Bag of Tricks for Long-Tailed Visual Recognition with Deep Convolutional Neural Networks

Yongshun Zhang[1], **Xiu-Shen Wei**[2,3*], **Boyan Zhou**[4], **Jianxin Wu**[1]

[1]State Key Laboratory for Novel Software Technology, Nanjing University
[2]PCA Lab, Key Lab of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, School of Computer Science and Engineering, Nanjing University of Science and Technology
[3]Jiangsu Key Lab of Image and Video Understanding for Social Security
[4]Megvii Research Nanjing
zhangys@lamda.nju.edu.cn, {weixs.gm, zhouboyan94, wujx2001}@gmail.com

Abstract

In recent years, visual recognition on challenging long-tailed distributions, where classes often exhibit extremely imbalanced frequencies, has made great progress mostly based on various complex paradigms (*e.g.*, meta learning). Apart from these complex methods, simple refinements on training procedure also make contributions. These refinements, also called *tricks*, are minor but effective, such as adjustments in the data distribution or loss functions. However, different tricks might conflict with each other. If users apply these long-tail related tricks inappropriately, it could cause worse recognition accuracy than expected. Unfortunately, there has not been a scientific guideline of these tricks in the literature. In this paper, we first collect existing tricks in long-tailed visual recognition and then perform extensive and systematic experiments, in order to give a detailed experimental guideline and obtain an effective combination of these tricks. Furthermore, we also propose a novel data augmentation approach based on class activation maps for long-tailed recognition, which can be friendly combined with re-sampling methods and shows excellent results. By assembling these tricks scientifically, we can outperform state-of-the-art methods on four long-tailed benchmark datasets, including ImageNet-LT and iNaturalist 2018. Our code is open-source and available at https://github.com/zhangyongshun/BagofTricks-LT.

Introduction

Computer vision has achieved great progress with the development of convolutional neural networks (CNNs) trained on balanced distributed datasets (Deng et al. 2009; Krizhevsky and Hinton 2009; Wah et al. 2011). But in real-world scenarios, large scale datasets (Zhou et al. 2017; Van Horn et al. 2018; Lin et al. 2014; Wang et al. 2020) naturally exhibit the imbalanced and long-tailed distributions, where a few categories (majority categories) occupy most of the data while most categories (minority categories) are under-represented. CNNs trained on these long-tailed datasets deliver poor recognition accuracy, especially for under-represented minority categories. Dealing with such long-tailed distributions is indispensable in real-world applications, such as object detection (Lin et al. 2017; Ouyang et al. 2016; Wang, Wang, and Wang 2020), instance segmentation (Wang et al. 2019; Gupta, Dollar, and Girshick 2019), visual recognition (Zhang et al. 2017; Zhong et al. 2019; Cao et al. 2019; Cui et al. 2019), etc. In this paper, we focus on the fundamental long-tailed visual recognition problem.

Recently, long-tailed visual recognition has attracted increasing attentions. Various methods belonging to different paradigms, *e.g.*, metric learning (Wang et al. 2018; Liu et al. 2019a; Cao et al. 2019), meta learning (Liu et al. 2019b; Peng et al. 2019; Jamal et al. 2020) and knowledge transfer (Wang, Ramanan, and Hebert 2017), have been successfully explored for long-tailed recognition. Although these methods bring a steady trend of accuracy improvements on long-tailed datasets, they often suffer from high sensitivity to hyper-parameters (Cao et al. 2019; Yan et al. 2019) or high complexity in the training procedures (Wang, Ramanan, and Hebert 2017; Liu et al. 2019b; Xiang, Ding, and Han 2020). Besides, it causes difficulties to efficiently apply these methods in various real-world scenarios. Apart from these methods, existing training tricks in long-tailed visual recognition also play a major role, which just make simple refinements to the vanilla training procedure, such as adjustments in loss functions or data sampling strategies. These tricks are simple but make big differences in recognition accuracy. However, different tricks might hurt each other during training when they were employed inappropriately. For instance, re-sampling (Buda, Maki, and Mazurowski 2018; Japkowicz and Stephen 2002) and re-weighting (Mikolov et al. 2013; Cui et al. 2019) are two commonly used tricks to alleviate the imbalance of long-tailed distributions. Re-sampling tries to get balanced datasets, and re-weighting assigns weights to categories determined by inversion of class frequencies.

*Corresponding author: X.-S. Wei (Nanjing University of Science and Technology). This research was supported by National Natural Science Foundation of China No. 61772256, 61921006 and the Fundamental Research Funds for the Central Universities, No. 30920041111.

