

P2SGrad: Refined Gradients for Optimizing Deep Face Models

Xiao Zhang¹ Rui Zhao² Junjie Yan² Mengya Gao² Yu Qiao³ Xiaogang Wang¹ Hongsheng Li¹

¹CUHK-SenseTime Joint Laboratory, The Chinese University of Hong Kong ²SenseTime Research

³SIAT-SenseTime Joint Lab, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences

zhangx9411@gmail.com hsl@ee.cuhk.edu.hk

Abstract

Cosine-based softmax losses [20, 29, 27, 3] significantly improve the performance of deep face recognition networks. However, these losses always include sensitive hyper-parameters which can make training process unstable, and it is very tricky to set suitable hyper parameters for a specific dataset. This paper addresses this challenge by directly designing the gradients for training in an adaptive manner. We first investigate and unify previous cosine softmax losses from the perspective of gradients. This unified view inspires us to propose a novel gradient called P2SGrad (Probability-to-Similarity Gradient), which leverages a cosine similarity instead of classification probability to control the gradients for updating neural network parameters. P2SGrad is adaptive and hyper-parameter free, which makes training process more efficient and faster. We evaluate our P2SGrad on three face recognition benchmarks, LFW [7], MegaFace [8], and IJB-C [16]. The results show that P2SGrad is stable in training, robust to noise, and achieves state-of-the-art performance on all the three benchmarks.

1. Introduction

Over the last few years, deep convolutional neural networks have significantly boosted the face recognition accuracy. State-of-the-art approaches are based on deep neural networks and adopt the following pipeline: training a classification model with different types of softmax losses and use the trained model as a feature extractor to test unseen samples. Then the cosine similarities between testing faces' features, are exploited to determine whether these features belong to the same identity. Unlike other vision tasks, such as object detection, where training and testing have the same objectives and evaluation procedures, conventional face recognition systems were trained with softmax losses but tested with cosine similarities. In other words, there is a gap between the softmax probability in training and inner product similarity in testing.

This problem is not well addressed in the classical softmax cross-entropy loss function (softmax loss for short

in the remaining part), which mainly considers probability distributions of training classes and ignores the testing setup. In order to bridge this gap, cosine softmax losses [28, 13, 14] and their angular margin based variants [29, 27, 3] directly use cosine distances instead of inner products as the input raw classification scores, namely logits. Specially, the angular margin based variants aim to learn the decision boundaries with a margin between different classes. These methods improve the face recognition performance in the challenging setup.

In spite of their successes, cosine-based softmax loss is only a trade-off: the supervision signals for training are still classification probabilities, which are never evaluated during testing. Considering the fact that the similarity between two testing face images is only related to themselves while the classification probabilities are related to all the identities, cosine softmax losses are not the ideal training measures in face recognition.

This paper aims to address these problems from a different perspective. Deep neural networks are generally trained with Stochastic Gradient Descent (SGD) algorithms where gradients play an essential role in this process. In addition to the loss function, we focus on the gradients of cosine softmax loss functions. This new perspective not only allows us to analyze the relations and problems of previous methods, but also inspires us to develop a novel form of adaptive gradients, P2SGrad, which mitigates the problem of training-testing mismatch and further improves the face recognition performance in practice.

To be more specific, P2SGrad optimizes deep models by well-designed gradients. Compared with the conventional gradients in cosine-based softmax losses, P2SGrad uses cosine distances to replace the probabilities in the original gradients. P2SGrad decouples gradients from hyperparameters and the number of classes, and matches testing targets.

This paper mainly contributes in the following aspects:

1. We analyze the recent cosine softmax losses and their angular-margin based variants from the perspective of gradients, and propose a general formulation to unify different cosine softmax cross-entropy losses;
2. With this unified model, we propose an adaptive

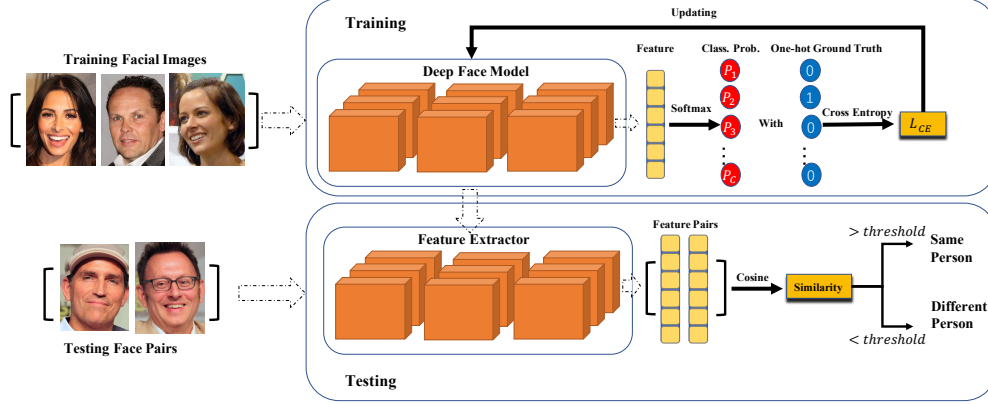


Figure 1. Pipeline of current face recognition system. In this general pipeline, deep face models trained on classification tasks are treated as feature extractors. Best viewed in color.

hyperparameter-free gradient method - P2SGrad for training deep face recognition networks. This method reserves advantages of using cosine distances in training but replaces classification probabilities with cosine similarities in the backward propagation;

3. We conduct extensive experiments on large-scale face datasets. Experimental results show that P2SGrad outperforms state-of-the-art methods on the same setup and clearly improves the stability of the training process.

2. Related Works

The accuracy improvements of face recognition [9, 6, 18, 25] enjoy the large-scale training data, and the improvements of neural network structures. Modern face datasets contain a huge number of identities, such as LFW [7], PubFig [10], CASIA-WebFace [32], MS1M [4] and MegaFace [17, 8], which enable the effective training of very deep neural networks. A number of recent studies demonstrated that well-designed network architectures lead to better performance, such as DeepFace [26], DeepID2, 3 [22, 23] and FaceNet [21].

