This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Image Classification on Hypersphere Loss

Hao Wang, Jinpeng Cao, Zhang-Lei Shi, Chi-Sing Leung, *Senior Member, IEEE*, Ruibin Feng, Wenming Cao, *Senior Member, IEEE*, and Yuxin He

***Abstract*—The effectiveness of an image classification system depends on the following two key components: 1) the feature learning module and 2) the classification module. A well-designed loss function can not only enhance the classification ability of the latter but also improve the feature extraction capabilities of the former. This article devises a novel hypersphere loss function, which enhances the intraclass compactness and interclass separability of feature vectors given by the feature learning module. Furthermore, a new generalized class center is introduced into the loss function to handle the inevitable variability in samples (such as illumination, background, blurriness, low resolution, etc.) within the same class. Then, an alternative learning strategy is employed to optimize trainable parameters and class centers. Specifically, we first fix the trainable parameters of the deep learning model and calculate class centers using the exponentially weighted moving average method. Subsequently, we fix the generalized class centers and update the model's trainable parameters using mini-batch stochastic gradient descent. The proposed algorithm is evaluated on a range of typical tasks, including standard image classification, face verification, object detection, and retail product checkout. The results demonstrate that our proposed algorithm outperforms several state-of-the-art approaches.**

***Index Terms*—Constrained optimization, hypersphere, image classification, metric learning.**

Manuscript received 17 October 2023; revised 14 December 2023; accepted 19 December 2023. This work was supported by the National Natural Science Foundation of China under Grant 62206178 and Grant 72301180. Paper no. TII-23-4045. *(Corresponding author: Yuxin He.)*

Hao Wang, Jinpeng Cao, and Wenming Cao are with the Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University, Shenzhen 518060, China, also with the State Key Laboratory of Radio Frequency Heterogeneous Integration, Shenzhen 518060, China, and also with the Guangdong Multimedia Information Service Engineering Technology Research Center, Shenzhen University, Shenzhen 518060, China (e-mail: haowang@szu.edu.cn; 2210433012@email.szu.edu.cn; wmcao@szu.edu.cn).

Zhang-Lei Shi is with the College of Science, China University of Petroleum, Qingdao 266580, China (e-mail: zlshi@upc.edu.cn).

Chi-Sing Leung is with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, China (e-mail: eeleungc@cityu.edu.hk).

Ruibin Feng is with the Department of Medicine, Stanford, CA 94304 USA (e-mail: ruibin@stanford.edu).

Yuxin He is with the College of Urban Transportation and Logistics, Shenzhen Technology University, Shenzhen 518118, China (e-mail: heyuxin@sztu.edu.cn).

The code of our proposed method can be found at: https://github.com/TCCofWANG/Hypersphere_loss.

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TII.2023.3347759.

Digital Object Identifier 10.1109/TII.2023.3347759

I. Introduction

INCORPORATING intraclass compactness and interclass separability is crucial for improving the performance of image classification algorithms. From the perspective of feature representation, image classification algorithms typically contain two primary steps. In the first step, a feature learning module maps an image into a low-dimensional feature vector. Many feature learning methods have been proposed, including linear discriminant analysis (LDA) [1], principal component analysis (PCA) [2], and neural networks [3], [4]. Among these, deep neural networks, especially the convolutional neural network (CNN), have achieved remarkable successes in image classification tasks [5], [6]. CNN is a typical multilevel representation method that stacks convolutional and pooling layers to extract spatial structure information from images. Based on different configurations of convolutional layers, a wide range of CNN models have been proposed, such as ResNet [7], DenseNet [8], EfficientNet [9], MobileNet [10], and more. In the second step, the output vector from the feature learning module is used for image classification. For CNN-based methods, the output is typically a low-dimensional feature vector that captures the spatial information from the original image. Subsequently, the feature vector is multiplied by weight vectors for each class, and the resulting values are used to calculate the probability of each class using the softmax loss. The combination of CNN and softmax is prevalent in image classification tasks [7]. The softmax loss not only acts as a classifier but also can help adjust the feature vectors to improve the performance of the feature learning module. As a softened max operator, softmax will push the feature vectors to fill the whole feature space [11]. In this process, the magnitude of features from the same class may not be equally amplified, resulting in imbalanced magnitudes among feature vectors of the same category.

In order to improve classification accuracy, we hope that the feature vectors generated by the feature learning module have interclass separability and intraclass compactness. However, empirical experiments have revealed that the intraclass similarity of feature vectors obtained from CNN and softmax loss may be lower than their interclass similarity [12]. Furthermore, it has been observed that there are significant variations among feature vectors within a specific class. These issues can potentially have a negative effect on the classification accuracy of these methods.

To acquire discriminative features, an alternative approach is to explore metric learning techniques [13] or design specialized classification losses [5], [14], [15]. For instance, a center penalty function known as center loss (CL) is introduced alongside the original softmax loss [16]. The CL method and its variants

1551-3203 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

can enforce the feature vectors of samples in the same class gathered around their class centers. These class centers are trainable parameters in this neural network and can be directly learned from the training set. The CL method can improve the intraclass compactness of feature vectors. However, it is worth noting that CL may lead to training instability [17] and does not guarantee the interclass separability of feature vectors given by this approach.

Numerous studies have highlighted the potential benefits of conducting classification tasks in the angular domain. For instance, the approach in [18] combined the softmax loss with a feature normalization process to address imbalanced magnitudes of feature vectors within the same class. The article [3], [11] concurrently applied the normalization process on both feature vectors and weight vectors. The article [14] proposed an angular softmax loss, assuming that the weight vectors in the last layer can represent class centers in angular space. Consequently, it penalizes the angles between the features and their corresponding weights. The large margin cosine loss in [15] reformulated the softmax loss as a cosine loss by l_2 normalizing both features and weight vectors, and introduced a cosine margin term to further maximize the decision margin in angular space. An exclusive regularization was proposed to improve the interclass separability by penalizing the angle between a weight vector and its nearest neighbor in [5]. These studies demonstrated various effective approaches for tackling classification challenges in the angular domain, highlighting the importance of considering angular representations to enhance classification performance.

Nonetheless, recent studies indicate that by conducting classification in the angular domain and incorporating the magnitudes of feature vectors into this process, the performance may be further improved. For instance, Meng et al. [19] proposed a novel loss that performs classification in the angular domain while utilizing the magnitude of the feature vector to measure the quality of input images. The experimental results presented in the article demonstrate that the proposed method achieves exceptional performance in face recognition tasks. In another study [12], a constrained center loss was proposed. Its objective combines the standard softmax loss with a center loss, while also introducing constraints on the magnitudes of the class center vectors into the model. By selecting an appropriate magnitude for the class centers, further enhancements in classification performance can be achieved. However, it is important to note that this method incurs a high computational cost for updating class centers, and there is no guarantee of interclass separability.

In this article, the hypersphere loss is proposed, and the class center in our method is no longer a point but a small hypersphere. Consequently, all feature vectors gather around their respective class centers, which we refer to as **feature hypersphere** throughout this article. A feature hypersphere serves as a generalized class center that can tolerate the inevitable variability for samples within the same class. Notably, the centers of all feature hyperspheres are situated on the surface of a larger hypersphere called **class center hypersphere**. To ensure separation between the feature hyperspheres of different classes, we introduce a class center separation function. The hypersphere loss given in this article is constrained. Hence, an alternative learning strategy is developed for training the model. In this strategy, we first fix the trainable parameters in the deep learning model and calculate the class centers based on the exponentially weighted moving average (EWMA) method. Subsequently, we keep the generalized class centers fixed and update the trainable parameters in the deep learning model by mini-batch stochastic gradient descent (SGD).

The rest of this article is organized as follows. The backgrounds of image classification model, center loss, and constrained center loss are presented in Section II. In Section III, we propose a hypersphere loss based on the geometry analysis of $d-1$-sphere. The numerical results for algorithm evaluation and comparison are shown in Section IV. Finally, Section V concludes this article.

