# Simple Triplet Loss Based on Intra/Inter-class Metric Learning for Face Verification

Zuheng Ming Joseph Chazalon Muhammad Muzzamil Luqman Muriel Visani
Jean-Christophe Burie
L3I, University of La Rochelle, La Rochelle, France
{zuheng.ming, joseph.chazalon, muhammad_muzzamil.luqman, muriel.visani, jcburie}@univ.lr-fr

## Abstract

*Recently, benefiting from the advances of the deep convolution neural networks (CNNs), significant progress has been made in the field of the face verification and face recognition. Specially, the performance of the FaceNet has overpassed the human level performance in terms of the accuracy on the datasets "Labeled Faces in the Wild (LFW)"and "Youtube Faces in the Wild (YTF)". The triplet loss used in the FaceNet has proved its effectiveness for face verification. However, the number of the possible triplets is explosive when using a large scale dataset to train the model. In this paper, we propose a simple class-wise triplet loss based on the intra/inter-class distance metric learning which can largely reduce the number of the possible triplets to be learned. However the simplification of the classic triplet loss function has not degraded the performance of the proposed approach. The experimental evaluations on the most widely used benchmarks LFW and YTF show that the model with the proposed class-wise simple triplet loss can reach the state-of-the-art performance. And the visualization of the distribution of the learned features based on the MNIST dataset has also shown the effectiveness of the proposed method to better separate the classes and make the features more discriminative in comparison with the other state-of-the-art loss function.*

## 1. Introduction

The face verification and face recognition problems represent a sub-domain of the more general problem of visual object recognition or classification. In recent years, thanks to the great development of the deep CNNs the effective and powerful high-level features have been learned which can very well represent the images, and the state-of-the-art of visual object classification and recognition has been significantly improved [10, 24, 12, 7, 8]. Generally, the deep CNNs mainly develops in the following three directions: 1) constructing deeper networks, such as the VGG networks [18] which has 19 layers, and the RsNet series [7, 8] which has even more that 1000 layers; 2) constructing wider networks, such as the Inception networks [24, 25] which contain more than one branch by extending one layer with several different convolution blocks or the maxpool module; 3) constructing the networks by fusing the two structures from 1) and 2) such as the InceptionV4 [23] which fuses the inception module into a very deep RsNet networks aiming to take advantage of the depth and width of the networks. Given the success of these networks obtained in the field of the classification or recognition of the visual objects, these powerful networks have also been applied to the face recognition problem. With the very deep and wide CNNs, this is first time that the accuracy of face verification/recognition has overpassed the human-level performance, as evaluated on some benchmarks such as LFW [9] and YTF [16]. Considering the good representation capacity of the Inception networks and the residual learning framework making it possible to train a deep networks without the problem of vanishing gradient, in this work we propose to use a deep networks based on the Inception-RsNet structure. Beside the architecture of the networks, another important factor for both general image classification and face verification problem is the design of the loss function. The loss function not only controls the object of the optimization of the deep networks but also affects their efficiency during the training of the model. In the field of face recognition, the state-of-the-art FaceNet proposes to use the triplet-loss as the loss function to train a deep CNNs for establishing an embedding space, in which the face images of the same identity should be more close to the face images of the different persons. Ideally, the triplet loss would compare all the possible pairs of the images in the dataset during the training. This is unpractical since the number of the possible pairs will be explosive when the size of the dataset increases. Thus a complicated sampling strategy was proposed in [16], which only selects the hard or semi-hard samples to train the model. However even with the vast com-

putation resources at Google, it took hundreds of hours to train the model. Inspired by the center-loss in [27] and the idea of Linear Discriminative Analysis (LDA), we propose a simple class-wise triplet loss based on the intra/inter-class distance metric learning to employ the idea of the triplet loss on the level of classes instead of the individual samples. The loss function that we have proposed in this work, aims to decrease the distance of the samples to the center within the same class and enlarge the distances to the centers of inter-classes by enforcing a margin between intra-class and inter-class distances. Specifically, we use the centers of the classes, instead of the individual samples, as the possible positives and negatives in the triplet pairs. Since the class-wise loss function only considers the distance of a sample to the intra- and inter-classes centers, the number of the triplets used for training the model can be largely reduced, which results in a decrease of the computation cost of the training processing. Our main contributions are summarized as follows.

- We propose a simple class-wise triplet loss based on intra/inter-class distance metric learning. The proposed class-wise triplet loss aims to minimize the intra-class distance of the features meanwhile maximize the inter-class distance. By using the centers instead of the individual samples as the possible positives and negatives in the triplets, the class-wise triplet loss can largely reduce the number of the triplets to be learned which can consequently simplify the training procedure.

- The visualization of the distributions of the features learned by the different loss functions shows the advantage of the proposed approach which can better separate the classes and make the data more discriminative.

