Key Point Sensitive Loss for Long-Tailed Visual Recognition

Mengke Li, Yiu-Ming Cheung, *Fellow, IEEE*, and Zhikai Hu

Abstract—For long-tailed distributed data, existing classification models often learn overwhelmingly on the head classes while ignoring the tail classes, resulting in poor generalization capability. To address this problem, we thereby propose a new approach in this paper, in which a key point sensitive (KPS) loss is presented to regularize the key points strongly to improve the generalization performance of the classification model. Meanwhile, in order to improve the performance on tail classes, the proposed KPS loss also assigns relatively large margins on tail classes. Furthermore, we propose a gradient adjustment (GA) optimization strategy to re-balance the gradients of positive and negative samples for each class. By virtue of the gradient analysis of the loss function, it is found that the tail classes always receive negative signals during training, which misleads the tail prediction to be biased towards the head. The proposed GA strategy can circumvent excessive negative signals on tail classes and further improve the overall classification accuracy. Extensive experiments conducted on long-tailed benchmarks show that the proposed method is capable of significantly improving the classification accuracy of the model in tail classes while maintaining competent performance in head classes.

Index Terms—Long-tailed classification, long-tailed visual recognition, class imbalance, imbalance learning

1 Introduction

Visual recognition problems have achieved immense success, thanks to the advent of deep convolutional neural networks (CNNs) and the availability of large-scale, high-quality annotated datasets such as ImageNet ILSVRC 2012 [1] and Places [2]. In such datasets, both the training and testing data have been artificially balanced. That is, each class has roughly the same number of training samples. However, from the practical point of view, the number of samples for different classes varies greatly due to the different difficulties in data collection. As a result, real-world datasets generally have skewed distributions with a long tail [3], [4]. That is, a few dominant categories (called head classes) occupy most of the samples, while most of the remaining categories (called tail classes) are associated with rarely few samples. Nevertheless, small sample size does not mean that the tail classes are unimportant. For example, when classifying mammals, training samples of endangered animals such as tigers and snow leopards are more difficult to obtain than those of common animals like cats and dogs. It is still need to be correctly classified when the given query belongs endangered animal. Therefore, in order to show the equal importance of each class, their sample size should be roughly the same during the test stage even if the training data is long-tailed. Unfortunately, training on long-tailed data will raise a problem, i.e., a biased learning process for the classification model, because the instance-rich head classes usually contribute to an overwhelmingly large quantity of negative samples for tail classes. Consequently, the learned classification model tends to have serious poor performance in the tail classes during testing.

A straightforward solution to address the issue of long-tailed data is re-balancing the distribution of different classes to mitigate the extreme imbalance of the training set. In the literature, two such representative techniques are re-sampling and re-weighting. Re-sampling methods usually sample the training images of different classes with different variants of sampling rates to make the class-wise sample sizes roughly balanced. Existing commonly used re-sampling methods include under-sampling ones (e.g., see [5], [6], [7], [8]), which randomly remove training samples from head classes, and over-sampling ones (e.g., see [9], [10], [11], [12]), which randomly replicate training samples from tail classes. Re-weighting methods (e.g., see [13], [14], [15], [16]) balance the contribution proportion of each class to the classification model through the multiplicative parameters on the loss function. To increase the impact of tail classes in the training process, the multiplicative parameters are inversely proportional to the number of class samples. It is expected that these two techniques can make the distribution of training data closer to that of the testing data which is uniformly distributed. One limitation of the re-balancing methods, however, is over-fitting on tail classes because of duplicated training on the tail class samples that provide essentially insufficient information. There are several attempts, such as class-level re-weighting [17], [18], [19] and re-margining [20], [21], [22], to alleviate this issue. Recently, Cui et al. [14] proposed to re-weight the softmax cross-entropy loss by the "effective number" of each class. Cao et al. [20] adopted both re-weighting and re-sampling techniques to train a

• *The authors are with the Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China. E-mail: {csmkli, ymc, cszkhu}@comp.hkbu.edu.hk.*

Manuscript received 4 October 2021; revised 5 March 2022; accepted 29 July 2022. Date of publication 3 August 2022; date of current version 6 March 2023.
This work was supported in part by NSFC/Research Grants Council (RGC) Joint Research Scheme under Grant N_HKBU214/21, in part by General Research Fund of RGC under Grant 12201321, and in part by Hong Kong Baptist University under Grant RC-FNRA-IG/18-19/SCI/03.
(Corresponding author: Yiu-Ming Cheung.)
Recommended for acceptance by O. Russakovsky.
Digital Object Identifier no. 10.1109/TPAMI.2022.3196044

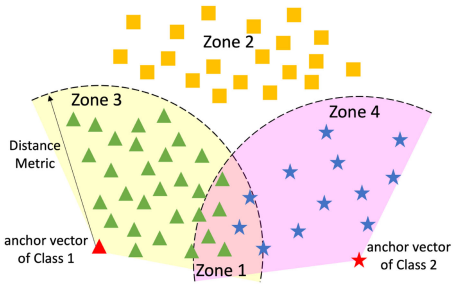
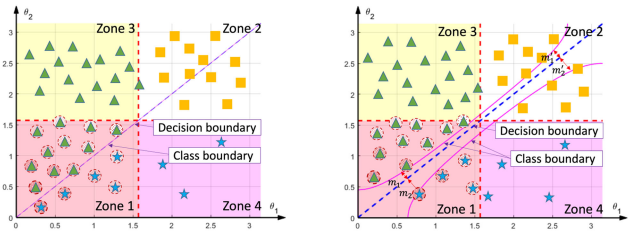


Fig. 1. Different kinds of points in feature space. *Key points*: the points located in Zone 1 have small distance from the anchor vectors of both Class 1 and Class 2. *Non-key points*: the points falling in Zone 2 are far away from the anchor vectors of Class 1 and Class 2. *Simple points*: the points located in Zone 3 (or Zone 4) have smaller distance with the anchor vector of Class 1 (or Class 2).



(a) Class boundary of softmax cross-entropy loss function.

(b) Class boundary of the loss function in Cao *et al.*'s work [20].

Fig. 2. Class boundaries of different loss functions, where angular distance is used. The θ_1 axis represents angular distance between sample features and the anchor vector of Class 1, while the θ_2 axis is for the angular distance of Class 2. Shaded points with red textures represent key points. The denser the texture, the more important this is.

neural network. They used a label-dependent regularizer to re-weight the tail classes stronger than the head classes and trained the network by a deferred re-sampling strategy.

Empirical studies have shown the success of the aforementioned approaches in their application domains, but they all ignore the differences between the points in embedding space and treat them equally. In fact, different kinds of points in embedding space may contribute differently to the classification model from the practical viewpoint. Hence, we define three kinds of points based on their importance. Based on the fact that if the two classes that are most likely to be misclassified from each other can be predicted correctly, the remaining classes are relatively easy to be classified, we use these two classes for illustration without loss of generality. We name these two most difficult classes Class 1 and Class 2, and the remaining classes are other classes. As shown in Fig. 1, when assigning labels for the points in Class 1 and Class 2, the three kinds of points are:

- *Key point*: The points located in Zone 1 have a small distance from the anchor vectors of both Class 1 and Class 2. These points are most likely to be misclassified, because they are close to the anchor vectors of both Class 1 and Class 2. We call this kind of point as *key point*, which should be paid more attention to.
- *Non-key point*: The points falling in Zone 2 are far away from the anchor vectors of Class 1 and Class 2. This kind of point belongs to other classes, because it is far away from the anchor vectors of Class 1 and Class 2. We need not pay much attention to them. We call the points in Zone 2 as *non-key points*.
- *Simple point*: The points located in Zone 3 (or Zone 4) have a smaller distance with the correct anchor vector. This kind of point is relatively easy to be classified. We name the points in Zone 3 and Zone 4 as *simple points*.

Increasing the overall classification accuracy of the classification model require to make key points more separable. Therefore, key points should be regularized more strongly.

Based on the above analysis, we introduce the margins which are the distance between the class boundary to the decision boundary[1] to treat the aforementioned three kinds of points differently. In order to improve the separability of key points, their margin needs to be larger than that of non-key points. Unfortunately, most existing methods ignore this. Fig. 2 shows the margins in the existing methods. For a clear visualization, we use the angular distance as the metric and transform the points into the coordinate axis. We can see that most previous methods (for example, see [8], [23], [24]) use the basic softmax cross-entropy loss, which ignores margins. Their class boundary coincides with the decision boundary, which is shown in Fig. 2a. In the literature, Cao et al. 's work [20] take the margins into consideration, but it allocates the same margins between key and non-key points, namely $m_1 = m_1'$ and $m_2 = m_2'$, as shown in Fig. 2b.

In this paper, we propose a key point sensitive (KPS) loss that makes the key points more separable as well as increases the model performance on tail classes. To make the samples with key points easier to separate, we regularize the key points strongly by multiplying the proposed label-dependent factors on the predicted scores of the points. In order to increase the classification accuracy on tail classes, the proposed loss simultaneously encourage relatively large margins between points of tail versus head classes through the class-based margins. Besides the modification of the loss function, we also design a gradient adjustment optimization strategy. In the loss of softmax cross-entropy form, those instances in head classes contribute a large quantity of negative samples to tail classes. In this case, the penalty signal overwhelmingly suppresses stimulus signal for tail classes, causing bias in the learning process of the classifier. To balance the gradient signals, we propose the class-based scale parameters to narrow down the overwhelming gradients of negative samples for the tail classes. Experiments on benchmark datasets have shown that the proposed KPS loss and optimization strategy can obtain higher classification accuracy for long-tailed data.

In summary, the main contributions of this paper are as follows.