II. Background

A. Basic Structure of Image Classification Model

A typical image classification model based on deep learning typically consists of the following two modules.

1) Feature learning module: This module is responsible for extracting a discriminative low-dimensional feature vector from each image.
2) Classification module: In this module, the feature vectors are matched with their corresponding labels.

Suppose that we are given a training set $\mathcal{D} = \{\{\boldsymbol{x}_i, y_i\}_{i=1}^N\}$, where $\boldsymbol{x}_i$ denotes the ith image sample, $y_i \in \{1, \ldots, C\}$ is the corresponding class label, and C is the number of classes in $\mathcal{D}$. The input of the feature learning module is an image sample $\boldsymbol{x}_i$, and the output is the corresponding feature vector $\boldsymbol{f}_i \in \mathbb{R}^d$. The classification module gives the predicted classification result based on the feature vector $\boldsymbol{f}_i$. The softmax loss is often used to make decisions in the classification module. The mathematical expression of softmax loss is given by

$$\min_{\boldsymbol{w}_k, \boldsymbol{\Theta}} \mathcal{L}_{\text{softmax}} = \min_{\boldsymbol{w}_k, \boldsymbol{\Theta}} -\sum_{i=1}^{N} \log \frac{\exp(\boldsymbol{w}_{y_i}^\top \boldsymbol{f}_i)}{\sum_{k=1}^{C} \exp(\boldsymbol{w}_k^\top \boldsymbol{f}_i)} \tag{1}$$

where $\boldsymbol{w}_k \in \mathbb{R}^d$ denotes the weight vector for the kth class in the classification module, $\boldsymbol{\Theta}$ is the collection of all trainable parameters in the feature learning module. From (1), we know that the feature learning module will give a discriminative feature vector $\boldsymbol{f}_i$ by adjusting the trainable parameters in $\boldsymbol{\Theta}$, while the classification module will minimize the softmax loss by searching an appropriate weight vector for each class. For solving the problem in (1), the mini-batch SGD method is utilized.

B. Center Loss

From empirical experiments, we see that although a model with softmax loss can generate interclass separable feature vectors to a certain extent, it cannot ensure the intraclass compactness of these feature vectors. This characteristic may influence the effectiveness of the softmax loss.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

To enhance the intraclass compactness, a center penalty was proposed in [16], which is defined as

$$\mathcal{L}_{\text{center}} = \frac{1}{2\,N}\sum_{i=1}^{N}\|\boldsymbol{f}_i - \boldsymbol{c}_{y_i}\|_2^2 \tag{2}$$

where $\boldsymbol{c}_{y_i} \in \mathbb{R}^d$ is a trainable parameter, which represents the corresponding class center of y_i, and $\boldsymbol{c}_{y_i} \in [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_C]$. The center penalty enhances the intraclass compactness by penalizing the Euclidean distance between the feature vectors and their corresponding centers. Combining the softmax loss and the center penalty, the objective of the center loss approach is given as

$$\mathcal{L}_{\text{softmax}} + \lambda\mathcal{L}_{\text{center}} \tag{3}$$

where λ is a tradeoff parameter. Similar to the softmax approach, all parameters in this method can be optimized by mini-batch SGD.

C. Constrained Center Loss

From [18], we know that the l_2-normalization can reduce the variation in feature magnitudes. With the l_2-norm constraint, the softmax loss can be modified as

$$\min_{\boldsymbol{w}_k,\boldsymbol{\Theta}} \mathcal{L}_{\text{softmax}} \quad s.t. \|\boldsymbol{f}_i\| = r \quad \forall i = 1, \ldots, N \tag{4}$$

where r is a positive constant that denotes the magnitudes of features. However, solving the problem in (4) is difficult due to the nature of feature vectors, which is a very complex function involving all parameters in the feature learning module.

To circumvent the intricate nonlinear optimization problem in (4), the article in [18] introduced a feature normalization process. As a result, the softmax loss can be rewritten as

$$\min_{\boldsymbol{w}_k,\boldsymbol{\Theta}} -\sum_{i=1}^{N}\log\frac{\exp\left(r\boldsymbol{w}_{y_i}^{\top}\frac{\boldsymbol{f}_i}{\|\boldsymbol{f}_i\|}\right)}{\sum_{k=1}^{C}\exp\left(r\boldsymbol{w}_k^{\top}\frac{\boldsymbol{f}_i}{\|\boldsymbol{f}_i\|}\right)}. \tag{5}$$

To further improve the performance, the articles in [11] and [3] also applied the normalization process to the weight vector $\boldsymbol{w}_k$ in (4). Their objective function is defined as follows:

$$\min_{\boldsymbol{w}_k,\boldsymbol{\Theta}} -\sum_{i=1}^{N}\log\frac{\exp\left(r\frac{\boldsymbol{w}_{y_i}^{\top}\boldsymbol{f}_i}{\|\boldsymbol{w}_{y_i}\|\|\boldsymbol{f}_i\|}\right)}{\sum_{k=1}^{C}\exp\left(r\frac{\boldsymbol{w}_k^{\top}\boldsymbol{f}_i}{\|\boldsymbol{w}_k\|\|\boldsymbol{f}_i\|}\right)}. \tag{6}$$

In addition, other relevant research studies [14], [15] have shown that conducting classification in the angular domain can reduce classification difficulty.

Inspired by the aforementioned research, the constrained center loss (CCL) was proposed in [12], and its objective is formulated as

$$\min_{\boldsymbol{w}_k,\boldsymbol{c}_k,\boldsymbol{\Theta}} -\sum_{i=1}^{N}\log\frac{\exp(\boldsymbol{w}_{y_i}^{\top}\boldsymbol{f}_i)}{\sum_{k=1}^{C}\exp(\boldsymbol{w}_k^{\top}\boldsymbol{f}_i)} + \frac{\lambda}{2\,N}\sum_{i=1}^{N}\|\boldsymbol{f}_i - \boldsymbol{c}_{y_i}\|_2^2,$$
$$\text{s.t.} \quad \|\boldsymbol{c}_k\| = r \quad \forall k = 1, \ldots, C \tag{7}$$

where $\boldsymbol{c}_k$ denotes the class center of kth class, and r is the radius of a hypersphere where the class centers are located. The CCL can constrain all class centers to the surface of a hypersphere and make each feature vector gather around their respective class centers. Due to the constraints of class centers, the image classification within this model can be regarded as being conducted in the angular domain. Furthermore, the adjustment of the hypersphere radius incorporates the magnitude of features into this method.

To solve this problem, an alternative training strategy is used. First, the class centers are updated by solving the following optimization problem:

$$\min_{\boldsymbol{c}_k} \frac{1}{2\,N}\sum_{i=1}^{N}\|\boldsymbol{f}_i - \boldsymbol{c}_{y_i}\|_2^2, \quad \text{s.t.} \quad \|\boldsymbol{c}_k\| = r. \tag{8}$$

Based on the Lagrange multiplier method, the class center of the kth class can be updated as

$$\boldsymbol{c}_k = r\frac{\frac{1}{N}\sum_{i=1}^{N}\tau(y_i,k)\boldsymbol{f}_i}{\|\frac{1}{N}\sum_{i=1}^{N}\tau(y_i,k)\boldsymbol{f}_i\|} \tag{9}$$

where $\tau(y_i, k)$ is an indicator function defined by

$$\tau(y_i,k) = \begin{cases} 1, & \text{if } y_i = k, \\ 0, & \text{otherwise.} \end{cases}$$

Second, class centers are fixed in (7), and the trainable parameters $\boldsymbol{w}_k$ and $\boldsymbol{\Theta}$ are updated using mini-batch SGD. Finally, the first and second steps are repeated until convergence.

While the CCL approach enhances classification accuracy, it still has three noteworthy weaknesses as follows.

