Face Recognition via Centralized Coordinate Learning

Xianbiao Qi, Lei Zhang

Abstract—Owe to the rapid development of deep neural network (DNN) techniques and the emergence of large scale face databases, face recognition has achieved a great success in recent years. During the training process of DNN, the face features and classification vectors to be learned will interact with each other, while the distribution of face features will largely affect the convergence status of network and the face similarity computing in test stage. In this work, we formulate jointly the learning of face features and classification vectors, and propose a simple yet effective centralized coordinate learning (CCL) method, which enforces the features to be dispersedly spanned in the coordinate space while ensuring the classification vectors to lie on a hypersphere. An adaptive angular margin is further proposed to enhance the discrimination capability of face features. Extensive experiments are conducted on six face benchmarks, including those have large age gap and hard negative samples. Trained only on the small-scale CASIA Webface dataset with 460K face images from about 10K subjects, our CCL model demonstrates high effectiveness and generality, showing consistently competitive performance across all the six benchmark databases.

Index Terms—Face Recognition, Cross Age, Similar Looking, Large-scale Face Identification.

---◆---

1 INTRODUCTION

Face recognition [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12] has a broad range of applications in our daily life, including access control, video surveillance, public safety, online payment, image search and family photo album management. As a classical yet active topic, face recognition has been extensively studied. One essential problem in face recognition is how to obtain discriminative face features. In the past, handcrafted local image descriptors, such as SIFT [13], HOG [14], LBP [15] and its variants [16], [17], have been widely used to extract local face features. Meanwhile, Principal Component Analysis (PCA) [1], Linear Discriminative Analysis (LDA) [18], and Sparse Representation (SR) [19], [20], [21], [22], [23] are also popular methods to construct a global face representation.

Face recognition recently has made a breakthrough due to the rapid advancement of deep neural network (DNN) techniques, especially convolutional neural networks (CNNs) [9], [24], [25], [26], [27], [28], [29], and the availability of many large scale face databases [30], [31], [32], [33], [34], [35]. There are mainly two key issues in deep face recognition: designing effective network structure for face representation and constructing discriminative loss functions for feature learning. Some representative and successful deep CNNs for face recognition include Deep-Face [6], DeepIDs [7], [8], [36], [37], FaceNet [38], Deep FR [39], Center Loss [10], and SphereFace [12].

Given the network structure and optimization method, the design of loss functions largely determines the final recognition performance. Many discriminative loss functions [6], [7], [8], [10], [12], [36], [37], [38] have been proposed to provide effective supervision information for the DNN

Xianbiao Qi and Lei Zhang are with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China. Contact e-mails: (qixianbiao@gmail.com, cslzhang@comp.polyu.edu.hk).

training. Based on the different focuses of loss design, these works can be roughly categorized into three groups:

- Methods focusing on the form of final face feature, such as L_2-constrained face representation [40], L_2 Normface [41], and Coco loss [42].
- Methods investigating the importance of classification vector, such as Large-margin softmax loss [43] and SphereFace [12].
- Methods providing additional types of supervision information, such as Center loss [10] and Triplet loss in FaceNet [38].

During the training process of DNN, the face feature $\mathbf{x}$ and the classification vector $\mathbf{w}$ will interact with each other. The distribution of features obtained in one training stage will largely affect the output of classification vectors, which will in turn pull the samples of one specific person gathering together in one region of the coordinate space. Many previous works [12], [40], [41], [42], [43] focus on either the formulation of feature $\mathbf{x}$ or the formulation of classification vector $\mathbf{w}$, while they ignore the impact of the distribution of $\mathbf{x}$ on $\mathbf{w}$ as well as the final convergence of network. If in one training stage of DNN, most face features lie in a certain quadrant, the final features and their corresponding classification vectors will very likely converge to that quadrant. This will make the separation of different face subjects less effective.

In this paper, we consider simultaneously the formulations of feature $\mathbf{x}$ and classification vector $\mathbf{w}$. To make the faces of different subjects more separable, we argue that the learned face features should span dispersedly across the whole coordinate space centered on the origin, and the classification vectors should lie on a hypersphere manifold to make the cosine-similarity computation consistent in the training and test stages. To this end, we propose a simple yet effective method, namely centralized coordinate learning

(CCL), where we centralize the face features $\mathbf{x}$ to the origin of the space. As illustrated in Fig. 2(e), the centralization operator can make the features span more dispersedly across the whole coordinate space. Meanwhile, we introduce an adaptive angular margin (AAM) to further improve the face feature discrimination capability, as illustrated in Fig. 2(f).

The proposed CCL method demonstrates high effectiveness and generality. Trained only on the CASIA Webface [32], which has 460K face images from 10K subjects, CCL consistently shows competitive results with state-of-the-arts on six face benchmarks, including Labeled Face in the Wild (LFW) [30], Cross-Age LFW (CALFW) [44], Cross-Age Celebrity Database (CACD) [45], Similar-Looking LFW (SLLFW) [46], YouTube Face (YTF) [31] and MegaFace [34] datasets. Even without enforcing the proposed AAM operator, the learned CCL model still exhibits leading performance.

2 RELATED WORKS

In the study of DNN based face recognition, there are two key factors to the final performance: network structure and loss function. In the following, we briefly review the relevant works from these two aspects.

2.1 Network Structure

Motivated by the success of AlexNet [25] in ImageNet competition [47], researchers promptly adopted DNN in the field of face recognition. DeepFace [6] and DeepIDs [7], [8] are the first attempts to employ deep CNNs with a large amount data for face recognition. DeepFace adopts an AlexNet-like structure with similar input size, similar number of layers, large convolution kernel size, Rectified Linear Unit (ReLU) [48] and Dropout [49]. Comparatively, DeepIDs [7], [8] use a smaller network structure. DeepID1 directly trains an 8-layer network on 10K classes with input image size $39 \times 31 \times 1$, and DeepID2 combines the verification loss and identification loss and changes the network structure with input image size $55 \times 47 \times 3$. VGGFace [39] uses a bigger network structure of 19 layers with 2.6M training images. FaceNet [38] uses the deep GoogleNet as the base network and adopts a triplet loss function. The development of residual network (ResNet) [28] enables the training of extremely deep networks possible. SphereFace [12] makes use of a 64-layer ResNet with a newly proposed large angular margin loss. The Coco loss [42] applies a 128-layer ResNet for face recognition.

In DNN based face recognition, the selection of network structures often depends on the number and size of images as well as the computational power. There is no general agreement on which network structures should be used. Residual module and Google Inception module are the two most popular modules. In this work, we adopt the Google Inception_ResNet_V1 [29] model as our network structure due to its high efficiency and effectiveness.

2.2 Loss Function

Softmax Loss. Softmax loss [24], [25] is a standard multi-class classification loss function in DNN. It projects an input feature, learned by deep neural network, into a probability distribution. In softmax loss, the predicted posterior probability for the k-th class is defined as follows:

$$p_k = \frac{\exp(\mathbf{w}_k^T \mathbf{x} + b_k)}{\sum_{l=1}^{K} \exp(\mathbf{w}_l^T \mathbf{x} + b_l)}, \quad (1)$$

where $\mathbf{x}$ is the feature vector in the last layer, $\mathbf{w}_l$ and b_l are classification vector and bias of the l-th class, K is the total number of classes.

With the probability distribution of the feature $\mathbf{x}_i$ of the i-th input sample, a cross-entropy loss can be calculated as follows:

$$\mathcal{L}_s = \sum_{i}^{N} -\log(p_{y_i}), \quad (2)$$

where y_i is the label of the i-th sample and N is the total number of training samples.

The softmax loss has been used in DeepFace [6] and DeepID [7] for face recognition. When the training of the DNN is done, the loss function will be removed and only the trained feature extractor is used in the deployment stage.

Triplet Loss. Triplet loss was firstly introduced in FaceNet [38] to improve face recognition. It is originally proposed in the Large Margin Nearest Neighbor (LMNN) method [50]. Given a triplet $(\mathbf{x}_a, \mathbf{x}_p, \mathbf{x}_n)$ where $\mathbf{x}_a$ is the anchor sample, $\mathbf{x}_p$ denotes the positive sample, $\mathbf{x}_n$ represents the negative sample, the formulation of the triplet loss is defined as:

$$\mathcal{L}_t = \sum_{i \in N} \left[\|\mathbf{x}_{a_i} - \mathbf{x}_{p_i}\|_2^2 - \|\mathbf{x}_{a_i} - \mathbf{x}_{n_i}\|_2^2 + \epsilon \right]_+ ,$$

where $[\tau]_+$ denotes $\max(\tau, 0)$, ϵ is a preset margin, and N is the number of triplets. In triplet loss, the first term $\|\mathbf{x}_{a_i} - \mathbf{x}_{p_i}\|_2^2$ is to pull samples from the same class together, and the second term $-\|\mathbf{x}_{a_i} - \mathbf{x}_{n_i}\|_2^2$ is to push samples from different classes away.

Center Loss. Center loss [10] learns a center for each class by minimizing the distance between the sample features and their corresponding class centers. The definition of center loss is defined as follows:

$$\mathcal{L}_c = \sum_{i}^{N} \|\mathbf{x}_i - \mathbf{c}_{y_i}\|_2^2 ,$$

where $\mathbf{x}_i$ is the i-th sample feature vector, $\mathbf{c}_{y_i}$ is the center vector of the y_i-th class, and N is the number of samples. In practice, a weighted version of the softmax loss $\mathcal{L}_s$ and the above-mentioned center loss is usually used:

$$\mathcal{L}_{sc} = \mathcal{L}_s + \lambda \mathcal{L}_c ,$$

where λ is a balance parameter.