- The evaluations on the widely used benchmarks LFW and YTF show the state-of-art performance of the proposed class-wise loss function, even with a small training dataset the model can reach a comparable state-of-art performance.

- The deep CNNs networks based on the Inception-RsNet is proposed to implement the proposed loss function.

In the following parts of this paper: in Section 2 we review the related works in the area of face verification/recognition; in Section 3 we conduct a preliminary study based on the dataset MNIST to have an intuitive idea and then elaborate the formulations of the proposed class-wise triplet loss function based on intra/inter-loss metric learning; Section 4 describes the deep CNNs networks used in this work; and Section 5 presents the datasets used for the training and the evaluations; finally in Section 6 and 7 we present the experimental evaluations of our proposed model and the conclusion, respectively.

## 2. Related Work

The face verification and recognition problem have always received the great interests of the researchers. Before the deep learning, the classification methods for face recognition are mainly based on the well-designed handcrafted features extracted by the feature engineering. To make a distinction from the deep neural architecture, these models are so called "shallow" models. In order to represent the face image, many local descriptor have been proposed for face recognition task, such as LBP, HOG, Gabor-LBP, SIFT [4, 28, 19, 14]. Later, the Fisher Vector [17, 28] has been proposed to employ a fusion mechanism to integrate the different features into an overall face descriptor. Recently, face verification or recognition has achieved a series breakthrough via the deep neural networks and especially the deep CNNs architecture.

DeepFace [26] firstly introduced a siamese networks architecture for the face verification problem. Siamese networks consists of two identical CNNs, in parallel, which are fed by two images in the pair to be distinguished. Two high-level features extracted from the two CNNs are employed as the descriptors of the images. A metric learning based on the L2-norm distance of the two extracted features is used to train the model, in which the model minimizes the Euclidean distance of the images of the same identity and maximizes the distance of the images from the different person. Besides, a 3D to 2D alignment prep-processing is applied to align the different poses of the face images. Thus, in addition to a deep CNNs model, a 3D-based pose alignment model has also been adopted in DeepFace. Training on a private dataset including 4 million examples of 4000 identities, DeepFace has achieved 97.35% on the LFW and 91.4% on the YTF.

DeepID [21, 20, 22] series continue the work of the DeepFace. The significant feature of the DeepID series is using more than 200 CNNs to form the so-called multi-scale ConNets for face verification. However, DeepID [21] and DeepID2 [20] still keep the structure of siamese architecture using the Joint Bayesian [2] technique for face verification. Unlike the DeepFace, DeepID use a simple 2D alignment instead of the 3D alignment and DeepID was trained on the public datasets. Benefiting from a very complicated structure, DeepID series have reached the state-of-the-art performance (99.15% on LFW).

FaceNet [16] is proposed by the Google's researchers which still keeps the state-of-the-art results for face verification and recognition on the benchmarks LFW and YTF. It proposes to use the triplet loss on the sampled triplet face images including a pair of images from the same person and

an image from the different person. A distance metric learning was employed in the triplet-loss, which aims to make the images from the same person closer than the ones from the different person in terms of the Euclidean distance. Since it is impossible to check all possible triplets in the dataset, the FaceNet uses some strategies to limit the samples which are so-called "hard samples" or "semi-hard samples". It means only the samples most-violating or second-most-violating the optimization goal have been selected to train the model. The triplet loss function is applied to train several different deep CNNs based on Inception structure aiming to adapt the model to the different use cases. Even with the sampling strategies to limit the training samples, the training cost is impressive (hundreds of hours for training) based on their private massive datasets which has about 200 million images spanning 8 million identities. For face verification task, FaceNet achieved 99.63% (overpassing human-level 97.5%) on LFW and 95.12% on YTF.

VGG face [18] implements the triplet loss on the VGG networks and trains the model on the datasets collected by their proposed protocol with about 2.6 million images spanning 2622 celebrities. VGG face also received a state-of-the-art result for face recognition.

In [27], the center loss joint with the cross entropy loss of softmax is proposed to use for face recognition. Unlike the triplet loss, the center loss tries to decrease the distances of the samples to their within-class centers to make the data more discriminative. It does not need the sample strategy as used in the triplet loss. The model uses a combination of the public datasets including CASIA-WebFace [29], CACD2000 [1], Celebrity+ [13] to train their deep CNNs networks and also achieved the state-of-the-art performance.

## 3. Proposed Simple Triplet Loss

Triplet loss is proven to be very effective for face verification/recognition and also in the related domain such as person re-identification [3]. By enforcing a margin between the pairs of faces of the same identity and the ones of the different identities, the triplet loss tries to keep the faces of the same identity closer than the faces from the different identities in the embedding space. This allows the faces for one identity to live on a manifold while still enforcing the distance and thus discriminating to other identities [16]. However, in order to describe the entire distribution of the dataset the classic triplet loss should implement on all possible triplet pairs denoted by <anchor, positive, negative>, in which anchor is an input sample, positive is an image sample belonging to the same identity while the negative is a sample from the different identity. In this way the number of the possible triplet pairs will grow exponentially when a large-scale dataset is provided.