1) The loss function in (7) encourages feature vectors within the same category to gather around their respective class center. Nevertheless, it cannot ensure that feature vectors from different categories are adequately separated.
2) The update of all class centers relies on (9). It is worth noting that all samples in the training set are needed to update centers. Notably, for large training sets, this method demands substantial computational resources and memory. In addition, it is inconvenient to combine this method with the neural network training methods, such as mini-batch SGD.
3) The CCL approach uses a single point as the class center for features within the same class and makes a very strong assumption that all feature vectors of the same category should converge around this point. It overlooks the potential influence of external factors like illumination, shooting angles, occlusion, and others on a class center.

III. Methodology

Inspired by the geometric structure of the hypersphere, we devise a novel objective in this article, which can overcome the abovementioned shortcomings of CCL and further improve its performance.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

A. Geometry Analysis of d-1-Sphere

Due to the constraints in CCL, all class centers are distributed on the surface of a d-1-sphere with radius r[1], which is called **class center hypersphere** in this article.

To improve the interclass discrepancy, the class centers should be scattered as much as possible on the surface of the class center hypersphere. Moreover, to improve the intraclass compactness, the feature vectors of the same class should be compactly centered around their class center. For such purposes, we assume that the feature vectors of the same class will be distributed in a hypersphere with radius a centered at its corresponding class center, which is called **feature hypersphere** in this article. Therefore, for an image classification problem with C categories, the centers of C feature hyperspheres should be constructed on the surface of a d-1-sphere, and these feature hyperspheres are discrete and do not overlap. Thus, the original classification problem is transformed into the problem of constructing C maximum nonoverlapping feature hyperspheres with centers on the surface of the class center hypersphere.

For convenience, let us consider the low-dimensional scenario illustrated in Fig. 1(a), where $d = 3$. In this figure, class centers are distributed on the surface of the white class center hypersphere with a radius of r. Each class corresponds to a feature hypersphere centered at the respective class center, with a radius of a. To ensure interclass separability, class centers should be distributed as uniformly as possible on the surface of the class center hypersphere. In addition, to ensure that the feature hyperspheres do not overlap, adjacent feature hyperspheres must either be positioned far apart from each other or be tangent to each other. To obtain the maximum radius a for the feature hyperspheres, we assume that all adjacent feature hyperspheres are tangent to each other. For instance, in Fig. 1(a), two adjacent blue feature hyperspheres, which are tangent to each other, have their centers at points A and B, with an included angle of 2θ between them. Fig. 1(b) shows a similar scenario in 2-D space. From Fig. 1, we see that by adjusting the position of class centers and their radius a, we can certainly construct C largest nonoverlapping feature hyperspheres.

In the following, we will discuss how to estimate the radius a for the feature hyperspheres. For convenience, we analyze this in a 2-D space, as illustrated in Fig. 1(b). To ensure interclass separability, it is crucial to prevent overlap among the feature hyperspheres. To maximize the radius a of the feature hyperspheres, we assume that all adjacent feature hyperspheres are tangent to each other. In this scenario, each feature hypersphere truncates a hyperspherical cap from the class center hypersphere, exemplified by the green shaded region in Fig. 1(b). Any such hyperspherical cap's area must be less than $1/C$ of the area of the class center hypersphere; otherwise, interclass overlap would ensue. Given that the diameter of the feature hypersphere is $2a$, it can be observed that the radius of the base of the green hyperspherical cap, denoted as $dis(E, F)/2$, is at most a. Consequently, the area of the green hyperspherical cap is smaller than or equal to that of the yellow hyperspherical cap in Fig. 1(b). Thus, the size of feature hyperspheres is appropriate as long as the area of the yellow hyperspherical cap is less than or equal to the $1/C$ area of the class center hypersphere. Following this logic, we can infer the size of a. The details are given in Appendix A of the Supplementary Material.

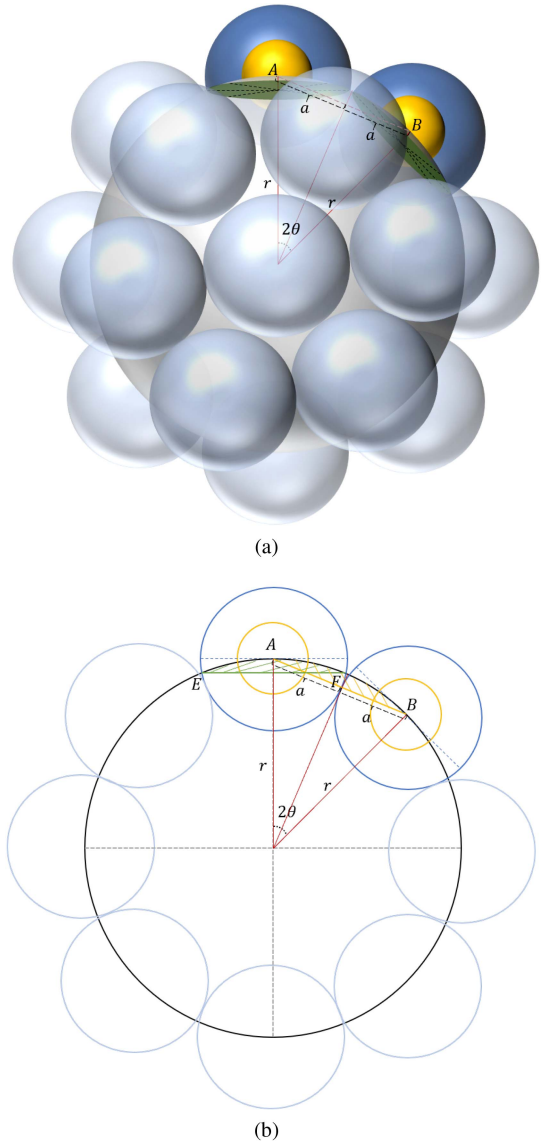

Fig. 1. Schematic diagram of class center hypersphere and feature hyperspheres in low-dimensional space. (a) In 3-D space. (b) In 2-D space.

Besides, from Fig. 1, it is observed that all feature hyperspheres contain a small yellow concentric hypersphere inside. These small concentric hyperspheres generalize the class center from a point to a hypersphere. In this way, the generalized class centers can tolerate some disturbances caused by unavoidable external factors. Therefore, it is helpful to improve the representation ability and stability of class centers.

B. Hypersphere Loss

In the following, we propose the objective called hypersphere loss (HL) based on the geometry analysis of d-1-sphere and

[1] An d-1-sphere with radius r can be defined as $S^{d-1}(r) = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\| = r\}$.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

devise the method to train models with this objective. For an image classification problem, the formulation of hypersphere loss is given by

$$\min_{\boldsymbol{w}_k,\boldsymbol{c}_k,\boldsymbol{\Theta}} -\sum_{i=1}^{N}\log\frac{\exp(\boldsymbol{w}_{y_i}^{\top}\boldsymbol{f}_i)}{\sum_{k=1}^{C}\exp(\boldsymbol{w}_k^{\top}\boldsymbol{f}_i)}+\frac{\beta}{N(C-1)}\sum_{i=1}^{N}\sum_{l\neq y_i}^{C}\boldsymbol{c}_{y_i}^{\top}\boldsymbol{c}_l + \frac{\lambda}{2N}\sum_{i=1}^{N}[\iota(\|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2)\|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2]^2, \quad \text{s.t.} \quad \|\boldsymbol{c}_k\|=r \quad \forall k=1,\ldots,C \tag{10}$$

where $\boldsymbol{c}_k$ denotes the class center of kth class, r is the radius of the class center hypersphere, λ and β are tradeoff parameters, $\boldsymbol{w}_k \in \mathbb{R}^d$ is the weight vector for the kth class, $\boldsymbol{\Theta}$ is the collection of all trainable parameters in the feature learning module. $\iota(\|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2)$ is an indicate function defined as

$$\iota(\|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2)=\begin{cases}0, & \|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2<\epsilon a,\\ 1, & \|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2\geq\epsilon a\end{cases} \tag{11}$$

where a is the radius of feature hypersphere, $\epsilon \in [0,1]$ is a tradeoff parameter. A small ϵ will improve the interclass separability and intraclass compactness, but a large ϵ will generalize the class center from a point to a big region, which can enhance the representation ability of the class center. We let $\epsilon = 0.05$ in our experiments.

