IMPROVING CALIBRATION FOR LONG-TAILED RECOGNITION

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep neural networks often perform poorly when training datasets are heavily class-imbalanced. Recently, two-stage methods greatly improve the performances by decoupling representation learning and classifier learning. In this paper, we discover that networks trained on long-tailed datasets are more prone to miscalibrated and over-confident. The two-stage models suffer the same issue as well. We design two novel methods to improve calibration and performance in such scenarios. Motivated by the predicted probability distributions of classes are highly related to the numbers of class instances, we propose a label-aware smoothing to deal with the different degrees of over-confidence for different classes and improve classifier learning. Noting that there is a dataset bias between these two stages because of different samplers, we further propose a shifted batch normalization to solve the dataset bias in the decoupling framework. Through extensive experiments, we also observe that mixup can remedy over-confidence and improve representation learning but has a negative or negligible effect on classifier learning. Our proposed methods set new records on multiple popular long-tailed recognition benchmarks including LT CIFAR 10/100, ImageNet-LT, Places-LT, and iNaturalist 2018.

1 INTRODUCTION

With numerous available large-scale and high-quality datasets such as ImageNet (Russakovsky et al., 2015), COCO (Lin et al., 2014), and Places (Zhou et al., 2017), deep convolutional neural networks (CNNs) have made notable breakthroughs in various computer vision tasks such as image recognition (Krizhevsky et al., 2012; He et al., 2016), object detection (Ren et al., 2015) and semantic segmentation (Cordts et al., 2016). These delicate datasets are usually artificially balanced with respect to the number of instances for each object/class. However, in real-world applications, data often follows an unexpected long-tailed distribution, where the numbers of instances for different classes are seriously imbalanced. When training CNNs on such long-tailed datasets, the performances extremely degrade. Motivated by this phenomenon, a number of works have recently emerged that try to explore long-tailed recognition.

Recently, many two-stage approaches have achieved significant improvement comparing with one-stage methods. Concretely, DRS and DRW (Cao et al., 2019) first train CNNs in a normal way in Stage-1. DRS finetunes CNNs on datasets with class-balanced resampling while DRW finetunes CNNs by assigning different weights to different classes in Stage-2. Zhou et al. (2020) proposed BBN with one-stage to simulate the process of DRS by dynamically combining the instance-balanced sampler and the reverse-balanced sampler. Kang et al. (2020) proposed two-stage decoupling models, cRT and LWS, to further boost the performance: Decoupling models freeze the backbone and just finetune the classifier with class-balanced resampling in Stage-2.

Confidence calibration (Niculescu-Mizil & Caruana, 2005; Guo et al., 2017) – the problem of predicting probability estimates representative of the true correctness likelihood – is important for recognition models in many applications (Bojarski et al., 2016; Jiang et al., 2012). In this study, we discover that networks trained on long-tailed datasets are more miscalibrated and over-confident: We draw the reliability diagrams with 15 bins in Fig. 1, which compares the plain model trained on the original CIFAR-100 dataset, the plain model, cRT, and LWS trained on long-tailed CIFAR-100 with imbalanced factor (IF) 100. We observe that networks trained on long-tailed datasets have higher expected calibration errors (ECEs). The two-stage models, cRT and LWS, suffer over-confidence as

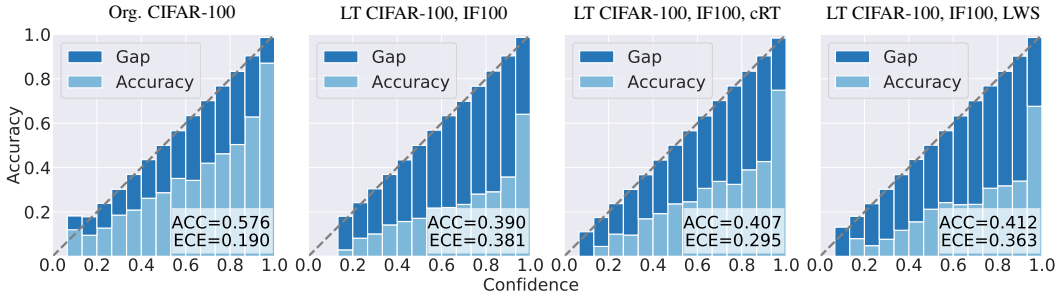


Figure 1: Reliability diagrams of ResNet-32. From left to right: the plain model trained on the original CIFAR-100 dataset, the plain model, cRT, and LWS trained on long-tailed CIFAR-100 with IF=100.

well. Moreover, Fig. 7 and Fig. 8 (the first two plots) in Appendix B depict that this phenomenon also commonly exists on other long-tailed datasets such as LT CIFAR-10 and ImageNet-LT.

Another issue is that two-stage decoupling methods ignore the dataset bias or domain shift (Quionero-Candela et al., 2009) between these two stages. Concretely, two-stage models are first trained on the instanced-balanced dataset $\mathcal{D}_{\mathrm{I}}$ in Stage-1. Then, models are trained on the class-balanced dataset $\mathcal{D}_{\mathrm{C}}$ in Stage-2. Obviously, $P_{\mathcal{D}_{\mathrm{I}}}(\boldsymbol{x}, y) \neq P_{\mathcal{D}_{\mathrm{C}}}(\boldsymbol{x}, y)$, the distributions of the dataset with different sampling manners are inconsistent. Motivated by the transfer learning methods (Li et al., 2018; Wang et al., 2019), we focus on the batch normalization (Ioffe & Szegedy, 2015) layer to deal with the dataset bias problem.

In this work, we propose a **Mi**xup **S**hifted **L**abel-**A**ware **S**moothing model (MiSLAS) to effectively solve the above issues. Our key contributions are as follows: (i) We discover that models trained on long-tailed datasets are much more miscalibrated and over-confident than them trained on balanced datasets. The two-stage models suffer the same problem as well. (ii) We find that mixup can remedy over-confidence and have a positive effect on representation learning but a negative or negligible effect on classifier learning. To further enhance classifier learning and calibration, we propose a label-aware smoothing to handle the different degrees of over-confidence for different classes. (iii) We are the first to note the dataset bias or domain shift in two-stage resampling methods for long-tailed recognition. To deal with the dataset bias in the decoupling framework, we propose shift learning on the batch normalization layer, which can greatly improve the performance.

We extensively validate our MiSLAS on multiple long-tailed recognition benchmark datasets, i.e., LT CIFAR-10, LT CIFAR-100, ImageNet-LT, Places-LT, and iNaturalist 2018. Experimental results manifest that the effectiveness and our method yields new state-of-the-art.