SphereFace Loss. Liu *et al.* [12] pointed out that the original softmax loss $\mathcal{L}_s$ in Eq. 2 is rational only for close-set problems, but have disadvantages for face recognition, which is an open-set problem[1]. They reformulated the original softmax loss into a modified softmax form:

---

1. In an open-set problem, unknown classes may occur in the test stage. In a close-set problem, all test classes are known in the training stage.

$$\mathcal{L}_{ms} = \sum_i^N -\log\left(\frac{\exp\left(\frac{\mathbf{w}_{y_i}^T \mathbf{x}_i}{\|\mathbf{w}_{y_i}\|}\right)}{\sum_{k=1}^K \exp\left(\frac{\mathbf{w}_k^T \mathbf{x}_i}{\|\mathbf{w}_k\|}\right)}\right).$$

The bias term b is removed and the weight $\mathbf{w}$ is normalized by its L_2 norm. The normalized classification vector $\frac{\mathbf{w}}{\|\mathbf{w}\|}$ will lie on a hypersphere. The modified softmax loss $\mathcal{L}_{ms}$ reduces the inconsistency of similarity measures in training and test stages brought by the bias term b and the magnitude of classification vector $\mathbf{w}$.

The modified softmax loss $\mathcal{L}_{ms}$ can be rewritten into an angular version:

$$\mathcal{L}_{as} = \sum_i^N -\log\left(\frac{\exp(\|\mathbf{x}_i\|\cos(\theta_{y_i,i}))}{\sum_{k=1}^K \exp(\|\mathbf{x}_i\|\cos(\theta_{k,i}))}\right), \quad (3)$$

which is called A-Softmax. Liu *et al.* further introduced an angular margin to A-Softmax:

$$\mathcal{L}_{sphere} = \sum_i^N -\log\left(\frac{\omega(\mathbf{x}_i, y_i)}{\omega(\mathbf{x}_i, y_i) + \psi(\mathbf{x}_i, y_i)}\right), \quad (4)$$

where

$$\omega(\mathbf{x}_i, y_i) = \exp(\|\mathbf{x}_i\|\cos(m\theta_{y_i,i})),$$
$$\psi(\mathbf{x}_i, y_i) = \sum_{k\neq y_i}\exp(\|\mathbf{x}_i\|\cos(\theta_{k,i})).$$

This angular margin version of A-Softmax is named as SphereFace [12]. The SphereFace loss provides a clear understanding on how the classification vector $\mathbf{w}$ affects face recognition; however, it does not discuss the importance of the distribution of $\mathbf{x}$.

Remarks. In addition to the above-mentioned loss functions, some recent works [40], [41], [42] impose an L_2-constraint on the features so that the feature vectors are restricted to lie on a hypersphere. The L_2-constrained features could be written as follows:

$$\hat{\mathbf{x}} = \alpha\frac{\mathbf{x}}{\|\mathbf{x}\|}, \quad (5)$$

where α is a preset or learnable parameter.

The L_2-constraint operator normalizes the features to a fixed hypersphere so that each sample will contribute equally to the final cross-entropy loss function for network updating. However, one possible problem is that the loss function will become sensitive to the choice of α. If α is small, the distribution of projected probabilities p_k via softmax operator in Eq. 1 will become very flat, reducing the discrimination of the cross-entropy loss function in Eq. 2. If α is large, the probabilities p_k will vary much, reducing the stability of network training.

Our work is inspired by the SphereFace loss. However, we observe that in the training process of a DNN, the final convergence status is largely dependent on the distribution of $\mathbf{x}$, and the similarity matching in the test stage is basically determined by $\mathbf{x}$ as well. Therefore, different from SphereFace which focuses on formulating $\mathbf{w}$, we aim to formulate simultaneously $\mathbf{x}$ and $\mathbf{w}$. In SphereFace, the angular margin is an extremely important factor for the final performance of face recognition. In contrast, our model works consistently well on different face recognition tasks even without using any angular margin.

3 CENTRALIZED COORDINATE LEARNING

We first present our formulation of the loss function, then introduce the centralized feature learning strategy. After that, we discuss the relation of our model to previous works. Finally, an adaptive angular margin is proposed to further improve the discriminative power of the learned features.

3.1 Formulation of Loss Function

In a standard DNN for classification, an inner product operator is employed in the softmax operation for probability projection (refer to Eq. 1). The inner product is defined as:

$$z = \mathbf{w}_k^T\mathbf{x} + b_k, \quad (6)$$

where $\mathbf{w}_k$ and b_k are the classification vector and the bias for the k-th class, and $\mathbf{x}$ is the output feature at the last layer of the neural network.

Both the features $\mathbf{x}$ and the classification vectors $\mathbf{w}$ are variables to be learned by the DNN. They are iteratively and alternatively updated in the training process. Given the optimization method and the hyper parameters, the convergence of the network mostly depends on the setting of $\mathbf{w}$ and the distribution of $\mathbf{x}$. Therefore, how to formulate the features $\mathbf{x}$ and the classification vectors $\mathbf{w}$ is crucial to the loss function design and consequently the learning outputs. An improper formulation of $\mathbf{w}$ may induce a classification gap between the training and test stages, while a less effective formulation of $\mathbf{x}$ may lead to less discriminative distribution of learned features. Researchers have proposed several schemes [12], [40], [41], [42], [43] to transform either $\mathbf{w}$ or $\mathbf{x}$ before performing the inner product. In this paper, we propose to transform both $\mathbf{w}$ and $\mathbf{x}$ for a more effective network training and face matching.

In open-set problems such as face recognition, the bias term b may make the face similarity computation inconsistent in the training and test stages. As in [12], [43], we disable this term in Eq. 6. We then introduce two transformations $\Psi(\cdot)$ and $\Phi(\cdot)$ on $\mathbf{w}$ and $\mathbf{x}$, respectively, resulting in the following inner product:

$$z = \Psi(\mathbf{w})^T\Phi(\mathbf{x}). \quad (7)$$

The choices of $\Psi(\cdot)$ and $\Phi(\cdot)$ will largely influence the softmax probability p_k of a sample, and thus impact the final loss function. Note that some recent works [12], [40], [41], [42], [43] can be written as Eq. 7 with specific forms of $\Psi(\cdot)$ and $\Phi(\cdot)$.

In [12], Liu *et al.* discussed the formulation of $\mathbf{w}$, and they let $\Psi(\mathbf{w}) = \frac{\mathbf{w}}{\|\mathbf{w}\|}$. Such an L_2 normalization eliminates the influence of the varying magnitude of $\mathbf{w}$ on face matching in the test stage. Note that in the training stage of DNN, we compute $z = \Psi(\mathbf{w})^T\Phi(\mathbf{x})$ to update the network, but in the test stage we often employ the cosine similarity to match two face feature vectors $\Phi(\mathbf{x}_1)$ and $\Phi(\mathbf{x}_2)$:

$$s = \frac{\Phi(\mathbf{x}_1)^T}{\|\Phi(\mathbf{x}_1)\|}\frac{\Phi(\mathbf{x}_2)}{\|\Phi(\mathbf{x}_2)\|}, \quad (8)$$

(a) The mean of r.
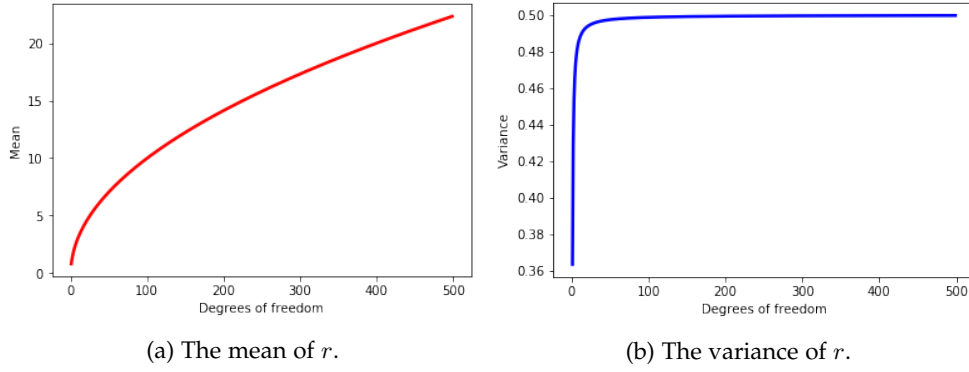

(b) The variance of r.

Fig. 1: Mean and variance of variable r (Eq. 12) with respect to its degrees of freedom.

where the trained $\mathbf{w}$ is not involved. Normalizing the classification vectors $\mathbf{w}$ in DNN training could avoid the case that some classification vectors with big magnitudes dominate the training of face features $\mathbf{x}$.