1) We propose a key point sensitive loss, which significantly improves tail class classification accuracy while compromises little on head classes. Considering that key points have a greater impact on overall

1 Some papers, for example, see [25], [26] and [27], name the boundary determined by training objective as decision boundary, while the decision boundary in [20] means the boundary defined by the standard cross-entropy function during testing. To make them distinguishable, we call the boundary determined by the training objective as class boundary and that defined by the standard cross-entropy function which is used during testing as decision boundary.

accuracy than non-key points and a large margin helps improve the classification accuracy, KPS loss assigns large margins for key points and tail classes, which can increase the model classification accuracy on long-tailed datasets.

2) We propose a gradient adjustment optimization strategy to prevent tail class from overwhelming suppression. In the softmax cross-entropy loss function, the penalty signal overwhelmingly suppresses stimulus signal for tail classes. We utilize class-based scale parameters to re-balance positive and negative gradient signals for each class, which can further improve the overall classification accuracy.
3) We conduct extensive experimentation on several benchmarks. The results show that the new loss with the proposed gradient adjustment optimization strategy can significantly improve the performance of the model in tail classes, meanwhile maintaining the competent performance of the head classes.

The rest of this paper is organized as follows. We make an overview of long-tailed classification approaches in Section 2. Then, the proposed KPS loss and the gradient adjustment optimization strategy are described in details in Section 3. Experimental results and discussions are provided in Section 4. Finally, we give the concluding remarks in Section 5.

2 Related Work

In this section, we make an overview of class re-balancing methods, data augmentation methods and two-stage methods that are widely used to alleviate the class-imbalanced problem in long-tailed datasets.

2.1 Class Re-Balancing Methods

The most commonly used class re-balancing methods balance the impact caused by class distribution differences, which include re-sampling, re-weighting and re-margining, to name a few.

The most important two manners of re-sampling methods include under-sampling the head classes [9], [11], [12] and over-sampling the tail classes [6], [7], [8]. However, under-sampling that discards a large amount of samples in head classes will deface the generalization ability when the imbalance ratio is extreme. Over-sampling which duplicates samples of tail classes usually causes over-fitting [23].

Re-weighting methods direct the network to allocate more attention to the samples in tail classes than head classes through the loss function through assigning large weights to tail classes [15]. Some methods like focal loss [28], Meta-Weight-Net [29] and cost-sensitive SVM [16], [30] can achieve fine-grained control though the sample dependent costs. Some methods, for example, see [15] and [31], assign weights inversely proportional to the number of samples of different classes. However, re-weighting methods yield poor performance on head classes [14] and lead to optimization difficulty under the case of extremely imbalanced data and large-scale scenarios [32], [33]. Accordingly, Cui et al. [14] proposed to replace the sample frequency with the *effective number* of samples to re-weight the loss function. Recently, many works [17], [18], [19], [34], [35], [36] re-weight the loss function based on the gradients of different classes to overcome the problem of negative gradient over-suppression. For example, to reduce the influence of negative samples for tail classes, Tan et al. proposed equalization loss [34] and equalization Loss v2 [17], which introduced a weight term for each class on loss function and gradient, respectively.

Re-margining methods [20], [21], [22] adjust the decision margin towards different classes. For example, LDAM [20] increases the margin for tail classes and decreases it for head classes based on class frequency. Feng et al. [21] increased the margin for rare class by the approximate mean classification score.

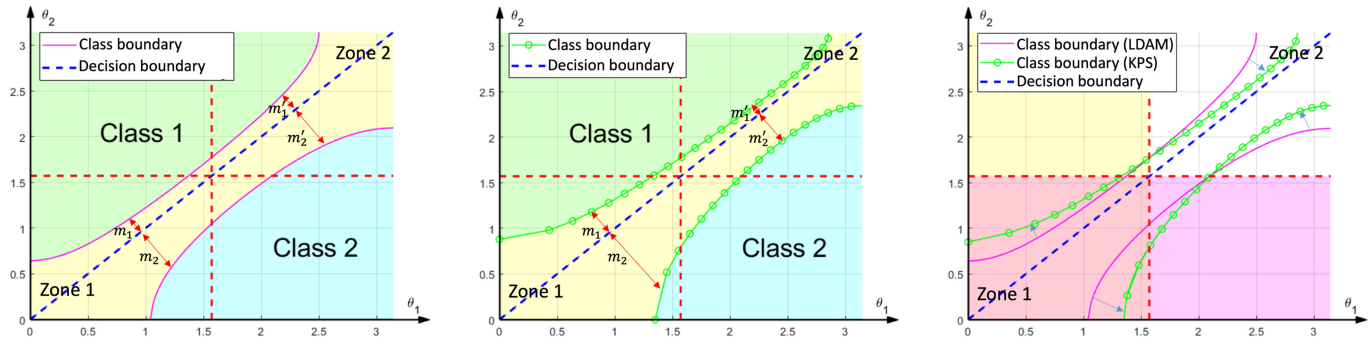
The re-weighting and re-margining methods all have superior performance to the vanilla empirical risk minimization. Nevertheless, the loss functions of these methods do not consider the different influences of key and non-key points.

2.2 Data Augmentation Methods

Naively re-balancing the objective usually results in harsh over-fitting to tail classes because it is inherently unable to deal with the lack of tail class information. To alleviate the issue of fewer samples in tail classes, a canonical solution is to augment tail classes for more training samples. Traditional methods including rotation, horizontal flipping, and erasing are widely employed for maintaining the prediction invariance of CNN models [37], [38], [39]. Mixup is another kind of data augmentation technique, which convexly combines a pair of inputs and the associated labels to obtain new samples to train the network. Training image mixup [40] generates new training samples by weighted linear interpolating two randomly sampled examples and their labels, which can reduce adversarial perturbations in CNNs and improve model robustness [41]. In contrast, manifold mixup [42], CutMix [43] and Remix [24] conduct mixup strategy on two random samples in the feature space. They exploit semantic interpolations. The diversity of augmented samples is inherently limited by the small amount of training data in tail classes because these methods are typically performed based on modification and/or combination of input images. A recently proposed implicit semantic data augmentation (ISDA) technique [44] also performs data augmentation in feature space. Different from manifold mixup, ISDA transforms deep features towards certain meaningful semantic directions. Nevertheless, ISDA may lead to inferior performance in the long-tailed scenario, because scarce data in tail classes is insufficient to obtain reasonable covariance for the method to augment tail classes.

2.3 Two-Stage Training Methods

Two-stage training strategy includes imbalanced training and balanced fine-tuning [45]. The first stage, namely the imbalanced training, utilizes the original long-tailed dataset to train the network. The second stage usually uses re-sampling or re-weighting to fine-tune the classifier and should be applied with a small learning rate. Recently, Cao et al. [20] proposed LDAM to encourage large margin to the tail classes to improve the generalization. This method trains the network with the original dataset in the first stage and applies deferred re-weighting (DRW) in the second stage. LDAM with DRW significantly improves the performance



(a) LDAM (b) KPS (c) Comparison

Fig. 3. Margins of different loss functions and their comparison under binary-classes scenarios, where the θ_1 and θ_2 axes represent the angular distance between the sample features and the class anchor vectors of $\mathbf{w}_1$ and $\mathbf{w}_2$, respectively, and $m_{1,2}$ and $m'_{1,2}$ represent class margins.

on tail classes. Decoupling learning [46] and Bilateral-Branch Network (BBN) [23] both proposed to decouple the learning of representation and classifier. Decoupling learning first uses imbalanced data for representation obtaining and then exploits a balanced sampling strategy to adjust the classifier. BBN takes a two-branch structure with an adaptive fusion operation at the end of the network. One branch of this model focuses on the head classes through directly sampling from the original imbalanced data, and the other branch focuses on the tail classes through reversed sampling the data. Such decoupling of representation and classifier learning is another fruitful avenue of exploration.

In addition, many other methods of different learning paradigms, e.g., ensemble learning [47], [48], [49], [50], [51], meta-learning [52], [53], and knowledge transfer [32], [54], are also proposed to address long-tailed problems. These methods all show effectiveness, but significantly increase model parameters or optimization difficulty.

3 Proposed Method

3.1 Basic Notation and Two Lemmas

This section defines the notation used in this paper. Let $\{x, y\}$ denote a training sample from the training set $\mathcal{T}$, where x is the input training image and $y \in \{1, \ldots, C\}$ is the corresponding label. We use n_j to represent the number of training samples of class j and use $N = \sum_{j=1}^{C} n_j$ to represent the total number of training samples. $\mathbf{f} \in \mathbb{R}^D$ denotes the representation in feature space of x and is the input of the last fully connected layer. D is the dimension of the feature. We use $\mathbf{z} \in \mathbb{R}^C$ to represent the output of the last fully connected layer of the CNN model. $\mathbf{W} \in \mathbb{R}^{D \times C}$ represents the classifier weight matrix, then, $\mathbf{z} = \mathbf{W}^T\mathbf{f}$. $\mathbf{z}$ is the class score vector. $\mathbf{w}_j$ $(j \in \{1, \ldots, C\})$ represents the jth column of $\mathbf{W}$ and is called class anchor vector. $z_j = \mathbf{w}_j^T\mathbf{f}$ is the predicted score of class j. We use the name *target score*[2] and *non-target score* to represent $z_y = \mathbf{w}_y^T\mathbf{f}$ and $z_j = \mathbf{w}_j^T\mathbf{f}, j \neq y$, respectively. The subscript y represents the ground truth class label here.

Two lemmas used in this paper are introduced as follows:

Lemma 1. *[55]* When the number of classes C is smaller than twice the feature dimension D (i.e., $C < 2D$), any two class anchors can be distributed at least $\pi/2$ apart on a hypersphere of dimension D.