2 Related Works

Re-sampling and re-weighting. There are two groups of re-sampling strategies: over-sampling the tail-class images (Shen et al., 2016; Buda et al., 2018; Byrd & Lipton, 2019) and under-sampling the head-class images (Japkowicz & Stephen, 2002; Buda et al., 2018). Over-sampling is regularly useful on large datasets and often suffers from heavy over-fitting to tail classes especially on small datasets. For under-sampling, it discards a large portion of data, which inevitably causes degradation of the generalization ability of deep models. Re-weighting (Huang et al., 2016; Wang et al., 2017) is another prominent strategy. It assigns different weights for classes and even instances. The vanilla re-weighting method gives class weights in reverse proportion to the number of samples of classes. However, with large-scale data, re-weighting makes the deep models difficult to optimize during training. Cui et al. (2019) relieved the problem using the effective numbers to calculate the class weights. Another line of work is to adaptively re-weight each instance, e.g., Focal loss (Lin et al., 2017) assigned smaller weights for well-classified samples.

Network calibration and regularization. Calibrated confidence is significant for classification models in many applications. The calibration of modern neural networks is first discussed in Guo et al. (2017). The authors discovered that model capacity, normalization, and regularization have strong effects on network calibration. mixup (Zhang et al., 2018) is a regularization technique that is proposed to train with interpolations of inputs and labels. mixup inspires several follow-ups like manifold mixup (Verma et al., 2019), CutMix (Yun et al., 2019), and Remix (Chou et al., 2020) that

Table 1: Top-1 accuracy of the decoupling models (cRT and LWS) for ResNet families trained on the ImageNet-LT dataset. We vary the augmentation strategies (with or without mixup $\alpha = 0.2$) on both two stages.

Training setup for two stages	ResNet-10		ResNet-50		ResNet-101		ResNet-152	
	cRT	LWS	cRT	LWS	cRT	LWS	cRT	LWS
Stage-1 (no mixup)	36.8	36.8	45.8	45.8	47.3	47.3	48.7	48.7
Stage-1 (mixup)	35.7	35.7	45.6	45.6	47.7	47.7	48.4	48.4
Stage-1 (no mixup) + Stage-2 (no mixup)	**43.3**	**43.5**	**50.3**	**51.2**	51.4	**52.3**	52.7	**53.8**
Stage-1 (no mixup) + Stage-2 (mixup)	43.0	43.3	50.2	51.1	51.4	52.2	**52.8**	53.6
Stage-1 (mixup) + Stage-2 (no mixup)	**43.4**	**42.9**	**51.7**	**52.0**	**53.1**	53.5	**54.2**	**54.6**
Stage-1 (mixup) + Stage-2 (mixup)	43.3	42.8	51.6	51.9	53.0	53.5	54.1	54.5

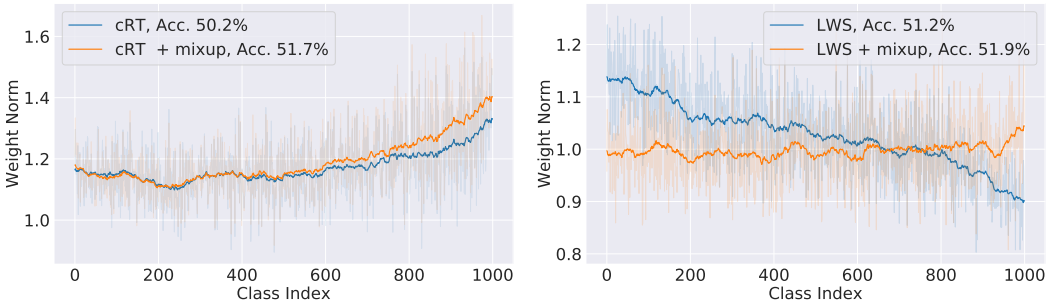


Figure 2: Classifier weight norms for the ImageNet-LT validation set when classes are sorted by descending values of N_j. Left: weight norms of cRT with or without mixup. Right: weight norms of LWS with or without mixup. (light shade: true norm, dark lines: smooth version)

have shown significant improvement over mixup. Thulasidasan et al. (2019) found that CNNs trained with mixup are significantly better calibrated. Label smoothing (Szegedy et al., 2016) is another regularization technique that encourages the model to be less over-confident. Unlike cross-entropy computes loss upon the ground truth labels, label smoothing computes loss upon a soft version of the label, which can relieve the over-fitting and increase calibration and reliability (Müller et al., 2019).

Two-stage methods. Cao et al. (2019) first proposed deferred re-weighting (DRW) and deferred re-sampling (DRS) that are superior to conventional one-stage methods: Stage-2, starting from better features, adjusts the decision boundary and locally fine-tunes the features. Recently, Kang et al. (2020) and Zhou et al. (2020) concluded that although class re-balance strategies matter when jointly training representation and classifier, instance-balanced sampling gives more general representations. Based on this observation, Kang et al. (2020) achieved state-of-the-art results by decomposing representation and classifier learning, i.e., first train the deep models with instance-balanced sampling, then fine-tune the classifier with class-balanced sampling while keeping parameters of representation learning fixed. Similarly, Zhou et al. (2020) integrated mixup training into the proposed cumulative learning strategy with which they bridged the representation learning and classifier re-balancing. The cumulative learning strategy requires dual samplers: instance-balanced and reversed instance-balanced sampler.

3 Main Approach

3.1 Improving calibration and representation learning by mixup

For the two-stage learning framework, Kang et al. (2020) and Zhou et al. (2020) found that instance-balanced sampling gives the most generalizable representations among other sampling methods. Thulasidasan et al. (2019) found that networks trained with mixup are better calibrated. When using instance-balanced sampling, to further improve the representation generalization and relieve over-confidence, we explore the effect of mixup in the two-stage decoupling framework.

Here, we train two two-stage models, i.e. cRT and LWS, on ImageNet-LT for 180 epochs in Stage-1 and finetune for 10 epochs in Stage-2, respectively. We vary the training setup (with/without mixup

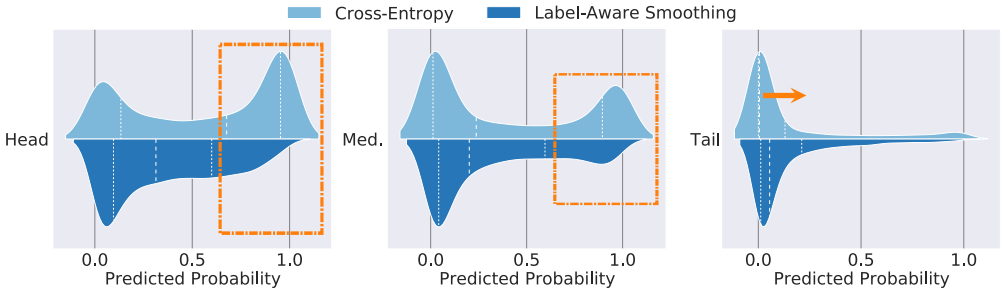


Figure 3: Violin plot of predicted probability distributions for different parts of classes, head (more than 100 images), medium (20 to 100 images), and tail (less than 20 images) on LT CIFAR-100, IF=100. The upper half part in light blue: LWS (cross-entropy). The bottom half part in deep blue: LWS (label-aware smoothing).

