discriminative Analysis). Using multiplication, only one parameter  $\lambda$  is needed for adjusting the impact of Gico Std. Using addition, two parameters are needed for the two parts of Gico Std respectively. Roughly speaking, Gico Std is the quotient of the average inter-class distance and the average intra-class distance as shown in Equation (13). Both denominator and nominator have limits, and they are mutually constrained; thus, their quotient does not lead to instability. We checked the loss curves, and confirm that the cases of instability did not happen.
2. **The improvements on recognition accuracy are somewhat incremental?**  
Our observation is that incremental improvements are common in General Face Recognition (GFR). GFR has reached a very high level of performance so the scope of improvement is limited. Most of the recent GFR methods have marginal improvement or even worse than the state-of-the-art but are aimed to solve specific problems. For example, Sphereface+ [9], Center loss [1] and CosFace [15] have improvements from  $-0.19\%$  to  $0.31\%$  on LFW dataset.
3. **What are the highlights of the proposed method?**  
Our method creates two "firsts". It is the first loss function that simultaneously satisfies all five properties in Table 1 and is the first to use global information as feedback.



Therefore, the proposed loss has its own merits, will encourage others to carefully consider the use of global information and will create opportunities for new research.

4. **Cross entropy loss separates the samples of different classes, but does not enlarge the margin between neighbor classes". What's the difference?**

These two cases correspond to two kinds of features: separable features and discriminative features. Separable features are able to separate classes by decision boundaries. Discriminative features are further required to have better intra-class compactness and inter-class separability to enhance predictivity. The Example can be found in Figure 1 of [1].

5. **Using global information is better than just using mini-batch? Why is global information introduced?**

No, both of them are necessary for training a deep learning model. All practitioners are aware that using mini-batch SGD (Stochastic Gradient Descent) makes the neural network generalize better than using standard gradient descent that takes the entire dataset as input, as the randomness helps the network jump out of some local minima which is beneficial to the generalization. Therefore, the proposed deep model is trained by the mini-batch data on one hand. On the other hand, the proposed methods also introduce global information, as the mini-batch data cannot provide the loss functions with precise measurement information, like the positions of the class center and the class edge in Gico loss. Introducing global information makes the measurement information precise, thus improve the final recognition accuracy.

### 3. Experiments

#### 3.1. Experiment Settings

Our network models are implemented by Tensorflow with Inception-ResNet-v1 [23] as the trunk network. We combine Inception-ResNet-v1 with different losses resulting in five different combinations: (1) ResNet+Softmax; (2) ResNet+AM-Softmax; (3) ResNet+Gico Lite A; (4) ResNet+Gico Lite B; and (5) ResNet+Gico Std.

In all experiments, we set 320 as the epoch size, 120 as the batch size,  $5 \times 10^4$  as the weight decay, 0.4 as the keep probability of the fully connected layer, 512 as the embedding size and 0.01 as the shrink rate. We manually optimize the hyperparameter  $\lambda$ . Since it is not sensitive to the performance, we just try multiple different values on the verification set and choose the value that leads to the minimum total loss. The initial learning rate is set to 0.05 and is reduced by a factor of 10 every 100,000 iterations. Table 3 summarizes all experimental simulation parameters.

**Table 3.** Experimental simulation parameters.

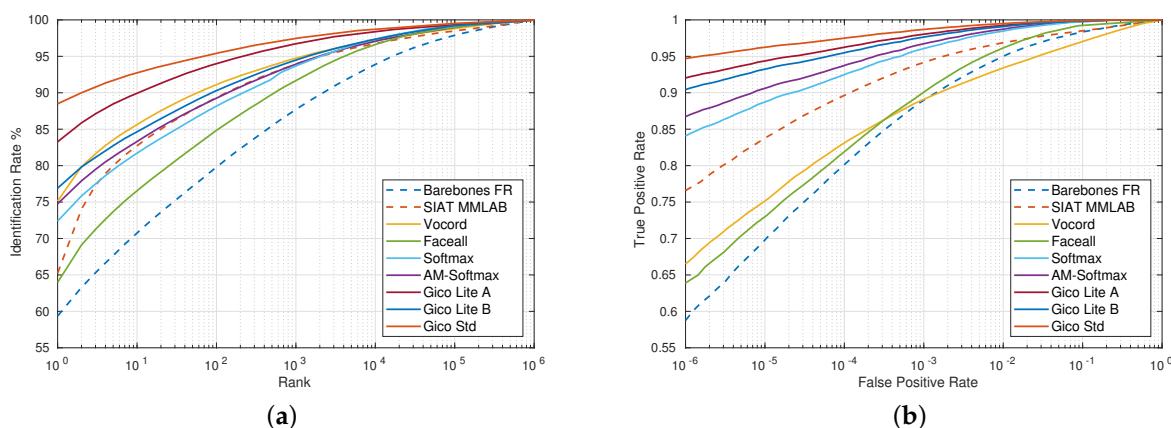
Parameters	Values
epoch size	320
batch size	120
weight decay	$5 \times 10^4$
keep probability of the fully connected layer	0.4
embedding size	512
shrink rate	0.01
initial learning rate	0.05