A problem for applying the triplet loss is how to sample the triplet pairs efficiently. Inspired by the center loss and Linear Discriminant Analysis (LDA), we would employ a triplet loss idea on the level of the class rather than the individual sample by employing the centers of the classes to represent the overall distribution the classes. Specifically, we let the input sample closer to their within-class centers but further to the centers of the other classes in embedding space (see Figure 1). Since we use the centers of the classes to represent the global distribution of the classes rather than the individual samples, we only have $k-1$ triplets for each sample, namely <anchor, intra-class center, inter-class center>, where $k$ is the number of the classes. Thus the proposed class-center based triplet loss can largely decrease the number of the possible triplets for each input sample comparing to the classic triplet loss method. For instance, assuming a dataset with $k$ classes and $n$ samples in each class, there are $n(n-1)k$ possible triplets for each sample for the classic triplet loss method, while for the proposed class-center based triplet loss we have only $k-1$ triplets (see Figure 2). The significant decrease of the number of the triplets can consequently reduce the computation cost for training the model.

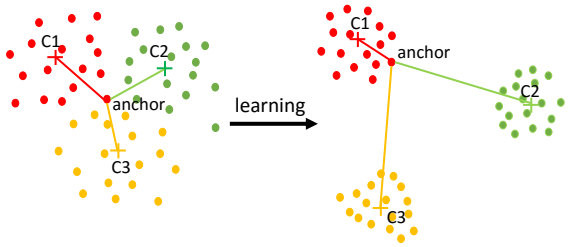



Figure 1: The proposed class-wise triplet loss enforces the input sample (i.e. anchor) closer to the intra-class center and further to the inter-class centers.

### 3.1. Preliminary study on MNIST

Before we elaborate the formulations of the proposed approach, we present an intuitive example based on the MNIST dataset [11] to illustrate how the proposed class-wise triplet loss to effect the distribution of the features learned by a simple CNNs networks. Figure 3 shows the simple CNNs networks with only 4 hidden layers applied in the toy experiment. The proposed class-wise triplet loss is calculated based on the bottleneck layer which is the last hidden fully connected layer fc2. The last layer of the networks is the softmax layer which can help the networks converge fast to make the features discriminable preliminarily. In order to provide a more intuitive visualization of the distribution of the learned features, the 2D features are given in the bottleneck layer for calculating the proposed triplet loss. Table 1 shows the details of the networks. Since the

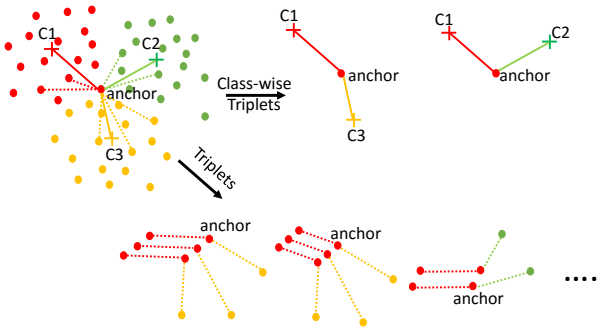



Figure 2: The class-wise triplets of the proposed method and the triplets of the classic triplet loss method. Unlikely the classic triplet loss method, the number of the class-wise triplets only depends on the centers of the classes but independent with the samples within the classes, which can largely decrease the number of the possible triplets to be learned.

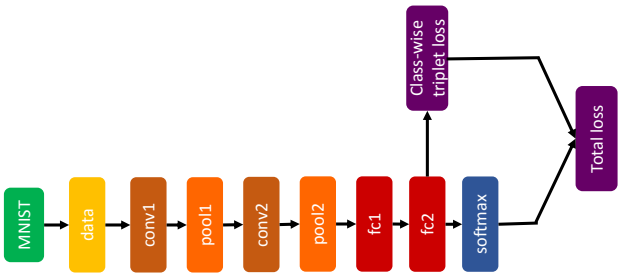



Figure 3: The simple CNNs networks for the toy example based on the MNIST. The proposed class-wise triplet loss is computed based on the bottleneck layer, i.e. the last hidden full connect layer. The last layer of the networks is the softmax layer.

| layer | conv1 | pool1 | conv2 | pool2 | fc1 | fc2 |
|---|---|---|---|---|---|---|
| kernel | 5*5 | 2*2 | 5*5 | 2*2 | 7*7 | 1024 |
| filters | 32 | 1 | 64 | 1 | 1024 | **2** |

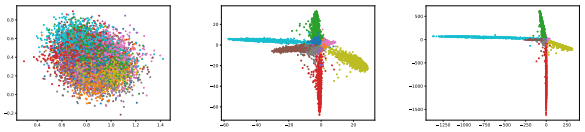
Table 1: The simple CNNs networks applied in the toy example. The 2D features are given by the fc2 layer for computing the class-wise triplet loss. The rectified linear unit is employed as the nonlinear activation function in the networks.

experiment is based on the MNIST, the output of the softmax has 10 classes. In the following illustration, we will use 10 colors to represent the 10 classes. The Stochastic Gradient Descent (SGD) was employed to optimize the gradients based on the mini-batch with the learning rate 1e-4.