The formulation (10) consists of the following four parts:

1) the softmax loss $-\sum_{i=1}^{N}\log\frac{\exp(\boldsymbol{w}_{y_i}^{\top}\boldsymbol{f}_i)}{\sum_{k=1}^{C}\exp(\boldsymbol{w}_k^{\top}\boldsymbol{f}_i)}$;
2) the generalized center penalty function $\frac{\lambda}{2N}\sum_{i=1}^{N}\{[\iota(\|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2)\|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2]^2\}$;
3) the class center separation function $\frac{\beta}{N(C-1)}\sum_{i=1}^{N}\sum_{l\neq y_i}^{C}\boldsymbol{c}_{y_i}^{\top}\boldsymbol{c}_l$;
4) the hypersphere surface constraints $\|\boldsymbol{c}_k\|=r(k=1,\ldots,C)$.

The generalized center penalty function will make all feature vectors converge toward their corresponding centers. For any feature vector $\boldsymbol{f}_i$, if the distance between the feature vector and its corresponding center is larger than ϵa, the generalized center penalty function will penalize the distance between them. For any feature vector $\boldsymbol{f}_i$, the class center separation function will penalize the sum of the inner product between the class center $\boldsymbol{c}_{y_i}$ of $\boldsymbol{f}_i$ and other class centers. Combined with the constraints $\|\boldsymbol{c}_k\|=r, \forall k=1,\ldots,C$, we can increase the angular separation between all class centers.

The loss function in (10) is a constrained optimization problem, and an alternative strategy is introduced to solve it. The alternative strategy consists of the following two steps: 1) class centers are updated; 2) all class centers are fixed, and we update the weight vectors $\boldsymbol{w}_k$'s in the classification module and all trainable parameters $\boldsymbol{\Theta}$ in the feature learning module by mini-batch SGD. The details of the alternative training approach are given as follows.

1) Update Class Centers: In the first step, the weight vectors $\boldsymbol{w}_k$'s in the classification module and all trainable parameters $\boldsymbol{\Theta}$ in the feature learning module are fixed. Thus, the class center vectors $\boldsymbol{c}_k$'s become the only parameters to be optimized in this step, and the formulation (10) can be rewritten as

$$\min_{\boldsymbol{c}_k}\frac{\lambda}{2N}\sum_{i=1}^{N}[\iota(\|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2)\|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|_2]^2 + \frac{\beta}{N(C-1)}\sum_{i=1}^{N}\sum_{l\neq y_i}^{C}\boldsymbol{c}_{y_i}^{\top}\boldsymbol{c}_l, \quad \text{s.t.} \quad \|\boldsymbol{c}_k\|=r \quad \forall k=1,\ldots,C. \tag{12}$$

Based on the Lagrange multiplier method and some convex optimization concepts, we can deduce that

$$\boldsymbol{c}_k = r\frac{\frac{1}{N}\sum_{i=1}^{N}\delta(k,y_i)\left(\boldsymbol{f}_i-\frac{\beta}{C-1}\sum_{l\neq y_i}^{C}\boldsymbol{c}_l\right)}{\|\frac{1}{N}\sum_{i=1}^{N}\delta(k,y_i)\left(\boldsymbol{f}_i-\frac{\beta}{C-1}\sum_{l\neq y_i}^{C}\boldsymbol{c}_l\right)\|_2} \tag{13}$$

where $k=1,\ldots,C$, $\delta(k,y_i)$ is an indicator function given by

$$\delta(k,y_i)=\begin{cases}1, & \text{if } y_i=k \text{ and } \|\boldsymbol{f}_i-\boldsymbol{c}_{y_i}\|\geq\epsilon a\\ 0, & \text{otherwise}\end{cases}. \tag{14}$$

The derivation details of (13) are shown in Appendix B of the Supplementary Material.

From the equation in (13), it is evident that all samples in the training set are used during the class center update process. In this case, this approach incurs a substantial computational cost. If we only use a mini-batch of samples to update the class centers, it may result in significant deviations of the computed class centers from their actual positions. Consequently, this could pose challenges to the convergence of the proposed algorithm.

In this article, we proposed an efficient class center updating method based on mini-batch samples. The training process unfolds in two phases. Before the t_1th epoch, we employ the standard softmax loss to train the model. After the t_1th epoch, we introduce the proposed class center updating method. In our experiments, we set $t_1 = 5$. This approach follows a warm-start technique [13], which can be used to avoid the excessive loss caused by random values in the initial state.

To enable the update of class centers based on mini-batch samples rather than the entire training set, we have devised two modification strategies as follows. 1) First, we ignore the setting of the generalized class center in this step. In other words, the indicator function $\delta(k,y_i)$ is replaced by $\tau(y_i,k)$. In this way, all samples in a mini-batch are used to update the class centers. Consequently, the stability of our proposed approach is improved by increasing the number of samples used for updating class centers. 2) Second, the EWMA is utilized to calculate class centers. Thus, given a mini-batch with m samples, an auxiliary variable $\boldsymbol{v}_i$ is calculated according to

$$\boldsymbol{v}_i=\frac{1}{1-\phi^{N_c}}\left(\phi\boldsymbol{c}_{y_i}+(1-\phi)\boldsymbol{f}_i\right),\ i\in[1,\ldots,m] \tag{15}$$

where $0<\phi<1$ is a hyperparameter of EWMA (we set $\phi=0.99$ in our experiments), N_c represents the cumulative updates of class centers, initially set to 0 and incremented by 1 after all class centers have completed an update. The term $\frac{1}{1-\phi^{N_c}}$ serves to mitigate the substantial bias introduced by the EWMA method during the initial class center updates. As N_c increases,

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Algorithm 1: Classification based on hypershpere loss.

while $t \in [1, \ldots, T]$ **do**
  Randomly divide the training data into Q mini-batches, where $Q = \lfloor \frac{N}{m} \rfloor + 1$.
  if $1 \leq t \leq t_1$ **then**
    for $q \in [1, \ldots, Q]$ **do**
      Use the softmax loss in (1) as the objective and update the trainable parameters by mini-batch SGD.
    end for
  else if $t_1 < t \leq T$ **then**
    for $q \in [1, \ldots, Q]$ **do**
      (1) Randomly select a mini-batch of samples;
      (2) Update auxiliary variable $\boldsymbol{v}_i$'s for all samples in the mini-batch according to (15);
      (3) Update all class centers based on (16);
      (4) $N_c = N_c + 1$;
      (5) Based on the objective in (17), update the weight vectors $\boldsymbol{w}_k$'s in the classification module and all trainable parameters $\boldsymbol{\Theta}$ in the feature learning module by mini-batch SGD.
    end for
  end if
end while

$\frac{1}{1-\phi^{N_c}}$ approaches 1, and its impact becomes negligible. After t_1th epoch, when we get $\boldsymbol{v}$'s for all samples in a mini-batch, we can update the kth class center based on

$$\boldsymbol{c}_k = r \frac{\sum_{i=1}^{m} \tau(k, y_i) \left(\boldsymbol{v}_i - \frac{\beta}{C-1} \sum_{l \neq y_i}^{C} \boldsymbol{c}_l \right)}{\| \sum_{i=1}^{m} \tau(k, y_i) \left(\boldsymbol{v}_i - \frac{\beta}{C-1} \sum_{l \neq y_i}^{C} \boldsymbol{c}_l \right) \|_2} \tag{16}$$

where $k \in [1, \ldots, C]$.