Lemma 2. *[56]* For the binary classification case ($C = 2$), the loss function with margin of the target class is:

$$\begin{aligned} \ell(+, \mathbf{f}) &= \tfrac{W_+}{T} \log\left(1 + e^{m_+} \cdot e^{-\tilde{z}}\right) \\ \ell(-, \mathbf{f}) &= \tfrac{W_-}{T} \log\left(1 + e^{m_-} \cdot e^{\tilde{z}}\right) \end{aligned}, \tag{1}$$

where $W_\pm, T > 0$ and $m_\pm \in \mathbb{R}$ are weights, temperature parameter, and margins. The loss in Eq. (1) is Fisher consistent for the balanced error iff

$$\frac{W_+}{W_-} \cdot \frac{\sigma(T \cdot m_+)}{\sigma(T \cdot m_-)} = \frac{1-p}{p}, \tag{2}$$

where $\sigma(z) = (1 + \exp(-z))^{-1}$, $p = \mathbb{P}(y = +)$ is the prior probability of $y = +$.

3.2 Proposed Model: Key Points Sensitive Loss

3.2.1 Intuition

To simplify the interpretation, we use the two classes that are most likely to be confused by the classification model to elucidate the direct intuition of our KPS loss, because the remaining classes will be relatively easy to be classified when these two classes are separated. We name these two classes Class 1 and Class 2, respectively. Without loss of generality, we set Class 1 and Class 2 as head and tail classes (i.e. $n_1 \gg n_2$), respectively. Considering we have a sample x_2 from Class 2, correctly classifying x_2 requires:

$$z_2 > z_1. \tag{3}$$

Eq. (3) can incorporate a margin to increase the class separability:

$$z_2 - m_2 > z_1, \tag{4}$$

where m_2 can be the label distribution dependent aware margin (LDAM) [20] that is chosen as $m_2 = C \cdot {n_2}^{-1/4}$ based on the empirical Rademacher complexity [59], C is a constant. Different with Eq. (3), m_2 helps to assign relatively larger margin to tail class. From Eq. (4), we find that a large predicted score $z_i, (i = 1, 2)$ will weaken the function of the margin m_2. In order to maintain the consistency of the influence of margin on different predicted scores, motivated by

2 Some literature, e.g., see [57] and [26], names z_i as logit, but the logit function is actually the natural log of the odds. Here, z_i has yet to become a ratio and is without the natural log operation. In order to make the meaning of z_i clearer, we adopt the name 'score' used in [58].

[25], we can normalize the output of the linear classifier:

$$\mathrm{Norm}(z_i) = \frac{\mathbf{w}_i^T \mathbf{f}}{\|\mathbf{w}_i^T\|_2 \|\mathbf{f}\|_2} = \cos\theta_i, \tag{5}$$

where θ_i means the angle between class anchor vector W_i and feature f. Eq. (5) means that cosine similarity is used to measure the distance between features and class anchor vectors. Then, Eq. (4) can be changed to:

$$\cos\theta_2 - m_2 > \cos\theta_1. \tag{6}$$

Eq. (6) decides the class boundary of LDAM for two classes, which can be seen in Fig. 3a. It can be observed that the margins of different classes have two properties:

1) The points fall in tail class (Class 2) has larger margins than head class, i.e., $m_2 > m_1$ and $m_2' > m_1'$.
2) The points with different importance are both assigned the same margins, i.e., $m_1 = m_1'$ and $m_2 = m_2'$.

Intuitively, points with different importance should be treated differently. Because non-key point has the angler distance larger than $\pi/2$ to both class anchor vectors (Zone 2 in Fig. 3), it may not actually fall into either class as Lemma 1 indicated. Thus, there is no need to pay much attention to this kind of point. In contrast, the key points are close to both class anchor vectors (Zone 1 in Fig. 3). If these points can be classified correctly, the classification accuracy can be improved. Therefore, we assign larger margins to the key points to make them easier to be classified. However, blindly increasing the margins results in compressing the feature space area of each class. In the extreme case, all samples in each class shrink to one point, where the diversity of the features will disappear, leading to worse generalization. Take this into consideration, we set relatively small margin for non-key points, namely $m_1 > m_1'$ and $m_2 > m_2'$. Non-key points are too far from the anchor vectors of both classes to be assigned to either class according to Lemma 1. Therefore, assigning small margins of these points does not affect the overall classification accuracy. We can move the class boundary as Fig. 3b shown without compressing the feature space area and reducing feature diversity. Fig. 3c shows the comparison before and after the movement of class boundary.

3.2.2 Considering KPS Loss for Two Classes

We can observe that the margins of Eqs. (3), (4) and (6) are the same for all points in the score space because their slopes of the class boundary in their corresponding score space are equal to 1. We prefer to assign a larger margin for key points, i.e. those points that are close to the anchor vectors of both classes. That is, points with larger scores have larger margins, thus we introduce a label-dependent factor r_i called radius to make the slope of the class boundary non-1. As a result, m_i $(i = 1, 2)$ shown in Fig. 3 can be enlarged and the key points can be more separable. Accordingly, the class boundary function is rewritten as:

$$r_2 \cdot \cos\theta_2 - m_2 > r_1 \cdot \cos\theta_1. \tag{7}$$

The class boundary defined by Eq. (7) is shown in Fig. 3b. Suppose $\psi(n)$ represents a non-decreasing function. The margin $m_i \propto 1/\psi(n_i)$ can encourage relative large margins for tail classes. The property 1, namely $m_2 > m_1$ and $m_2' > m_1'$, can be maintained to ensure the good performance of the model in the tail class. We expect that the features of tail class samples can be clustered tightly around the class anchor vector so that they can be correctly classified relatively easily. Therefore, $\cos\theta_i$ for tail classes should be relatively large. To encourage large $\cos\theta_i$ for tail classes and regularize key points strongly at the same time, we set $r_i \propto \psi(n_i)$. The class boundary leaves larger margin for key points, namely $m_1 > m_1'$ and $m_2 > m_2'$.

3.2.3 Selection of Margin and Radius

According to Cao et al. [20], the margin can be chosen as $m_i = \frac{C}{n_i^{1/4}}$ for class i. However, Menon et al. [56] pointed out that this margin is not fisher consistent in minimizing the balanced error. We consider to obtain the expression of m_i based on Lemma 2. Then, the following equation can be obtained:

$$\begin{aligned} \tfrac{1+e^{-m_1}}{1+e^{-m_2}} &= \tfrac{1-p}{p} \Leftrightarrow \\ \tfrac{1+e^{-m_1}}{1+e^{-m_2}} &= \tfrac{n_1/N}{n_2/N} \Leftrightarrow \\ \tfrac{1+e^{-m_1}}{1+e^{-m_2}} &= \tfrac{n_1}{n_2}, \end{aligned} \tag{8}$$

where $p = \mathbb{P}(y = 2)$. Suppose $C_t \cdot n_1 \gg 1$, we can have:

$$\begin{aligned} m_1 &= -\log\left(C_t \cdot n_1 - 1\right) \\ &\approx -\log\left(C_t \cdot n_1\right). \end{aligned} \tag{9}$$

Since $m_{1,2} > 0$, we add a constant $C' > 0$ to Eq. (9):

$$\begin{aligned} m_1 &= -\log\left(C_t \cdot n_1\right) + C' \\ &= C_m - \log n_1, \end{aligned} \tag{10}$$

where C_m should satisfy $C_m \in \mathbb{R}$ and $C_m \geq \log n_{max}$, n_{max} is the largest class size. Eventually, we choose the general expression of margin m_i as:

$$m_i = C_m - \log n_i. \tag{11}$$

Since radius r_i has opposite monotonicity with m_i, for simplicity, we set:

$$r_i = \log n_i + C_r, \tag{12}$$

for a constant C_r.

3.2.4 Extending KPS Loss to Multiple Classes

We use $\tilde{z}_j$ to represent the modified score of the jth class in Eq. (7) for convenience, i.e.,

$$\tilde{z}_j = r_j \cos\theta_j, \tag{13}$$

where r_j is the class-based hyper-parameter and θ_j is learned via a CNN model. To extend the loss to multi-class case, we first extend the binary classification loss to hinge loss, and then use LogSumExp function to replace the max

function and softplus function $\log(1+e^x)$ to smoothly relax $\max(x,0)$. The KPS loss for multiple classes can be expressed as follows:

$$\begin{aligned} L_{KPS}(x,y) &= \log\left(1+e^{\log\left(\sum_{j=1,i\neq y}^{C} e^{\tilde{z}_j}\right)-\tilde{z}_y+m_y}\right) \\ &= \log\left(1+\frac{\sum_{j=1,i\neq y}^{C} e^{\tilde{z}_j}}{e^{\tilde{z}_y-m_y}}\right) \\ &= -\log\frac{e^{\tilde{z}_y-m_y}}{e^{\tilde{z}_y-m_y}+\sum_{j=1,j\neq y}^{C} e^{\tilde{z}_j}}. \end{aligned} \tag{14}$$

This is in the form of the well-known cross-entropy loss.