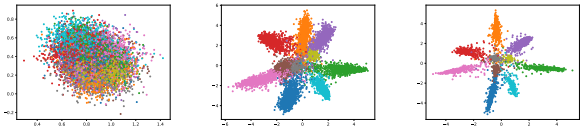
Meanwhile, the center loss focusing on the intra-class distance metric and the softmax loss measuring the probability similarity of the classes have been also carried out to compare with the class-wise triplet loss. Thus actually three different models were trained in the toy experiment. The frameworks of the three models are the same except the configurations of the loss functions: the cross entropy loss is served as the total loss of the softmax, while the center loss and the class-wise triplet loss are joint with the softmax loss as the total loss respectively. Figure 4 has shown the distributions of the 2D features extracted by the models learned with the three different metrics. Note that the models used for extracting the features are trained in advance based on the training dataset of the MNIST, and then the features of the test dataset of the MNIST were extracted and their distributions are shown in the figure. From the Figure 4 we can see that the softmax can only partly separate the features where 5 classes of 10 are separated apparently by the model trained after 40000 iterations. Comparing to the pure softmax, the center loss joint with the softmax is much better. Benefiting from the optimization of the intra-class distances to their centers, the learned features of the center loss are much more centralized and discriminative. The majority part of the classes, i.e. 7 classes in 10, have been well separated. However the center loss only optimizes the intra-class distance disregarding of the inter-class distance in the loss function, and 3 classes (in the color of gray, brown and olive) are still gathering together. By adding the measurement of the inter-class distance to the centers of the other classes, the proposed class-wise triplet loss has further separated the classes that 8 classes in 10 have been separated effectively. In addition, by enforcing the margin between the intra- and inter-class distance, the class-wise loss enables the margins of the separated classes are indeed greater than the center loss which can help to discriminate the learned features. This point is also demonstrated in the evaluation of the three models in terms of the accuracy of the classification (see Figure 5), in which we can see that the model trained with the proposed class-wise triplet loss can converge faster and obtain a slightly better result.

### 3.2. Simple class-wise triplet loss metric

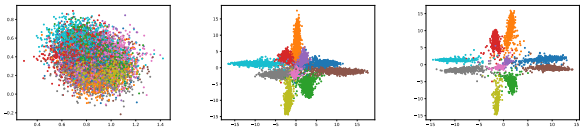
In this section, we describe the proposed simple class-wise triplet loss in detail. The basic idea of the triplet loss is enforcing the input sample as an anchor being closer to the positive (the sample within the class) than the negative (the sample belonging to the other classes). While in this work, the proposed class-wise triplet loss using the centers of the classes as the possible positives or negatives instead of the individual samples in the classic triplet loss. Thus the class-wise triplet loss for a triplet <anchor, positive, negative> can be described as:



(a) iters 0 (b) iters 5000 (c) iters 40000



(d) iters 0 (e) iters 5000 (f) iters 40000



(g) iters 0 (h) iters 5000 (i) iters 40000

Figure 4: The distributions of the 2D features learned by the different metrics based on the dataset MNIST. The rows from top to bottom are corresponding to the distributions of the features learned by the metrics of the softmax, center loss and proposed class-wise triplet loss respectively. The columns from left to right corresponding to the distributions of the features extracted by the models trained in different stages from 1 to 40000 iterations. In particularly the sub-figure corresponding to the iteration 0 means the distribution of the input data, and the input data for the three different models are the same. The 10 classes in MNIST are represented by 10 colors in the figures.

$$d_{inter} \geq d_{intra} + \beta_0 \tag{1}$$

Where, $d_{intra}$ is the intra-class distance of an anchor to its center within the class, $d_{inter}$ is the inter-class distance of an anchor to the center of the other class, $\beta_0$ is the margin between the intra- and inter-class distances. Thus the class-wise triplet loss $l_c$ of a triplet is given by:

$$l_c = \max(d_{intra} + \beta_0 - d_{inter}, 0) \tag{2}$$

For a given anchor $\mathbf{x_i} \in \mathbb{R}^d$ with all possible class-wise triplets corresponding to $k$ classes, the class-wise triplet loss of $\mathbf{x_i}$ is given by:

$$L_c^i = \sum_{l=1, l\neq y_i}^{k} \max(d_{y_i,i} + \beta_0 - d_{l,i}, 0) \tag{3}$$

where $L_c^i$ is the class-wise triplet loss of anchor $\mathbf{x_i}$, $k$ is the number of the classes, $d_{y_i,i}$ is the distance of the anchor $\mathbf{x_i}$ to the center of the $y_i$th class corresponding to the $\mathbf{x_i}$, i.e. the intra-class distance of $\mathbf{x_i}$, and $d_{l,i}$ is the inter-class distance of the anchor $\mathbf{x_i}$ to the center of the $l$th class.