Table 1: Top-1 error rates on long-tailed benchmark datasets. Our bag of tricks obtains significant accuracy gains compared with state-of-the-art methods. (Best results are marked in bold.)

Datasets	CIFAR-10-LT		CIFAR-100-LT		iNaturalist 2018	ImageNet-LT
	Imbalance factor					
	100	50	100	50		
Backbones	ResNet-32		ResNet-32		ResNet-50	ResNet-10
Baseline (Vanilla ResNet)	30.18	24.78	61.73	57.90	39.89	65.99
Focal loss (Lin et al. 2017)	29.62	24.75	61.90	57.56	39.70	67.36
CB Focal (Cui et al. 2019)	25.43	20.73	60.40	53.79	38.88	–
Feature space augmentation (Chu et al. 2020)	–	–	–	–	34.09	64.80
Meta-learning (Jamal et al. 2020)†	20.00	17.77	55.92	50.84	32.45	70.10
LDAM with DRW (Cao et al. 2019)	22.97	20.70	57.96	54.92	32.00	63.97
Decoupling learning (Kang et al. 2020)	–	–	–	–	30.70	58.20
Multi-experts (Xiang, Ding, and Han 2020)†‡	–	–	57.70	–	–	61.20
BBN (Zhou et al. 2020)	20.18	17.82	57.44	52.98	30.38	–
Baseline + tricks (Ours)	**19.97**	**16.41**	**52.17**	**48.31**	**29.13**	**56.87**

† : Results on CIFAR-10-LT and CIFAR-100-LT are obtained by incorporating LDAM (Cao et al. 2019).

‡ : Results on ImageNet-LT are obtained by incorporating OLTR (Liu et al. 2019b).

Since both re-sampling and re-weighting try to enlarge the influence of minority categories, applying re-weighting and re-sampling simultaneously will obtain similar or even worse accuracy than using them alone.

Similar to re-weighting and re-sampling, when we apply two or more long-tail related tricks, it would be great to know which of them can be combined synergistically and also which of them might conflict with others. Yet, no guideline is available in the literature. Although there are several good surveys about class imbalance learning (More 2016; Buda, Maki, and Mazurowski 2018; Japkowicz and Stephen 2002), they could be further comprised of effective tricks in the deep learning era. More importantly, they lack the comprehensive empirical studies of combining and evaluating a set of long-tail related tricks quantatively.

In this paper, we focus on exploring commonly used, easily equipped, and hyper-parameters insensitive tricks in long-tailed visual recognition. Also, we conduct extensive experiments to provide valuable practical guidelines for future researches. These long-tail related tricks are separated into four families, *i.e.*, re-weighting, re-sampling, mixup training, and two-stage training. Particularly, we add mixup training (Zhang et al. 2018; Verma et al. 2019) into long-tail related tricks because we find that mixup training delivers good results in long-tailed visual recognition, especially when combined with re-sampling. In each trick family, we introduce commonly used tricks and compare the results on long-tailed benchmark datasets. Furthermore, to overcome the lack of discriminative information in existing re-sampling methods, we propose a novel data augmentation approach based on class activation maps (CAM) (Zhou et al. 2016), which is tailored for two-stage training and generates discriminative images by transferring foregrounds while keeping backgrounds unchanged. It can be friendly combined with existing re-sampling methods and exhibits excellent results, which is termed as "CAM-based sampling". Also, we explore the conflicts between tricks of different families to find the optimal combination of tricks, named *bag of tricks*. Top-1 error rates on long-tailed CIFAR and two large scale datasets (*e.g.*, ImageNet-LT and iNaturalist 2018) are shown in Table 1, which shows significant accuracy gains of our bag of tricks compared with state-of-the-art methods.

The major contributions of our work can be summarized:

- We comprehensively explore existing simple, hyper-parameters insensitive, long-tail related tricks and provide a valuable practical guideline for future researches.
- We propose a novel CAM-based sampling approach tailored for two-stage training, which is simple but effective for long-tailed visual recognition.
- We conduct extensive experiments and find the optimal combination of tricks. Our bag of tricks achieves outperforming recognition results compared with state-of-the-art methods on four long-tailed benchmark datasets without introducing extra FLOPs.

Datasets and baseline settings

In this section, we describe the long-tailed datasets used in experiments as well as baseline training settings, *e.g.*, backbone network, data augmentation, etc. For fair comparisons, we keep our experimental settings consistent with previous works (Cao et al. 2019; Cui et al. 2019; Zhou et al. 2020).