In face recognition, feature representation normalization, which restricts features to lie on a fixed-radius hypersphere, is a common operation to enhance models' final performance. COCO loss [13, 14] and NormFace [28] studied the effect of normalization through mathematical analysis and proposed two strategies through reformulating softmax loss and metric learning. Coincidentally, L2-softmax [20] also proposed a similar method. These methods obtain the same formulation of cosine softmax loss from different views.

Optimizing auxiliary metric loss function is also a popular choice for boosting performance. In the early years, most face recognition approaches utilized metric loss functions, such as triplet loss [30] and contrastive loss [2], which use Euclidean margin to measure distance between features. Taking advantages of these works, center loss [31] and

range loss [33] were proposed to reduce intra-class variations through minimizing distance within target classes [1].

Simply using Euclidean distance or Euclidean margin is insufficient to maximize the classification performance. To circumvent this difficulty, angular margin based softmax loss functions were proposed and became popular in face recognition. Angular constraints were added to traditional softmax loss function to improve feature discriminativeness in L-softmax [12] and A-softmax [11], where A-softmax applied weight normalization but L-softmax [12] did not. CosFace [29], AM-softmax [27] and ArcFace [3] also embraced the idea of angular margins and employed simpler as well as more intuitive loss functions compared with aforementioned methods. Normalization is applied to both features and weights in these methods.

3. Limitations of cosine softmax losses

In this section we will discuss limitations caused by the mismatch between training and testing of face recognition models. We first provide a brief review of the workflow of cosine softmax losses. Then we will reveal the limitations of existing loss functions in face recognition from the perspective of forward and backward calculation respectively.

3.1. Gradients of cosine softmax losses

In face recognition tasks, the cosine softmax cross-entropy loss has an elegant two-part formulation, softmax function and cross-entropy loss.

We discuss softmax function at first. Assuming that the vector \vec{x}_i denotes the feature representation of a face image, the input of the softmax function is the logit $f_{i,j}$, i.e.,

$$f_{i,j} = s \cdot \frac{\langle \vec{x}_i, \vec{W}_j \rangle}{\|\vec{x}_i\|_2 \|\vec{W}_j\|_2} = s \cdot \langle \hat{\mathbf{x}}_i, \hat{\mathbf{W}}_j \rangle = s \cdot \cos \theta_{i,j}, \quad (1)$$

where s is a hyperparameter and $f_{i,j}$ is the classification score (logit) that \vec{x}_i is assigned to class j , and \vec{W}_j is the weight vector of class j . $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{W}}_j$ are normalized vectors of \vec{x}_i and \vec{W}_j respectively. $\theta_{i,j}$ is the angle between

feature x_i and class weight W_j . The logits $f_{i,j}$ are then input into the softmax function to obtain the probability $P_{i,j} = \text{Softmax}(f_{i,j}) = \frac{e^{f_{i,j}}}{\sum_{k=1}^C e^{f_{i,k}}}$, where C is the number of classes and the output $P_{i,j}$ can be interpreted as the probability of \vec{x}_i being assigned to a certain class j . If $j = y_i$, then P_{i,y_i} is the class probability of \vec{x}_i being assigned to its corresponding class y_i .

Then we discuss the cross-entropy loss associated with the softmax function, which measures the divergence between the predicted probability P_{i,y_i} and ground truth distributions as

$$\mathcal{L}_{\text{CE}}(\vec{x}_i) = -\log P_{i,y_i} = -\log \frac{e^{f_{i,y_i}}}{\sum_{k=1}^C e^{f_{i,k}}}, \quad (2)$$

where $\mathcal{L}_{\text{CE}}(\vec{x}_i)$ is the loss of input feature \vec{x}_i . The larger probability P_{i,y_i} is, the smaller loss $\mathcal{L}_{\text{CE}}(\vec{x}_i)$ is.

In order to decrease the loss $\mathcal{L}_{\text{CE}}(\vec{x}_i)$, the model needs to enlarge P_{i,y_i} and thus enlarge f_{i,y_i} . Then θ_{i,y_i} becomes smaller. In summary, cosine softmax loss function maps θ_{i,y_i} to the probability P_{i,y_i} and calculates the cross-entropy loss to supervise the training.

In the backward propagation process, classification probabilities $P_{i,j}$ play key roles for optimization. The gradient of \vec{x}_i and \vec{W}_j in cosine softmax losses are calculated as

$$\begin{aligned} \frac{\partial \mathcal{L}_{\text{CE}}(\vec{x}_i)}{\partial \vec{x}_i} &= \sum_{j=1}^C (P_{i,j} - \mathbb{1}(y_i = j)) \nabla f(\cos \theta_{i,j}) \cdot \frac{\partial \cos \theta_{i,j}}{\partial \vec{x}_i}, \\ \frac{\partial \mathcal{L}_{\text{CE}}(\vec{x}_i)}{\partial \vec{W}_j} &= (P_{i,j} - \mathbb{1}(y_i = j)) \nabla f(\cos \theta_{i,j}) \cdot \frac{\partial \cos \theta_{i,j}}{\partial \vec{W}_j}, \end{aligned} \quad (3)$$

where the indicator function $\mathbb{1}(j = y_i)$ returns 1 when $j = y_i$ and 0 otherwise. $\frac{\partial \cos \theta_{i,j}}{\partial \vec{x}_i}$ and $\frac{\partial \cos \theta_{i,j}}{\partial \vec{W}_j}$ can be computed respectively as:

$$\begin{aligned} \frac{\partial \cos \theta_{i,j}}{\partial \vec{x}_i} &= \frac{1}{\|\vec{x}_i\|_2} (\hat{\vec{W}}_j - \cos \theta_{i,j} \cdot \hat{\vec{x}}_i), \\ \frac{\partial \cos \theta_{i,j}}{\partial \vec{W}_j} &= \frac{1}{\|\vec{W}_j\|_2} (\hat{\vec{x}}_i - \cos \theta_{i,j} \cdot \hat{\vec{W}}_j), \end{aligned} \quad (4)$$

where $\hat{\vec{W}}_j$ and $\hat{\vec{x}}_i$ are unit vectors of \vec{W}_j and \vec{x}_i , respectively. $\frac{\partial \cos \theta_{i,j}}{\partial \vec{W}_j}$ is visualized as the red arrow in Fig. 2. This gradient vector is updating directions of class weights \vec{W}_j . Intuitively, we expect the updating of \vec{W}_j makes \vec{W}_{y_i} close to \vec{x}_i , and makes \vec{W}_j for $j \neq y_i$ away from \vec{x}_i . Gradient $\frac{\partial \cos \theta_{i,j}}{\partial \vec{W}_j}$ is vertical to \vec{W}_j and points toward \vec{x}_i . Thus it is the fastest and optimal direction for updating \vec{W}_j .