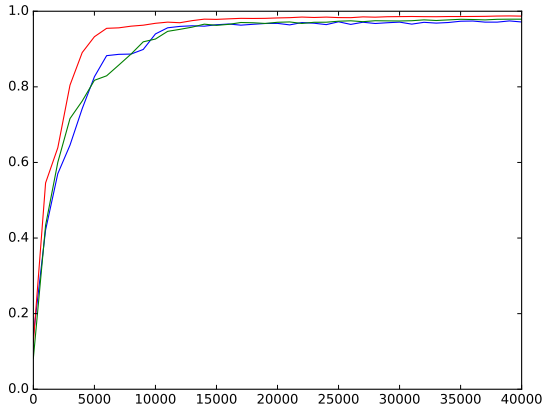



Figure 5: The accuracies of the classification of the models trained with the three metrics of the softmax (blue), center loss (green) and proposed class-wise triplet loss (red) based on the dataset MNIST. The horizontal axis is the number of the iterations for training the models.

Since the training of the deep networks is normally based on the mini-batch, the class-wise triplet loss $L_c$ of the mini-batch with $m$ samples is given by:

$$\begin{aligned} L_c &= \sum_{i=1}^{m} L_c^i \\ &= \sum_{i=1}^{m} \sum_{l=1, l\neq y_i}^{k} \max(d_{y_i,i} + \beta_0 - d_{l,i}, 0) \\ &= \max(k \sum_{i=1}^{m} d_{y_i,i} + m(k-1)\beta_0 - \sum_{i=1}^{m}\sum_{l=1}^{k} d_{l,i}, 0) \\ &= \max(kD_{intra} + \beta - D_\psi, 0) \end{aligned} \tag{4}$$

Where, $D_{intra}$ is the sum of the intra-class distances of all the samples in the mini-batch and $D_\psi$ is the sum of the distances of all the samples to the centers of the classes. $D_{intra}$ is given by:

$$D_{intra} = \frac{1}{2} \sum_{i=1}^{m} \|\mathbf{x_i} - \mathbf{c}_{y_i}\|_2^2, \quad \mathbf{x_i}, \mathbf{c}_{y_i} \in \mathbb{R}^d \tag{5}$$

where the $\mathbf{c}_{y_i}$ is the center of $y_i$th class corresponding to the deep feature $\mathbf{x_i}$. $D_\psi$ is given by:

$$D_\psi = \frac{1}{2} \sum_{i=1}^{m} \sum_{l=1}^{k} \|\mathbf{x}_i - \mathbf{c}_l\|_2^2, \quad \mathbf{x_i}, \mathbf{c}_l \in \mathbb{R}^d \tag{6}$$

In practice, the $D_\psi$ is weighted by $\theta$ in the proposed class-wise loss function. So the class-wise triplet loss can be degraded to the center loss when the $\beta = 0$ and $\theta = 0$ .

$$L_c = \max(kD_{intra} + \beta - \theta D_\psi, 0) \tag{7}$$

The updating of the centers of the classes can be calculated simply by averaging the features of the corresponding class of the mini-batch. Nonetheless, this is inclined to have large perturbations caused by the mislabeled samples in the dataset. As proposed in [27], we use a weight $\gamma$ to control the learning rate of the updating:

$$\mathbf{c}_l^{t+1} = \mathbf{c}_l^t - \gamma \Delta \mathbf{c}_l^t \tag{8}$$

where $t$ is the number of the iterations, and $\Delta c_l^t$ is the variation of the centers during the updating, $\gamma$ is the learning rate for updating. The variation of the center $\Delta \mathbf{c}_l$ is given by:

$$\Delta c_l^t = \frac{\sum_{i=1}^{m} 1\{y_i = l\} \cdot (\mathbf{c_l}^t - \mathbf{x_i}^{t+1})}{\sum_{i=1}^{m} 1\{y_i = l\}} \tag{9}$$

where $1\{\cdot\}$ is the indicator function, i.e.$1\{$a true statement$\}$ = 1, and $1\{$a false statement$\}$ = 0. In order to separate the features rapidly, the cross-entropy loss function of the softmax is also jointed with the proposed loss function. For a mini-batch having $m$ features, the cross-entropy loss of softmax with $k$ classes is given by:

$$L_s = -\sum_{i=1}^{m}\sum_{j=1}^{k} 1\{y_i = j\} log \frac{e^{\mathbf{W}_j^T \mathbf{x}_i} + b_{y_i}}{\sum_{l=1}^{k} e^{\mathbf{W}_l^T \mathbf{x}_i} + b_l} \tag{10}$$

Finally, the total loss function of this work is given by:

$$L = L_s + \alpha L_c \tag{11}$$

where $\alpha$ is the weight used to trade off the class-wise triplet loss and the softmax loss in the total loss.