$\alpha = 0.2$) for both two stages. Top-1 accuracy results of these variants are listed in Table 1. From it, we conclude that: (i) When applying mixup, the performance improvements of Stage-1 are ignorable but the performances of Stage-2 are greatly enhanced for both cRT and LWS. (ii) Applying additional mixup in Stage-2 has no obvious improvement or even damages the performance, which means that mixup encourages representation learning but has a negative or negligible effect on classifier learning.

We also draw the final classifier weight norms of these variants in Fig. 2. We show the L_2 norms of the weight vectors for all classes, as well as the training data distribution sorted in a descending manner concerning the number of instances. We observe that when applying mixup (orange line), the weight norms of the tail classes uniformly tend to become larger and the weight norms of the head classes are decreased, which means mixup may be more friendly to the tail classes.

The analysis of calibration for networks whether adding mixup will be discussed in our experiment part (Sec. 4.2). Due to the poor and unsatisfied enhancement of mixup for classifier learning, we further propose a label-aware smoothing to improve both the calibration and classifier learning.

3.2 Improving Calibration and Classifier Learning by Label-Aware Smoothing

As discussed in the introduction part and Sec. 3.1, two-stage models suffer serious over-confidence and there is no significant improvement for classifier learning when adding additional mixup. In this subsection, we try to analyze and deal with these two issues. Suppose that the weight of the classifier is $\boldsymbol{W} \in \mathbb{R}^{M \times K}$, where M is the number of features and K is the number of classes. The cross-entropy encourages the whole network to be over-confident on the head classes: Concretely, the cross-entropy loss after the softmax activation is $l(y, \boldsymbol{p}) = -\log(\boldsymbol{p}_y) = -\boldsymbol{w}_y^\top \boldsymbol{x} + \log(\sum \exp(\boldsymbol{w}_i^\top \boldsymbol{x}))$, where $y \in \{1, 2, ..., K\}$ is the label, $\boldsymbol{x} \in \mathbb{R}^M$ is the feature vector send to classifier and $\boldsymbol{w}_i$ is the i-th column vector of $\boldsymbol{W}$. The optimal solution is $\boldsymbol{w}_y^{*\top} \boldsymbol{x} = \inf$ while keeping others $\boldsymbol{w}_i^\top \boldsymbol{x}$, $i \neq y$, small enough. Because the head classes contain much more training examples, the network makes the weight norm $\|\boldsymbol{w}\|$ of the head classes become larger to near the optimal solution as much as possible, which results that their predicted probabilities mainly concentrate near 1.0 (see Fig. 3, the upper half part showing in light blue). Another fact we can get from Fig. 3 is that the distributions of predicted probability are severely related to the instance numbers. Unlike balanced recognition, we claim that applying different strategies for different classes is extremely necessary for the long-tailed problem.

Here, we propose a label-aware smoothing to solve the over-confidence in cross-entropy and the different distributions of predicted probability issue. The mathematical computation of label-aware smoothing is:

$$l(\boldsymbol{q}, \boldsymbol{p}) = -\sum_{i=1}^{K} \boldsymbol{q}_i \log \boldsymbol{p}_i, \quad \boldsymbol{q}_i = \begin{cases} 1 - \epsilon_y = 1 - f(N_y), & i = y, \\ \frac{\epsilon_y}{K-1} = \frac{f(N_y)}{K-1}, & \text{otherwise}, \end{cases} \tag{1}$$

where ϵ_y is a small label smoothing factor for Class-y and relates to its class number N_y. Now the optimal solution becomes:

$$\boldsymbol{w}_i^{*\top} \boldsymbol{x} = \begin{cases} \log\left(\frac{(K-1)(1-\epsilon_y)}{\epsilon_y}\right) + c, & i = y, \\ c, & \text{otherwise}, \end{cases} \tag{2}$$

where c can be an arbitrary real number. Comparing with the infinite optimal solution in cross-entropy, the label-aware smoothing encourages a finite output, which can get more generalized results and remedy over-fitting. We suppose the labels of the long-tailed dataset are assigned in a descending manner concerning the number of instances, i.e., $N_1 \geq N_2 \geq ... \geq N_K$. Because the head classes contain more various and diverse examples, the predicted probabilities are more promising than them of tail classes. Thus, we suggest classes with larger instance numbers should be penalized larger label smoothing factors, that is, the related function $f(N_y)$ should be negatively correlated to N_y. We define three types of related function $f(N_y)$:

$$\epsilon_y = f(N_y) = \begin{cases} \text{(Concave)} & \epsilon_K + (\epsilon_1 - \epsilon_K) \sin\left[\frac{\pi(N_y - N_K)}{2(N_1 - N_K)}\right], \quad y = 1, 2, ..., K, \\ \text{(Linear)} & \epsilon_K + (\epsilon_1 - \epsilon_K) \frac{N_y - N_K}{N_1 - N_K}, \quad y = 1, 2, ..., K, \\ \text{(Convex)} & \epsilon_1 + (\epsilon_1 - \epsilon_K) \sin\left[\frac{3\pi}{2} + \frac{\pi(N_y - N_K)}{2(N_1 - N_K)}\right], \quad y = 1, 2, ..., K, \end{cases} \tag{3}$$

where ϵ_1 and ϵ_K are two hyperparameters. If we set $\epsilon_1 \geq \epsilon_K$, then we can get $\epsilon_1 \geq \epsilon_2 \geq ... \geq \epsilon_K$. It means that if the instance number N_y for Class-y is larger, label-aware smoothing will allocate a larger smoothing factor and lower the fitting probability to relieve the over-confidence because the head and medium classes are more likely to be over-confident than the tail classes (see Fig. 3).