By setting $\Psi(\mathbf{w}) = \frac{\mathbf{w}}{\|\mathbf{w}\|}$ in Eq. 7, the softmax loss in Eq. 2 can be rewritten as:

$$\mathcal{L}_{sf} = \sum_i^N -\log\left(\frac{\exp(\frac{\mathbf{w}_{y_i}^T \Phi(\mathbf{x}_i)}{\|\mathbf{w}_{y_i}\|})}{\sum_{k=1}^K \exp(\frac{\mathbf{w}_k^T \Phi(\mathbf{x}_i)}{\|\mathbf{w}_k\|})}\right). \tag{9}$$

According to the definition of cosine similarity, Eq. 9 can be reformulated into an angular version as follows:

$$\mathcal{L}_{sf} = \sum_i^N -\log\left(\frac{\exp(\|\Phi(\mathbf{x}_i)\| \cos(\theta_{y_i,i}))}{\sum_{k=1}^K \exp(\|\Phi(\mathbf{x}_i)\| \cos(\theta_{k,i}))}\right), \tag{10}$$

where $\theta_{y_i,i}$ is the intersection angle between $\Phi(\mathbf{x}_i)$ and $\frac{\mathbf{w}}{\|\mathbf{w}\|}$, and the range of $\theta_{y_i,i}$ is $[0, \pi]$. One can see that $\|\Phi(\mathbf{x}_i)\|$ and $\cos(\theta_{y_i,i})$ will impact the loss function in Eq. 10, and both of them depend on the form of $\Phi(\mathbf{x}_i)$.

3.2 Centralized Feature Learning

From the analysis in Section 3.1, we can see that the formulation of $\Phi(\mathbf{x})$ will be the key factor to the success of face feature learning. On one hand, it largely affects the training of DNN via loss function in Eq. 10. If $\|\Phi(\mathbf{x}_i)\|$ is small, the softmax projected probabilities of all samples $\Phi(\mathbf{x}_i)$ will become similar so that the loss function will be less discriminative. If $\|\Phi(\mathbf{x}_i)\|$ is large, the probabilities may vary much and make the learning of DNN less stable. On the other hand, in the test stage, the cosine similarity of two face feature vectors, as given in Eq. 8, is computed for face recognition. Ideally, $\Phi(\mathbf{x})$ is expected to distribute dispersedly across the whole coordinate space so that two face feature vectors from different subjects can be more separable with big angles.

In this paper, we propose to centralize the face features to the origin of the space during the learning process. Specifically, for each dimension j of the feature vector $\mathbf{x}$, we define $\Phi(\mathbf{x}(j))$ as:

$$\Phi(\mathbf{x}(j)) = \frac{\mathbf{x}(j) - \mathbf{o}(j)}{\boldsymbol{\sigma}(j)}, \tag{11}$$

where $\mathbf{o} = \mathbb{E}[\mathbf{x}]$ is the mean vector of $\mathbf{x}$, and $\boldsymbol{\sigma}(j)$ is the standard deviation of $\mathbf{x}(j)$. Clearly, the transform $\Phi(\cdot)$ will centralize each dimension of $\mathbf{x}$ to the origin so that the features $\Phi(\mathbf{x})$ will span across all quadrants of the coordinate space. Meanwhile, each dimension of $\Phi(\mathbf{x})$ will have the same unit variance so that each dimension will contribute equally to the discrimination of faces instead of using only several strong dimensions for face recognition. Actually, several recent works [40], [41], [42] have been proposed to normalize $\mathbf{x}$ into a hypersphere as $\hat{\mathbf{x}} = \alpha \frac{\mathbf{x}}{\|\mathbf{x}\|}$. However, the normalization operator will not change the quadrant of the feature $\mathbf{x}$ and the selection of parameter α is not a trivial work.

Let's then analyze the L_2 norm of $\Phi(\mathbf{x})$, i.e., $\|\Phi(\mathbf{x})\|$. With the transform in Eq. 11, it is reasonable to assume that $\Phi(\mathbf{x}(j)), j = 1, 2, ..., D$, are i.i.d. variables and each variable follows a standard Gaussian distribution $\mathcal{N}(0, 1)$. Then the L_2 norm of $\Phi(\mathbf{x})$, defined as

$$r = \|\Phi(\mathbf{x})\| = \sqrt{\sum_{j=1}^D (\Phi(\mathbf{x}(j)))^2}, \tag{12}$$

will follow the χ-distribution with D degrees of freedom [51]. The mean and variance of r, denoted by μ_r and σ_r^2, are

$$\mu_r = \mathbb{E}[r] = \sqrt{2}\frac{\Gamma((D+1)/2)}{\Gamma(D/2)}, \tag{13}$$

$$\sigma_r^2 = \mathbb{V}\mathrm{ar}[r] = D - \mu_r^2, \tag{14}$$

where $\Gamma(\cdot)$ is a gamma function. Both μ_r and σ_r^2 are functions of D. In Figs. 1(a) and 1(b), we plot the curves of μ_r and σ_r^2 with respect to the degrees of freedom D.

As can be seen from Fig. 1(a), μ_r increases with the increase of D. Actually, μ_r is very close to $\sqrt{D}$. For example, when $D = 400$, $r = 19.987$. As we discussed before, a small $\|\Phi(\mathbf{x})\|$ will make the softmax projected probabilities of face samples similar to each other so that the discriminability will be reduced, while a large $\|\Phi(\mathbf{x})\|$ will reduce the stability of network training. Based on our experience, setting D between 350 and 400 will be a rational choice. (In our implementation, D is set to 374.) From Fig. 1(b), we can see that σ_r^2 is around 0.5 when D is larger than 20. It is more than one order smaller than μ_r. This is a

very desirable property because it implies that for most of the face samples, their corresponding $\|\Phi(\mathbf{x})\|$ values will not vary much. Therefore, each sample will approximately contribute equally to the final cross-entropy loss function for network updating.

Having analyzed the properties of the proposed transform $\Phi(\mathbf{x})$, now let us discuss how to learn the origin vector $\mathbf{o}$ and the standard deviation $\boldsymbol{\sigma}$ in Eq. 11. Given two face feature vectors $\mathbf{x}_1$ and $\mathbf{x}_2$, their cosine similarity by taking $\mathbf{o}$ as the coordinate origin is:

$$s_{\mathbf{o}}(\mathbf{x}_1, \mathbf{x}_2) = \frac{(\mathbf{x}_1 - \mathbf{o})^T(\mathbf{x}_2 - \mathbf{o})}{\|\mathbf{x}_1 - \mathbf{o}\| \|\mathbf{x}_2 - \mathbf{o}\|} \qquad (15)$$

The similarity between $\mathbf{x}_1$ and $\mathbf{x}_2$ is sensitive to the origin $\mathbf{o}$. For instance, let $\mathbf{x}_1$ be [0.1, 0.1, 0.2] and $\mathbf{x}_2$ be [0.2, 0.2, 0.1], if the origin is [0, 0, 0], the intersection angle between $\mathbf{x}_1$ and $\mathbf{x}_2$ is $35.3°$ and $s_{\mathbf{o}}(\mathbf{x}_1, \mathbf{x}_2) = 0.816$. If the origin moves a little to [-0.01, -0.01, -0.01], then the intersection angle becomes $33.1°$ and $s_{\mathbf{o}}(\mathbf{x}_1, \mathbf{x}_2) = 0.837$. Thus, in each update during DNN training, we should not change $\mathbf{o}$ and $\boldsymbol{\sigma}$ too much. We use a moving average strategy with a large decay factor to update $\mathbf{o}$ and $\boldsymbol{\sigma}$:

$$\begin{cases} \mathbf{o}_{new} &= \rho \cdot \mathbf{o}_{old} + (1 - \rho) \cdot \mathbf{o}_b, \\ \boldsymbol{\sigma}_{new} &= \rho \cdot \boldsymbol{\sigma}_{old} + (1 - \rho) \cdot \boldsymbol{\sigma}_b, \end{cases} \qquad (16)$$

where ρ is the decay factor, and $\mathbf{o}_b$ and $\boldsymbol{\sigma}_b$ are the mean and standard deviation vectors generated by the current mini-batch. In our implementation, ρ is set as 0.995 to ensure that $\mathbf{o}$ and $\boldsymbol{\sigma}$ do not change too much in each iteration.

3.3 Relation to Previous Works

Overall, the proposed formulation of centralized coordinate learning (CCL) can be written as:

$$z = \frac{\mathbf{w}^{\mathbf{T}}}{\|\mathbf{w}\|}\Phi(\mathbf{x}), \qquad (17)$$

where $\Phi(\mathbf{x})$ is defined in Eq. 11. In CCL, the normalization on classification vector $\mathbf{w} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$ is the same as SphereFace [12], which can be considered as a simplified version of *Weight Normalization* (WN) [52]. However, the normalization is only applied to the classification layer instead of each layer of the network as in WN.

As for the feature transformation $\Phi(\mathbf{x})$, it shares a similar form to the widely used *Batch Normalization* (BN) [53] if we add scaling and shifting terms to it:

$$\Phi(\mathbf{x}(j)) = \boldsymbol{\gamma}(j)\frac{\mathbf{x}(j) - \mathbf{o}(j)}{\boldsymbol{\sigma}(j)} + \boldsymbol{\beta}(j). \qquad (18)$$

However, BN is applied to each layer after the convolution operation, while CCL is only applied to the last classification layer before softmax operation. Furthermore, the parameters $\boldsymbol{\gamma}(j)$ and $\boldsymbol{\beta}(j)$ in BN may destroy the advantages of $\Phi(\mathbf{x})$ analyzed in Section 3.2. We will make more discussions on this Section 4.2.