Algorithm 1 shows the main procedure of the training algorithm.

**Algorithm 1:** The class-wise triplet loss training algorithm

**Input :** Training samples $\{\mathbf{I_i}\}$, i.e. the input images
**Output:** The networks parameters $\{\mathbf{w}\}$

**while** $t \leq T$ **do**
- t ← t+1
- Calculate the features $\mathbf{x_i}$ by forward propagation
- Calculate the total loss $L = L_s + \alpha L_c$
- Update the centers of the classes in the mini-batch: $\mathbf{c}_l^{t+1} = \mathbf{c}_l^t - \gamma \Delta \mathbf{c}_l^t$
- Calculate the $\frac{\partial L_s}{\partial \mathbf{x_i}}$, $\frac{\partial L_c}{\partial \mathbf{x_i}}$ by back propagation
- Update the parameters of the softmax (the output layer) $\mathbf{W^{t+1}} = \mathbf{W^t} - \lambda^\mathbf{t} \frac{\partial \mathbf{L_s}}{\partial \mathbf{W^t}}$
- Update the parameters of the networks $\mathbf{w}^{t+1} = \mathbf{w}^t - \lambda^t (\frac{\partial Ls}{\partial \mathbf{x_i}} \cdot \frac{\partial \mathbf{x_i}}{\partial \mathbf{w^t}} + \frac{\partial Lc}{\partial \mathbf{x_i}} \cdot \frac{\partial \mathbf{x_i}}{\partial \mathbf{w^t}})$

**end**

## 4. Deep Inception-ResNet Networks

In this section, we describe the deep CNNs that we have used in this work. Overall the deep CNNs based on the Inception-RsNet architecture has 32 layers in terms the depth and 4 branches of the width. As mentioned before, in order to take advantage of the depth and width of the networks, the inception structure has been adopted. Meanwhile using the residual networks RsNet to avoid the problem of gradient vanish. Although the deep CNNs has more than thirty layers, several simplification techniques are introduced by the Inception module, such as using the 1x1 convolution to reduce the dimension of the convolutions, and also factorizing the standard nxn convolution into 1xn and nx1 modules which reduce the grid-size of the networks while expands the filter banks to keep the representation capability [25, 23]. The total number of the parameter of the networks is about 10 millions, which is 14 times fewer than [30] having 140 millions parameters of standard convolution with 22 layers deep or 6 times fewer than AlexNet [10] having 60 millions parameters with total 9 layers by using the standard convolution. In practice, it spends only about 12 hours to train the networks on dataset CASIA-WebFace with only one GPU (Nvidia TitanX). The architecture of the deep Inception-RsNet CNNs used in this work is shown in Figure 6.

## 5. Datasets

### 5.1. Datasets for training

Two public datasets of different scales are used separately to train the model in this work.

**CASIA-WebFace** dataset is a public dataset which has almost 0.5 million images of about 10 thousands identities. It was one of the largest public dataset when it was introduced. However comparing to the datasets used in Facenet (200 millions) or the one used in DeepFace (SFC, 4 millions), even comparing to the other public datasets proposed recently e.g MS-Celeb-1M [6], it is a relative small dataset now. We mainly trained our model on the CASIA-WebFace.

**MS-Celeb-1M** dataset is a public dataset established by MSR. MS-Celeb-1M is much more larger than CASIA-WebFace which includes almost 10 millions web images covering about 1 million celebrities. The images are collected automatically by the search engine providing the most approximate images of the given celebrities. We also trained the model on MS-Celeb-1M to see the effect of the