As the form of label-aware smoothing is more complicated than cross-entropy, we propose a more generalized classifier learning framework to fit it. Here we give a quick review about cRT and LWS: cRT tries to learn a new classifier weight, which contains KM learnable parameters. LWS is restricted to learn the weight scaling vector $\boldsymbol{s} \in \mathbb{R}^K$, which contains only K learnable parameters. By contrast, cRT has more learnable parameters. It means cRT has a more powerful representation ability. LWS tends to obtain better validation losses and performances on large-scale datasets (refer to the experiment part in Kang et al. (2020)). It means LWS has a better generalization property. To combine the advantages of both cRT and LWS, we redesign the classifier framework in Stage-2:

$$\boldsymbol{z} = \text{diag}(\boldsymbol{s}) \left(a\boldsymbol{W} + \Delta\boldsymbol{W}\right)^\top \boldsymbol{x}. \tag{4}$$

In Eqn. (3), we fix the original classifier weight $\boldsymbol{W}$ in Stage-2. If we make the learnable scaling vector $\boldsymbol{s}$ fixed, set $\boldsymbol{s} = \mathbf{1}$, $a = 0$, and just learn the new classifier weight $\Delta\boldsymbol{W} \in \mathbb{R}^{M \times K}$, Eqn. (4) will degrade to cRT. Because LWS fixes the original classifier weights $\boldsymbol{W}$ and only learns the scaling $\boldsymbol{s}$, Eqn. (4) will degrade to LWS if we set $a = 1$ and $\Delta\boldsymbol{W} = \mathbf{0}$. In most cases, LWS generally achieves better results than cRT on large scale datasets. Thus, we let $\boldsymbol{s}$ learnable and set $a = 1$. We also make $\Delta\boldsymbol{W}$ learnable to improve the representation ability but optimize $\Delta\boldsymbol{W}$ by a different learning rate. $\Delta\boldsymbol{W}$ can be viewed as doing a shift transformation on $\boldsymbol{W}$. This transformation can change the direction of the original weight vector $\boldsymbol{w}$ in $\boldsymbol{W}$, which is what LWS cannot do.

3.3 Shift learning on batch normalization

In the two-stage training framework, models are first trained with instance-balanced sampling in Stage-1 and then trained with class-balanced sampling in Stage-2. Since the framework involves two samplers, or two datasets: the instance-balanced dataset $\mathcal{D}_\text{I}$ and the class-balanced dataset $\mathcal{D}_\text{C}$, we can regard this two-stage training framework as a derivative of transfer learning approaches. However, if we view the two-stage decoupling training framework from the transfer learning perspective, fixing the backbone part and just fine-tuning the classifier in Stage-2 will be clearly unreasonable, especially for the batch normalization (BN) layers.

Concretely, we suppose that the input of the network is $\boldsymbol{x}_i$, the input feature of some BN layer is $g(\boldsymbol{x}_i)$, and the mini-batch size is m. The running mean and the running variance of Channel-j for these two stages are:

$$\boldsymbol{\mu}_\text{I}^{(j)} = \frac{1}{m}\sum_{i=1}^{m} g(\boldsymbol{x}_i)^{(j)}, \quad \boldsymbol{\sigma}_\text{I}^{2\,(j)} = \frac{1}{m}\sum_{i=1}^{m}\left[g(\boldsymbol{x}_i)^{(j)} - \boldsymbol{\mu}_\text{I}^{(j)}\right]^2, \quad \boldsymbol{x}_i \sim P_{\mathcal{D}_\text{I}}(\boldsymbol{x}, y), \tag{5}$$

$$\boldsymbol{\mu}_\text{C}^{(j)} = \frac{1}{m}\sum_{i=1}^{m} g(\boldsymbol{x}_i)^{(j)}, \quad \boldsymbol{\sigma}_\text{C}^{2\,(j)} = \frac{1}{m}\sum_{i=1}^{m}\left[g(\boldsymbol{x}_i)^{(j)} - \boldsymbol{\mu}_\text{C}^{(j)}\right]^2, \quad \boldsymbol{x}_i \sim P_{\mathcal{D}_\text{C}}(\boldsymbol{x}, y). \tag{6}$$

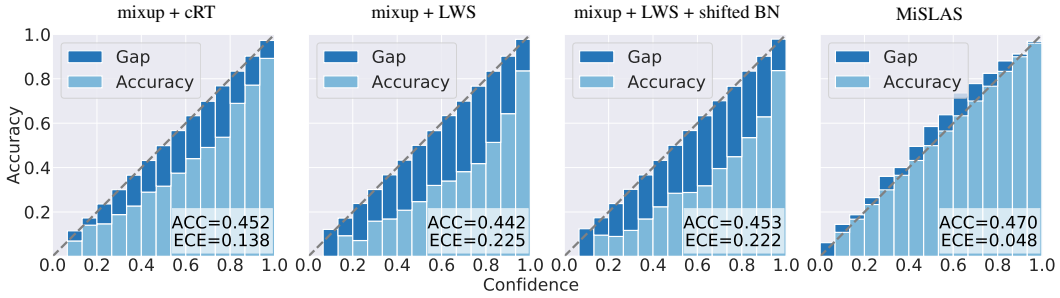


Figure 4: Reliability diagrams of ResNet-32 trained on LT CIFAR-100, IF=100. From left to right: cRT with mixup, LWS with mixup, LWS with mixup and shifted BN, and MiSLAS. It is better to look together with Fig. 1.

Due to the different sampling strategies, the composition ratios of the head, medium, and tail classes are also totally different, which leads to $P_{\mathcal{D}_\mathrm{I}}(\boldsymbol{x}, y) \neq P_{\mathcal{D}_\mathrm{C}}(\boldsymbol{x}, y)$. Calculated by Eqn. (5) and (6), there exist some biases in $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ under two sampling strategies, i.e., $\boldsymbol{\mu}_\mathrm{I} \neq \boldsymbol{\mu}_\mathrm{C}$, and $\boldsymbol{\sigma}_\mathrm{I}^2 \neq \boldsymbol{\sigma}_\mathrm{C}^2$. Thus, it is clearly infeasible for the decoupling framework that BN shares mean and variance across datasets with two sampling strategies. Motivated by AdaBN (Li et al., 2018) and TransNorm (Wang et al., 2019), we unfreeze the update procedures of the running mean $\boldsymbol{\mu}$ and running variance $\boldsymbol{\sigma}$ but fix the learnable linear transformation parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ for a better normalization in Stage-2.

4 Experiments

4.1 Experimental setup

Our experimental setup including the implementation details and evaluation protocol mainly follows Cao et al. (2019) for LT CIFAR-10 and LT CIFAR-100, and Kang et al. (2020) for ImageNet-LT, Places-LT, and iNuturalist 2018. Please see Appendix A for further details.

4.2 Ablation study

Improving calibration. Here we show the reliability diagrams with 15 bins of our methods in Fig. 4. Comparing with Fig. 1 in the introduction part, both the mixup and label-aware smoothing can not only largely enhance the network calibration (even lower ECEs than them on balanced datasets) but also greatly improve the performance for long-tailed recognition. The similar trends can also be found on LT CIFAR-10, ImageNet-LT, and Places-LT (please see the figures in Appendix B for detail), which proves the powerful effects of the proposed method on calibration. According to all experiment results, training networks on imbalanced datasets leads to more severe over-confidence. Since the conventional mixup and label-smoothing both contain the operation of softening the ground truth labels, which may suggest that training with hard labels is likely to be another contributing factor leading to network over-confidence.