In all experiments, VGGFace2 [24] is used as the training data. To guarantee the reliability of the results, we removed the identities which might be overlapped with the testing sets from VGGFace2 but we did not do data cleaning, as VGGFace2 is a very clean dataset. Finally, there are 3.05 million face images in the preprocessed training set. For testing, we use diverse public benchmark datasets: LFW [17], SLLFW [18], YTF [19], MegaFace [20] and FaceScrub [21] datasets. For image preprocessing, we applied the same pipeline of processes on every raw image in the training set and the testing sets. At first, MTCNN [25] is employed for face detection. MTCNN occasionally fails to detect the face.

If this occurs for a training image, the image is simply abandoned. If it occurs for a testing image, we use the provided official landmarks or bounding boxes instead. All the face images are cropped to the size of  $160 \times 160$ . To strengthen the randomness of the training data, random horizontal flipping is performed on the training images. The final features of a testing image are generated by concatenating the features of the original image and the features of its horizontally flipped counterpart so as to improve the recognition accuracy.

### 3.2. MegaFace Challenge 1 on FaceScrub

In this section, we evaluate the performance of the proposed Gico loss on the MegaFace dataset [20] and the FaceScrub dataset [21]. Following the experimental protocol of MegaFace Challenge 1, we use the MegaFace dataset as the distractor set and set 1 million distractors. FaceScrub dataset is used as the testing set. The evaluation is conducted with the officially provided code [20]. Figure 2a,b report the CMC curves and the ROC curves of different methods with 1M distractors on MegaFace Set 1, respectively. The results of the benchmark methods (including Barebones FR, SIAT MMLAB, Vocord and Faceall) are generated with the evaluation code and features provided by MegaFace team <http://megaface.cs.washington.edu/participate/challenge.html> accessed on 30 June 2021. From Figure 2a, we can observe that the three versions of Gico loss outperform Softmax, AM-Softmax and other benchmark methods on the Rank1 identification rate by 5% to 22%. On Rank10, the best-performing comparable method is Vocord, but Gico Std still outperforms it by 7%. On all the values of rank, Gico Std shows better performance than Gico Lite B and Gico Lite A, while Gico Lite A performs better than Gico Lite B. Figure 2b shows the verification performance, where all three versions of Gico loss significantly outperform the other methods with the change of False Positive Rate. Specifically, the proposed Gico loss has a higher True Positive Rate than the other methods by at least 4% when the False Positive Rate is  $10^{-6}$ . Gico Std still shows better performance than Gico Lite B and Gico Lite A. These results on the FaceScrub dataset demonstrate the effectiveness of the proposed Gico loss.



**Figure 2.** (a) The CMC curves of different methods with 1 million distractors on MegaFace Set 1. (b) The ROC curves of different methods with 1 million distractors on MegaFace Set 1.

### 3.3. Results on LFW, YTF and SLLFW

In this section, the proposed methods and the state-of-the-art methods are evaluated on the LFW, YTF and SLLFW dataset. The LFW [17] face image dataset is collected from the web. It contains 13,233 face images with large variations in facial paraphernalia, pose and expression. Following the standard experimental protocol of “unrestricted with labeled outside data” [26], 6000 face pairs are tested according to the given pair list. The YTF [19] face video dataset contains 3425 videos and is obtained from YouTube. We also follow the

standard experimental protocol of “unrestricted with labeled outside data” to evaluate the relevant methods on the given 5000 video pairs.