However, the modified score $\tilde{z}_j$ is small, which is not conducive to convergence. As suggested in [60] and [61], we use a large number s to scale $\tilde{z}_j$. Eventually, the loss function is expressed as:

$$L_{KPS}(x,y) = -\log\frac{e^{s\cdot(r_y\cos\theta_y-m_y)}}{e^{s\cdot(r_y\cos\theta_y-m_y)}+\sum_{j=1,j\neq y}^{C} e^{s\cdot r_j\cos\theta_j}}. \tag{15}$$

3.2.5 Analysis of KPS Loss and Comparison With Previous Methods

Bring the radius (Eq. (12)) and the modified score (Eq. (13)) back to KPS loss for multiple classes (Eq. (14)), because

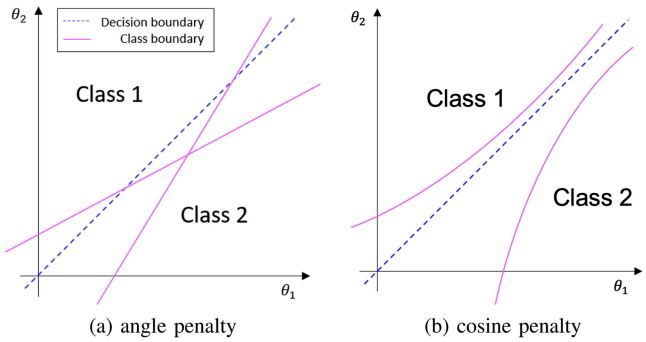
$$\begin{aligned} e^{r_i\cdot\cos\theta_i} &= e^{\log n_i\cdot\cos\theta_i+C_r\cos\theta_i} \\ &= {n_i}^{\cos\theta_i}\cdot e^{C_r\cos\theta_i} \\ &= \left(n_i\cdot e^{C_r}\right)^{\cos\theta_i}. \end{aligned} \tag{16}$$

Therefore, we have

$$\begin{aligned} L_{KPS}(x,y) &= -\log\frac{e^{r_y\cos\theta_y-m_y}}{e^{r_y\cos\theta_y-m_y}+\sum_{j=1,j\neq y}^{C} e^{r_j\cos\theta_j}} \\ &= -\log\frac{\left(n_y\cdot e^{C_r}\right)^{\cos\theta_y}\cdot e^{-m_y}}{\left(n_y\cdot e^{C_r}\right)^{\cos\theta_y}\cdot e^{-m_y}+\sum_{j=1,j\neq y}^{C}\left(n_j\cdot e^{C_r}\right)^{\cos\theta_j}} \\ &\propto -\log\frac{\left(n_y\cdot e\right)^{\cos\theta_y}\cdot e^{-m_y}}{\left(n_y\cdot e\right)^{\cos\theta_y}\cdot e^{-m_y}+\sum_{j=1,j\neq y}^{C}\left(n_j\cdot e\right)^{\cos\theta_j}} \end{aligned} \tag{17}$$

We can notice that KPS loss re-weights the loss through the output probability of different classes. In Eq. (17), the item $n_j\cdot e$ acts as an attractor that requires the tail classes to be clustered closer than the head classes. The item e^{-m_y} leaves larger margin for tail classes. It is intuitively more reasonable because the hidden space capacity is finite.

In comparison with the previous methods, the proposed KPS loss has three characteristics that make it more appropriate for long-tail datasets: (1) KPS loss adopts class-based margin which assigns relatively large margin for tail class. Previous methods [26], [58], [62] do not consider severe class imbalance data and set the same margin for all classes. (2) KPS loss adopts cosine similarity to avoid bias caused by feature norm and class anchor norm. The margin modification losses [18], [63] utilize the inner product to measure the similarity between the feature and class anchor, but a feature or an anchor vector with large norm will cause privilege. (3) KPS loss makes the slopes of different class boundaries different in the score space, so that larger margins can be assigned to the points with higher scores (i.e., the key points). In this way, samples that are prone to misclassification can be further separated. The prior margin modification methods [18], [20], [56], [63] utilize additive margin. The slopes of the different class boundaries are all equal to 1. The margins are the same for different kinds of points.



(a) angle penalty (b) cosine penalty

Fig. 4. Schematic plot of angle penalty and cosine penalty.

In addition, inspired by [26], the margin and radius can also be applied to the angle. Let us take the binary classification case as an example. Accordingly, the class boundary is then expressed as:

$$\hat{r}_2\theta_2+\hat{m}_2 \;<\; \hat{r}_1\theta_1. \tag{18}$$

We can combine the angle penalty and cosine penalty in a unified framework with r_i, $\hat{r}_i$, m_i and $\hat{m}_i$, $(i=1,2)$ as the hyper-parameters. Hence, we have:

$$r_2\cdot\cos\left(\hat{r}_2\theta_2+\hat{m}_2\right)-m_2 \;>\; r_1\cdot\cos\left(\hat{r}_1\theta_1\right). \tag{19}$$

The class boundaries of angle penalty and cosine penalty are shown in Fig. 4. For cosine penalty, we can use the margin m_i to avoid the intersection of class boundary and decision boundary, but the class boundary of angle penalty inevitably intersects the decision boundary. In addition, the cosine penalty is easier to calculate than the angle penalty, because the acquisition of the angle θ_i needs to calculate the arc cosine. Thus, we choose the cosine penalties, i.e., Eq. (7), as the class boundary which is equivalent to $\hat{r}_i=1$ for $i=1,2,\hat{m}_2=0$ in Eq. (19).

3.3 Gradient Adjustment Optimization Strategy

3.3.1 Analysis of Gradient

For a given sample x with the label y, we use $\hat{z}_i$ to represent the final score of sample x belonging to class i, namely

$$\hat{z}_i = \begin{cases} r_i\cos\theta_i-m_i, & i=y \\ r_i\cos\theta_i, & i\neq y \end{cases}. \tag{20}$$

The loss function is:

$$L_{KPS}(x,y) = -\log\left(p_y\right), \text{with}\, p_y = \frac{e^{s\cdot\hat{z}_y}}{\sum_{j=1}^{C} e^{s\cdot\hat{z}_j}}. \tag{21}$$

For a given training sample, the gradients on $\hat{z}_i$ is:

$$\frac{\partial L_{KPS}}{\partial \hat{z}_i} = \begin{cases} s \cdot (p_i - 1), & i = y \\ s \cdot p_i, & i \neq y \end{cases}. \tag{22}$$

It shows that samples of class y punish the classifier of other classes w.r.t. $s \cdot p_i, (i \neq y)$. If class y belongs to the head class, it will contain an enormously greater instance number than that of tail classes. Then, the classifier of tail classes will receive penalties in most samples and rarely get positive signals. In this case, the predicted probabilities of tail classes are severely suppressed, leading to low classification accuracy.

To alleviate this gradient over-suppression problem, some previous works (e.g., see [19], [34], [36]) directly ignore the gradient from samples of head class for the tail class by the weights based on class sizes. The predicted probability $\tilde{p}_i$ of class i given by the loss function of these methods is:

$$\tilde{p}_i = \frac{e^{z_i}}{\sum_{j=1}^{C} \tilde{w}_j e^{z_j}}, \tag{23}$$

where $\tilde{w}_j, j \neq i$ is 1 for head class and less than 1 for tail class. Eq. (23) can achieve: (1) for tail classes, only positive samples propagate gradients; (2) for head classes, the negative gradients propagated to tail classes are reduced. However, the sum of predicted probability for all classes of one training sample is not equal to 1, i.e., $\sum_{i=1}^{C} \tilde{p}_i \neq 1$. It does not conform to the property of probability.

In this paper, we introduce an adjustable scale parameter s_i for class i to re-balance the gradient signal of each class. Because we want to decrease the gradients of head classes and increase that of the tail classes in Eq. (22) during optimization, s_i should be set to a decreasing number with respect to n_i, which has the same trend with m_i. Accordingly, we set:

$$s_i = C_s \cdot m_i, \tag{24}$$

for the constant $C_s > 0$. The KPS loss with gradient adjustment (GA) optimization strategy is:

$$L_{KPS-GA}(x, y) = -\log(\hat{p}_y), \text{with } \hat{p}_y = \frac{e^{s_y \cdot \hat{z}_y}}{\sum_{j=1}^{C} e^{s_y \cdot \hat{z}_j}}. \tag{25}$$

The predicted probability given by Eq. (25) satisfies that $\sum_{i=1}^{C} \hat{p}_i = 1$.

3.3.2 Analysis of Scale Parameters

For the original $p_i = \frac{e^{s \cdot \hat{z}_i}}{\sum_{j=1}^{C} e^{s \cdot \hat{z}_j}}$, it is well known that the smaller the scale parameter s is, the more uniformly p_i is distributed. As s increases, p_i will quickly decay for non-maximum scores and approach 1 for the maximum score. The extreme cases are: (1) when $s = 0$, $p_i = 1/C$ for $i = \{1, 2, \cdots, C\}$; (2) when $s \to \infty, p_i = \begin{cases} 1, & i = M \\ 0, & i \neq M \end{cases}$., M is the class with the maximum score. If s is relatively small, there will be more classes providing gradients, which means more non-target classes will vote for the class boundary. When s is large, only classes with high scores will provide gradients. That is to say, the samples that are prone to be misclassified will participate in class boundary voting. In the early stage of training, we hope that all samples participate in the training to the same degree in order to converge quickly. Besides, simply scaling down the gradients of head class samples increases the risk of false positives for tail classes, because samples of other classes that are misclassified into tail classes receive less penalty. Therefore, we set s_i at the same value for all classes in the early stage of training. When the model converges to a certain extent, we calculate s_i by Eq. (24) for class i. The network will give more positive signals to the tail class in this way. At the same time, the classifier can pay more attention to distinguish the samples from tail classes that are prone to be confused.

The overall training procedure is summarized in Algorithm 1.

Algorithm 1. KPS Loss With Gradient Adjustment Optimization

Require: Training dataset $\mathcal{T}$;
A CNN network $\phi((x, y); \omega)$ which is parameterized by ω.
for $iter = 1$ to I_0 **do**
  Sample batch samples $\mathcal{B}$ from $\mathcal{T}$ with batch size of b;
  $\mathcal{L}((x, y); \omega) = \frac{1}{b} \sum_{(x,y) \in \mathcal{B}} L_{KPS}(x, y)$;
  $\omega = \omega - \alpha \nabla_\omega \mathcal{L}((x, y); \omega)$;
end for
for $iter = I_0 + 1$ to I_1 **do**
  Sample batch samples $\mathcal{B}$ from $\mathcal{T}$ with batch size of b;
  $\mathcal{L}((x, y); \omega) = \frac{1}{b} \sum_{(x,y) \in \mathcal{B}} L_{KPS-GA}(x, y)$;
  $\omega = \omega - \alpha \nabla_\omega \mathcal{L}((x, y); \omega)$;
end for

3.4 Time-Complexity Analysis

For a given sample, the time-complexity of the original softmax cross-entropy loss is $\mathcal{O}(C)$ where C is the number of classes. It is linear with the input dimension of the loss function. Eqs. (15) and (25) show that KPS loss only adds scalar addition and multiplication compared with the original softmax cross-entropy loss. Therefore, KPS loss has $\mathcal{O}(C)$ time-complexity, which only adds negligible burden on the training process.