Further analysis of label-aware smoothing. In our label-aware smoothing, there are two hyper-parameters in Eqn. (3), ϵ_1 and ϵ_K, which control the penalties of classes. In recognition system, if the predicted probability of some Class-y is larger than 0.5, the classifier will classify the input to Class-y. Thus, to ensure reasonability, we limit $0 \leq \epsilon_K \leq \epsilon_1 \leq 0.5$. Here we conduct a comparing experiment for varying ϵ_1 and ϵ_K both from 0.0 to 0.5 on LT CIFAR-100 with imbalanced factor 100. We plot the performance matrix upon ϵ_1 and ϵ_k in Fig. 5 for all possible variants. From it, the classification accuracy can be further improved by 0.9% comparing with the conventional cross-entropy ($\epsilon_1 = 0$, $\epsilon_K = 0$, green square) when we pick $\epsilon_1 = 0.4$, $\epsilon_K = 0.1$ (orange square) for label-aware smoothing. A more surprising improvement (growing by 3.3%) can be found on LT CIFAR-10 (see Appendix D.1 for detail). We also find that the concave related function $f(\cdot)$ in Eqn. (3) achieves the best performance but the gain is quite limited (refer Appendix D.2 for detail).

To visualize the change in predicted probability distributions, we train two LWS models, one with cross-entropy and the other with label-aware smoothing on long-tailed CIFAR-100 with imbalanced factor 100. The cross-entropy-based distributions of the head, medium, and tail classes are showing in the upper half part of Fig. 3 in light blue. The label-aware smoothing-based distributions are showing in the bottom half part in deep blue. We observe that the over-confidence of head and medium classes relieve greatly, and the whole distribution of the tail classes slightly moves right (a larger mean) when using label-aware smoothing. This empirical visualization is consistent with our analysis mentioned in Sec. 3.2.

Table 2: Ablation study for all proposed modules on long-tailed CIFAR-100, IF=100. MU: applying mixup just in Stage-1. SL: shift learning on batch normalization. LAS: label-aware smoothing.

Module			LT CIFAR-100		
MU	SL	LAS	100	50	10
☒	☒	☒	41.2	46.0	58.5
☑	☒	☒	44.2	50.6	62.2
☑	☑	☒	45.3	51.4	62.8
☑	☑	☑	**47.0**	**52.3**	**63.0**

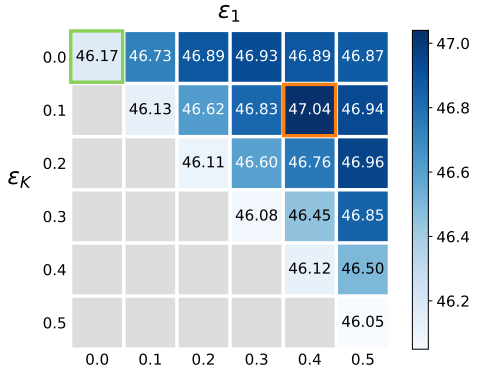


Figure 5: Ablation study of two hyperparameters ϵ_1 and ϵ_K in label-aware smoothing.

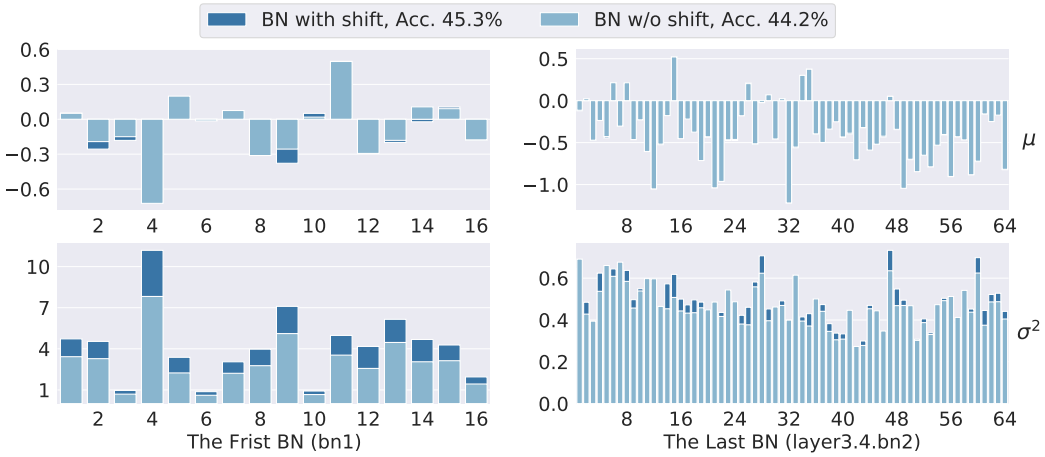


Figure 6: Visualization of the changes in the running mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}^2$. The ResNet-32 based model is trained on LT CIFAR-100 with imbalanced factor 100. Left: $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ in the first BN of ResNet-32, which contains 16 channels. Right: $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ in the last BN of ResNet-32, which contains 64 channels.

Further analysis of shift learning. In this part, we conduct an empirical experiment to show the effectiveness and reasonability of shift learning on BN. We train the LWS model on long-tailed CIFAR-100 with imbalanced factor 100. After 10 epochs finetuning in Stage-2, the model trained with BN shifting achieves accuracy at 45.3%, which is 1.1% higher than it without BN shifting. We also draw a visualization of the change in BN. As shown in Fig. 6, we see that there indeed exist biases in $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ between the dataset using different sampling strategies. Due to the composition ratios of the head classes, medium classes and tail classes are different in terms of different sampling strategies, the statistic running mean $\boldsymbol{\mu}$ and running variance $\boldsymbol{\sigma}^2$ are certainly different. We also find some interesting phenomenons need for future exploration: (i) The changes in the running variance $\boldsymbol{\sigma}^2$ are larger than the changes in the running mean $\boldsymbol{\mu}$. (ii) The changes of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ in deep BN layers are quite smaller than them in shallow BN layers.

Overall, Table 2 shows the ablation investigation on the effects of mixup (adding mixup in Stage-1, MU), shift learning on batch normalization (SL), and label-aware smoothing (LAS). From it, each proposed module can further improve the performances on long-tailed CIFAR-100 for all commonly-used imbalanced factors, which firmly demonstrates the effectiveness.

4.3 Comparison with the state-of-the-art

In this subsection, we compare the proposed method against previous one-stage methods, such as Range Loss (Zhang et al., 2017), LDAM Loss (Cao et al., 2019), FSLwF (Gidaris & Komodakis, 2018), and OLTR (Liu et al., 2019), and against previous two-stage methods, such as DRS-like, DRW-like (Cao et al., 2019), LFME (Xiang & Ding, 2020), cRT, and LWS (Kang et al., 2020). For fair comparisons, we also add mixup on the LWS and cRT models. Remix (Chou et al., 2020) is a

Table 3: Top-1 accuracy (%) for ResNet-32 models trained on long tailed CIFAR-10 and CIFAR-100.