Then we consider the gradient $\nabla f(\cos \theta_{i,j})$. In conventional cosine softmax losses [20, 28, 13], classification score $f(\cos \theta_{i,j}) = s \cdot \cos \theta_{i,j}$ and thus $\nabla f(\cos \theta_{i,j}) = s$. In angular margin-based cosine softmax losses [27, 29, 3],

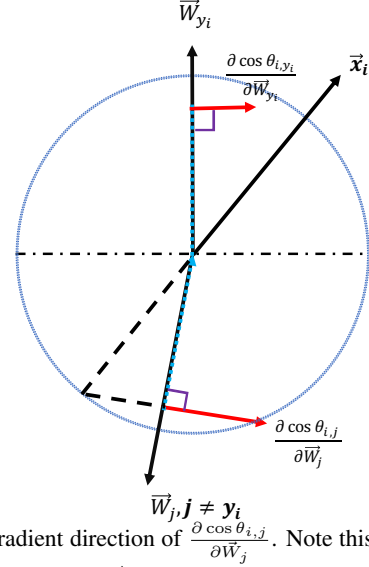


Figure 2. Gradient direction of $\frac{\partial \cos \theta_{i,j}}{\partial \vec{W}_j}$. Note this gradient is the updating direction of \vec{W}_j . The red pointed line shows that the gradient of \vec{W}_j is vertical to \vec{W}_j itself and in the plane spanned by \vec{x}_i and \vec{W}_j . This can be seen as the fastest direction for updating \vec{W}_{y_i} to be close to \vec{x}_i and for updating $\vec{W}_j, j \neq y_i$ to be far away from \vec{x}_i . Best viewed in color.

however, the gradient of $f_{\text{margin}}(\cos \theta_{i,y_i})$ for $j = y_i$ depends on where the margin parameter m is. For example, in CosFace [29] $f(\cos \theta_{i,y_i}) = s \cdot (\cos \theta_{i,y_i} - m)$, thus $\nabla f(\cos \theta_{i,y_i}) = s$ and in ArcFace [3] $f(\cos \theta_{i,y_i}) = s \cdot \cos(\theta_{i,y_i} + m)$, thus $\nabla f(\cos \theta_{i,y_i}) = s \cdot \frac{\sin(\theta_{i,y_i} + m)}{\sin \theta_{i,y_i}}$. In general, gradient $\nabla f(\cos \theta_{i,j})$ is always a scalar related to parameters s, m and $\cos \theta_{i,j}$.

Based on the aforementioned discussions, we reconsider gradients of class weights \vec{W}_j in Eq. (3). In $\frac{\partial \mathcal{L}_{\text{CE}}}{\partial \vec{W}_j}$, the first part $(P_{i,j} - \mathbb{1}(y_i = j)) \cdot \nabla f(\cos \theta_{i,j})$ is a scalar, which decides the length of gradient, while the second part $\frac{\partial \cos \theta_{i,j}}{\partial \vec{W}_j}$ is a vector which decides the direction of gradient. Since the directions of gradients for various cosine softmax losses remain the same, the essential difference of these cosine softmax losses is the different lengths of gradients, which significantly affect the optimization of model. In the following sections, we will discuss the suboptimal gradient length caused by forward and backward process respectively.

3.2. Limitations in probability calculation

In this section we discuss the limitations of the forward calculation of cosine softmax losses in deep face networks and focus on the classification probability $P_{i,j}$ obtained in the forward calculation.

We first revisit the relation between $P_{i,j}$ and $\theta_{i,j}$. The classification probability $P_{i,j}$ in Eq. (3) is a part of gradient length. Hence $P_{i,j}$ significantly affects the length of gradient. Probability $P_{i,j}$ and logit $f_{i,j}$ are positively correlated. For all cosine softmax losses, logits $f_{i,j}$ measure $\theta_{i,j}$ between feature \vec{x}_i and class weight \vec{W}_j . A larger $\theta_{i,j}$ pro-

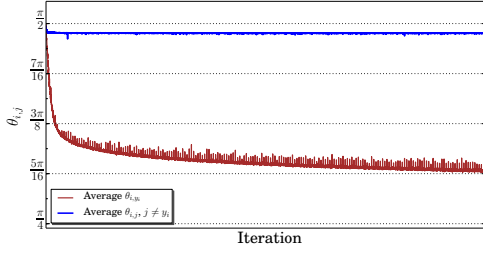


Figure 3. The change of average $\theta_{i,j}$ of each mini-batch when training on WebFace dataset. (Red) average angles in each mini-batch for non-corresponding classes, $\theta_{i,j}$ for $j \neq y_i$. (Brown) average angles in each mini-batch for corresponding classes, θ_{i,y_i} .

duces lower classification probability $P_{i,j}$ while a smaller $\theta_{i,j}$ produces higher $P_{i,j}$. It means that $\theta_{i,j}$ affects gradient length by its corresponding probability $P_{i,j}$. The equation sets up a mapping relation between $\theta_{i,j}$ and $P_{i,j}$ and makes $\theta_{i,j}$ affects optimization. Above analysis is also the reason why cosine softmax losses are effective on face recognition tasks.