4 Experiments

4.1 Datasets

4.1.1 Long-Tailed CIFAR

The original CIFAR datasets [64] consist of 50,000 color images of size 32×32 for training and 10,000 images with the same size for testing. CIFAR-10 and CIFAR-100 consist of 10 classes and 100 classes, respectively. To create the long-tailed versions (CIFAR-10-LT and CIFAR-100-LT), we follow Cui et al. 's setting [14] by down-sampling training images per class with the exponential function $n_i = n_{o_i} \times \mu^i$, where i is the class index (0-indexed), n_{o_i} is the number of training samples in original CIFAR and $\mu \in (0, 1)$. The test set remains unchanged. The imbalance ratio

TABLE 1
SUMMARY OF DATASET

Dataset	CIFAR-10-LT	CIFAR-100-LT	ImageNet-L T	iNaturalist 2018
♯ Classes	10	100	1000	8142
Imbalance ratio	100,50	100, 50	256	500
Tail class size	50, 100	5, 10	5	2
Head class size	5000	500	1280	1000

is defined as $\rho = \max(\mathbf{n})/\min(\mathbf{n}), \mathbf{n} = \{n_1, n_2, \cdots, n_C\}$. In the literature, the most widely used ρ are 50 and 100.

4.1.2 Long-Tailed ImageNet

The original ImageNet-2012 [1] is a large-scale real-world dataset for classification and localization, which contains more than 1.2 million images for training and 150,000 images for validation and testing. We follow Liu et al. 's work [52] to construct the long-tailed version of ImageNet-2012 (ImageNet-LT) by truncating a subset with the Pareto distribution from the balanced version. Overall, this long-tailed version dataset has 1,000 categories with maximally 1,280 images and minimally 5 images per class. The original balanced validation data containing 50,000 images are used for testing in our experiments.

4.1.3 iNaturalist 2018

The 2018 version of iNaturalist species classification and detection dataset [65] is a real-world long-tailed dataset suffering from extremely imbalanced distributions and thus is the most challenging version. It includes 437,513 training images from 8,142 categories. We follow the official training and validation splits of iNaturalist 2018 in our experiments.

The datasets used in our experiments and their imbalance ratios are summarized in Table 1.

4.2 Basic Setting

We use Pytorch [66] to implement and train all backbones from scratch by stochastic gradient descent (SGD) with momentum of 0.9.

4.2.1 Basic Setting on CIFAR-10/100-LT

For CIFAR-10-LT and CIFAR-100-LT datasets, the same data augmentation strategies with [67] are utilized, namely randomly cropping a 32×32 region from the image that is flipped with 0.5 probability and padded with 4 pixels on each side. ResNet-32 is chosen as the backbone network with weight decay of 2×10^{-4}. The number of training epochs is 200 with a mini-batch size of 64. Learning rate is initialized to 0.1 and multiplied by 0.01 at the 160th and 180th epoch, respectively. Linear warm-up learning rate [68] is used in the first five epochs.

4.2.2 Basic Setting on ImageNet-LT and iNaturalist 2018

For ImageNet-LT and iNaturalist 2018, we follow the data augmentation strategies in [68], namely scaling the shorter dimension to 256 followed by randomly cropping a 224×224 patch from the augmented image or its horizontal flip.

TABLE 2
SUMMARY OF BASIC SETTING

Dataset	CIFAR-10-LT	CIFAR-100-LT	ImageNet-LT	iNaturalist 2018
Backbone	ResNet-32	ResNet-32	ResNet-50, ResNeXt-50	ResNet-50
Mini-batch size	64	64	128	512
Initial l_r	0.1	0.1	0.1	0.2
Weight decay	2×10^{-4}	2×10^{-4}	1×10^{-4}	1×10^{-4}
l_r warm-up	Yes	Yes	No	No
Maximum epochs	200	200	180	180
l_r decay ratio	0.01	0.01	0.1	0.1
l_r decay epochs	{160,180}	{160,180}	{120,160}	{120,160}

For fair comparisons, we adopt the most common practices which are ResNet-50 [67] and ResNeXt-50 [69] for ImageNet-LT, and ResNet-50 for iNaturalist 2018, respectively.

The weight decay in all experiments on ImageNet-LT and iNaturalist 2018 is set as 1×10^{-4}. The number of training epochs is 180. During training, we decay the learning rate by 0.1 at the 120th and 160th epoch, respectively. The mini-batch size is set as 128 and the learning rate is initialized to 0.1 for ImageNet-LT. As for iNaturalist 2018 dataset, we use the mini-batch size of 512 to speed up training. The learning rate should be increased by the square root of the mini-batch size according to [70], so the learning rate is initialized to 0.2.

The basic setting used in our experiments is summarized in Table 2.

4.2.3 Hyper-Parameters Selection

There are hyper-parameters in r_i, m_i and s_i. In order to simplify the selection of hyper-parameters for different datasets, we adopt the following strategy.

For r_j, we first normalize the minimum number in the class number list $\mathbf{n} = \{n_1, n_2, \cdots, n_C\}$ to a preset number n_{base} to get a temp list $\mathbf{r}' = \{r'_1, r'_2, \cdots, r'_C\}$:

$$r'_i = \log\left(n_i \times \frac{n_{base}}{\min(\mathbf{n})}\right). \tag{26}$$

Then, we obtain the final list $\mathbf{r} = \{r_1, r_2, \cdots, r_C\}$ by normalizing the minimum number in $\mathbf{r}$ to 1:

$$r_i = r'_i \times \frac{1}{\min(\mathbf{r}')}. \tag{27}$$

For m_i, we first get a temp list $\mathbf{m}' = \{m'_1, m'_2, \cdots, m'_C\}$ based on Eq. (11) by:

$$\mathbf{m}'_i = max(\mathbf{r}) + 1 - r_i. \tag{28}$$

Then, the final margin list $\mathbf{m} = \{m_1, m_2, \cdots, m_C\}$ is obtained by normalizing the maximum number in $\mathbf{m}'$ to m_{max}:

$$m_i = m'_i \times \frac{m_{max}}{\max(\mathbf{m}')}. \tag{29}$$

For s_i, we obtain the scale parameter list $\mathbf{s} = \{s_1, s_2, \cdots, s_C\}$ by:

$$\mathbf{s} = s_{base} \cdot \mathbf{m}', \tag{30}$$

TABLE 3
TOP-1 ERROR RATES (%) COMPARISON ON LONG-TAILED CIFAR DATASETS

Dataset	CIFAR-10-LT		CIFAR-100-LT	
Backbone	ResNet-32			
Imbalance ratio	100	50	100	50
CE loss	29.64	24.78	63.32	56.15
Focal loss [28]	29.62	24.75	62.75	55.68
CosFace [62]	27.92	22.60	60.79	56.89
ArcFace[26]	26.24	21.81	60.94	56.60
LDAM-DRW [20]	22.97	18.97	57.96	53.41
CB-Focal [14]	25.43	20.73	60.40	53.79
Equalization loss* [34]	26.02	-	57.26	-
LA loss [56]	**19.08**	-	56.11	-
BBN [23]	20.18	17.82	57.44	52.98
Meta-learning† [53]	20.00	17.77	55.92	50.84
Meta-learning‡ [53]	21.10	**17.12**	**55.30**	**49.92**
KPS loss (Ours)§	**18.77**	**15.41**	**54.97**	**50.82**

***Note**: The best and the second best results are shown in **underline bold** and **bold**, respectively.*
*: *Results are obtained by coping from LA loss [56].*
†: *Results are obtained by incorporating LDAM [20].*
‡: *Results are obtained by incorporating focal loss [28].*
§: *GA strategy is utilized.*

TABLE 4
TOP-1 ERROR RATES (%) COMPARISON ON IMAGENET-LT AND INATURALIST 2018

Dataset	ImageNet-LT		iNaturalist 2018
Backbone	ResNet-50	ResNeXt-50	ResNet-50
CE loss	55.49	53.35	39.89
Focal loss [28]	54.20	-	39.70
CosFace [62]	55.05	53.22	33.28
ArcFace[26]	55.46	51.51	36.14
LDAM-DRW [20]	51.20	-	32.00
CB-Focal [14]	-	-	38.88
Equalization loss [34]	52.70	50.90	38.37
LA loss [56]	**51.11**	-	31.56
BBN [23]	55.30	-	**30.38**
Meta-learning* [53]	-	-	32.45
Decoupling [46]	52.30	**50.10**	30.70
KPS loss (Ours)†	**48.72**	**47.17**	**29.65**

***Note**: The best and the second best results are shown in **underline bold** and **bold**, respectively.*
*: *Results are obtained by incorporating CE loss.*
†: *GA strategy is utilized.*

where s_{base} is the scale parameter in the first stage, which is the same for all classes.

In this way, the parameters that need to be preset are: n_{base}, m_{max} and s_{base}. We choose $n_{base} = 50$ and $s_{base} = 15$ experimentally for all datasets. The only hyper-parameter that need to be adjusted towards different dataset is m_{max}. For CIFAR-10/100-LT, the total classes C are much smaller than the dimension of features. So the margin for CIFAR-10/100-LT can be set relatively large. We choose $m_{max} = 0.5$ for these two datasets. As for the large-scale datasets, namely ImageNet-LT and iNaturalist 2018, we select $m_{max} = 0.35$ and 0.3, respectively.