Method	Long-tailed CIFAR-10			Long-tailed CIFAR-100		
	100	50	10	100	50	10
CE	70.4	74.8	86.4	38.4	43.9	55.8
mixup	73.1	77.8	87.1	39.6	45.0	58.2
LDAM+DRW	77.1	81.1	88.4	42.1	46.7	58.8
BBN$_{\text{(include mixup)}}$	79.9	82.2	88.4	42.6	47.1	59.2
Remix+DRW$_{\text{(300 epochs)}}$	79.8	-	89.1	46.8	-	61.3
cRT+mixup	79.1	84.2	89.8	45.1	50.9	62.1
LWS+mixup	76.3	82.6	89.6	44.2	50.6	62.2
MiSLAS	**82.1**	**85.8**	**89.9**	**47.0**	**52.3**	**63.0**

Table 4: Top-1 accuracy (%) on ImageNet-LT (left), iNaturalist 2018 (center) and Place-LT (right).

Method	ResNet-50
CE	44.6
CE+DRW	48.5
Focal+DRW	47.9
LDAM+DRW	48.8
CRT+mixup	51.7
LWS+mixup	52.0
MiSLAS	**52.7**

(a) ImageNet-LT

Method	ResNet-50
CB-Focal	61.1
LDAM+DRW	68.0
BBN$_{\text{(include mixup)}}$	69.6
Remix+DRW	70.5
cRT+mixup	70.2
LWS+mixup	70.9
MiSLAS	**71.6**

(b) iNaturalist 2018

Method	ResNet-152
Range Loss	35.1
FSLwF	34.9
OLTR	35.9
OLTR+LFME	36.2
cRT+mixup	38.3
LWS+mixup	39.7
MiSLAS	**40.4**

(c) Place-LT

recently proposed augmentation method for long-tail recognition. Because BBN (Zhou et al., 2020) has double samplers and is trained in a mixup-like manner, we directly compare our method with it.

Experimental results on CIFAR-LT. We conduct extensive experiments on long-tailed CIFAR-10 and CIFAR-100 with imbalanced factors of 10, 50, and 100, which is the same as the previous setting (Cao et al., 2019; Zhou et al., 2020). The experimental results are summarized in Table 3. Compared with previous methods (+mixup, one/two-stage), our MiSLAS outperforms all previous methods by a large margin. Moreover, this superiority of the proposed method holds for all imbalanced factors on both long-tailed CIFAR-10 and CIFAR-100.

Experimental results on ImageNet-LT, iNaturalist 2018, and Place-LT. We further verify the effectiveness of our method on three large-scale imbalanced datasets, i.e., ImageNet-LT, iNaturalist 2018, and Place-LT. Table 4 lists experimental results on ImageNet-LT (left), iNaturalist 2018 (center), and Places-LT (right). Notably, our MiSLAS still outperforms all competing approaches and sets new state-of-the-art records for all three large-scale long-tailed benchmarks. More detailed results about the split class accuracies and different backbones on these three datasets are listed in Appendix C.

5 Conclusion

In this paper, we discover that models trained on long-tailed datasets are more miscalibrated and over-confident than them trained on balanced datasets. The two-stage models suffer the same issue as well. To relieve over-confidence, we propose two solutions: (i) We find that mixup can remedy over-confidence and have a positive effect on representation learning but a negative or negligible effect on classifier learning. (ii) To further improve classifier learning and calibration, we propose label-aware smoothing to handle the different degrees of over-confidence for different classes. We are the first to note the dataset bias or domain shift in two-stage resampling methods for long-tailed recognition. To solve the dataset bias producing by different re-sampling in the decoupling framework, we propose shift learning on the batch normalization layer and this novel model can greatly improve the performance. Extensive quantitative and qualitative experiments on multiple benchmark datasets show that our MiSLAS achieves superior performances over the state-of-the-art methods.

REFERENCES

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *ICML*, pp. 872–881, 2019.

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *NeurIPS*, pp. 1567–1578, 2019.

Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: Rebalanced mixup. In *ECCVW*, 2020.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes dataset for semantic urban scene understanding. In *CVPR*, pp. 3213–3223, 2016.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, pp. 9268–9277, 2019.

Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, pp. 4367–4375, 2018.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *ICML*, pp. 1321–1330, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *CVPR*, pp. 5375–5384, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

Xiaoqian Jiang, Melanie Osl, Jihoon Kim, and Lucila Ohno-Machado. Calibrating predictive model estimates to support personalized medicine. *Journal of the American Medical Informatics Association*, 19(2):263–274, 2012.

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *ICLR*, 2020.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pp. 1097–1105, 2012.

Yanghao Li, Naiyan Wang, Jianping Shi, Xiaodi Hou, and Jiaying Liu. Adaptive batch normalization for practical domain adaptation. *Pattern Recognition*, 80:109–117, 2018.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, pp. 740–755, 2014.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pp. 2980–2988, 2017.

Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, pp. 2537–2546, 2019.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *ICLR*, 2017.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? In *NeurIPS*, pp. 4694–4703, 2019.

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *ICML*, pp. 625–632, 2005.

Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. The MIT Press, 2009.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeurIPS*, pp. 91–99, 2015.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

Li Shen, Zhouchen Lin, and Qingming Huang. Relay backpropagation for effective learning of deep convolutional neural networks. In *ECCV*, pp. 467–482, 2016.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pp. 2818–2826, 2016.

Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *NeurIPS*, pp. 13888–13899, 2019.

Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The iNaturalist species classification and detection dataset. In *CVPR*, pp. 8769–8778, 2018.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, pp. 6438–6447, 2019.

Ximei Wang, Ying Jin, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Transferable normalization: Towards improving transferability of deep neural networks. In *NeurIPS*, pp. 1953–1963, 2019.

Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. In *NeurIPS*, pp. 7029–7039, 2017.

Liuyu Xiang and Guiguang Ding. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *ECCV*, 2020.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, pp. 6023–6032, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ICLR*, 2018.

Xiao Zhang, Zhiyuan Fang, Yandong Wen, Zhifeng Li, and Yu Qiao. Range loss for deep face recognition with long-tailed training data. In *ICCV*, pp. 5409–5418, 2017.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE TPAMI*, 40(6):1452–1464, 2017.

Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. BBN: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *CVPR*, pp. 9719–9728, 2020.

A EXPERIMENT SETUP

A.1 DATASETS EXPLANATION

CIFAR-10-LT and CIFAR-100-LT. CIFAR-10 and CIFAR-100 both have 60,000 images 50,000 for training and 10,000 for validation with 10 categories and 100 categories, respectively. For a fair comparison, we use the long-tailed versions of CIFAR datasets with the same setting as those used in Cao et al. (2019). They control the degrees of data imbalance with an imbalanced factor β. $\beta = \frac{N_{\max}}{N_{\min}}$, where $N_{\max}$ and $N_{\min}$ are the numbers of training samples for the most frequent class and the least frequent class. Following Cao et al. (2019) and Zhou et al. (2020), we conduct experiments with imbalanced factors 100, 50, and 10.