2) Update Trainable Parameters: In the second step, we fix all class centers $\boldsymbol{c}_k$'s and update the weight vectors $\boldsymbol{w}_k$'s in the classification module and all trainable parameters $\boldsymbol{\Theta}$ in the feature learning module. For the current mini-batch, the loss function is

$$\min_{\boldsymbol{w}_k, \boldsymbol{\Theta}} - \sum_{i=1}^{m} \log \frac{\exp(\boldsymbol{w}_{y_i}^{\top} \boldsymbol{f}_i)}{\sum_{k=1}^{C} \exp(\boldsymbol{w}_k^{\top} \boldsymbol{f}_i)} + \frac{\lambda}{2m} \sum_{i=1}^{m} \{ [\iota(\| \boldsymbol{f}_i - \boldsymbol{c}_{y_i} \|_2) \| \boldsymbol{f}_i - \boldsymbol{c}_{y_i} \|_2]^2 \}. \tag{17}$$

In objective (17), all $\boldsymbol{c}_{y_i}$'s are known, $\boldsymbol{w}_k$'s and $\boldsymbol{\Theta}$ together give all trainable parameters. Therefore, we can directly use mini-batch SGD to train the model and update these parameters.

In conclusion, we summarize the framework of our proposed algorithm in Algorithm 1.

IV. Experiments

In this section, we conduct several experiments to evaluate the performance of our proposed algorithm. The datasets and settings are given in Section IV-A. The properties of our method are investigated in Sections IV-B–IV-D. Afterward, an ablation study is shown in Section IV-E. Subsequently, the comparison with other approaches is demonstrated in Sections IV-F–IV-H. In addition, a supplementary experiment is shown in Appendix C of the Supplementary Material to illustrate the intraclass compactness and interclass separability of our method.

TABLE I
Datasets

Dataset	Image size	Total samples	Samples in test set	Classes
CIFAR-10	32×32	60 000	10 000	10
CIFAR-100	32×32	60 000	10 000	100
Tiny ImageNet	224×224	120 000	20 000	200
CASIA-WebFace	112×96	494 414	0	10 575
LFW	112×96	13 233	13 233	5749
CFP-FP	112×96	7000	7000	500
AgeDB-30	112×96	12 240	12 240	570
VOC 07+12	600×600	16 492	4952	20
RPC	600×600	5422	542	113

A. Datasets and Settings

First, the proposed algorithm is evaluated on three standard image classification datasets: CIFAR-10 [20], CIFAR-100 [20], Tiny ImageNet [21]. The settings of the training set and test set follow the common practice in these datasets, which are given in Table I. For CIFAR-10 and CIFAR-100, ResNet18 is equipped as the backbone, while ResNet34 is used for Tiny ImageNet. The mini-batch SGD with batch size 256, momentum 0.9, and weight decay 5×10^{-4} is used in all experiments. For all these three datasets, all algorithms are trained for 240 epochs. The initial learning rate is 0.1 and divided by 10 at the 120th and 180th epoch.

To further investigate the performance of our proposed algorithm, we conduct face verification on three different datasets. Unlike standard image classification tasks, face verification is an open-set problem. Specifically, models are trained for image classification on the CASIA-WebFace dataset [22]. After that, the trained feature learning modules will be used to conduct face verification on LFW [12], CFP-FP [23], and AgeDB-30 [24]. The details of the face verification datasets are also given in Table I. For this task, the 20-layer SphereFace [6] structure is equipped as the backbone. The training process includes 40 epochs for all methods. Moreover, the learning rate starts from 0.1 and is divided by 10 at the 20th and 30th epoch.

Besides, we also investigate the performance of our proposed algorithm in two practical applications: instance-level object detection and retail product checkout. For instance-level object detection, the Pascal VOC dataset is utilized. The training set is formed by combining the trainval 2007 and trainval 2012 datasets, while the test2007 dataset is used for evaluation [25]. The retail product checkout (RPC) dataset[2] is used in another practical experiment. The details of these datasets are also presented in Table I. In both tasks, Faster R-CNN structure [26] is employed as the backbone. The training process includes 100 epochs. The learning rate, initially set at 0.0001, is multiplied by 0.6 every 10 epochs.

[2]https://aistudio.baidu.com/datasetdetail/91732/0

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE II
AVERAGE ACCURACY (%) OF 5 TRIES FOR VARIOUS λ'S AND β'S ON CIFAR-10 AND CIFAR-100 DATASETS

Accuracy			β			
			0.001	0.01	0.1	1
CIFAR-10	λ	0.001	95.34	95.46	95.46	95.51
		0.002	95.38	95.32	**95.76**	95.35
		0.005	95.47	95.10	95.10	95.30
		0.01	95.20	94.74	95.16	94.98
		0.02	95.52	94.37	94.81	95.05
		0.05	95.20	93.03	94.12	94.36
		0.1	95.25	91.70	92.14	91.43
Accuracy			β			
			0.001	0.01	0.1	1
CIFAR-100	λ	0.001	78.23	78.28	78.12	78.13
		0.002	77.56	77.77	**78.59**	78.51
		0.005	77.8	77.74	78.08	78.19
		0.01	77.69	77.02	77.57	76.4
		0.02	77.61	76.27	76.65	68.66
		0.05	77.18	72.77	72.12	09.12
		0.1	77.4	68.48	66.78	11.01

In all experiments, each method was tested 5 times. According to the setting in [12], we let $r = 40$ in our proposed algorithm and the comparison methods that utilize this hyperparameter. The feature dimension $d = 512$ in image classification tasks and face verification tasks, while for the practical applications, the feature dimension $d = 2048$.

B. Hyperparameter Discussion

λ and β are two tradeoff parameters in the hypersphere loss, which can be used to balance the softmax loss, the generalized center penalty function, and the class center separation function. The datasets CIFAR-10 and CIFAR-100 are used to discuss the influence of λ and β. We try $\lambda \in \{0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1\}$ and $\beta \in \{0.001, 0.01, 0.1, 1\}$. The corresponding results are shown in Table II. For both CIFAR-10 and CIFAR-100, the proposed algorithm achieves the best result when $\lambda = 0.002$ and $\beta = 0.1$. Hence, we use this setting in all the following experiments. For CIFAR-10, the tradeoff parameter λ can be selected from 0.001 to 0.05, and β can be selected from 0.001 to 1. For CIFAR-100, the λ can be adjusted from 0.001 to 0.01 and the corresponding β can be selected from 0.001 to 1. As a result, our proposed algorithm exhibits relatively stable performance across these parameter variations. This suggests that, in practical applications, we can efficiently select suitable values for λ and β.

C. Performance Comparison on Different Backbones

In this experiment, the influence of hypersphere loss on various feature learning modules is investigated. We combine the hypersphere loss (or the original softmax) with nine different feature learning modules, including ResNet18, DenseNet121 [8], DLA [27], EfficientNet [9], MobileNet [10], PreActResNet18 [28], PyramidNet [29], SimpleDLA [27], and Transformer [30]. The corresponding results are shown in Table III. In this table, "Transformer w/ PT" and"Transformer w/o PT," respectively, denote a Transformer model pretrained with the ImageNet-21 k dataset and a Transformer model without pretraining. It is evident that the proposed hypersphere loss can be integrated with various backbone models and consistently outperforms the original softmax in all cases for both CIFAR-10 and CIFAR-100. Specifically, for CIFAR-10, the hypersphere loss enhances the performance of DLA modules with softmax from 95.62% to 96.07%, and it also significantly boosts the performance of EfficientNet modules with softmax from 90.65% to 93.95%. Similar improvements are observed on CIFAR-100. For example, the performance of DLA modules with softmax loss is 76.84%, but when the softmax is replaced by hypersphere loss, the mean accuracy increases to 78.61%. For transformer models, whether pretrained or not, the use of hypersphere loss can improve their performance to some extent.