Table 4 shows the experimental results of different methods on the LFW and YTF datasets. As we follow the same experimental protocol and settings, the results shown in the upper part of the table are cited from their original papers. From Table 4, it can be observed that Gico Std shows higher verification accuracy on LFW than Softmax, AM-Softmax, Gico Lite A and Gico Lite B by about 0.1%. Gico Std ties with FaceNet for first place on LFW. However, Gico Std utilizes only 3.05 million images for training, whilst FaceNet utilizes 200 million images for training. Gico Std also beats the other benchmark methods by 2.28% to 0.11% on LFW, most of which are published in leading computer vision conferences. As for the results on the YTF dataset, all three versions of Gico loss have a better performance than the comparable methods by at most 3.42%, which demonstrates the state-of-the-art performance of the Gico loss.

**Table 4.** Verification performance of state-of-the-art methods on LFW and YTF datasets.

Methods	Images	LFW(%)	YTF(%)
ICCV17' Range Loss [3]	1.5M	99.52	93.7
CVPR15' DeepID2+ [27]		99.47	93.2
CVPR14' Deep Face [28]	4M	97.35	91.4
CVPR15' Fusion [29]	500M	98.37	
ICCV15' FaceNet [4]	200M	99.63	95.1
ECCV16' Center Loss [1]	0.7M	99.28	94.9
NIPS16' Multibatch [30]	2.6M	98.20	
ECCV16' Aug [31]	0.5M	98.06	
ICML16' L-Softmax [8]	0.5M	98.71	
CVPR17' A-Softmax [9]	0.5M	99.42	95.0
Softmax	3.05M	99.50	95.22
AM-Softmax	3.05M	99.57	95.62
<b>Gico Lite A</b>	3.05M	99.60	95.70
<b>Gico Lite B</b>	3.05M	99.62	95.78
<b>Gico Std</b>	3.05M	99.63	95.82

LFW is a popular face dataset. However, more and more methods are gradually touching its theoretical upper limit. Consequently, it becomes more and more difficult to differentiate different methods on LFW. To confirm the performance of the proposed methods, we conducted an additional experiment on SLLFW [18]. SLLFW uses the same positive pairs as LFW for testing, but in SLLFW, 3000 similar-looking face pairs are deliberately selected out from LFW by human crowdsourcing to replace the random negative pairs in LFW. SLLFW adds more challenges to the testing, causing the accuracy of the same state-of-the-art methods to drop by about 10–20%.

From Table 5, we can see the verification accuracy of different methods on SLLFW. The results of some benchmark methods are shown in the top half of the table, which are provided by the SLLFW team [32] and are publicly accessible <http://www.whdeng.cn/SLLFW/index.html#reference> accessed on 30 June 2021. As shown in Table 5, Gico loss achieves considerably higher verification accuracy on SLLFW when it is compared with other methods. In the top half of Table 5, the accuracy of the benchmark methods drops by between 4.68% and 16.75% from LFW to SLLFW. By comparison, the accuracy of the proposed Gico loss drops by between 1.45% and 1.49%. The experimental results on SLLFW further confirm the effectiveness of the proposed methods.

**Table 5.** Verification performance of different methods on SLLFW.

Method	Images	LFW(%)	SLLFW(%)
Deep Face [28]	0.5M	92.87	78.78
DeepID2 [5]	0.2M	95.00	78.25
VGG Face [33]	2.6M	96.70	85.78
DCMN [32]	0.5M	98.03	91.00
Noisy Softmax [34]	0.5M	99.18	94.50
Softmax	3.05M	99.50	96.17
AM-Softmax	3.05M	99.57	98.02
<b>Gico Lite A</b>	3.05M	99.60	98.15
<b>Gico Lite B</b>	3.05M	99.62	98.13
<b>Gico Std</b>	3.05M	99.63	98.17