3.4 Adaptive Angular Margin

Angular margin was firstly introduced in L-Softmax [43] and SphereFace [12] to make the classification boundary more compact. It has proved to be an effective way to further improve face recognition performance. However, the angular margin function introduced in [12], [43] is difficult to train and is sensitive to parameters. To ease this issue, we propose a simple adaptive angular margin (AAM) function as follows:

$$\mathcal{L}_{AAM} = \sum_i^N -\log(p_{y_i}^{AAM}), \qquad (19)$$

where

$$p_{y_i}^{AAM} = \frac{\exp(\|\Phi(\mathbf{x}_i)\|\cos(\eta\theta_{y_i,i}))}{\exp(\|\Phi(\mathbf{x}_i)\|\cos(\eta\theta_{y_i,i}) + \sum_{k \neq y_i}\exp(\|\Phi(\mathbf{x}_i)\|\cos(\theta_{k,i}))}, \qquad (20)$$

where η is an adaptive parameter and it is set based on the value of $\theta_{y_i,i}$:

$$\eta = \begin{cases} 1, & \pi/3 < \theta_{y_i,i} <= \pi; \\ \frac{\pi/3}{\theta_{y_i,i}}, & \pi/30 < \theta_{y_i,i} <= \pi/3; \\ 10, & \theta_{y_i,i} <= \pi/30. \end{cases} \qquad (21)$$

As shown in Eq. 21, we partition the range of $\theta_{y_i,i}$ into three intervals, based on which η is set. When $\pi/3 < \theta_{y_i,i} <= \pi$, the angle is big enough to provide enough gradient information for back-propagation, thus, adding a stronger angular margin is not necessary. For example, if $\theta_{y_i,i} = \frac{2\pi}{3}$ and η is set to 2, then the quadrant will jump from the second quadrant ($\frac{2\pi}{3}$) to the third quadrant ($\frac{4\pi}{3}$), which will bring in wrong gradient information. Therefore, when $\pi/3 < \theta_{y_i,i} <= \pi$, we set $\eta = 1$. When $\pi/30 < \theta_{y_i,i} <= \pi/3$, the angle is relatively small and it is necessary to introduce additional angular margin, and we set $\eta = \frac{\pi/3}{\theta_{y_i,i}}$, i.e., the smaller the angle $\theta_{y_i,i}$, the more angular margin we introduce. When $\theta_{y_i,i} <= \pi/30$, we fix the parameter $\eta = 10$ since a too big η may make the back-propagation vibrate too much.

The AAM loss defined above can pull the face feature vectors from the same subjects more compactly distributed, but using it alone to train the DNN may not be stable. In practice, we leverage a weighted version of the softmax loss in Eq. 10 and the AAM loss in Eq. 19 for training:

$$\mathcal{L} = \frac{\lambda\mathcal{L}_{sf} + \mathcal{L}_{AAM}}{\lambda + 1.0}, \qquad (22)$$

where λ is a constant to balance $\mathcal{L}_{sf}$ and $\mathcal{L}_{AAM}$. In our implementation, we empirically set $\lambda = 3$.

In Fig. 2, we illustrate the differences of different loss functions on the convergence of face features, including the original softmax loss $\mathcal{L}_s$ in Eq. 2, the A-Softmax loss in Eq. 3, the SphereFace loss in Eq. 4, the CCL loss $\mathcal{L}_{sf}$ in Eq. 10, and the loss of CCL with AAM in Eq. 19. From Fig. 2, we have the following comments:

- As shown in Fig. 2(b), the original softmax loss $\mathcal{L}_s$ may make the cosine similarity of face vectors inconsistent in the training and test stages, due to the bias term b
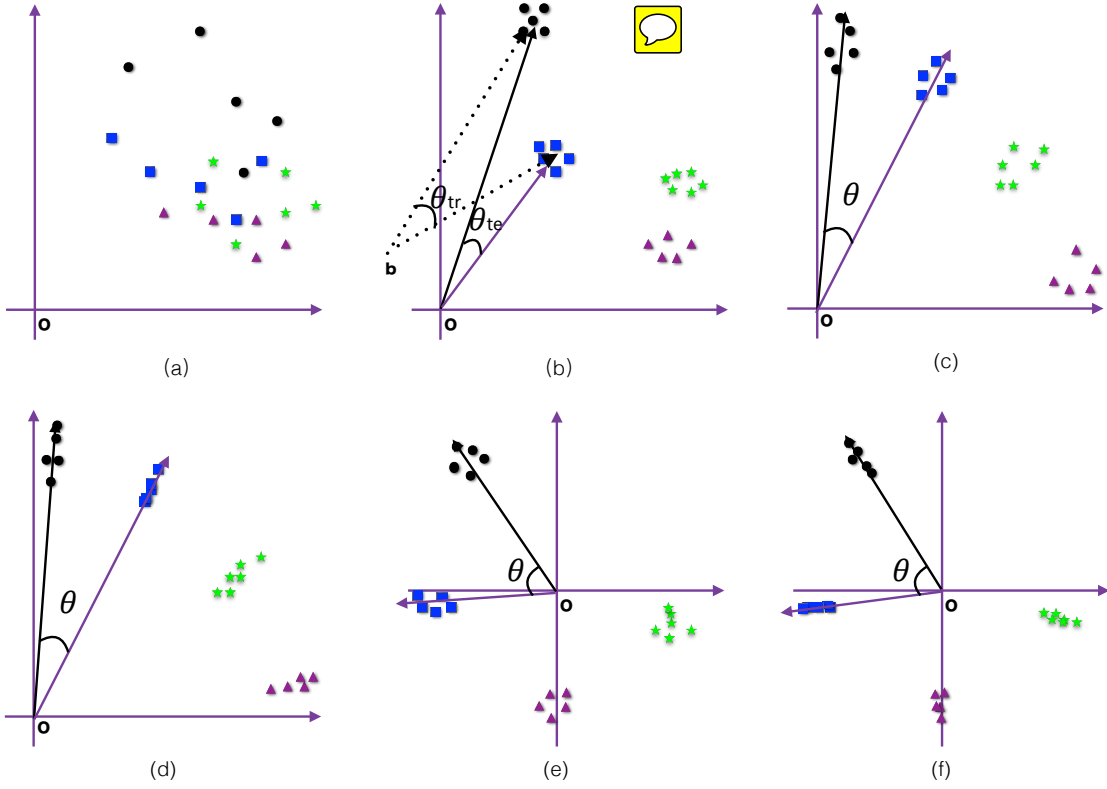
Fig. 2: Illustration of the effects of different loss functions. The "black dots", "blue squares", "green stars" and "pink triangles" represent samples of four face classes. (a). Original data distribution. (b). Converged face features by using the original softmax loss. (c). Converged face features by using the A-Softmax loss. (d) Converged face features by using the SphereFace loss. (e) Converged face features by using the CCL loss. (f). Converged face features by using the CCL loss with AAM.

and the different magnitudes of $\mathbf{w}$ (i.e., the lengths of different class centers can be very different). The angle θ_{tr} between two classes (denoted by the black circle and blue square, respectively) in the training stage may be different from the angle θ_{te} in the test stage.

- The A-Softmax loss can eliminate this similarity inconsistency in training and test stages by removing the bias term b and normalizing the magnitude of classification vectors $\mathbf{w}$. As shown in Fig. 2(c), the centers of different classes lie nearly in a unit hypersphere. The SphereFace loss, by introducing a large angular margin, can enforce the face samples from the same subject to get closer to each other, as shown in Fig. 3(d). This will enhance the discrimination of face features.

- Suppose that the original face features lie in the first quadrant, as shown in Fig. 2(a), the softmax loss, A-Softmax loss and SphereFace loss will pull each class of face images to its center, while push different classes away from each other. However, most of the face features will still lie in the first quadrant. By using the proposed CCL based loss, the learned face features will be centralized to a common origin so that they will span across all the four quadrants. Intuitively, such a disperse distribution of face features tends to make the neighboring classes have larger angles, improving their separability, as illustrated in Fig. 2(e). With the proposed AAM, the intra-class variations can be further reduced,

as shown in Fig. 2(f). In this way, the AAM can further improve the discrimination capability of face features.

4 EXPERIMENTS

We evaluate our method on six face datasets: Labeled Face in the Wild (LFW) [30], Cross-Age Celebrity Dataset (CACD) [45], Cross-Age LFW (CALFW) [44], Similar-Looking LFW (SLLFW) [46], YouTube Face (YTF) [31] and MegaFace [34]. On each dataset, we compare our method with the state-of-the-art results reported in literature.

4.1 Experimental Details

Training data. In this paper, we use only the CASIA Web-Face dataset [32] to train our CCL model. CASIA Web-Face consists of 494,414 images of 10,575 subjects and it is widely used to train DNNs. In the original dataset, there are some annotation errors. A corrected version of CASIA Webface was later released, which has 455,594 images of 10,575 subjects. Compared to some previous works, such as DeepFace [6] (4M), VGGFace [39] (2M), FaceNet [9] (200M) and Coco loss [42] (half MS-Celebrity), the scale of CASIA WebFace (around 0.46M) is small. In addition, the number of face images in each class is uneven. Some classes have several hundreds of images, while some classes have only around 10 images. Regardless of the small scale and uneven distribution of face images in the CASIA WebFace dataset,

TABLE 1: Inception_ResNet_V1 network structure used in this paper.