TABLE III
AVERAGE INACCURACIES (%) OF HYPERSPHERE LOSS AND SOFTMAX LOSS WITH DIFFERENT FEATURE LEARNING MODULES ON CIFAR-10 AND CIFAR-100 DATASETS

Average accuracy (%)		Hypersphere loss	Softmax loss
CIFAR-10	ResNet18	**95.76**	93.68
	DenseNet121	**95.91**	95.63
	DLA	**96.07**	95.62
	EfficientNet	**93.95**	90.65
	MobileNet	**92.71**	91.77
	PreActResNet18	**94.92**	94.56
	PyramidNet	**95.77**	95.40
	SimpleDLA	**95.32**	95.02
	Transformer w/ PT	**98.24**	98.21
	Transformer w/o PT	**73.04**	71.97
Average accuracy (%)		Hypersphere loss	Softmax loss
CIFAR-100	ResNet18	**78.59**	75.66
	DenseNet121	**79.84**	78.96
	DLA	**78.61**	76.84
	EfficientNet	**71.89**	70.32
	MobileNet	**70.93**	69.00
	PreActResNet18	**78.10**	76.14
	PyramidNet	**79.76**	78.86
	SimpleDLA	**77.36**	76.17
	Transformer w/ PT	**90.62**	90.54
	Transformer w/o PT	**48.45**	44.81

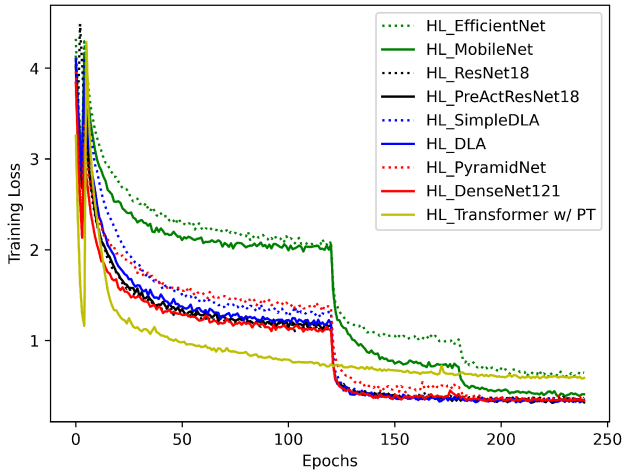

Fig. 2. Convergence on CIFAR-100.

D. Convergence and Complexity Analysis

In the following, we investigate the convergence of our proposed algorithm. The training losses of our proposed method on different backbones are illustrated in Fig. 2. It can be observed

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE IV
COMPLEXITY ANALYSIS ON TINY IMAGENET

Method	Training time (h)	Extra param. (K)	Extra FLOPs (M)	Max. GPU memory (G)
Softmax	7.77	100	50.00	5.261
ASL	8.03	100	52.18	5.307
ER	7.98	100	89.77	5.673
CL	7.97	300	50.81	5.675
CCL	14.17	200	50.13	5.914
HML	11.89	100	50.00	5.261
arcface	8.23	200	101.18	5.589
cosface	9.67	200	100.84	5.675
MML	11.96	200	101.28	5.636
HL	10.20	200	51.46	5.277
HL(10)	7.80	200	50.46	5.274

that the proposed approach, regardless of the backbone used, consistently converges within 240 epochs.

Next, we perform a complexity analysis for each algorithm using two NVIDIA GeForce RTX 3090 graphics cards. All experiments are conducted on the Tiny ImageNet dataset using a ResNet34 backbone. The results presented in Table IV are the averages of five independent replicate trials. In this table, the first column lists the methods for complexity analysis, and their full names are provided in Section IV-F. The second column shows the training time of various algorithms. It is worth noting that their inference time is similar to each other, so we omit them. On average, the proposed algorithm requires approximately 10.2 h for training. HL(10) is the proposed algorithm that updates the class center at every ten mini-batch iterations. It takes roughly 7.8 h on average for training. However, in this case, the mean accuracy of HL(10) decreases from 65.88% to 65.72%.

The third and fourth columns of TABLE IV present the number of extra parameters and floating-point operations (FLOPs) introduced by the classification module of each method. In this experiment, the feature learning module (ResNet34) contains approximately 21.55 M parameters, and the total number of FLOPs during its forward propagation is around 941.69 G. As the table shows, the extra parameters and FLOPs of all classification modules are much smaller than that of the feature learning module. For instance, the HL method introduces approximately 200 K extra parameters, which is roughly 0.1% of the backbone's parameter count, and its extra FLOPs are 51.46 M, roughly 0.005% of the backbone's FLOPs. In addition, compared to the original softmax, all other algorithms have a certain increase in the number of extra parameters and FLOPs, but this does not significantly affect the overall complexity.

The last column of the table provides the maximum GPU memory usage for each method. It is observed that, compared to the softmax, none of the other methods significantly increase the maximum GPU memory consumption. The CCL method has the highest GPU memory consumption, and the experiments show that inappropriate hyperparameter settings for CCL may easily lead to memory overflow.

E. Ablation Studies

The main characteristics of the proposed method include the following.

TABLE V
ABLATION STUDIES ON CIFAR-10

Generalized class center	Class center separation	Class center updating way	Mean accuracy
×	×	Per epoch using all training set	95.24
×	×	Per epoch by EWMA	94.29
×	✓	Per epoch by EWMA	94.42
×	✓	Per mini-batch by EWMA	94.95
✓	×	Per epoch by EWMA	95.09
✓	×	Per mini-batch by EWMA	95.28
✓	✓	Per epoch by EWMA	95.39
✓	✓	Per mini-batch by EWMA	**95.76**

1) The class center separation term is introduced.
2) The generalized class center is devised.
3) The class centers are updated by EWMA in each mini-batch instead of using all samples in the training set to update class centers in each epoch.

In this section, we will conduct ablation studies on the proposed algorithm. CIFAR-10 is used as the evaluation dataset, and ResNet18 is used as the backbones. The results are shown in Table V. It is observed that when the class centers are updated using the equation in (9) with all samples in the training set per epoch, the verification accuracy is 95.24%. However, if we update the class centers by EWMA per epoch, the performance will decrease to 94.29%. By introducing the class center separation term, we can further increase the accuracy to 94.42%. Next, by increasing the update frequency of class centers, the accuracy of the proposed algorithm can be further improved to 94.95%. Finally, by introducing the generalized class center term, the performance of the proposed algorithm can achieve 95.76%. Specifically, in Table V, the only difference between the experiments in row 2 and row 5 is whether to use the generalized class center. The comparative analysis of the two rows indicates that the generalized class center outperforms the traditional class center. This observation is further corroborated by comparing the outcomes of rows 3 and 7, and rows 4 and 8, respectively. Moreover, comparative analysis of rows 2 and 3, 5 and 7, and 6 and 8, reveals that the introduction of the class center separation term also contributes to enhancing the performance of the proposed algorithm. In summary, EWMA is utilized to improve the efficiency of the proposed algorithm, but it may influence its accuracy performance. By introducing the class center separation term and the generalized class center, the performance of our proposed algorithm is largely improved.