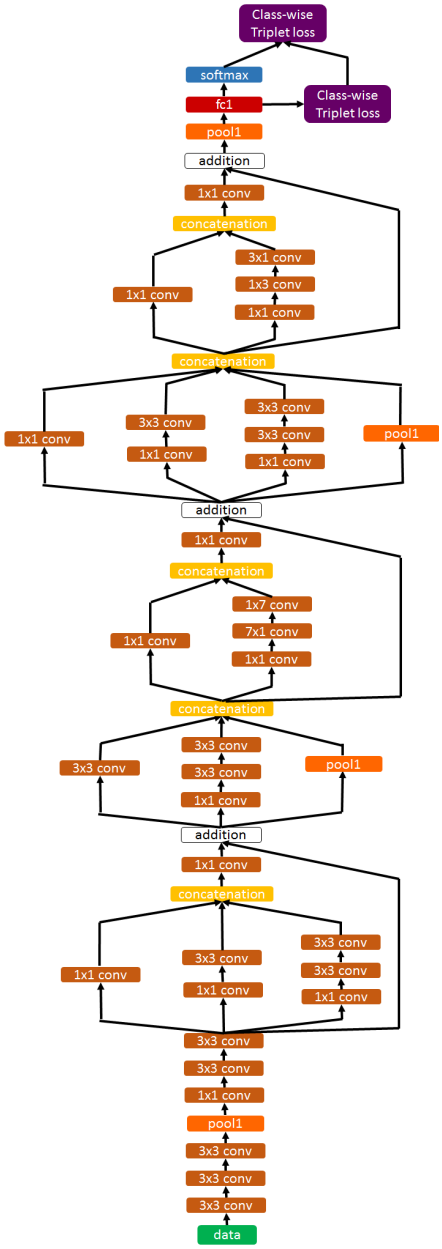



Figure 6: The deep CNNs networks based on the Inception-RsNet used in this work. The proposed class-wise loss is calculated based on the bottleneck layer and the high-level features are extracted from the bottleneck layer.

different scale of dataset. In this work we only use a subset of the MS-Celeb-1M to train the model.

### 5.2. Datasets for evaluation

**LFW** dataset [9] is the most wildly used dataset for evaluating the face verification algorithms. LFW contains 13,233 web-collected images from 5749 different identities, with large variations in pose, expression and illuminations.

**YTF** dataset [28] is the only dataset consisting of videos. It includes 3,425 videos of 1,595 different people, with an average of 2.15 videos per person. The clip durations vary from 48 frames to 6,070 frames, with an average length of 181.3 frames.

## 6. Experiments and results

### 6.1. Training configuration

The model has been mainly trained on the dataset CASIA-WebFace with a relative small scale. In order to verify the effectiveness of the class-wise triplet loss, we propose three configurations for training the model.

**Configuration A** In this configuration, the softmax loss has been only included in the total loss function.

**Configuration B** Using the joint loss function of the center loss and the softmax loss as the total loss function to train the model.

**Configuration C** Using the joint loss function of proposed class-wise triplet loss and the softmax loss to train the model. In this configuration, we also trained a model on the dataset MS-Celeb-1M to see how the different scales of datasets will affect the model.

The networks of the different configurations are the same except the different loss function used in the model.

### 6.2. Evaluation protocol and details

For both evaluation and training procedure, the faces have been detected by the method described in [31], in which a cascade multi-task CNNs framework has been employed to detect and align the faces in the images. The detected face images are aligned to the 180x180 pixels images and used for training the deep CNNs networks. Before inputting the detected face images into the networks, the processing of the data augmentation has been applied to the detected face images:

- **Data filtering** Since the noise in the dataset is prone to degrade the performance of model, it is crucial to filter the data before feeding it to the model for the training. In particularly, the samples in the dataset MS-Celeb-1M are collected automatically by the search engine without any manual checking, thus the data filtering for the MS-Celeb-1M is essential for the training. In this work, the data filtering is based on the L2 distances between the image features and their corresponding centers. The $p\%$ percents image samples corresponding to the extreme large distance will be filtered out from the training dataset. In this work, the 5% samples with the largest distance will be filtered for both datasets MS-Celeb-1M and CASIA-WebFace. Note that a pre-

liminary trained model is provided for producing the deep features of the images.

- **Random crop** In random crop processing, a specific size of patch has been cropped randomly from the original image aiming to augment the variety of the training samples. In this work, the 160x160 pixels image patch has been randomly cropped from the 180x180 pixels detected face image.

- **Random left-to-right flip**. In random left-to-right flip processing, the image patch has been randomly (i.e. with 1/2 chance) flipped horizontally from left. This can make the model more robust for the flipping images.

After the preprocessings for the data augmentation, the image patches are fed to the model for training. Since the last layer of the network is the softmax layer, the high-level features learned from the deep networks have been extracted from the second last full connection (FC) layer, i.e. the bottleneck layer. Then the learned features follow a L2 normalization to make $\|\cdot\| = 1$, which maps the learned features into the embedding space for the later face verification or recognition tasks. For face verification, the distance between the two embeddings has been compared. If the distance is larger than a known threshold we classify the two face images as a negative pair which means the identities of the two face images are different and vice versa. The threshold is searched during the evaluation by the 10-folds cross validation in this work.

The SGD and the mini-batches of 90 samples with standard back propagation [15] are used to train the deep CNNs in this work. The momentum coefficient is set to 0.99 [11]. The learning rate is started from 0.1, and divided by 10 at the 60K, 80K iterations respectively. The model is regularized by using the dropout with the probability of 0.8 and the weight decay of 5e-5. The weights of the filters in the CNNs were initialized by Xavier [5]. Biases were initialized to zero. The weight of the class-wise triplet loss $\alpha$ is set to 1e-4, the margin $\beta$ is set to 10, and the weight of the inter-class distance $\theta$ in the class-wise triplet loss function is set to 0.5.