4.3 Comparison Methods

We compare our KPS loss to the following three groups of competing methods:

4.3.1 Baseline Methods

Besides vanilla training with cross-entropy loss (CE loss), we also employ the recent proposed focal loss [28] that focuses training on difficult samples as one of our baselines.

4.3.2 Loss Modification Methods

Four recently proposed loss function modification methods are compared: CosFace [62], ArcFace [26], Label-Distribution-Aware Margin Loss with deferred re-weighting (LDAM-DRW)[3] [20], Class-Balanced focal loss (CB-Focal) [14], Equalization loss [34] and Logit Adjustment loss (LA loss) [56].

3 LDAM-DRW trains the backbone with the original dataset at the first stage and fine-tunes the classifier with DRW at the second stage. It is also a two-stage method.

4.3.3 Two-Stage Methods

We compare with the recently proposed Bilateral-Branch Network (BBN) [23] and meta-learning [53], which adopt two-stage fine-tuning strategy and a domain adaptation strategy, respectively. These two methods achieve good performance on those four aforementioned commonly used long-tailed datasets. For ImageNet-LT and iNaturalist 2018 datasets, we also compare with Decoupling learning [46], which takes the strategy with post-hoc normalization of the classification weights and achieves high accuracy on large-scale datasets.

4.4 Comparison Results

Top-1 error rates of baseline and comparison methods are shown in Tables 3 and 4. We use the results reported in references ([14], [20], [23], [28], [34], [53], [56]) except the baseline method. Our re-implemented results are mostly consistent with references. The slightly inconsistent ones might be caused by the running environment (e.g., the version of CUDA and precision of device), since we keep the experimental settings consistent with references.

4.4.1 Experimental Results on CIFAR-10/100-LT

We conduct extensive experiments on CIFAR-10-LT and CIFAR-100-LT with different imbalance ratios. The performance comparison of various methods is shown in Table 3. Our proposed KPS loss achieves the best results in most cases compared to other methods, including loss modification strategies (i.e., LDAM-DRW, CB-Focal, Equalization loss and LA loss), two-stage fine-tuning strategy (i.e., BBN), and also two-stage domain adaptation strategy (i.e., meta-learning). These methods are all recently proposed state-of-the-arts.

Except for meta-learning, our method obtains significant improvement across all the datasets compared with other comparison methods. Comparing KPS loss with baseline methods (CE loss and Focal loss), we find that KPS loss significantly improves the CE loss and Focal loss. Under the setting of $\rho = 100$, for instance, KPS loss reduces the

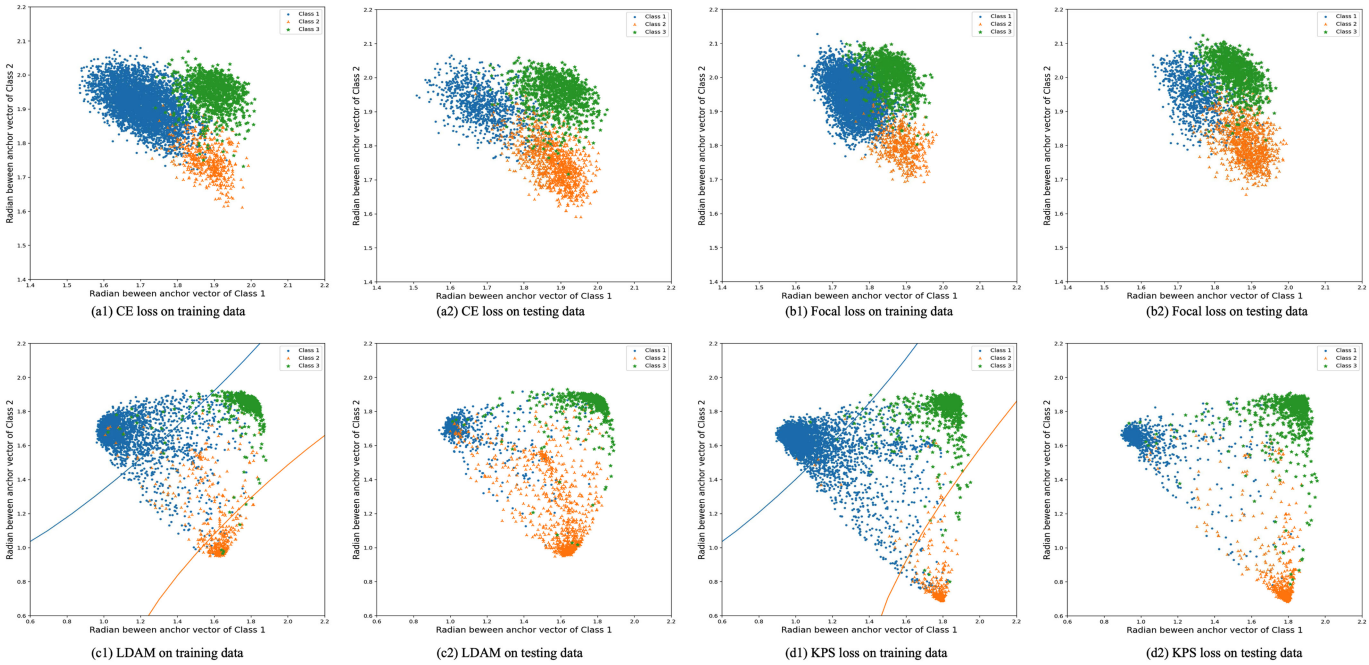


Fig. 5. Feature distribution of different loss functions. A ResNet-32 is trained on 3 classes from CIFAR-10. 5000, 500 and 1000 samples for Class 1, Class 2 and Class 3 are randomly selected, respectively. (a1)-(d1) are the feature distributions obtained by different loss functions on training data, and (a2)-(d2) are on testing data.

baseline methods by more than 10% and 6% top-1 error rate on CIFAR-10-LT and CIFAR-100-LT, respectively. Compared with the loss modification methods, our proposed method outperforms these methods in each session by notable margins. Especially, compared with the most related method, namely LDAM-DRW, KPS loss reduces the top-1 error rate by 4.20% and 3.56% on CIFAR-10-LT and 2.99% and 2.59% on CIFAR-100-LT under $\rho = 100$ and 50. We can also clearly observe the superiority of KPS loss compared with two-stage methods that achieve good performance. Even though meta-learning incorporating focal loss gets the best result on CIFAR-100-LT with $\rho = 50$, our proposed KPS loss outperforms it on other datasets. And our method gets the second best on CIFAR-100-LT with $\rho = 50$. Furthermore, meta-learning adopts two training stages and needs to explicitly estimate the class-conditioned distribution differences using a separate network. Our method which only modifies the loss function based on label distribution is simpler to implement compared with meta-learning.

Additionally, the following observation can be obtained from Table 3. The two-stage training strategies (i.e. BBN and meta-learning) are also effective in most cases, since they could obtain comparable or even better results compared with other state-of-the-art methods. BBN takes two branches network to learn different distributions training samples and meta-learning uses a network to estimate the conditional distribution $P(x|y)$ from head classes and then transfer it to the tail classes. These strategies are enlightening.

4.4.2 Experimental Results on ImageNet-LT and iNaturalist 2018

Table 4 shows the results on two large-scale long-tailed datasets. As shown in Table 4, our KPS loss still outperforms baselines and the competing approaches across these two datasets, which is consistent with the observation of CIFAR-10/100-LT. Compared with the baseline methods, our method reduces the top-1 error rate on ImageNet-LT and iNaturalist 2018 by more than 6% and 10%, respectively, which is a significant performance improvement. Additionally, compared to LDAM-DRW, we have achieved notable improvements, i.e., 2.48% and 2.35% improvements on ImageNet-LT and iNaturalist 2018, respectively. Even compared with the two-stage methods that have achieved significant improvements, our method can surpass them.

Moreover, the two-stage fine-tuning strategies also perform well. But meta-learning does not perform as well as long-tailed CIFAR on these two large-scale datasets, probably because of the loss function that meta-learning incorporates with.

4.5 Ablation Experiment

1) Why is KPS effective? In order to better explain the proposed KPS loss, we use a toy experiment to visualize feature distributions trained with different loss functions. We train a ResNet-32 with 3 classes from CIFAR-10 with 30 epochs. The number of training samples for Class 1, Class 2 and Class 3 are 5000, 500 and 1000, respectively. The visualization is shown in Fig. 5. We can see that most of the points of Class 3 are far away from Class 1 and Class 2 than $\pi/2$ in feature space. The points of Class 3 are non-key points for class 1 and class 2. In Figs. 5a1, 5a2, 5b1, and 5b2, we can observe that CE loss and focal loss have no margins, leading to difficulty in classifying each class. In Figs. 5c1 and 5c2, LDAM assigns margins for different classes and can make each class more separable. But there are some key points clustered together, so that they are difficult to be classified. From Figs. 5d1 and 5d2, we can see that our proposed KPS loss can well cluster the samples in each class and make

TABLE 5
TOP-1 ERROR RATES (%) ON LONG-TAILED CIFAR DATASET

Dataset	CIFAR-10-LT		CIFAR-100-LT	
Imbalance ratio	100	50	100	50
KPS w/o GA and DRW	19.85	16.40	55.31	50.92
KPS-DRW	19.09	16.11	58.95	54.30
KPS-GA	**18.77**	**15.41**	**54.97**	**50.82**

most points become simple points. And the key points obtained by KPS loss become more separable.