F. Performance Comparison on Image Classification

In the following, we compare our proposed algorithm with several baselines or state-of-the-art methods, including the original softmax, angular softmax loss (ASL) [14], exclusive regularity (ER) [5], center loss (CL) [16], constrained center

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE VI
VERIFICATION ACCURACIES (%) FOR VARIOUS ALGORITHMS ON THREE IMAGE CLASSIFICATION DATASETS OVER 5 RUNS

Method	CIFAR-10			CIFAR-100			Tiny-ImageNet		
	Best	Mean	Std.	Best	Mean	Std.	Best	Mean	Std.
softmax	94.31	93.68	0.5568	75.86	75.66	0.3215	63.38	63.11	0.0907
ASL	95.46	95.27	0.1366	75.47	74.88	0.5410	62.47	62.12	0.2754
ER	95.53	95.40	0.1162	77.29	77.05	0.2292	65.62	65.37	0.2247
CL	95.37	95.30	0.0693	77.51	77.42	0.1415	65.60	65.34	0.2272
CCL	95.42	95.24	0.1609	78.11	77.74	0.4067	65.67	65.51	0.1704
MML	93.99	93.90	0.0699	75.53	75.13	0.2501	62.14	61.82	0.2308
HML	94.31	94.20	0.1010	73.84	72.44	1.5092	62.45	61.44	0.8229
Cosface	94.23	93.93	0.2180	75.45	75.03	0.2631	62.30	61.91	0.4230
Arcface	94.93	94.78	0.0907	77.82	77.40	0.2858	64.99	64.83	0.1527
HL	**95.95**	**95.76**	0.0964	**78.82**	**78.59**	0.2518	**66.02**	**65.88**	0.2203

TABLE VII
ACCURARIES (%) FOR VARIOUS ALGORITHMS ON THREE FACE VERIFICATION DATASETS OVER 5 RUNS

Method	LFW			CFP-FP			AgeDB-30		
	Best	Mean	Std.	Best	Mean	Std.	Best	Mean	Std.
softmax	98.33	98.25	0.0681	92.56	92.39	0.2157	89.13	88.66	0.2973
ASL	98.58	98.48	0.0714	94.05	93.67	0.2361	88.95	88.50	0.2991
ER	98.47	98.39	0.0545	92.82	92.59	0.1844	88.88	88.71	0.1872
CT	99.13	99.07	0.0505	95.74	95.63	0.1151	91.50	91.41	0.0770
CCL	99.07	98.97	0.1037	96.07	95.37	0.7396	91.46	91.10	0.2200
MML	99.13	99.03	0.0825	94.06	93.83	0.1999	91.10	90.92	0.1571
HML	98.02	97.89	0.1090	91.57	91.12	0.3720	87.25	87.00	0.3156
Cosface	98.68	98.55	0.1149	93.32	93.16	0.1277	89.32	89.13	0.2072
Arcface	98.50	98.32	0.1970	93.18	92.87	0.3054	89.35	88.82	0.3558
MagFace	**99.35**	**99.23**	0.0887	94.85	94.58	0.2037	92.42	92.16	0.2149
HL	99.25	99.12	0.1120	**96.19**	**95.91**	0.0229	**92.49**	**92.33**	0.1142

loss (CCL) [12], Arcface [14], Cosface [15], hard-mining loss (HML) [31], and multimargin loss (MML) [32]. In all comparison experiments, we run the experiments five times and report the results in "best accuracy, mean accuracy, standard deviation" as in [7]. The results on CIFAR-10, CIFAR-100, and Tiny ImageNet are given in Table VI. The best result is marked in bold for each dataset, while the second-best result is marked with an underline. From Table VI, our proposed classification module outperforms all other methods on all these three datasets. For instance, the proposed algorithm's best accuracy and mean accuracy are 95.95% and 95.76% on CIFAR-10. The second best approach is achieved by the ER method; its best accuracy and mean accuracy are, respectively, 95.53% and 95.40%. For CIFAR-100 and Tiny ImageNet, the performance of the hypersphere loss is also better than other approaches. The second-best accuracies for these two datasets are given by CCL. In addition, our proposed algorithm exhibits low standard deviation, indicating its effectiveness and stability across all datasets.

G. Performance Comparison on Face Verification

In this section, we investigate the performance of our proposed algorithm on face verification, which is an open-set problem. The training is conducted on the CASIA-WebFace dataset, and the face verification on LFW, CFP-FP, and AgeDB-30 is used for testing. The standard protocol of unrestricted with labeled outside data [33] is utilized in this experiment. We compare the proposed algorithm with various methods, including the original softmax, ASL [14], ER [5], CL [16], CCL [12], MML [32], HML [31], Cosface [15], Arcface [14], and Magface [19]. We also run the experiments 5 times and report the results in "best accuracy, mean accuracy, standard deviation." The results are shown in Table VII. For LFW, MagFace achieves the best result with the best accuracy of 99.35% and mean accuracy of 99.23%, while the second best approach is our proposed algorithms with the best accuracy of 99.25% and mean accuracy of 99.12%. For CFP-FP, our proposed approach obtains the best result 96.19%, while the second best method is CCL with 96.07%. For AgeDB-30, the proposed approach is the best with an accuracy of 92.49%, while the second best method is MagFace with 92.42%.

H. Performance Comparison on Practical Applications

In the following, the effectiveness of our algorithm is evaluated in two practical tasks: 1) instance-level object detection and 2) retail product checkout. Both tasks require identifying appropriate candidate boxes in target images and classifying the objects within these candidate boxes. Our proposed method and other comparative algorithms can be used to improve classification accuracy in this process. We evaluate the algorithms using two criteria. First, the standard mean of average precisions (AP) at $IoU = 0.5$[3] is introduced to evaluate the overall model performance. Second, the classification accuracy of the most suitable candidate boxes for each target object in the image is used to evaluate the classification modules. The experimental results, including best values, average values, and standard deviations

[3]Intersection over Union (IoU): In object detection, it is used to determine how well a predicted bounding box aligns with the ground truth bounding box for an object.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE VIII
PERFORMANCE ON OBJECT DETECTION

Method	$AP_{0.5}$ (%)			Accuracy (%)		
	Best	Mean	Std.	Best	Mean	Std.
softmax	78.26	78.17	0.0896	85.11	84.90	0.1757
ASL	80.09	79.95	0.1014	65.46	63.90	1.1234
ER	81.41	80.93	0.3962	87.78	87.44	0.3990
CL	80.66	80.51	0.2168	86.54	86.30	0.1801
CCL	81.88	81.74	0.1143	88.74	88.51	0.1965
MML	75.77	75.58	0.1370	87.02	86.82	0.2666
HML	69.98	69.32	0.6871	69.07	68.63	0.3088
Cosface	78.54	77.95	0.4337	84.00	83.83	0.1320
Arcface	80.61	80.19	0.3516	80.47	80.05	0.3561
HL	**82.35**	**81.95**	0.2859	**89.13**	**88.92**	0.2299

TABLE IX
PERFORMANCE ON RETAIL PRODUCT CHECKOUT

Method	$AP_{0.5}$ (%)			Accuracy (%)		
	Best	Mean	Std.	Best	Mean	Std.
Softmax	96.46	96.24	0.2085	72.27	71.74	0.6685
ASL	98.21	98.17	0.0294	50.49	49.86	0.4813
ER	98.83	98.70	0.1420	84.63	83.77	0.6685
CL	98.89	98.81	0.0655	74.48	73.24	0.9501
CCL	98.74	98.69	0.0685	83.78	83.72	0.0450
MML	96.86	96.74	0.0834	68.03	66.27	1.6139
HML	89.70	89.41	0.2499	54.09	53.95	0.1053
CosFace	98.08	97.96	0.1281	73.18	72.68	0.4749
ArcFace	98.17	98.10	0.0544	76.61	75.51	0.9793
HL	**99.06**	**99.03**	0.0245	**86.50**	**86.23**	0.3091

from five trials, are presented in Tables VIII and IX. The best result is marked in bold for each criterion, while the second-best result is underlined. It is evident from these tables that our proposed algorithm outperforms other methods in both $AP_{0.5}$ and classification accuracy. For instance, in the instance-level object detection, HL achieves the highest $AP_{0.5}$ of 82.35% and the best accuracy of 89.13%. The corresponding results of the second-best approach, CCL, are 81.88% and 88.74%, respectively. Under the same conditions, the best value of $AP_{0.5}$ and classification accuracy for the softmax method are 78.26% and 85.11%. Furthermore, the small standard deviation of the HL algorithm, as shown in Tables VIII and IX, indicates its stability in both the object recognition and retail product checkout tasks. Visualizations of the results for these practical tasks can be found in Appendix D of the Supplementary Material.