ImageNet-LT and Places-LT. ImageNet-LT and Places-LT were proposed by Liu et al. (2019). ImageNet-LT is a long-tailed version of the large-scale object classification dataset ImageNet (Russakovsky et al., 2015) by sampling a subset following the Pareto distribution with power value $\alpha = 6$. It contains 115.8K images from 1,000 categories, with class cardinality ranging from 5 to 1,280. Places-LT is a long-tailed version of the large-scale scene classification dataset Places (Zhou et al., 2017). It consists of 184.5K images from 365 categories with class cardinality ranging from 5 to 4,980.

iNaturalist 2018. iNaturalist 2018 (Van Horn et al., 2018) is one species classification dataset, which is on a large scale and suffers from extremely imbalanced label distributions. It is composed of 437.5K images from 8,142 categories. In addition to the extreme imbalance, the iNaturalist 2018 dataset also confronts the fine-grained problem.

A.2 EVALUATION PROTOCOL

Following Liu et al. (2019) and Kang et al. (2020), we report the commonly used top-1 accuracy over all classes on the balanced test/validation datasets, denoted as *All*. In more detail, further report accuracy on three splits of the set of classes: *Head-Many* (more than 100 images), *Med.-Medium* (20 to 100 images) and *Tail-Few* (less than 20 images).

A.3 IMPLEMENTATION DETAILS

For all experiments, we use the SGD optimizer with momentum 0.9 to optimize networks.

For long-tailed CIFAR, we mainly follow Cao et al. (2019). We train all MiSLAS models with the ResNet-32 backbone on one GPU and use the multistep learning rate schedule, which decreases the learning rate by 0.1 at the 160^{th} epoch and the 180^{th} epochs in Stage-1.

For ImageNet-LT, Place-LT, and iNaturalist 2018, We mainly follow Kang et al. (2020) and use the cosine learning rate schedule (Loshchilov & Hutter, 2017) to train all MiSLAS models with the ResNet-10/50/101/152 backbones on 4 GPUs.

Table 5: Detailed experiment settings on five benchmark datasets. LR: learning rate, BS: batch size, WD: weight decay, and LRS: learning rate schedule, $\Delta \boldsymbol{W}$: learning rate ratio of $\Delta \boldsymbol{W}$.

Dataset	Common			Stage-1		Stage-2				
	LR	BS	WD	Epochs	LRS	Epochs	LRS	ϵ_1	ϵ_K	$\Delta \boldsymbol{W}$
LT CIFAR-10	0.1	128	2e-4	200	*multi.*	10	*cosine*	0.3	0.0	0.5x
LT CIFAR-100	0.1	128	2e-4	200	*multi.*	10	*cosine*	0.4	0.1	0.2x
ImageNet-LT	0.1	256	5e-4	180	*cosine*	10	*cosine*	0.4	0.1	0.05x
Places-LT	0.1	256	5e-4	90	*cosine*	10	*cosine*	0.4	0.1	0.05x
iNaturalist'18	0.1	256	1e-4	200	*cosine*	30	*cosine*	0.4	0.1	0.05x

B CALIBRATION

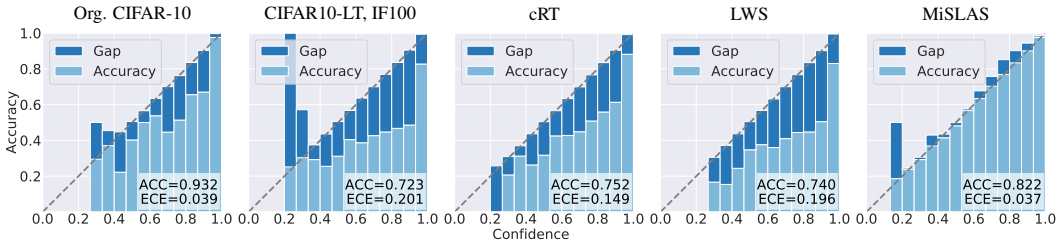


Figure 7: Reliability diagrams on CIFAR10 with 15 bins. From left to right: plain ResNet-32 model trained on the original CIFAR-10 dataset, plain model, cRT, LWS, and MiSLAS trained on long-tailed CIFAR-10 with imbalanced factor 100.

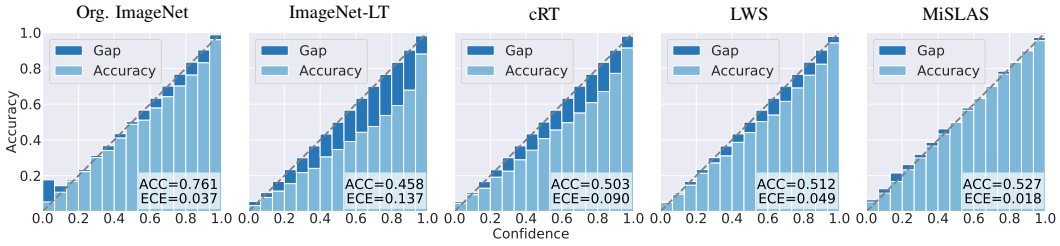


Figure 8: Reliability diagrams on ImageNet with 15 bins. From left to right: plain ResNet-50 model trained on the original ImageNet dataset, plain model, cRT, LWS, and MiSLAS trained on ImageNet-LT.

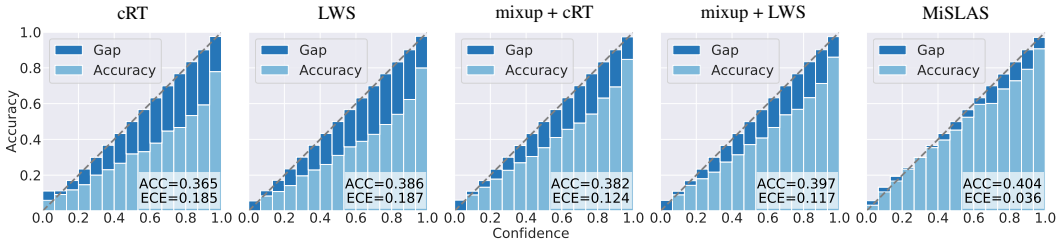


Figure 9: Reliability diagrams of ResNet-152 trained on Places-LT with 15 bins. From left to right: cRT, LWS, cRT with mixup, LWS with mixup, and MiSLAS.

C MORE DETAILED RESULTS ON IMAGENET-LT, PLACES-LT AND INATURALIST 2018

Table 6: Comprehensive accuracy results on ImageNet-LT with different backbone networks (ResNet-50, ResNet-101 & ResNet-152) and training 180 epochs.