#### 4. Conclusions

This paper presents a novel loss function—Global Information-based Cosine Optimal loss (i.e., Gico loss). To the best of our knowledge, Gico loss is the first attempt to use global information as the feedback in face recognition. We propose a novel algorithm to learn the cosine similarity between the class center and the class edge so as to break through the constraint and make Gico loss possible. In addition, the advantages of the best losses proposed in recent years are also integrated into the Gico loss. Extensive experiments are conducted on the LFW, SLLFW, YTF, MegaFace and FaceScrub datasets. The experimental results show that the proposed Gico loss outperforms all comparable methods on all datasets. Especially in the FaceScrub dataset, the three versions of Gico loss outperform the comparable methods on the Rank1 identification rate by 5% to 22%. The results demonstrate the effectiveness of the Gico loss and show that we achieve a state-of-the-art performance. However, since the class center and the class range used in Gico loss are obtained through a learning process, there is a time lag, which leads to a longer time to complete convergence. Future work will focus on reducing the convergence time while ensuring the learning accuracy of the class center and class range.

**Author Contributions:** Conceptualization, X.W.; Data curation, X.W.; Investigation, X.W.; Methodology, X.W.; Project administration, W.M.; Writing—original draft, X.W.; Writing—review and editing, W.D., X.H. and J.H. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the National Natural Science Foundation of China (Grant No.6210021948, 62076117) and Jiangxi key Laboratory of Smart City (Grant No. 20192BCD40002).

**Data Availability Statement:** The data presented in this study are available on request from the first author.

**Conflicts of Interest:** The authors declare no conflict of interest.

#### References

1. Wen, Y.; Zhang, K.; Li, Z.; Qiao, Y. A Discriminative Feature Learning Approach for Deep Face Recognition. In *Computer Vision—ECCV (Lecture Notes in Computer Science)*; Springer: Cham, Switzerland, 2016; pp. 499–515.
2. Deng, J.; Zhou, Y.; Zafeiriou, S. Marginal Loss for Deep Face Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*; IEEE: New York, NY, USA, 2017; pp. 60–68.
3. Zhang, X.; Fang, Z.; Wen, Y.; Li, Z.; Qiao, Y. Range loss for deep face recognition with long-tailed training data. In *Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017*; pp. 5409–5418.
4. 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.
5. Sun, Y.; Chen, Y.; Wang, X.; Tang, X. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*; The Chinese University of Hong Kong: Hong Kong, China, 2014; pp. 1988–1996.
6. Deng, J.; Guo, J.; Zafeiriou, S. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. *arXiv* **2018**, arXiv: 1801.07698.
7. Liu, B.; Deng, W.; Zhong, Y.; Wang, M.; Hu, J.; Tao, X.; Huang, Y. Fair loss: Margin-aware reinforcement learning for deep face recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019*; pp. 10052–10061.