V. CONCLUSION

In this article, a hypersphere loss was proposed, which can be used to improve intraclass compactness and interclass separability of feature vectors in image classification tasks. To improve the interclass separability, a class center separation function was devised. A generalized class center was utilized to handle the inevitable disturbances of samples in the same class. Furthermore, the constraints in the hypersphere loss can make class centers distribute on the surface of a hypersphere with a controllable radius. An alternative learning strategy was utilized for training the model. First, all trainable parameters in the deep learning model were fixed, and we calculated class centers based on EWMA method. Second, the generalized class centers given by the first step were fixed, and we updated the trainable parameters in the deep learning model by mini-batch SGD. We conducted experiments to analyze our algorithm's properties and compared its performance across various standard image classification tasks, face verification tasks, and two practical applications. The experimental results demonstrated that hypersphere loss, as a general framework, could be applied to diverse tasks that involve image classification. In the future, we will research techniques for adaptively adjusting feature hypersphere radii based on feature distribution to enhance model performance. In addition, we will explore innovative methods for computing class centers, aiming to mitigate the impact of approximate methods like EWMA on accuracy.

REFERENCES

[1] J. Wen et al., "Robust sparse linear discriminant analysis," *IEEE Trans. Circle Syst. Video Technol.*, vol. 29, no. 2, pp. 390–403, Feb. 2019.
[2] X. Chang et al., "Convex sparse PCA for unsupervised feature learning," *ACM Trans. Knowl. Discl. From Data*, vol. 11, no. 1, pp. 1–16, 2016.
[3] X. Zhe, S. Chen, and H. Yan, "Directional statistics-based deep metric learning for image classification and retrieval," *Pattern Recognit.*, vol. 93, pp. 113–123, 2019.
[4] X. Sun, S. Yang, and C. Zhao, "Lightweight industrial image classifier based on federated few-shot learning," *IEEE Trans. Ind. Inform.*, vol. 19, no. 6, pp. 7367–7376, Jun. 2022.
[5] K. Zhao, J. Xu, and M.-M. Cheng, "Regularface: Deep face recognition via exclusive regularization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1136–1144.
[6] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 212–220.
[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[8] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4700–4708.
[9] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.
[10] A. G. Howard et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.
[11] Y. Liu, H. Li, and X. Wang, "Rethinking feature discrimination and polymerization for large-scale recognition," 2017, *arXiv:1710.00870*.
[12] Z. Shi, H. Wang, and C.-S. Leung, "Constrained center loss for convolutional neural networks," *IEEE Trans. Neur. Netw. Learn. Syst.*, vol. 34, no. 2, pp. 1080–1088, Feb. 2021.
[13] O. Rippel et al., "Metric learning with adaptive density discrimination," 2015, *arXiv:1511.05939*.
[14] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4690–4699.
[15] H. Wang et al., "Cosface: Large margin cosine loss for deep face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5265–5274.
[16] Y. Wen et al., "A discriminative feature learning approach for deep face recognition," in *Proc. Eur. Conf. Comput. Vis*, 2016, pp. 499–515.
[17] W. Wang et al., "Push for center learning via orthogonalization and subspace masking for person re-identification," *IEEE Trans. Image Process.*, vol. 30, pp. 907–920, 2021.
[18] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: L2 hypersphere embedding for face verification," in *Proc. 25th ACM Int. Conf. Multimedia*, 2017, pp. 1041–1049.
[19] Q. Meng, S. Zhao, Z. Huang, and F. Zhou, "Magface: A universal representation for face recognition and quality assessment," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 14225–14234.
[20] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," M.S. thesis, Univ. Toronto, ON, Canada, 2009.
[21] O. Russakovsky et al., "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[22] D. Yi et al., "Learning face representation from scratch," *CoRR*, vol. abs/1411.7923, 2014.
[23] S. Sengupta, J.-C. Chen, C. Castillo, V. M. Patel, R. Chellappa, and D. W. Jacobs, "Frontal to profile face verification in the wild," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2016, pp. 1–9.
[24] S. Moschoglou, A. Papaioannou, C. Sagonas, J. Deng, I. Kotsia, and S Zafeiriou, "AGEDB: The first manually collected, in-the-wild age database," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 51–59.
[25] Y. Zhong, J. Wang, L. Wang, J. Peng, Y.-X. Wang, and L. Zhang, "DAP: Detection-aware pre-training with weak supervision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 4537–4546.
[26] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015.
[27] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2403–2412.
[28] K. He et al., "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis*, 2016, pp. 630–645.
[29] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2117–2125.
[30] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy
[31] Y. Srivastava, V. Murali, and S. R. Dubey, "Hard-mining loss based convolutional neural network for face recognition," in *Proc. 5th Int. Conf. Comput. Vis. Image Process.*, 2021, pp. 70–80.
[32] J. Li et al., "Combined angular margin and cosine margin softmax loss for music classification based on spectrograms," *Neural Comput. Appl.*, vol. 34, no. 13, pp. 10337–10353, 2022.
[33] G. B. Huang and E. Learned-Miller, "Labeled faces in the wild: Updates and new reporting procedures," Dept. Comput. Sci., Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep, vol. 14, no. 003, 2014.

Hao Wang received the Ph.D. degree in electronic engineering from the City University of Hong Kong, Hong Kong, in 2019.

He is with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen, China. His current research interests include deep learning, machine learning, and optimization.

Jinpeng Cao received the B.S. degree in electronic information engineering from the College of Electronics and Information Engineering, Shenzhen University, Shenzhen, China, in 2022, where he is currently working toward the master's degree in electronic information technology.

His current research interests include face recognition and metric learning.

Zhang-Lei Shi received the Ph.D. degree in electronic engineering from the Department of Electrical Engineering, City University of Hong Kong, Hong Kong SAR, China, in 2021.

He is currently a Lecturer with the College of Science, China University of Petroleum (East China), Qingdao, China. His current research interests include sparse optimization, neural networks, and machine learning.

Chi-Sing Leung (Senior Member, IEEE) received the Ph.D. degree in computer science from the Chinese University of Hong Kong, Hong Kong, in 1995.

He is currently a Professor with the Department of Electronic Engineering, City University of Hong Kong. He has authored or coauthored more than 120 journal papers in the areas of digital signal processing, neural networks, and computer graphics. His research interests include neural computing and computer graphics.

Dr. Leung was the recipient of the 2005 IEEE Transactions on Multimedia Prize Paper Award for his paper titled, "The Plenoptic Illumination Function." He was a Member of Organizing Committee of ICONIP2006. He was the Program Chair of ICONIP2009 and ICONIP2012. He is a governing board member of the Asian Pacific Neural Network Assembly (APNNA) and Vice President of APNNA. He is/was the Guest Editor for several journals, including *Neural Computing and Applications*, *Neurocomputing*, and *Neural Processing Letters*.

Ruibin Feng received the Ph.D. degree in electronic engineering from the City University of Hong Kong, Hong Kong, in 2017.

He is currently a Research Scientist with Stanford University, CA, USA. His primary research interests include designing and developing innovative AI algorithms and systems, and their application in reliable, rapid, and efficient diagnostic procedures and treatment guidance, with a special emphasis on patients suffering from cardiac arrhythmias.

Wenming Cao (Senior Member, IEEE) received the Ph.D. degree in automation from the School of Automation, Southeast University, Nanjing, China, in 2003.

He is currently a Professor with Shenzhen University, Shenzhen, China. He has authored or coauthored over 80 publications in top-tier conferences and journals. His research interests include pattern recognition, image processing, and visual tracking.

Yuxin He received the Ph.D. degree in data science from the City University of Hong Kong, Hong Kong, in 2020.

She is currently with the College of Urban Transportation and Logistics, Shenzhen Technology University, Shenzhen, China. Her current research interests include spatiotemporal data mining, forecasting modeling, and intelligent transportation.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 29,2024 at 05:40:21 UTC from IEEE Xplore. Restrictions apply.