Since θ_{i,y_i} is the direct measurement of the generalization but it can only indirectly affect gradient by corresponding P_{i,y_i} , setting a reasonable mapping relation between θ_{i,y_i} and P_{i,y_i} is crucial. However, there are two tricky problems in current cosine softmax losses: (1) classification probability P_{i,y_i} is sensitive to hyperparameter settings; (2) the calculation of P_{i,y_i} is dependent on class number, which is not related to face recognition tasks. We will discuss these problems below.

P_{i,y_i} is sensitive to hyperparameters. The most common hyperparameters in conventional cosine softmax losses [20, 28, 13] and margin variants [3] are the scale parameter s and the angular margin parameter m . We will analyze the sensitivity of probability P_{i,y_i} to hyperparameter s and m . For a more accurate analysis, we first look at the actual range of $\theta_{i,j}$. Fig. 3 exhibits how the average $\theta_{i,j}$ changes in training. Mathematically, $\theta_{i,j}$ could be any value in $[0, \pi]$. In practice, however, the maximum $\theta_{i,j}$ is around $\frac{\pi}{2}$. The blue curve reveals that $\theta_{i,j}$ for $j \neq y_i$ do not change significantly during training. The brown curve reveals that θ_{i,y_i} is gradually reduced. Therefore we can reasonably assume that $\theta_{i,j} \approx \frac{\pi}{2}$ for $j \neq y_i$ and the range of θ_{i,y_i} is $[0, \frac{\pi}{2}]$. Then P_{i,y_i} can be rewritten as

$$P_{i,y_i} = \frac{e^{f_{i,y_i}}}{\sum_{k=1}^C e^{f_{i,k}}} = \frac{e^{f_{i,y_i}}}{e^{f_{i,y_i}} + \sum_{k \neq y_i} e^{s \cdot \cos \theta_{i,k}}} \approx \frac{e^{f_{i,y_i}}}{e^{f_{i,y_i}} + \sum_{k \neq y_i} e^{s \cdot \cos \pi/2}} = \frac{e^{f_{i,y_i}}}{e^{f_{i,y_i}} + (C-1)}, \quad (5)$$

where f_{i,y_i} is logit that \vec{x}_i is assigned to its corresponding class y_i , and C is class number.

Theoretically, we can give the correspondence between probability P_{i,y_i} and angle θ_{i,y_i} under different hyperparameter settings. In state-of-the-art angular margin based

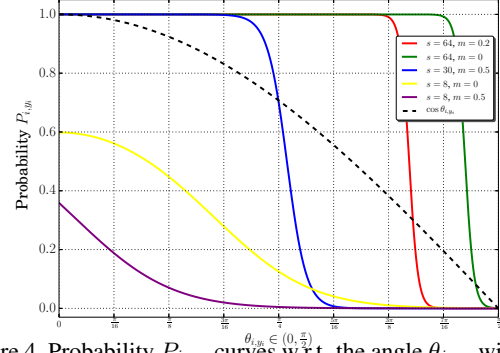


Figure 4. Probability P_{i,y_i} curves w.r.t. the angle θ_{i,y_i} with different hyperparameter settings.

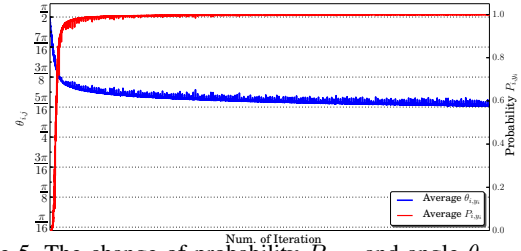


Figure 5. The change of probability P_{i,y_i} and angle θ_{i,y_i} as the iteration number increases with the hyperparameter setting $s = 35$ and $m = 0.2$. Best viewed in color.

losses [3], logit $f_{i,y_i} = s \cdot \cos(\theta_{i,y_i} + m)$. Fig. 4 reveals that different settings of s and m can significantly affect the relation between θ_{i,y_i} and P_{i,y_i} . Apparently, both the green curve and the purple curve are examples of unreasonable relations. The former is so lenient that even a very larger θ_{i,y_i} can produce a high $P_{i,y_i} \approx 1$. The later is so strict that even a very small θ_{i,y_i} can just produce a low P_{i,y_i} . In short, for a specific degree of θ_{i,y_i} , the difference of probability P_{i,y_i} under different settings is very large. This observation indicates that probability P_{i,y_i} is sensitive to parameters s and m .

To further confirm this conclusion, we take an example of correspondences between P_{i,y_i} and θ_{i,y_i} in real training. In Fig. 5, the red curve represents the change of P_{i,y_i} and the blue curve represents the change of θ_{i,y_i} during the training process. As we discussed above, $P_{i,y_i} \approx 1$ can produce very short gradients so that has little affection in updating. This setting is not ideal because P_{i,y_i} increases to 1 rapidly but θ_{i,y_i} is still large. Therefore classification probability P_{i,y_i} largely depends on the setting of hyperparameter.

P_{i,y_i} contains class number. In closed-set classification problems, probabilities $P_{i,j}$ become smaller as the growth of class number C because each class is assigned more or less probability (but not 0). This is reasonable in classification tasks. However, this is not suitable for face recognition, which is an open-set problem. Since θ_{i,y_i} is the direct measurement of generalization of \vec{x}_i while P_{i,y_i} is the indirect measurement, we expect that they have a consistent semantic meaning. But P_{i,y_i} is related to class number C while

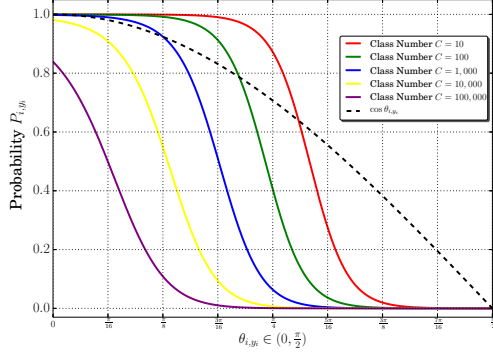


Figure 6. P_{i,y_i} with different class numbers. The hyperparameter setting is fixed to $s = 15$ and $m = 0.5$ for fair comparison. Best viewed in color.