Layer	size-in	size-out	kernel	stride,padding	params	ReLU_fn	scale
Conv_BN_ReLU	$160 \times 160 \times 3$	$79 \times 79 \times 32$	$3 \times 3 \times 3$	$2, 0$	$32 \times 3 \times 3 \times 3$	True	-
Conv_BN_ReLU	$79 \times 79 \times 32$	$77 \times 77 \times 48$	$3 \times 3 \times 32$	$1, 0$	$48 \times 3 \times 3 \times 32$	True	-
Conv_BN_ReLU	$77 \times 77 \times 48$	$77 \times 77 \times 64$	$3 \times 3 \times 48$	$1, 1$	$64 \times 3 \times 3 \times 48$	True	-
MaxPool2D	$77 \times 77 \times 64$	$38 \times 38 \times 64$	3×3	$2, -$	0	True	-
Conv_BN_ReLU	$38 \times 38 \times 64$	$38 \times 38 \times 80$	$1 \times 1 \times 64$	$2, 0$	$80 \times 1 \times 1 \times 64$	True	-
Conv_BN_ReLU	$38 \times 38 \times 80$	$36 \times 36 \times 192$	$3 \times 3 \times 80$	$1, 0$	$192 \times 3 \times 3 \times 80$	True	-
Conv_BN_ReLU	$36 \times 36 \times 192$	$17 \times 17 \times 256$	$3 \times 3 \times 192$	$2, 0$	$256 \times 3 \times 3 \times 192$	True	-
$5\times$ Inception_A	$17 \times 17 \times 256$	$17 \times 17 \times 256$	Inception_A	-	-	True	0.17
Reduction_A	$17 \times 17 \times 256$	$8 \times 8 \times 896$	Reduction_A	-	-	True	-
$10\times$ Inception_B	$8 \times 8 \times 896$	$8 \times 8 \times 896$	Inception_B	-	-	True	0.1
Reduction_B	$8 \times 8 \times 896$	$3 \times 3 \times 1792$	Reduction_B	-	-	True	-
$5\times$ Inception_C	$3 \times 3 \times 1792$	$3 \times 3 \times 1792$	Inception_C	-	-	True	0.2
Inception_C	$3 \times 3 \times 1792$	$3 \times 3 \times 1792$	Inception_C	-	-	False	1.0
AvgPool2d	$3 \times 3 \times 1792$	$1 \times 1 \times 1792$	3×3	-	0	-	-

it is a good platform to evaluate the effectiveness of a face learning algorithm without the influence of large-scale training data.

Face Detection and Face Alignment. Face detection [54], [55], [56], [57] is an important step for the following face recognition and analysis. Following previous works [10], [12], [43], in this paper we use MTCNN [54] for face detection in both training and test stages. In addition to providing the location of face, MTCNN also provides the positions of five face landmarks (the center of left eye, the center of right eye, nose, left mouth corner and right mouth corner). After the original image is inputted to the MTCNN face detector, if no face is detected in the image, we upsample the image by a factor of 2 and apply face detector again. If still no face is detected, a 182×182 region is cropped from the center of the image as the detection output.

Commonly used face alignment is to compute an affine transformation between the three landmarks detected by MTCNN and the three corresponding landmarks provided by a pre-defined face template such as OpenCV dlib library. However, simple affine transformation with three points correspondence may twist the profile faces. In this paper, we align the eyes to a horizontal line and make the center of three landmarks (the nose, the left mouth corner and the right mouth corner) in the middle of the image. A 182×182 image is cropped from the center of the aligned image. (Note that all face images in figures shown in this paper are cropped images with size 182×182.)

Preprocessing and Data Augmentation. Each pixel of the cropped face image is normalized to range $[-1, 1]$ by first subtracting 127.5 and then dividing by 127.5. In the training stage, an online data augmentation strategy is used. For each image, a sub-image of size 160×160 is randomly cropped from the pre-aligned 182×182 image. These face images are randomly horizontally flipped before being inputted to the network.

Network Structure. The proposed CCL method is implemented in PyTorch [58]. The Inception-ResNet network with a similar structure to the open FaceNet[2] implementation is employed as the base network. Detailed information of our base network is listed in Table 1, where "scale" refers to the scale factor used in residual block as $\mathbf{x} = \mathbf{x}+\text{scale} *\mathbf{f}(\mathbf{x})$. ReLU_fn denotes where ReLU is applied in the last

operator. For each image, we input itself and its horizontally flipped image into the based network, and obtain two 1,792-d feature vectors $\mathbf{f}$ and $\mathbf{f_{flip}}$ after the AvgPool2d operator as shown in Table 1. We concatenate the two features using two different concatenation forms as $[\mathbf{f}, \mathbf{f_{flip}}]$ and $[\mathbf{f_{flip}}, \mathbf{f}]$ to obtain two 3,584-d features. These two features are embedded into two individual 374-d features with the same embedding weight matrices. Finally, CCL operator is applied to both of them. In the training stage, the two features will lead to two individual loss values.

Learning Strategy. We set the batch size as 128. The learning rate starts from 0.1, and then is divided by 10 at the 80K-th and 110K-th iterations. The total number of iterations is 150K. Weight decay is set to 0.0002. In all convolution layers, the bias term is disabled. It takes around 36 hours to finish 150K iterations on a Titan X Pascal GPU.

Test settings. In the test stage, we input the 160×160 image cropped from the center of the pre-aligned image and its corresponding flipped image to the trained network, and generate two 374-d feature vectors. These two feature vectors are averaged and then normalized into a unit-length vector. The similarity score between two face images is computed by the cosine distance. Threshold comparison is used in face verification.

4.2 Exploratory Evaluation

4.2.1 Evaluation of Centralized Coordinate Learning

To better illustrate the effectiveness of the proposed CCL strategy, here we conduct experiments to evaluate the performance of original linear embedding (LE) (i.e., only full-connected layer is used before softmax operation), LE with BN, LE with BN but without β, and LE with CCL (CCL in short). The benchmark LFW database is used for the comparison experiments. The detailed information about LFW can be found in Section 4.3.

TABLE 2: Accuracy (%) on the LFW dataset.

Method	LFW
LE	98.367
LE with BN	99.133
LE with BN (no β)	99.217
CCL	99.467

2. https://github.com/davidsandberg/facenet

The experimental results are shown in Table 2. One can see that CCL significantly outperforms LE, improving its accuracy from 98.367% to 99.467%. We can also see that CCL performs better than LE with BN. We observe that LE with BN (with γ but without β) outperforms LE with original BN (with both γ and β), while both of them are behind CCL. We believe that the parameters γ and β in BN will bring certain negative effects on the distribution of features $\mathbf{x}$, which thus affect the final performance of instance-level face recognition problem.

4.2.2 Evaluation of Adaptive Angular Margin

We then evaluate the effectiveness of the proposed AAM (refer to Eq. 20) by learning CCL models with and without AAM loss. We compare our methods with SphereFace with and without angular margins. The experimental results on LFW database are listed in Table 3.

TABLE 3: Accuracy (%) of CCL with and without adaptive angular margin on LFW dataset.

Method	Accuracy
SphereFace without angular margin	97.88
SphereFace with angular margin ($m = 1$)	97.90
SphereFace with angular margin ($m = 2$)	98.40
SphereFace with angular margin ($m = 3$)	99.25
SphereFace with angular margin ($m = 4$)	99.42
CCL	99.467
CCL with AAM	99.583

From Table 3, it can be observed that the angular margin plays an important role in SphereFace. Without angular margin, SphereFace only has an accuracy of 97.88%. With angular margin, the performance of SphereFace can be improved to 99.42% ($m = 4$). In comparison, our method, even without using angular margin, outperforms the best results of SphereFace with strong angular margin. AAM can further improve the performance of CCL on LFW. The difficulties of training DNN with angular margin will increase with the number of output classes because angular margin enforces harder conditions for the loss function. We also found that enforcing a strong angular margin sometimes makes the network fail to converge. Therefore, how to apply angular margin is still a worth pondering problem.

4.3 Experiments on Six Benchmarks

Experiments on LFW. The LFW dataset [30] consists of 13,233 web-collected images from 5,749 different identities. Only 85 persons have more than 15 images, while 4,069 persons have only one image. There are 6,000 face pairs, including 3,000 positive pairs and 3,000 negative pairs. The 6,000 face pairs are divided into ten subsets, each having 300 positive pairs and 300 negative pairs. Note that the negative pairs are selected randomly. The images have different kinds of variations in pose, expression and illuminations. Some sample image pairs from LFW are shown in Fig. 3.

Following the standard protocol of unrestricted with labeled outside data, we test CCL on 6,000 face pairs in comparison with the state-of-the-art methods. The experimental results are reported in Table 4, from which we have the following observations:

Fig. 3: Sample images from the LFW dataset. The first two rows show six positive pairs, and the last two rows show six negative pairs.

TABLE 4: Accuracy (%) on the LFW dataset. ("*" denotes the ensemble of 25 models.)