2) What is the effectiveness of GA? To illustrate the effectiveness of our proposed gradient adjustment optimization strategies, we explore several different optimization strategies on CIFAR-10/100-LT with different imbalance ratios. Specifically, we test our proposed loss trained with both the basic SGD without any other optimization strategy (KPS w/ o GA and DRW) and with DRW [20] (KPS-DRW). The results are listed in Table 5. We can see that our proposed gradient adjustment strategy (KPS-GA) can yield better results than the other strategies. Since Gao et al. [20] observed that their loss benefits a lot from DRW. We expect that employing DRW can be similarly beneficial for our KPS loss. But the experiments show that DRW even hurts the performance compared with original SGD without optimization strategy on CIFAR-100-LT. To further analyze the results, we show the per-class error rate on CIFAR-10/100-LT with $\rho = 100$ in Fig. 6. On CIFAR-100-LT, in order to facilitate visualization, we aggregate the classes into ten groups according to their label frequency sort order. Group 1 corresponds to the top 10 most frequent classes. Group 2 comprises the second most frequent 10 classes, and so on. The most frequent 3 classes/groups in CIFAR-10/100-LT are named Head1 to Head3. The 4 classes/groups appearing with medium frequency are named Mid1 to Mid4 in order, and the rest 3 least frequent classes/groups are named Tail1 to Tail3 in order. As shown in Fig. 6, compared with the basic optimization strategy, DRW increases the performance on tail classes but meanwhile harms that on head classes. In contrast, the proposed GA improves the accuracy of the tail classes, and that of the head classes only slightly decreases or even improves in some cases (e.g., see Head2 in CIFAR-10-LT and Head1 in CIFAR-100-LT).

3) Can KPS loss be combined with other methods? KPS loss can be trained end-to-end and is easy to attach to other methods. We conduct the experiment of KPS combined with mixup strategy [40] and the most recent proposed two-stage method, i.e. MiSLAS [71]. The results are shown in Tables 6 and 7.

TABLE 6
Comparison of KPS Combined with mixup in Terms of Top-1 Error Rates (%)

Dataset	CIFAR-10-LT		CIFAR-100-LT		ImageNet-LT	iNaturalist 2018
Backbone	ResNet-32		ResNet-32		ResNet-50	ResNet-50
Imbalance ratio	100	50	100	50	-	-
KPS	18.77	15.41	54.97	50.82	48.72	29.65
KPS w. mixup	**17.84**	**15.07**	**54.66**	**49.45**	**47.05**	**28.12**

***Note**: GA strategy is utilized.*

TABLE 7
Comparison of KPS Combined with MiSLAS in Terms of Top-1 Error Rates (%)

Dataset	CIFAR-10-LT		CIFAR-100-LT		ImageNet-LT	iNaturalist 2018
Backbone	ResNet-32		ResNet-32		ResNet-50	ResNet-50
Imbalance ratio	100	50	100	50	-	-
MiSLAS	17.94	14.84	52.62	48.28	47.89	29.43
MiSLAS w. KPS	**17.26**	**14.36**	**52.28**	**47.59**	**47.74**	**29.12**

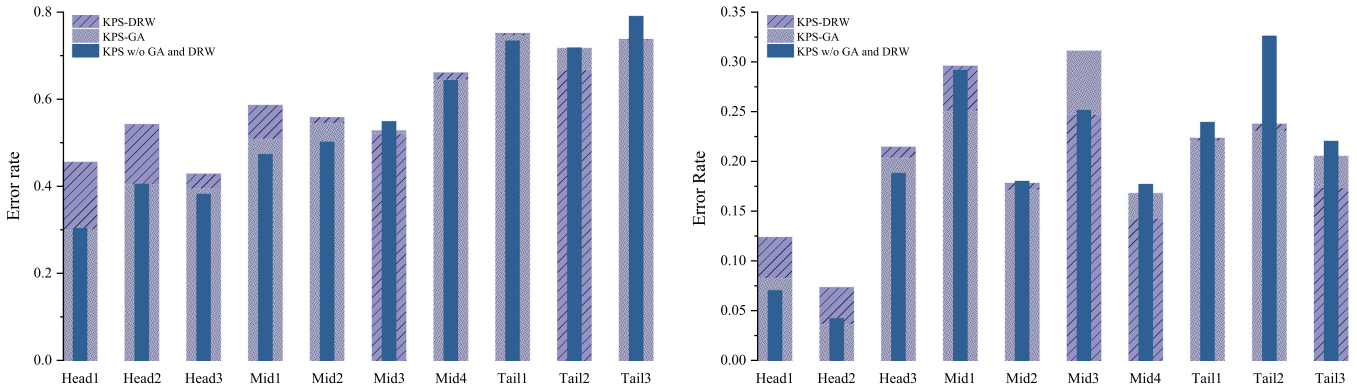
Table 6 shows that mixup is an effective strategy for boosting performance. It can be found that KPS with mixup, which can be trained end-to-end, is better than MiSLAS in most cases we have tried thus far. As for the two-stage method as shown in Table 7, using KPS in the second stage of MiSLAS can indeed further improve the performance.

4) How does KPS loss perform in the head, middle and tail classes? We use CIFAR-10-LT and CIFAR-100-LT with $\rho = 100$ to present the performance on each class/group. Besides KPS loss, the per-class/group error rates with baseline methods are shown for comparison. Since CosFace, ArcFace and LDAM-DRW are the most related methods to our work, Fig. 7 also breaks down the per-class/group error rate of these methods. Analogous to Fig. 6, on CIFAR-100-LT, we aggregate the classes into ten groups based on their frequency-sorted order for ease of visualisation. Then, we divide the 10 classes/groups into Head, Middle and Tail according to the sample frequency. The corresponding per-class/group accuracy is shown in Fig. 7. We can observe the following phenomena:

1) The overall error rate of all methods shows the trend of Head $<$ Mid $<$ Tail. This is in line with common sense because head classes have more sample diversity.
2) Compared with CE loss, CosFace and ArcFace perform well on balanced data. The methods for imbalanced data, i.e., Focal loss, LDAM-DRW and our KPS loss can decrease the error rate on tail classes. LDAM-DRW and KPS loss can also improve the performance on middle classes.
3) The methods for imbalanced data all sacrifice the performance of the head classes to a certain extent. The reduction in the head class of KPS loss is insignificant compared to the improvement in the tail class. For example, on CIFAR-100-LT with $\rho = 100$, KPS loss improves the error rate of Head1 by 1.7%, but reduces that of Tail1 by more than 10%.

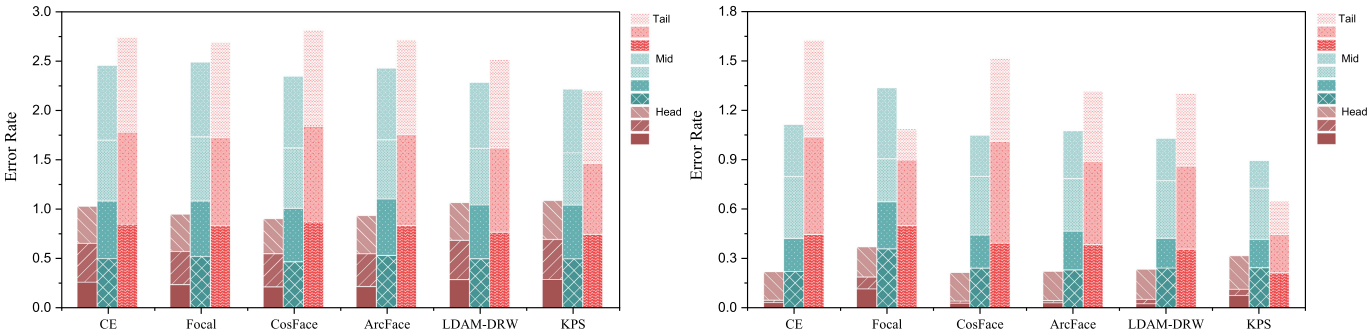
5 CONCLUDING REMARKS

In this paper, we have proposed a KPS loss to solve the classification problem of long-tailed distribution of training data, which can unify several recent proposals and overcome their limitation. This KPS loss is geometrically principled and has twofold effectiveness: 1) increase the margins of the samples with key points; 2) encourage a large relative margin between points of tail versus head classes. In order to further increase the classification accuracy, we have proposed a gradient adjustment (GA) optimization strategy that can compensate for the excessive punishment for tail



(a) Per-class error rates on CIFAR-10-LT (b) Per-group error rates on CIFAR-100-LT

Fig. 6. Per-class/group error rates obtained by different optimization strategies on CIFAR-10/100-LT with the imbalance ratio $\rho = 100$. Head classes are with low indices. Conversely, tailed-classes are with higher indices. For CIFAR-100-LT, we aggregate the classes into 10 groups.



(a) Per-class error rates on CIFAR-10-LT (b) Per-group error rates on CIFAR-100-LT

Fig. 7. Per-class/group error rates obtained by different techniques on CIFAR-10/100-LT datasets with $\rho = 100$. Head classes are with low indices. Conversely, tailed-classes are with higher indices. For CIFAR-100-LT, the classes are aggregated into 10 groups.

classes through the scale parameters based on label distribution. Such adjustment encourages a large relative gradient magnitude for tail classes. The extensive experiments have demonstrated that our KPS loss with GA achieves the best performance on long-tailed benchmarks, including the most challenging large-scale dataset.

Our KPS loss is able to increase the overall accuracy on each dataset compared with previous state-of-the-art methods. However, it has a limitation. As mentioned in Section 4.54, KPS loss significantly improves the performance of the middle and tail classes while sacrificing a small amount of head class accuracy. In our future work, we attempt to study the loss function that can improve the performance of tail as well as head classes.