θ_{i,y_i} is not, which causes the mismatch between them.

As shown in Fig. 6, we can summarize that the class number C is an important factor for P_{i,y_i} .

From the above discussion, we reveal that limitations exist in the forward calculation of cosine softmax losses. Both hyperparameters and the class number, which are unrelated to face recognition tasks, can determine the probability P_{i,y_i} , and thus affect the gradient length in Eq. (3).

3.3. Limitation in backward calculation of cosine softmax losses

In this section, we discuss the limitations in the backward calculation of the cosine softmax function, especially the angular-margin based softmax losses [3].

We revisit gradient $\nabla f(\cos \theta_{i,j})$ in Eq. (3). Besides P_{i,y_i} , the part of $\nabla f(\cos \theta_{i,j})$ also affects the length of gradient. Larger $\nabla f(\cos \theta_{i,j})$ produce longer gradients while smaller ones produce shorter gradients. So we expect θ_{i,y_i} and values of $\nabla f(\cos \theta_{i,j})$ to be positively correlated: small θ_{i,y_i} for small $\nabla f(\cos \theta_{i,j})$ and large θ_{i,y_i} for larger $\nabla f(\cos \theta_{i,j})$.

The logit f_{i,y_i} is different in various cosine softmax losses, and thus the specific form of $\nabla f(\cos \theta_{i,j})$ is different. Generally, we focus on simple cosine softmax losses [20, 28, 13] and state-of-the-art angular margin based loss [3]. Their $\nabla f(\cos \theta_{i,j})$ are visualized in Fig. 7, which shows that, under the factor of $\nabla f(\cos \theta_{i,j})$, the lengths of gradients in conventional softmax cosine losses [20, 28, 13] are constant. However in angular margin-based losses [3], the lengths of gradients and θ_{i,y_i} are negatively correlated, which is completely contrary to our expectations. Moreover, the correspondence between length of gradients in angular margin-based loss [3] and θ_{i,y_i} becomes tricky: when θ_{i,y_i} gradually reduced, P_{i,y_i} tends to shorten length of gradients but $\nabla f(\cos \theta_{i,j})$ tends to elongate the length. Therefore the geometric meaning of the gradient length becomes unexplained in angular margin-based cosine softmax loss.

3.4. Summary

In the above discussion, we first reveal that various cosine softmax losses have the same updating direction. Hence the main difference between the variants is their gradient lengths. For the length of gradient, there are two scalars that determine its value: the probability P_{i,y_i} in the forward process and the gradient $\nabla f(\cos \theta_{i,j})$. For P_{i,y_i} , we find that it can easily lose its semantic meaning with different hyperparameter settings and class numbers. For $\nabla f(\cos \theta_{i,j})$, its value depends on the definition of $f(\cos \theta_{i,y_i})$.

In summary, from the perspective of gradient, the widely used cosine softmax losses [20, 28, 13] and their angular margin variants [3] cannot produce optimal gradient lengths with well-explained geometric meanings.

4. P2SGrad: Change Probability to Similarity in Gradient

In this section, we propose a new method, namely P2SGrad, that determines the gradient length only by $\theta_{i,j}$ in training face recognition models. Formally, the gradient length produced by P2SGrad is hyperparameter-free and not related to the number of class C nor to a ad-hoc definition of logit f_{i,y_i} . P2SGrad does not need a specified formulation of loss function because gradients is well-designed to optimize deep models.

Since the main difference of state-of-the-art cosine softmax losses is the gradient length, reforming a reasonable gradient length is an intuitive thought. In order to decouple the length factor and direction factor of the gradients, we rewrite Eq. (3) as

$$\nabla \mathcal{L}_{\text{CE}}(\vec{x}_i) = \sum_{j=1}^C L(P_{i,j}, f(\cos \theta_{i,j})) \cdot D(\vec{W}_j, \vec{x}_i), \quad (6)$$

$$\nabla \mathcal{L}_{\text{CE}}(\vec{W}_j) = L(P_{i,j}, f(\cos \theta_{i,j})) \cdot D(\vec{x}_i, \vec{W}_j),$$

where the direction factors $D(\vec{W}_j, \vec{x}_i)$ and $D(\vec{x}_i, \vec{W}_j)$ are defined as

$$\begin{aligned} D(\vec{W}_j, \vec{x}_i) &= \frac{1}{\|\vec{x}_i\|_2} (\hat{\mathbf{W}}_j - \cos \theta_{i,j} \cdot \hat{\mathbf{x}}_i), \\ D(\vec{x}_i, \vec{W}_j) &= \frac{1}{\|\vec{W}_j\|_2} (\hat{\mathbf{x}}_i - \cos \theta_{i,j} \cdot \hat{\mathbf{W}}_j), \end{aligned} \quad (7)$$

where $\hat{\mathbf{W}}_j$ and $\hat{\mathbf{x}}_i$ are unit vectors of \vec{W}_j and \vec{x}_i , respectively. $\cos \theta_{i,j}$ is the cosine distances between feature \vec{x}_i and class weights \vec{W}_j . The direction factors will not be changed because they are the fastest changing directions, which are specified before. The length factor $|L(P_{i,j}, f(\cos \theta_{i,j}))|$ is defined as

$$|L(P_{i,j}, f(\cos \theta_{i,j}))| = \begin{cases} (1 - P_{i,y_i}) |\nabla f(\cos \theta_{i,y_i})| & j = y_i, \\ P_{i,j} \cdot |\nabla f(\cos \theta_{i,j})| & j \neq y_i. \end{cases} \quad (8)$$