Method	Training Images	Accuracy
DeepID2+ [36]	0.3M	98.70
DeepID2+ (25) [36]	0.3M*	99.47
DeepFace [6]	4M	97.35
Center Loss [10]	0.7M	99.28
UP loss [59]	1.2M	98.88
Marginal Loss [60]	4M	98.95
Noisy Softmax [61]	WebFace+	99.18
Range Loss [62]	1.5M	99.52
FaceNet [38]	200M	**99.65**
Coco Loss [42]	half MS-Cele [33]	**99.86**
Softmax Loss	CASIA	97.88
Center Loss	CASIA	99.05
Marginal Loss	CASIA	98.95
L_2 NormFace [41]	CASIA	99.20
L-Softmax [43]	CASIA	99.10
ReST [63]	CASIA	99.05
SphereFace [12]	CASIA	**99.42**
CCL	CASIA (0.46M)	**99.467**
CCL with AAM	CASIA (0.46M)	**99.583**

- The proposed CCL with AAM, trained on only the CASIA dataset with 0.46M samples, outperforms most state-of-the-art models, including those trained on much larger scale of data or the ensemble of multiple models. Its accuracy is only slightly lower than FaceNet [38] and Coco Loss [42], which are trained with 200M samples and half MS-Cele data (about 2M) [33], respectively.
- Compared with those models trained on the same CASIA dataset, including Center loss, L-Softmax, L_2 NormFace, and SphereFace, the proposed CCL models achieve the best accuracy, no matter the AAM loss is used or not.

Experiments on Cross-Age Celebrity Dataset. The CACD dataset [45] is a face dataset for age-invariant face recognition, containing 163,446 images from 2,000 celebrities with labelled ages. It includes varying illumination, pose varia-

Fig. 4: Sample images from the CACD dataset. The first two rows show six positive pairs, and the last two rows show six negative pairs.

tion, and makeup to simulate practical scenarios. However, the CACD dataset contains some incorrectly labelled samples and some duplicate images. Following the state-of-the-art configuration [60], [64], we test the proposed method on a subset of CACD, called CACD-VS. The CACD-VS consists of 4,000 image pairs (2,000 positive pairs and 2,000 negative pairs) and has been carefully annotated. The 4,000 image pairs are divided into ten folds. Identities in each fold are mutually exclusive. Sample images from the CACD-VS are shown in Fig. 4. We follow the ten-folds cross-validation rule to compute the face verification rate, and compare CCL with the existing methods on this dataset. The results are listed in Table 5. It should be noted that human performance on the CACD-VS is reported using Amazon Mechnical Turks.

TABLE 5: Accuracy (%) on the CACD dataset.

Method	Training Images	Accuracy
High-Dimensional LBP [65]	N/A	81.6
Hidden Factor Analysis [66]	N/A	84.4
Cross-Age Reference Coding [64]	N/A	87.6
LF-CNNs [67]	N/A	98.5
Human Average [64]	N/A	85.7
Human Voting [64]	N/A	94.2
Centre Loss [60]	CASIA	97.475
Marginal Loss [60]	4M	98.95
CCL	CASIA (0.46M)	**99.225**
CCL with AAM	CASIA (0.46M)	**99.175**

From Table 5, we have the following observations:

- CCL outperforms all the published results on this dataset, even surpassing the human-level performance by a clear margin. It achieves an accuracy of 99.225%, while that of human voting is only 94.2%. CCL with AAM has similar performance to CCL on this dataset.
- CCL shows strong robustness to age variations, though it is trained on CASIA WebFace whose data do not explicitly contain large age variation.
- The amount of our training data (0.46M) is significantly smaller than that (4M) used in Marginal Loss [60], yet our method achieves better accuracy. This validates the high effectiveness of our learning model.

To some extent, the CACD-VS is a good benchmark to evaluate the robustness of an algorithm to age variation. However, we observe that the average age gap between positive pairs in CACD-VS is not large enough, and the negative pairs are randomly selected, which are not hard enough either. Therefore, we further evaluate our method on a harder dataset, the Cross-Age LFW (CALFW) [44], which has larger age gap.

Fig. 5: Example positive pairs from the CALFW dataset. The first two rows have six positive male pairs and the last two rows consist of six positive female pairs.

Experiments on Cross-Age LFW Dataset. The CALFW dataset [44] is built to evaluate face verification algorithms under large age gap. It contains 4,025 individuals with each person having 2, 3, or 4 images. Similar to the original LFW dataset, CALFW defines 10 individual subsets of image pairs. Each subset has 300 positive pairs and 300 negative pairs. These 10 subsets are constructed according to their identities to ensure that each identity only occurs in one subset. Sample image pairs are shown in Fig. 5. One can see that there are very large age gaps between positive pairs. The age gap ranges from several years to 60 years. Large age gaps of positive pairs further increase intra-class variations. Meanwhile, only negative pairs with the same gender and race are selected to reduce the influence of attribute difference between positive/negative pairs. Overall, this dataset is very challenging because of the large age gap and hard negative pairs.

TABLE 6: Accuracy (%) on the CALFW dataset. ("+" denotes data expansion.)

Method	Training Images	Accuracy
SVM	N/A	65.27
ITML	N/A	68.82
KISSME	N/A	67.87
VGGFace [39]	2.6M	86.50
Noisy Softmax [61]	CASIA+	82.52
CCL	CASIA (0.46M)	**91.15**
CCL with AAM	CASIA (0.46M)	**90.83**

In [56], Zhang et al. used Dex [68] to estimate the age of each image, and then calculated the age gaps in both LFW and CALFW. The average age gaps of positive pairs and negative pairs are 4.94 and 14.85, respectively, on LFW,

Fig. 6: Example negative pairs from the SLLFW dataset. The first two rows show six male negative pairs and the last two rows consist of six female negative pairs. Some pairs are even hard for human to differentiate.

and 16.61 and 16.14, respectively, on CALFW. We compare our method with VGGFace [38] and Noisy Softmax [59], and the results are listed in Table 6. One can see that our method significantly outperforms its counterparts by a large margin. It surpasses Noisy Softmax by more than 8%. Although the verification accuracy on the original LFW is almost saturated, the performance on the CALFW is not good enough. There still has a long way to go for cross-age face recognition.

Experiments on Similar-Looking LFW Dataset. Conventional face verification addresses mainly large intra-class variations, such as pose, illumination, and expression. Zhang *et al.* [44] carefully inspected the LFW dataset. They found that the main reason for the saturated performance on LFW is that almost all negative pairs are rather easy to distinguish. They pointed out that the negative pairs were randomly selected from different individuals, and usually two randomly selected individuals will have large differences in appearance, even have different genders. In practice, however, when the gallery is large, there will be many similar-looking people as the query faces. To simulate this situation, Zhang *et al.* [44] designed the Similar-Looking LFW (SLLFW) dataset, where 3,000 similar-looking face pairs were deliberately selected from the original LFW image gallery by human crowdsourcing instead of random negative pairs selection. Fig. 6 shows some negative pairs in SLLFW. One can see that the negative pairs look very similar in gender, race, age and appearance.

TABLE 7: Accuracy (%) on the SLLFW dataset. ("+" denotes data expansion.)

Method	Training Images	Accuracy
DeepFace [6]	0.5M	78.78
DeepID2 [8]	0.2M	78.25
VGGFace [39]	2.6M	85.78
DCMN1 [46]	0.5M	91.00
Noisy Softmax [61]	CASIA+	94.50
Human	N/A	92.0
CCL	CASIA (0.46M)	**95.68**
CCL with AAM	CASIA (0.46M)	**96.43**

We evaluate our method on SLLFW and compare it with the state-of-the-art methods. The results are listed in Table 7. Our CCL methods show superior performance to the other approaches. Specifically, CCL with AAM improves the Noisy Softmax by nearly 2% using less training data. Compared with the original LFW, the performance on SLLFW drops a lot. This proves that with the increased difficulty in negative pairs, all methods will become less accurate. There is much room to improve on SLLFW.

Fig. 7: Example pairs from the YouTube face dataset. Images in each row come from one video. The first two rows are a positive pair, and the last two rows are a negative pair.

Experiments on YouTube Face Dataset. The YTF [31] dataset consists of 3,425 videos of 1,595 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 per clip. An essential challenge of YTF dataset is the low resolution of face images. In many videos, the size of face regions is around 40×40. Some sample images are shown in Fig. 7. One can see that some detected faces are very blurry. In contrast, the detected face regions in the LFW dataset are mostly larger than 120×120. The YTF has 5,000 video pairs which are divided into 10 subsets. Each subset has 500 pairs: 250 positive pairs and 250 negative pairs.

TABLE 8: Accuracy (%) on the YouTube Face dataset. ("*" denotes the ensemble of 25 models.)

Method	Training Images	Accuracy
DeepID2+ (25) [36]	0.3M*	93.2
DeepFace [6]	4M	91.4
FaceNet [38]	200M	95.1
Center Loss [10]	0.7M	94.9
Range Loss [62]	1.5M	93.70
Deep FR [39]	2.6M	**97.3**
ReST [63]	CASIA	**95.4**
Softmax Loss	CASIA	93.1
Softmax + Constrastive	CASIA	93.5
L_2 NormFace [41]	CASIA	94.24
L-Softmax [43]	CASIA	94.0
Center Loss	CASIA	94.4
SphereFace [12]	CASIA	95.0
CCL	CASIA (0.46M)	94.96
CCL with AAM	CASIA (0.46M)	**95.28**

We follow the unrestricted with labeled outside data protocol and report the results on 5,000 video pairs in Table 8. One can see that our method, trained only on the CASIA dataset, outperforms many state-of-the-art methods, including DeepFace, FaceNet and Center loss which are trained on larger dataset. Using the same CASIA training data, CCL with AAM is only slightly lower than ReST [63] and outperforms all the other competitors.