REFERENCES

[1] O. Russakovsky et al., "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
[2] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1452–1464, Jun. 2017.
[3] M. G. Kendall et al., *The Advanced Theory of Statistics*, Drury Lane, London, U.K.: Charles Griffin and Company Ltd., 1948.
[4] Y. Zhang, B. Kang, B. Hooi, S. Yan, and J. Feng, "Deep long-tailed learning: A survey," 2021, *arXiv:2110.04596*.
[5] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.
[6] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.
[7] Q. Zhong et al., "Towards good practices for recognition & detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2016, Art. no. 2.
[8] J. Byrd and Z. Lipton, "What is the effect of importance weighting in deep learning?," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 872–881.
[9] M. Kubat et al., "Addressing the curse of imbalanced training sets: One-sided selection," in *Proc. Int. Conf. Mach. Learn.*, 1997, pp. 179–186.
[10] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *Proc. Int. Conf. Artif. Intell.*, 2000, pp. 111–117.
[11] B. C. Wallace, K. Small, C. E. Brodley, and T. A. Trikalinos, "Class imbalance, redux," in *Proc. IEEE Conf. Data Mining*, 2011, pp. 754–763.
[12] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Netw.*, vol. 106, pp. 249–259, 2018.
[13] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3573–3587, Aug. 2018.
[14] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9268–9277.
[15] C. Huang, Y. Li, C. C. Loy, and X. Tang, "Learning deep representation for imbalanced classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5375–5384.
[16] A. Iranmehr, H. Masnadi-Shirazi, and N. Vasconcelos, "Cost-sensitive support vector machines," *Neurocomputing*, vol. 343, pp. 50–64, 2019.

[17] J. Tan, X. Lu, G. Zhang, C. Yin, and Q. Li, "Equalization loss V2: A new gradient balance approach for long-tailed object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1685–1694.

[18] J. Wang et al., "Seesaw loss for long-tailed instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9695–9704.

[19] T. Wang, Y. Zhu, C. Zhao, W. Zeng, J. Wang, and M. Tang, "Adaptive class suppression loss for long-tail object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3103–3112.

[20] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1565–1576.

[21] C. Feng, Y. Zhong, and W. Huang, "Exploring classification equilibrium in long-tailed object detection," in *Proc. Int. Conf. Conf. Comput. Vis.*, 2021, pp. 3417–3426.

[22] Z. Deng, H. Liu, Y. Wang, C. Wang, Z. Yu, and X. Sun, "PML: Progressive margin loss for long-tailed age classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 10503–10512.

[23] B. Zhou, Q. Cui, X.-S. Wei, and Z.-M. Chen, "BBN: Bilateral-branch network with cumulative learning for long-tailed visual recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9719–9728.

[24] H. Chou, S. Chang, J. Pan, W. Wei, and D. Juan, "Remix: Rebalanced mixup," in *Proc. Eur. Conf. Comput. Vis. Workshop*, 2020, pp. 95–110.

[25] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Process. Lett.*, vol. 25, no. 7, pp. 926–930, Jul. 2018.

[26] 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.

[27] B. Chen, W. Deng, and H. Shen, "Virtual class enhanced discriminative embedding learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1946–1956.

[28] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2999–3007.

[29] J. Shu et al., "Meta-weight-net: Learning an explicit mapping for sample weighting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1917–1928.

[30] H. Masnadi-Shirazi and N. Vasconcelos, "Risk minimization, probability elicitation, and cost-sensitive SVMs," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 759–766.

[31] S. Khan, M. Hayat, S. W. Zamir, J. Shen, and L. Shao, "Striking the right balance with uncertainty," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 103–112.

[32] Y.-X. Wang, D. Ramanan, and M. Hebert, "Learning to model the tail," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 7032–7042.

[33] C. Huang, Y. Li, C. C. Loy, and X. Tang, "Deep imbalanced learning for face recognition and attribute prediction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 11, pp. 2781–2794, Nov. 2019.

[34] J. Tan et al., "Equalization loss for long-tailed object recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11 662–11 671.

[35] T. Wu, Q. Huang, Z. Liu, Y. Wang, and D. Lin, "Distribution-balanced loss for multi-label classification in long-tailed datasets," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 162–178.

[36] T.-I. Hsieh, E. Robb, H.-T. Chen, and J.-B. Huang, "Droploss for long-tail instance segmentation," in *Proc. AAAI Nat. Conf. Artif. Intell.*, 2021, pp. 1549–1557.

[37] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," in *Proc. AAAI Nat. Conf. Artif. Intell.*, 2020, pp. 13 001–13 008.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, *arXiv:1409.1556*.

[39] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," *CoRR*, vol. abs/1712.04621, 2017. [Online]. Available: http://arxiv.org/abs/1712.04621

[40] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–13.

[41] M. Peng et al., "Trainable undersampling for class-imbalance learning," in *Proc. AAAI Nat. Conf. Artif. Intell.*, 2019, pp. 4707–4714.

[42] V. Verma et al., "Manifold mixup: Better representations by interpolating hidden states," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6438–6447.

[43] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proc. Int. Conf. Conf. Comput. Vis.*, 2019, pp. 6022–6031.

[44] Y. Wang, X. Pan, S. Song, H. Zhang, G. Huang, and C. Wu, "Implicit semantic data augmentation for deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 12 614–12 623.

[45] Y. Zhang, X. Wei, B. Zhou, and J. Wu, "Bag of tricks for long-tailed visual recognition with deep convolutional neural networks," in *Proc. AAAI Nat. Conf. Artif. Intell.*, 2021, pp. 3447–3455.

[46] B. Kang et al., "Decoupling representation and classifier for long-tailed recognition," 2020, *arXiv:1910.09217*.

[47] L. Xiang, G. Ding, and J. Han, "Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 247–263.

[48] X. Wang, L. Lian, Z. Miao, Z. Liu, and S. X. Yu, "Long-tailed recognition by routing diverse distribution-aware experts," in *Proc. Int. Conf. Learn. Represent.*, 2021.

[49] J. Cai, Y. Wang, and J.-N. Hwang, "ACE: Ally complementary experts for solving long-tailed recognition in one-shot," in *Proc. Int. Conf. Conf. Comput. Vis.*, 2021, pp. 112–121.

[50] Y. Zhang, B. Hooi, L. Hong, and J. Feng, "Test-agnostic long-tailed recognition by test-time aggregating diverse experts with self-supervision," 2021, *arXiv:2107.09249*.

[51] J. Cui, S. Liu, Z. Tian, Z. Zhong, and J. Jia, "ResLT: Residual learning for long-tailed recognition," 2021, *arXiv:2101.10633*.

[52] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, "Large-scale long-tailed recognition in an open world," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2537–2546.

[53] M. A. Jamal, M. Brown, M.-H. Yang, L. Wang, and B. Gong, "Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 7610–7619.

[54] Y. Zhong et al., "Unequal-training for deep face recognition with long-tailed noisy data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7812–7821.

[55] R. Ranjan, C. D. Castillo, and R. Chellappa, "L2-constrained softmax loss for discriminative face verification," *CoRR*, vol. abs/1703.09507, 2017. [Online]. Available: http://arxiv.org/abs/1703.09507

[56] A. K. Menon, S. Jayasumana, A. S. Rawat, H. Jain, A. Veit, and S. Kumar, "Long-tail learning via logit adjustment," 2021, *arXiv:2007.07314*.

[57] G. Pereyra, G. Tucker, J. Chorowski, Ł. Kaiser, and G. Hinton, "Regularizing neural networks by penalizing confident output distributions," 2017, *arXiv:1701.06548a*.

[58] 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. 6738–6746.

[59] S. M. Kakade, K. Sridharan, and A. Tewari, "On the complexity of linear prediction: Risk bounds, margin bounds, and regularization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 793–800.

[60] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: L_2 hypersphere embedding for face verification," in *Proc. ACM Int. Conf. Multimedia*, 2017, pp. 1041–1049.

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

[62] 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.

[63] J. Ren et al., "Balanced meta-softmax for long-tailed visual recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 4175–4186.

[64] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, Canada, Tech Rep., 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

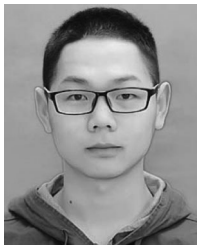
[65] G. Van Horn et al., "The inaturalist species classification and detection dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8769–8778.

[66] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.

[67] 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.
[68] P. Goyal et al., "Accurate, large minibatch SGD: Training imagenet in 1 hour," *CoRR*, vol. abs/1706.02677, 2017. [Online]. Available: http://arxiv.org/abs/1706.02677
[69] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5987–5995.
[70] E. Hoffer, I. Hubara, and D. Soudry, "Train longer, generalize better: Closing the generalization gap in large batch training of neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1731–1741.
[71] Z. Zhong, J. Cui, S. Liu, and J. Jia, "Improving calibration for long-tailed recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 16489–16498.

Mengke Li received the BSc degree in communication engineering from Southwest University, Chongqing, China, in 2015, and the MSc degree in electronic engineering from Xidian University, Xi'an, China, in 2018. She is currently working toward the PhD degree with the Department of Computer Science, Hong Kong Baptist University, Hong Kong, under the supervision of prof. Yiuming Cheung. Her current research interests include image restoration and imbalanced data learning.

Yiu-Ming Cheung (Fellow, IEEE) received the PhD degree from the Department of Computer Science and Engineering, Chinese University of Hong Kong, in Hong Kong. He is also a Fellow of AAAS, IET, and BCS. He is currently a chair professor (Artificial Intelligence) of the Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China. His research interests include machine learning and visual computing, as well as their applications in data science, pattern recognition, multi-objective optimization, and information security. He serves as an associate editor for *IEEE Transactions on Cybernetics*, *IEEE Transactions on Emerging Topics in Computational Intelligence*, *IEEE Transactions on Cognitive and Developmental Systems*, *IEEE Transactions on Neural Networks and Learning Systems (2014-2020)*, *Pattern Recognition, and Neurocomputing*, to name a few.



Zhikai Hu received the BS degree in computer science from China Jiliang University, Hangzhou, China, in 2015, and the MS degree in computer science from Huaqiao University, Xiamen, China, in 2019. He is currently working toward the PhD degree with the Department of Computer Science, Hong Kong Baptist University, Hong Kong, under the supervision of Prof. Yiu-ming Cheung. His present research interests include information retrieval, pattern recognition and data mining.