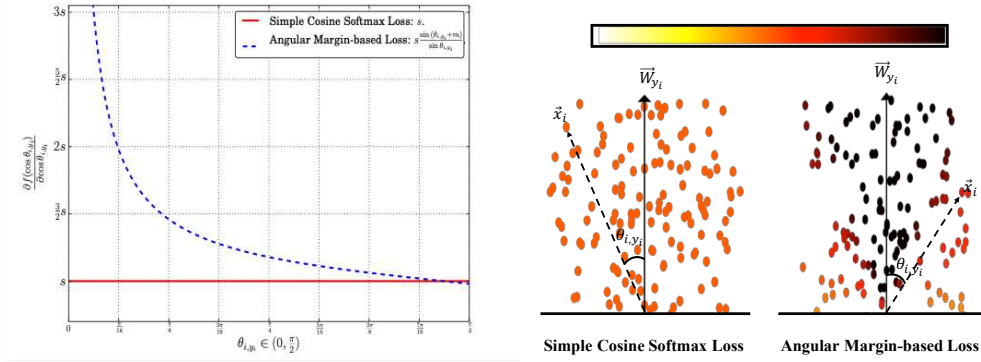


Figure 7. How $\nabla f(\cos \theta_{i,j})$ affects the length of gradients. (Left) the correspondence between θ_{i,y_i} and $\nabla f(\cos \theta_{i,j})$. The red curve means $\nabla f(\cos \theta_{i,j})$ is constant in conventional cosine softmax losses [20, 28, 13] while the blue curve means small θ_{i,y_i} can produce a very large $\nabla f(\cos \theta_{i,j})$. (Right) each point refers to a feature \vec{x}_i and the vertical vector is weight \vec{W}_{y_i} . The θ_{i,y_i} is angle between each \vec{x}_i and \vec{W}_{y_i} . The color from light to dark corresponds to the value of $\nabla f(\cos \theta_{i,j})$ from small to large. Hence for the factor of $\nabla f(\cos \theta_{i,j})$, the dark points produce longer gradients than the light points. Best viewed in color.

The length factor $|L(P_{i,j}, f(\cos \theta_{i,j}))|$ depends on the probability $P_{i,j}$ and $\nabla f(\cos \theta_{i,j})$ and is what we aim to reform.

Since we expect that the new length is hyperparameter-free, the cosine logit $f(\cos \theta_{i,j})$ will not have hyperparameters like s or m . Thus a constant $\nabla f(\cos \theta_{i,j})$ should be an ideal choice.

For the probability $P_{i,j}$, because it is hard to set a reasonable mapping function between $\theta_{i,j}$ and $P_{i,j}$, we can directly use $\cos \theta_{i,j}$ as a good alternative of $P_{i,j}$ in the gradient length term. Firstly, they have the same theoretical range of $[0, 1]$ where $\theta_{i,j} \in [0, \frac{\pi}{2}]$. Secondly, unlike $P_{i,j}$ which is adversely influenced by hyperparameter and the number of class, $\cos \theta_{i,j}$ does not contain any of these. It means that we do not need to select specified parameters settings for ideal correspondence between θ_{i,y_i} and P_{i,y_i} . Moreover, compared with $P_{i,j}$, $\cos \theta_{i,j}$ is a more natural supervision because cosine similarities are used in the testing phase of open-set face recognition systems while probabilities only apply for close-set classification tasks. Therefore, our reformed gradient length factor $\tilde{L}(\cos \theta_{i,j})$ can be defined as:

$$\tilde{L}(\cos \theta_{i,j}) = \cos \theta_{i,j} - \mathbb{1}(j = y_i), \quad (9)$$

where $\tilde{L}(\cos \theta_{i,j})$ is a function of $\cos \theta_{i,j}$. The reformed gradients $\tilde{G}_{\text{P2SGrad}}$ could then be defined as

$$\begin{aligned} \tilde{G}_{\text{P2SGrad}}(\vec{x}_i) &= \sum_{j=1}^C \tilde{L}(\cos \theta_{i,j}) \cdot D(\vec{W}_j, \vec{x}_i), \\ \tilde{G}_{\text{P2SGrad}}(\vec{W}_j) &= \tilde{L}(\cos \theta_{i,j}) \cdot D(\vec{x}_i, \vec{W}_j), \end{aligned} \quad (10)$$

where $\mathbb{1}$ is the indicator function. The full formulation can

be rewrite as

$$\begin{aligned} \tilde{G}_{\text{P2SGrad}}(\vec{x}_i) &= \sum_{j=1}^C (\cos \theta_{i,j} - \mathbb{1}(j = y_i)) \cdot \frac{\partial \cos \theta_{i,j}}{\partial \vec{x}_i}, \\ \tilde{G}_{\text{P2SGrad}}(\vec{W}_j) &= (\cos \theta_{i,j} - \mathbb{1}(j = y_i)) \cdot \frac{\partial \cos \theta_{i,j}}{\partial \vec{W}_j}, \end{aligned} \quad (11)$$

Although the analysis process is slightly complicated, the formulation of P2SGrad is not only succinct but reasonable. When $j = y_i$, the proposed gradient length and $\theta_{i,j}$ are positively correlated, when $j \neq y_i$, they are negatively correlated. More importantly, gradient length in P2SGrad only depends on $\theta_{i,j}$ and thus fits the testing metric of face recognition systems.

5. Experiments

In this section, we conduct a series of experiments to evaluate the proposed P2SGrad. We first verify advantages of P2SGrad in some exploratory experiments by testing the model's performance on LFW [7]. Then we evaluate P2SGrad on MegaFace [8] Challenge and IJBC 1:1 verification [16] with the same training configuration.

5.1. Exploratory Experiments

Preprocessing and training setting. We use CASIA-WebFace [32] as training data and ResNet-50 as the neural network architecture. Here WebFace [32] dataset is cleaned and contains about 450k facial images. RSA [15] is applied to images to extract facial areas and then aligns the faces similarity transformation. All images are resized to 144×144 . Also, we conduct pixel value normalization by subtracting 127.5 and then dividing by 128. For all exploratory experiments, the size of a mini-batch is 512 in every iteration.