Experiments on MegaFace. The MegaFace dataset [34] is a recently released highly challenging benchmark to evaluate the performance of face recognition methods at the million scale of distractors. The MegaFace evaluation sets consist of a gallery set and a probe set. The gallery set, as a subset of Flickr photos, consists of more than one million images from more than 690K individuals. The probe set descends from two existing databases: Facescrub and FGNet. The Facescrub dataset [69], which includes 100K photos of 530 celebrities, is available online. It has a similar number of male and female photos (55,742 photos of 265 males and 52,076 photos of 265 females) and a large variation across photos of the same individual. The probe set used in this paper consists of 3,530 images of 80 persons provided by the official MegaFace organizer.

To avoid any bias to the experimental result, we use exactly the same code to process both the probe set and the gallery set (face detection and face alignment). The experimental results are shown in Table 9, where "Large" means that the mount of used training images is larger than 500K. We can make the following observations.

TABLE 9: Accuracy (%) of different methods using Facescrub as the probe set on MegaFace with 1M distractors.

Method	Training Images	Accuracy
Coco Loss [42]	Large	**76.57**
NTechLAB - facenx large	Large	73.300
Vocord - DeepVo1	Large	75.127
Deepsense-large	Large	74.049
Shanghai Tech	Large	74.799
Google - FaceNet v8	Large	70.496
Beijing FaceAll_Norm_1600	Large	64.804
Beijing FaceAll_1600	Large	63.977
Deepsense-small	Small	70.983
SIAT_MMLAB	Small	65.233
barebones FR	Small	59.036
NTechLAB-facenx_small	Small	66.366
3DiVi Company-tdvm6	Small	36.927
Softmax Loss [12]	Small	54.855
Softmax+Contrastive [12]	Small	65.219
Triplet Loss [12]	Small	64.797
L-Softmax Loss [12]	Small	67.128
Softmax+Center Loss [12]	Small	65.494
SphereFace [12]	Small	**72.729**
CCL	Small	72.572
CCL with AAM	Small	**73.743**

- Among the methods which are trained on small scale data, the proposed CCL with AAM achieves the best accuracy. It largely outperforms Softmax loss, L-Softmax loss, Triplet loss, and Center Loss + Softmax loss. Without using AAM, CCL achieves similar performance

(72.572%) to SphereFace (72.729%) with large angular margin ($m = 4$). By using AAM, CCL outperforms SphereFace by about 1%.
- The proposed CCL with AAM also outperforms many methods trained on large scale data. The Coco Loss [41] method achieves the accuracy of 76.57%, about 2.8% higher than CCL with AAM, but it uses half MS-Celebrity (about 2M images) in training, while our methods uses only 0.46M images from 10,575 classes in training.

4.4 Discussions

Discussions about LFW, CACD, YTF, SLLFW and CALFW data sets. From our experiments in Section 4.3, one can see that the face recognition performance on different datasets varies much. We first compare the LFW, CACD, CALFW, SLLFW and YTF datasets by computing the statistics of similarity scores of positive pairs and negative pairs, respectively. The similarity distributions on the five datasets are drawn in Fig. 8. We can see that there are clear boundaries between the similarity distribution of positive pairs and the similarity distribution of negative pairs on the LFW and CACD datasets, which explains why the performance on these two datasets is already saturated. The two distribution curves have some overlaps on the YTF and SLLFW datasets, which implies that more efforts should be made to further improve the performance on these two datasets. For the CALFW dataset, the two distribution curves overlap much. This is mainly caused by the large age gap of face images from the same subject. Some positive pairs have lower similarity scores, and few positive pairs have a similarity score close to 1.0. CALFW is a challenging dataset to evaluate the cross-age face recognition algorithms.

Discussions about MegaFace dataset. Compare with the above 5 datasets, MegaFace is much bigger in scale and it also has a different yet more challenging test protocol. To visualize the results on MegaFace, we conduct a retrieval experiment. Each time, we select two images of one subject from the Facascrub probe set (3,540 images), one used as "Query" image and the other one used as "Target" image. The target image is mixed with the gallery set which has 1,027,060 images. We use the query image to retrieval the ten most similar faces from the 1,027,061 images. Some retrieval results are shown in Fig. 9.

Some observations can be made from the retrieval results in Fig. 9. In particular, it is very hard to decide whether some retrieval results, marked with purple rectangles, are truly from the same people as the query images. Since the gallery set is very large and the probe set is collected all from celebrities, the gallery set actually contain many images from the identities appearing in the probe set. Take the last two rows for example, the query images are from "Daniel Jacob Radcliffe" and "Jackie Chan"; however, we can find that the gallery set contain multiple images coming from the same identities. According to the "Rank 1" criterion, the retrieval results for "Daniel Jacob Radcliffe" and "Jackie Chan" are wrong, but we can observe that the top 4 results for "Daniel Jacob Radcliffe" and "Jackie Chan" are actually correct. The distractors in the gallery set do not exclude the images with the same subject as the query subject. Such a fact will affect the veracity of the accuracy criterion.
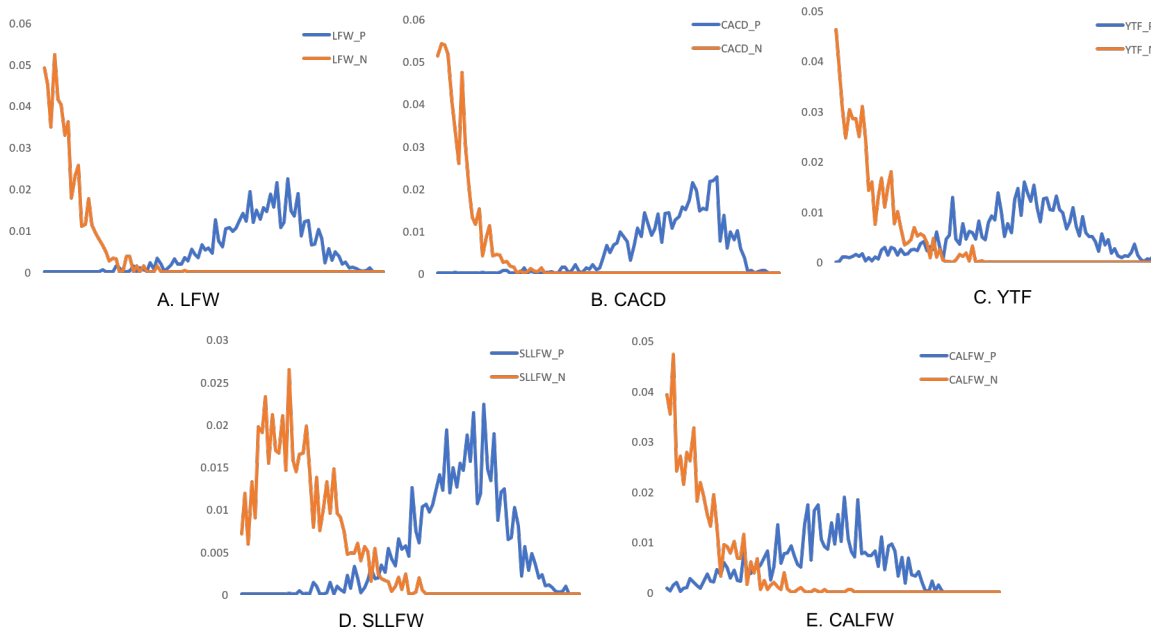
Fig. 8: Histogram distributions of similarity scores on LFW, CACD, YTF, SLLFW, and CALFW data sets. Similarities of positive pairs are marked blue color and similarities of positive pairs are marked orange color.

5 CONCLUSION

This paper presented a simple yet effective face feature learning method. With the help of deep convolutional neural networks (CNNs), we argued that a good face feature learner should push the face samples dispersedly distributed across the coordinate space centering on the origin so that the angles between different classes can be enlarged. Meanwhile, the classification vectors should lie on a hypersphere space to remove the influence of their varying magnitudes. To achieve this goal, we normalized the classification vector by its L_2 norm, and centralized each dimension of the face feature to zero mean with unit variance. An adaptive angular margin was also defined to further enhance the separability of neighboring classes. The proposed method, namely centralized coordinate learning (CCL), was trained on the CASIA Webface dataset, which has only 0.46M face images from about 10K persons. Extensive experiments on six benchmarks, including LFW, CACD, SLLFW, CALFW, YouTube Face and MegaFace, were conducted. CCL consistently exhibits competitive performance on all the six databases. It also outperforms many state-of-the-art models which are trained on much larger datasets.

REFERENCES

[1] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proceedings of Computer Vision and Pattern Recognition, 1991.* IEEE, 1991, pp. 586–591.

[2] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM computing surveys (CSUR)*, vol. 35, no. 4, pp. 399–458, 2003.

[3] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037–2041, 2006.

[4] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Fisher vector faces in the wild." in *British Machine Vision Conference*, 2013.

[5] C. Lu and X. Tang, "Surpassing human-level face verification performance on lfw with gaussianface." in *AAAI Conference on Artificial Intelligence*, 2015, pp. 3811–3819.

[6] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708.

[7] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1891–1898.

[8] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in Neural Information Processing Systems*, 2014, pp. 1988–1996.