Backbone	Method	Many	Medium	Few	All
ResNet-50	cRT	62.5	47.4	29.5	50.3
	LWS	61.8	48.6	33.5	51.2
	cRT+mixup	**63.9**	49.1	30.2	51.7
	LWS+mixup	62.9	49.8	31.6	52.0
	MiSLAS	61.7	**51.3**	**35.8**	**52.7**
ResNet-101	cRT	63.8	48.5	30.0	51.4
	LWS	63.1	49.9	33.8	52.3
	cRT+mixup	**65.2**	50.6	31.6	53.1
	LWS+mixup	64.5	51.2	34.1	53.5
	MiSLAS	64.3	**52.1**	**35.8**	**54.1**
ResNet-152	cRT	64.9	50.4	30.6	52.7
	LWS	64.1	51.8	35.5	53.8
	cRT+mixup	**66.5**	51.6	32.8	54.2
	LWS+mixup	66.1	52.2	34.5	54.6
	MiSLAS	65.4	**53.2**	**37.1**	**55.2**

Table 7: Comprehensive accuracy results on iNaturalist 2018 with ResNet-50 and training 200 epochs.

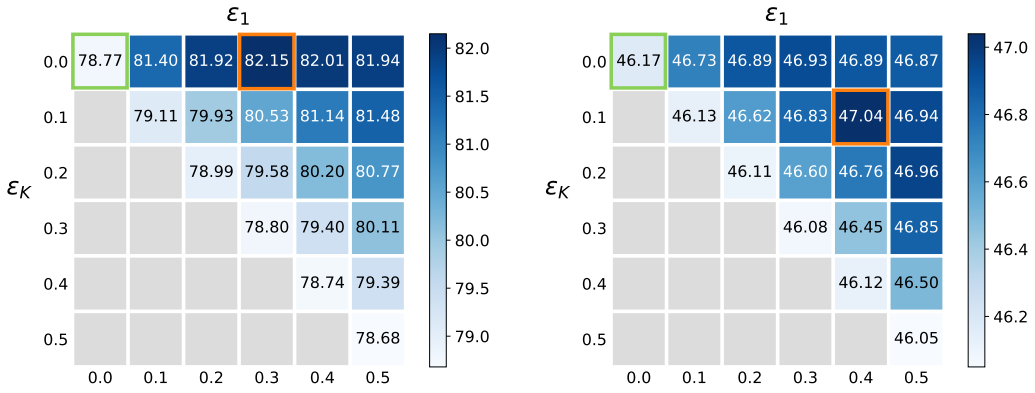
Backbone	Method	Many	Medium	Few	All
ResNet-50	cRT	73.2	68.8	66.1	68.2
	τ-normalized	71.1	68.9	69.3	69.3
	LWS	71.0	69.8	68.8	69.5
	cRT+mixup	**74.2**	71.1	68.2	70.2
	LWS+mixup	72.8	71.6	69.8	70.9
	MiSLAS	73.2	**72.4**	**70.4**	**71.6**

Table 8: Detailed accuracy results on Places-LT, starting from an ImageNet pre-trained ResNet-152.

Backbone	Method	Many	Medium	Few	All
ResNet-152	Lifted Loss	41.1	35.4	24.0	35.2
	Focal Loss	41.1	34.8	22.4	34.6
	Range Loss	41.1	35.4	23.2	35.1
	FSLwF	43.9	29.9	29.5	34.9
	OLTR	**44.7**	37.0	25.3	35.9
	OLTR+LFME	39.3	39.6	24.2	36.2
	cRT	42.0	37.6	24.9	36.7
	τ-normalized	37.8	40.7	31.8	37.9
	LWS	40.6	39.1	28.6	37.6
	cRT+mixup	44.1	38.5	27.1	38.1
	LWS+mixup	41.7	41.3	33.1	39.7
	MiSLAS	39.6	**43.3**	**36.1**	**40.4**

D ABLATION STUDY OF LABEL-AWARE SMOOTHING

D.1 MORE RESULTS ABOUT THE HYPERPARAMETERS ϵ_1 AND ϵ_K



LT CIFAR-10, IF=100 LT CIFAR-100, IF=100

Figure 10: Ablation study of two hyperparameters ϵ_1 and ϵ_K in label-aware smoothing. Our label-aware smoothing (orange square) outperforms cross-entropy (green square) by a large margin on both long-tailed CIFAR-10 (left) and long-tailed CIFAR-100 (right).

D.2 FORM OF THE RELATED FUNCTION $f(\cdot)$

As discussed in Sec. 3.2 and Sec. 4.2, the form of the related function $f(\cdot)$ may play a significant role for the final model performance. We draw the visualization of Eqn. (3) at the left part of Fig. 11. For the LT CIFAR-100 dataset with balanced factor 100, $N_1 = 500$ and $N_{100} = 5$. Based on the ablation study results of ϵ_1 and ϵ_K mentioned in Sec. 4.2 and above, we set $\epsilon_1 = 0.4$ and $\epsilon_{100} = 0.1$ here. After fintuning for 10 epochs in Stage-2, the accuracy of the concave model is the best. We also design a power-like related function, which can be written as:

$$\epsilon_y = f(N_y) = \epsilon_K + (\epsilon_1 - \epsilon_K)\left(\frac{N_y - N_K}{N_1 - N_K}\right)^p, \qquad y = 1, 2, ..., K, \tag{7}$$

where p is a hyperparameter to control the shape of the related function. For example, we will get concave related function if we set $p < 1$ and we will get convex related function if we set $p > 1$. The visualization of Eqn. (7) is shown at the right part of Fig. 11. However, comparing the accuracies of all variants, the influence of the related function form is quite limited for the final performance (just growing by 0.3%). Because the concave related function in Eqn. (3) achieves the best performance among all variants, we choose it as the default setting of the related function $f(\cdot)$ for other experiments.

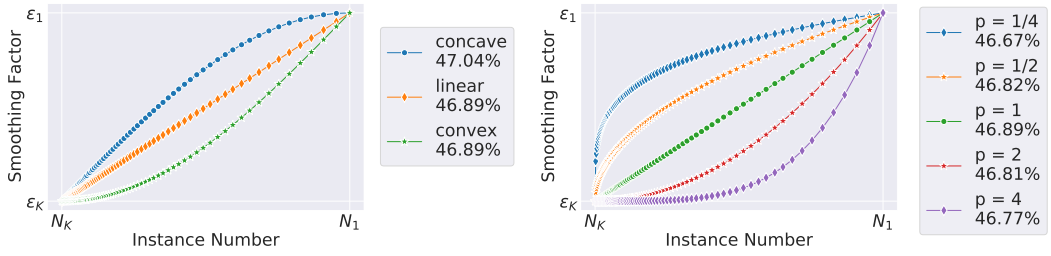


Figure 11: Function visualization and accuracy of Eqn. (3) (left) and Eqn. (7) (right).