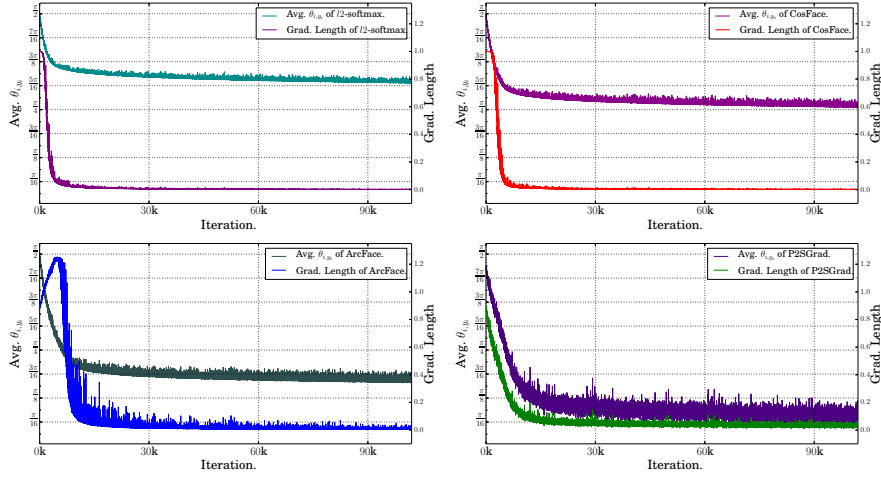


Figure 8. Curves of θ_{i,y_i} and gradient lengths w.r.t. iteration. Gradient lengths in existing cosine-based softmax losses (top-left, top-right, bottom-left) rapidly decrease to nearly 0 while gradient length produced by P2SGrad (bottom-right) can match θ_{i,y_i} between x_i and its ground truth class y_i . Best viewed in color.

Init. LR	Method			
	NormFace	CosFace	ArcFace	P2SGrad
10^{-1}	×	×	×	✓
10^{-2}	✓	×	×	✓
10^{-3}	✓	✓	✓	✓
10^{-4}	✓	✓	✓	✓

Table 1. The sensitiveness of initial learning rates. This table shows whether our P2SGrad and these cosine-based softmax loss are trainable under different initial learning rates.

The change of gradient length and θ_{i,y_i} w.r.t. iteration. Since P2SGrad aims to set up a reasonable mapping from θ_{i,y_i} to the length of gradients, it is necessary to visualize such mapping. In order to demonstrate the advancement of P2SGrad, we plot mapping curves of several cosine-based softmax losses in Fig. 8. This figure clearly shows that P2SGrad produces more optimal gradient length according to the change of θ_{i,y_i} .

Robustness of initial learning rates. An important problem of margin-based loss is that they are difficult to train with large learning rates. The implementation of L-softmax [12] and A-softmax [11] use extra hyperparameters to adjust the margin so that the models are trainable. Thus a small initial learning rate is important for properly training angular-margin-based softmax losses. In contrast, according to Table. 1, our proposed P2SGrad is stable with large learning rates.

Convergence rate. The convergence rate is important for evaluating an optimization method. We evaluated the trained model’s performance on Labeled Faces in the Wild (LFW) dataset of several cosine-based softmax losses and our P2SGrad method at different training periods. LFW dataset is an academic test set for unrestricted face verification. Its testing protocol contains about 13,000 images of about 1,680 identities. There are 3,000 positive matches

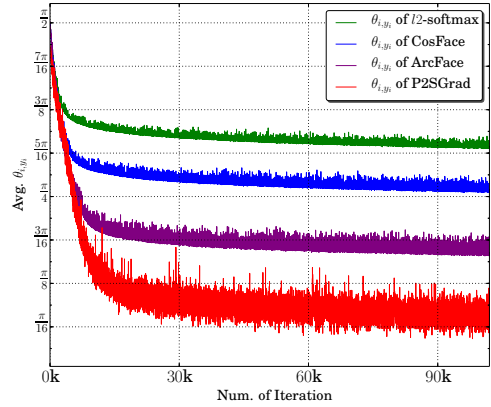


Figure 9. The change of average θ_{i,y_i} w.r.t. iteration number. θ_{i,y_i} represents the angle between x_i and the weight vector of its ground truth class y_i . Curves by the proposed P2SGrad, l_2 -softmax loss [20], CosFace [29] and ArcFace [3] are shown.

Method	Num. of Iteration		
	30k	60k	90k
l_2 -softmax [20]	81.50	91.27	97.92
CosFace [29]	83.63	93.58	99.05
ArcFace [3]	85.32	94.77	99.47
P2SGrad	91.25	97.38	99.82

Table 2. Convergence rates of P2SGrad and compared losses. With the same number of iterations, P2SGrad leads to the best performance.

and the same number of negative matches. Table. 2 shows the results with the same training configuration while Fig. 9 shows the decrease of average θ_{i,y_i} in P2SGrad is more quickly than other losses. These results reveal that our proposed P2SGrad can optimize neural network much faster.

Method	Size of MegaFace Distractor					
	10^1	10^2	10^3	10^4	10^5	10^6
l_2 -softmax [20]	99.73%	99.49%	99.03%	97.85%	95.56%	92.05%
CosFace [29]	99.82%	99.68%	99.46%	98.57%	97.58%	95.50%
ArcFace [3]	99.78%	99.65%	99.48%	98.87%	98.03%	96.88%
P2SGrad	99.86%	99.70%	99.52%	98.92%	98.35%	97.25%

Table 3. Recognition accuracy on MegaFace. Inception-ResNet [24] models trained with different compared softmax loss and the same cleaned WebFace [32] and MS1M [4] training data.

Method	True Acceptance Rate @ False Acceptance Rate						
	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
VggFace [18]	95.64%	87.13%	74.79%	59.75%	43.69%	32.20%	-
Crystal Loss [19]	99.06%	97.66%	95.63%	92.29%	87.35%	81.15%	71.37%
l_2 -softmax [20]	98.40%	96.45%	92.78%	86.33%	77.25%	62.61%	26.67%
CosFace [29]	99.01%	97.55%	95.37%	91.82%	86.94%	76.25%	61.72%
ArcFace [3]	99.07%	97.75%	95.55%	92.13%	87.28%	82.15%	72.28%
P2SGrad	99.03%	97.79%	95.58%	92.25%	87.84%	82.44%	73.16%

Table 4. TARs by different compared softmax losses on the IJB-C 1:1 verification task. The same training data (WebFace [32] and MS1M [4]) and Inception-ResNet [24] networks are used. Results of VggFace [18] and Crystal Loss [19] are from [19].