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

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

[11] Y. Sun, X. Wang, and X. Tang, "Sparsifying neural network connections for face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4856–4864.

[12] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," *arXiv preprint arXiv:1704.08063*, 2017.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[14] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[15] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution grayscale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[16] B. Zhang, S. Shan, X. Chen, and W. Gao, "Histogram of gabor phase patterns (hgpp): A novel object representation approach for face recognition," *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 57–68, 2007.

[17] L. Liu, P. Fieguth, Y. Guo, X. Wang, and M. Pietikäinen, "Local binary features for texture classification: Taxonomy and experimental study," *Pattern Recognition*, vol. 62, pp. 135–160, 2017.
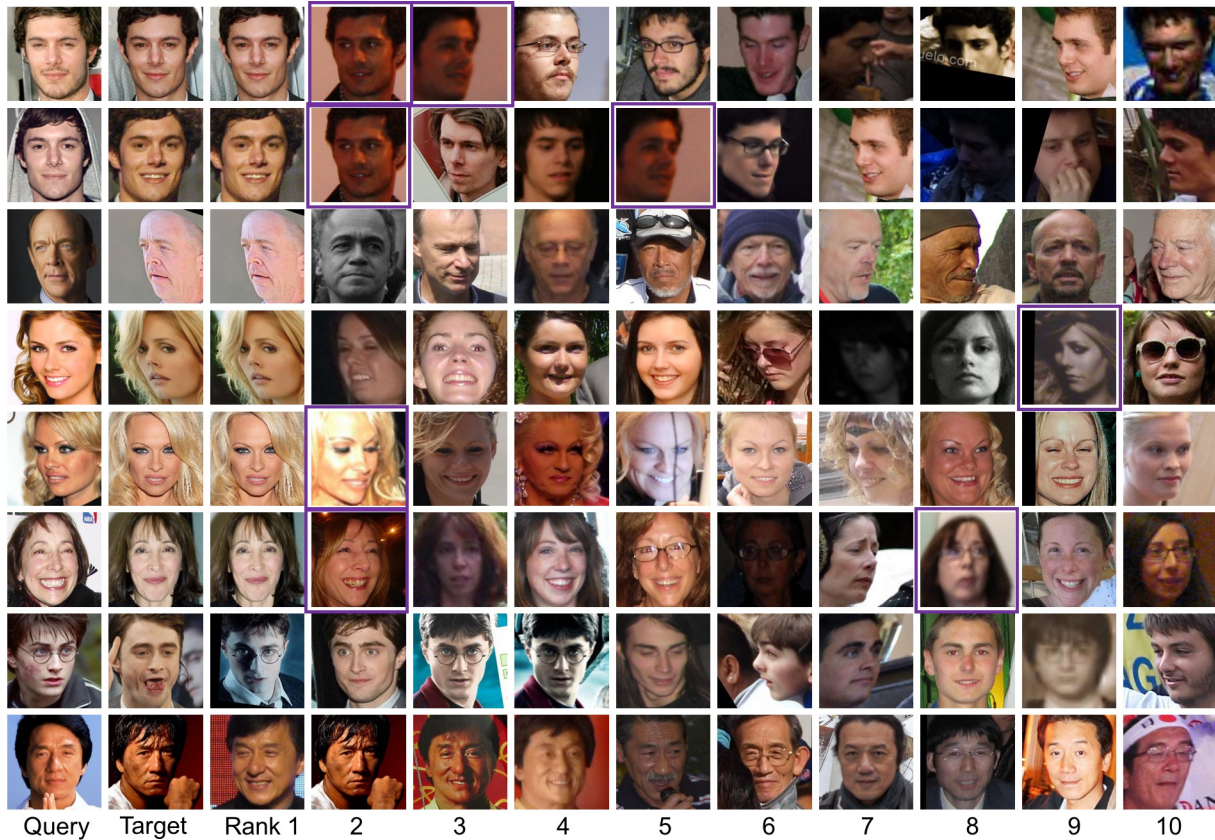
Fig. 9: Retrieval results on MegaFace. The first column is the query image, and the second column is the target image. The last ten columns are the top ten retrieval results.

[18] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[19] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[20] A. Wagner, J. Wright, A. Ganesh, Z. Zhou, H. Mobahi, and Y. Ma, "Toward a practical face recognition system: Robust alignment and illumination by sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, pp. 372–386, 2012.

[21] M. Yang, L. Zhang, J. Yang, and D. Zhang, "Robust sparse coding for face recognition," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 625–632.

[22] W. Deng, J. Hu, and J. Guo, "Extended src: Undersampled face recognition via intraclass variant dictionary," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1864–1870, 2012.

[23] L. Zhang, M. Yang, and X. Feng, "Sparse representation or collaborative representation: Which helps face recognition?" in *IEEE International Conference on Computer Vision*. IEEE, 2011, pp. 471–478.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

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

[27] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[29] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning." in *AAAI Conference on Artificial Intelligence*, 2017, pp. 4278–4284.

[30] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database forstudying face recognition in unconstrained environments," in *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*, 2008.

[31] 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*. IEEE, 2011, pp. 529–534.

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

[33] 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*. Springer, 2016, pp. 87–102.

[34] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard, "The megaface benchmark: 1 million faces for recognition at scale," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4873–4882.

[35] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," *arXiv preprint arXiv:1710.08092*, 2017.

[36] Y. Sun, X. Wang, and X. Tang, "Deeply learned face representations are sparse, selective, and robust," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2892–2900.

[37] Y. Sun, D. Liang, X. Wang, and X. Tang, "Deepid3: Face recognition with very deep neural networks," *arXiv preprint arXiv:1502.00873*, 2015.

[38] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings*

of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 815–823.

[39] O. M. Parkhi, A. Vedaldi, A. Zisserman *et al.*, "Deep face recognition." in *British Machine Vision Conference*, vol. 1, no. 3, 2015, p. 6.

[40] R. Ranjan, C. D. Castillo, and R. Chellappa, "L2-constrained softmax loss for discriminative face verification," *arXiv preprint arXiv:1703.09507*, 2017.

[41] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: l_2 hypersphere embedding for face verification," *arXiv preprint arXiv:1704.06369*, 2017.

[42] Y. Liu, H. Li, and X. Wang, "Rethinking feature discrimination and polymerization for large-scale recognition," *arXiv preprint arXiv:1710.00870*, 2017.

[43] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*. JMLR. org, 2016, pp. 507–516.

[44] T. Zheng, W. Deng, and J. Hu, "Cross-age lfw: A database for studying cross-age face recognition in unconstrained environments," *arXiv preprint arXiv:1708.08197*, 2017.

[45] B.-C. Chen, C.-S. Chen, and W. H. Hsu, "Cross-age reference coding for age-invariant face recognition and retrieval," in *European Conference on Computer Vision*. Springer, 2014, pp. 768–783.

[46] W. Deng, J. Hu, N. Zhang, B. Chen, and J. Guo, "Fine-grained face verification: Fglfw database, baselines, and human-dcmn partnership," *Pattern Recognition*, vol. 66, pp. 63–73, 2017.

[47] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[48] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 807–814.

[49] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[50] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, no. Feb, pp. 207–244, 2009.

[51] M. Abramowitz and I. A. Stegun, *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*. Courier Corporation, 1964, vol. 55.

[52] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 901–909.

[53] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the IEEE International Conference on Machine Learning*, 2015, pp. 448–456.

[54] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[55] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "Wider face: A face detection benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5525–5533.

[56] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li, "S³FD: Single shot scale-invariant face detector," *arXiv preprint arXiv:1708.05237*, 2017.

[57] P. Hu and D. Ramanan, "Finding tiny faces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 1522–1530.

[58] A. Paszke, S. Gross, S. Chintala, and G. Chanan, "Pytorch," 2017.

[59] Y. Guo and L. Zhang, "One-shot face recognition by promoting underrepresented classes," *arXiv preprint arXiv:1707.05574*, 2017.

[60] J. Deng, Y. Zhou, and S. Zafeiriou, "Marginal loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 60–68.

[61] B. Chen, W. Deng, and J. Du, "Noisy softmax: improving the generalization ability of dcnn via postponing the early softmax saturation," *arXiv preprint arXiv:1708.03769*, 2017.

[62] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao, "Range loss for deep face recognition with long-tail," *arXiv preprint arXiv:1611.08976*, 2016.

[63] W. Wu, M. Kan, X. Liu, Y. Yang, S. Shan, and X. Chen, "Recursive spatial transformer (rest) for alignment-free face recognition," in

The IEEE International Conference on Computer Vision (ICCV), Oct 2017.

[64] 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*, vol. 17, no. 6, pp. 804–815, 2015.

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

[66] D. Gong, Z. Li, D. Lin, J. Liu, and X. Tang, "Hidden factor analysis for age invariant face recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2872–2879.

[67] Y. Wen, Z. Li, and Y. Qiao, "Latent factor guided convolutional neural networks for age-invariant face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4893–4901.

[68] R. Rothe, R. Timofte, and L. Van Gool, "Dex: Deep expectation of apparent age from a single image," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 10–15.

[69] H.-W. Ng and S. Winkler, "A data-driven approach to cleaning large face datasets," in *Proceedings of the International Conference on Image Processing (ICIP)*. IEEE, 2014, pp. 343–347.