### 6.3. Results

Table 2 shows the evaluation results on datasets LFW and YTF. This evaluation aims to verify the effectiveness of the proposed class-wise loss and also to compare with the state-of-the-art performance.

Firstly, the results shown in the Table 2 prove the effectiveness of our proposed class-wise loss function. Either on LFW or YTF dataset, the performances of the configuration A are inferior to the configuration B. It means the class-wise loss function essentially works. Secondly, it shows that even the model trained on a relative small dataset, it can obtain a comparable state-of-art result, and when we enlarge the scale of the training dataset, the model can achieve the state-of-art performance. Although the class-wise loss function only evaluated for the face verification, it can be also applied for the face recognition. Moreover, it can be seen that enlarging the scale of the training dataset can help to improve the performance.

| Method | Images | Nets | LFW | YTF. |
|---|---|---|---|---|
| Fisher Faces [17] | - | - | 93.10 | 83.8 |
| DeepFace [26] | 4M | 3 | 97.35 | 91.4 |
| DeepID-2,3 [20, 22] | - | 200 | 99.47 | 93.2 |
| FaceNet [16] | 200M | 1 | 99.63 | 95.1 |
| VGGFace [18] | 2.6M | 1 | 98.95 | 91.6 |
| Centerloss [27] | 0.7M | 1 | 99.28 | 94.9 |
| A(softmax) | 0.46M | 1 | 96.00 | 89.20 |
| B(softmax+centerloss) | 0.46M | 1 | 98.40 | 93.10 |
| C(softmax+$L_c$) | 0.46M | 1 | **98.89** | **94.80** |
| C*(softmax+$L_c$) | 1.1M | 1 | **99.40** | **95.00** |

Table 2: Evaluation results on the LFW and YTF datasets. C* is the model of configuration C trained on the dataset MS-Celeb-1M.

## 7. Conclusion

In this work we have proposed a simple class-wise triplet loss function aiming to decrease the distances between the anchors and the intra-classes centers and enlarge the distances of the anchors to the inter-class centers. By using the centers to instead of the individual samples as the positives and negatives, the number of the possible triplets to be learned can be largely decreased which can effectively simplify the training process. However the simplification of the classic triplet loss hasn't degraded the performance of the proposed approach. Thanks to the optimization of the intra/inter-class distance simultaneously, the class-wise triplet loss can better separate the classes to make the features more discriminative in compare with the state-of-art center loss function. The preliminary experiment on the MNIST and the evaluations on the widely used benchmarks LFW and YTF prove the effectiveness of the proposed loss function. Indeed, the center loss can be treated as a special case of the class-wise triplet loss based on the intra/inter-class metric learning, which has been proved in the formulations of the class-wise triplet loss. In this work, the evaluation of the model only employed for the face verification task, while the model can be also used for the face recognition task or even more general classification problems.

**Acknowledgments**: The authors gratefully acknowledge the support of the project of MOBIDEM.

## References


[1] B.-C. Chen, C.-S. Chen, and W. H. Hsu. Face recognition and retrieval using cross-age reference coding with cross-age celebrity dataset. *IEEE Transactions on Multimedia*, 17(6):804–815, 2015.

[2] D. Chen, X. Cao, F. Wen, and J. Sun. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3025–3032, 2013.

[3] D. Cheng, Y. Gong, S. Zhou, J. Wang, and N. Zheng. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1335–1344, 2016.

[4] R. G. Cinbis, J. Verbeek, and C. Schmid. Unsupervised metric learning for face identification in tv video. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1559–1566. IEEE, 2011.

[5] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *13th International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[6] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision*, pages 87–102. Springer, 2016.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

[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 07-49, University of Massachusetts, Amherst, 2007.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[11] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[12] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[13] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *IEEE International Conference on Computer Vision*, pages 3730–3738, 2015.

[14] C. Lu and X. Tang. Surpassing human-level face verification performance on lfw with gaussianface. *arXiv preprint arXiv:1404.3840*, 2014.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[16] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[17] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman. Fisher vector faces in the wild. In *BMVC*, volume 2, page 4, 2013.

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

[19] J. Sivic, M. Everingham, and A. Zisserman. Person spotting: video shot retrieval for face sets. In *International Conference on Image and Video Retrieval*, pages 226–236, 2005.

[20] Y. Sun, Y. Chen, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.

[21] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1891–1898, 2014.

[22] Y. Sun, X. Wang, and X. Tang. Deeply learned face representations are sparse, selective, and robust. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2892–2900, 2015.

[23] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016.

[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[25] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[26] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[27] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515. Springer, 2016.

[28] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 529–534. IEEE, 2011.

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

[30] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[31] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *Signal Processing Letters*, 23(10):1499–1503, 2016.