5.2. Evaluation on MegaFace

Preprocessing and training setting. Besides the mentioned WebFace [32] dataset, we add another public training dataset, MS1M [4], which contains about 2.35M cleaned and aligned images. Here we use Inception-ResNet [5, 24] with a batch size of 512 for training.

Evaluation results. MegaFace 1 million Challenge [8] is a public identification benchmark to test the performance of facial identification algorithms. The distractor in MegaFace contains about 1,000,000 images. Here we follow the cleaned testing protocol in [3]. The results of P2SGrad on MegaFace dataset are shown in Table 3. P2SGrad exceeds other compared cosine-based losses on MegaFace 1 million challenge with every size of distractor.

5.3. Evaluation on IJBC 1:1 verification

Preprocessing and training setting. Same as 5.2.

Evaluation results. The IJB-C dataset [16] contains about 3,500 identities with a total of 31,334 still facial images and 117,542 unconstrained video frames. The entire IJB-C testing protocols are designed to test detection, identification, verification and clustering of faces. In the 1:1 verification protocol, there are 19,557 positive matches and 15,638,932 negative matches. Therefore we test True Acceptance Rates at very strict False Acceptance Rates. Table 4 exhibits that P2SGrad surpasses all other cosine-based losses.

6. Conclusion

we comprehensively discussed the limitation of the forward and backward processes in training deep model for face recognition. To deal with the limitations, we proposed a simple but effective gradient method, P2SGrad, which is hyperparameter free and leads to better optimization results. Unlike previous methods which focused on loss functions, we improve the deep network training by using

carefully designed gradients. Extensive experiments validate the robustness and fast convergence of the proposed method. Moreover, experimental results show that P2SGrad achieves superior performance over state-of-the-art methods on several challenging face recognition benchmarks.

Acknowledgements. This work is supported in part by SenseTime Group Limited, in part by the General Research Fund through the Research Grants Council of Hong Kong under Grants CUHK14202217, CUHK14203118, CUHK14205615, CUHK14207814, CUHK14213616, CUHK14208417, CUHK14239816, in part by CUHK Direct Grant, and in part by National Natural Science Foundation of China (61472410) and the Joint Lab of CAS-HK.

References

- [1] Peter N. Belhumeur, João P. Hespanha, and David J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720, 1997.
- [2] S. Chopra, R. Hadsell, and Y. Lecun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, pages 539–546 vol. 1, 2005.
- [3] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *arXiv preprint arXiv:1801.07698*, 2018.
- [4] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision*, pages 87–102. Springer, 2016.
- [5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [6] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *arXiv preprint arXiv:1709.01507*, 2017.

- [7] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik 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.
- [8] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4873–4882, 2016.
- [9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [10] Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 365–372. IEEE, 2009.
- [11] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. SpheroFace: Deep hypersphere embedding for face recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, 2017.
- [12] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, pages 507–516, 2016.
- [13] Yu Liu, Hongyang Li, and Xiaogang Wang. Learning deep features via congenerous cosine loss for person recognition. *arXiv preprint arXiv:1702.06890*, 2017.
- [14] Yu Liu, Hongyang Li, and Xiaogang Wang. Rethinking feature discrimination and polymerization for large-scale recognition. *arXiv preprint arXiv:1710.00870*, 2017.
- [15] Yu Liu, Hongyang Li, Junjie Yan, Fangyin Wei, Xiaogang Wang, and Xiaoou Tang. Recurrent scale approximation for object detection in cnn. In *IEEE International Conference on Computer Vision*, 2017.
- [16] Brianna Maze, Jocelyn Adams, James A Duncan, Nathan Kalka, Tim Miller, Charles Otto, Anil K Jain, W Tyler Niggel, Janet Anderson, Jordan Cheney, et al. Iarpa janus benchmark-c: Face dataset and protocol. In *11th IAPR International Conference on Biometrics*, 2018.
- [17] Aaron Nech and Ira Kemelmacher-Shlizerman. Level playing field for million scale face recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3406–3415. IEEE, 2017.
- [18] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In *BMVC*, volume 1, page 6, 2015.
- [19] Rajeev Ranjan, Ankan Bansal, Hongyu Xu, Swami Sankaranarayanan, Jun-Cheng Chen, Carlos D Castillo, and Rama Chellappa. Crystal loss and quality pooling for unconstrained face verification and recognition. *arXiv preprint arXiv:1804.01159*, 2018.
- [20] Rajeev Ranjan, Carlos D Castillo, and Rama Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017.
- [21] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.
- [22] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.
- [23] Yi Sun, Ding Liang, Xiaogang Wang, and Xiaoou Tang. Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015.
- [24] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.
- [25] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. *Cvpr*, 2015.
- [26] Yaniv Taigman, Ming Yang, Marc’Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.
- [27] Feng Wang, Weiyang Liu, Haijun Liu, and Jian Cheng. Additive margin softmax for face verification. *arXiv preprint arXiv:1801.05599*, 2018.
- [28] Feng Wang, Xiang Xiang, Jian Cheng, and Alan L Yuille. Normface: l_2 hypersphere embedding for face verification. *arXiv preprint arXiv:1704.06369*, 2017.
- [29] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Zhifeng Li, Dihong Gong, Jingchao Zhou, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. *arXiv preprint arXiv:1801.09414*, 2018.
- [30] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.
- [31] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515. Springer, 2016.
- [32] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.
- [33] Xiao Zhang, Zhiyuan Fang, Yandong Wen, Zhifeng Li, and Yu Qiao. Range loss for deep face recognition with long-tailed training data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5409–5418, 2017.