8. Liu, W.; Wen, Y.; Yu, Z.; Yang, M. Large-Margin Softmax Loss for Convolutional Neural Networks. In *International Conference on Machine Learning*; ICML: New York, USA, 2016; pp. 507–516.
9. Liu, W.; Wen, Y.; Yu, Z.; Li, M.; Raj, B.; Song, L. SphereFace: Deep Hypersphere Embedding for Face Recognition. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 21–26 July 2017; pp. 6738–6746. [[CrossRef](#)]
10. Wang, F.; Cheng, J.; Liu, W.; Liu, H. Additive margin softmax for face verification. *IEEE Signal Process. Lett.* **2018**, *25*, 926–930. [[CrossRef](#)]
11. Zhang, W.; Chen, Y.; Yang, W.; Wang, G.; Xue, J.-H.; Liao, Q. Class-Variant Margin Normalized Softmax Loss for Deep Face Recognition. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**. [[CrossRef](#)] [[PubMed](#)]
12. Zhong, Y.; Deng, W.; Hu, J.; Zhao, D.; Li, X.; Wen, D. SFace: Sigmoid-Constrained Hypersphere Loss for Robust Face Recognition. *IEEE Trans. Image Process.* **2021**, *30*, 2587–2598. [[CrossRef](#)] [[PubMed](#)]
13. Liu, Y.; Li, H.; Wang, X. Rethinking feature discrimination and polymerization for large-scale recognition. *arXiv* **2017**, arXiv:1710.00870.
14. Ranjan, R.; Castillo, C.D.; Chellappa, R. L2-constrained softmax loss for discriminative face verification. *arXiv* **2017**, arXiv:1703.09507.
15. Wang, H.; Wang, Y.; Zhou, Z.; Ji, X.; Gong, D.; Zhou, J.; Li, Z.; Liu, W. CosFace: Large Margin Cosine Loss for Deep Face Recognition. *arXiv* **2018**, arXiv:1801.09414.
16. Liu, W.; Zhang, Y.-M.; Li, X.; Yu, Z.; Dai, B.; Zhao, T.; Song, L. Deep hyperspherical learning. In *Proceedings of the Advances in Neural Information Processing Systems*, Long Beach, CA, USA, 4–9 December 2017; pp. 3950–3960.
17. Huang, G.-B.; Ramesh, M.; Berg, T.; Learned-Miller, E. *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*; Technical Report 07-49; University of Massachusetts: Amherst, MA, USA, 2007.
18. Zhang, N.; Deng, W. Fine-grained LFW database. In *Proceedings of the International Conference on Biometrics*, Niagara Falls, NY, USA, 1–6 September 2016; pp. 1–6.
19. Wolf, L.; Hassner, T.; Maoz, I. Face recognition in unconstrained videos with matched background similarity. In *Proceedings of the Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, CO, USA, 20–25 June 2011; pp. 529–534.
20. Kemelmacher-Shlizerman, I.; Seitz, S.M.; Miller, D.; Brossard, E. The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 27–30 June 2016; pp. 4873–4882.
21. Ng, H.W.; Winkler, S. A data-driven approach to cleaning large face datasets. In *Proceedings of the Image Processing (ICIP), 2014 IEEE International Conference on*, Toronto, ON, Canada, 27 April–2 May 2014; pp. 343–347.
22. Wei, X.; Wang, H.; Scotney, B.; Wan, H. Gicoface: Global Information-Based Cosine Optimal Loss for Deep Face Recognition. In *Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP)*, Taipei, Taiwan, 22–25 September 2019; pp. 3457–3461.
23. Szegedy, C.; Ioffe, S.; Vanhoucke, V. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arXiv* **2016**, arxiv1602.07261.
24. Cao, Q.; Shen, L.; Xie, W.; Parkhi, O.M.; Zisserman, A. VGGFace2: A dataset for recognising faces across pose and age. *arXiv* **2017**, arXiv:1710.08092.
25. Zhang, K.; Zhang, Z.; Li, Z.; Qiao, Y. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Process. Lett.* **2016**, *23*, 1499–1503. [[CrossRef](#)]
26. Huang, G.B.; Learned-Miller, E. Labeled faces in the wild: Updates and new reporting procedures. *Dept. Comput. Sci. Univ. Mass. Amherst Amherst MA USA Tech. Rep.* **2014**, *14*, 14–003.
27. Sun, Y.; Wang, X.; Tang, X. Deeply learned face representations are sparse, selective, and robust. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, 7–12 June 2015; pp. 2892–2900.
28. Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, Washington, DC, USA, 23–28 June 2014; IEEE Computer Society: Washington, DC, USA, 2014; pp. 1701–1708.
29. Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. Web-scale training for face identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, 7–12 June 2015; pp. 2746–2754.
30. Tadmor, O.; Rosenwein, T.; Shalev-Shwartz, S.; Wexler, Y.; Shashua, A. Learning a Metric Embedding for Face Recognition Using the Multibatch Method. In *Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS'16)*; Curran Associates Inc.: Red Hook, NY, USA, 2016; pp. 1396–1397.
31. Masi, I.; Tran, A.T.; Hassner, T.; Leksut, J.T.; Medioni, G. Do We Really Need to Collect Millions of Faces for Effective Face Recognition?. In *Computer Vision—ECCV*; (Lecture Notes in Computer Science); Springer: Cham, Switzerland, 2016; pp. 579–596.
32. Deng, W.; Hu, J.; Zhang, N.; Chen, B.; Guo, J. Fine-grained face verification: FGLFW database, baselines, and human-DCMN partnership. *Pattern Recognit.* **2017**, *66*, 63–73. [[CrossRef](#)]

- 
33. Parkhi, O.M.; Vedaldi, A.; Zisserman, A. Deep Face Recognition. In Proceedings of the 2015 British Machine Vision Conference (BMVC), Swansea, UK, 7–10 September 2015; Volume 1, p. 6.
  34. Chen, B.; Deng, W.; Du, J. Noisy softmax: Improving the generalization ability of dcnn via postponing the early softmax saturation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.