Datasets

Long-tailed CIFAR The long-tailed versions of CIFAR-10 and CIFAR-100 datasets (CIFAR-10-LT and CIFAR-100-LT) (Cui et al. 2019) are benchmark datasets for long-tailed recognition. As the original CIFAR datasets (Krizhevsky and Hinton 2009), the long-tailed versions contain the same categories. However, they are created by reducing the number of training samples per class according to an exponential function $n = n_t \times \mu^t$, where t is the class index (0-indexed) and n_t is the original number of training images with $\mu \in (0, 1)$. The test set remains unchanged. The imbalance factor of a long-tailed CIFAR dataset is defined as the number of training samples in the largest class divided by that of the smallest, which ranges from 10 to 200. In the literature, the

难受啊，这不是我的想法么？

Table 2: Top-1 error rates of reference implementations and our baseline.

Datasets	CIFAR-10-LT		CIFAR-100-LT		iNaturalist 2018	ImageNet-LT
	Imbalance factor					
	100	50	100	50		
Backbones	ResNet-32		ResNet-32		ResNet-50	ResNet-10
Baseline (Vanilla ResNet)	30.18	24.78	61.73	57.90	39.89	65.99
Reference (Cui et al. 2019; Liu et al. 2019b)	29.64	25.19	61.68	56.15	42.86	64.40

imbalance factor of 50 and 100 are widely used, with around 12,000 training images under each imbalance factor.

iNaturalist 2018 The iNaturalist species classification datasets (Van Horn et al. 2018) are large-scale real-world datasets that suffer from extremely imbalanced label distributions. The most challenging dataset of iNaturalist is the 2018 version, which contains 437,513 images from 8,142 categories. Besides the extreme imbalance, the iNaturalist datasets also face the fine-grained problem (Wei, Wu, and Cui 2019). We follow the official training and validation splits of iNaturalist 2018 in our experiments.

Long-tailed ImageNet The long-tailed ImageNet (ImageNet-LT) is derived from the original ImageNet-2012 (Deng et al. 2009) by sampling a subset following the Pareto distribution from 1,000 categories, with maximally 1,280 images per class and minimally 5 images per class. The test set is balanced by following (Liu et al. 2019b).

Baseline settings

Backbones We adopt deep residual networks (He et al. 2016) as backbones. Specifically, we follow (Cui et al. 2019) to use the residual network with 32 layers (ResNet-32) and the residual network with 50 layers (ResNet-50) for long-tailed CIFAR and iNaturalist datasets, respectively. For ImageNet-LT, according to (Liu et al. 2019b), we adopt ResNets with 10 layers (ResNet-10) for fair comparisons.

Data augmentation For long-tailed CIFAR datasets, we follow the data augmentation in (He et al. 2016). During training, each image is padded with 4 pixels on each side and randomly cropped out a 32-by-32 region. The cropped regions are flipped horizontally with 0.5 probability and then normalized by the per-color mean and standard deviation before training. During validation, for each image, we resize its shorter edge to 36 pixels while keeping its aspect ratio. Subsequently, we crop out the 32-by-32 region in the center and normalize RGB channels similar to the training procedure.

For iNaturalist 2018 and ImageNet-LT, we follow the data augmentation in (Goyal et al. 2017). During training, we use scale and aspect ratio data augmentation (Szegedy et al. 2015) and get a 224-by-224 randomly cropped region from an augmented image or its horizontal flip. The regions are normalized by the per-color mean and standard deviation before training. During validation, for each image, we resize its shorter edge to 256 pixels while keeping its aspect ratio and then the 224-by-224 region is cropped out in the center and normalized before validation.

Training details All backbones are trained from scratch. We adopt the initialization method in (He et al. 2015). The biases are initialized to 0 for all layers. We train ResNet-32 on long-tailed CIFAR datasets by stochastic gradient descent (SGD) with momentum of 0.9 and weight decay of 2×10^{-4}. The number of training epochs is 200 and the batch size is 128. Learning rate is initialized to 0.1 and divided by 100 at the 160^{th} and 180^{th} epoch, respectively. We use warm-up (Goyal et al. 2017) for the first five epochs.

For iNaturalist 2018 and ImageNet-LT, we follow the same training strategy with Goyal et al. (2017). Backbones are trained with batch size of 512. The number of training epochs is 90, and the learning rate is initialized to 0.2 and divided by 10 at the 30^{th}, 60^{th} and 80^{th} epoch without warm-up. SGD is adopted with momentum of 0.9 and weight decay of 1×10^{-4}.

Top-1 error rates of baseline training are shown in Table 2, and our results are mostly consistent with references (Cui et al. 2019; Liu et al. 2019b). For slightly inconsistent ones, such as iNaturalist 2018 and ImageNet-LT, they might be caused by running environment (*e.g.*, the version of CUDA and deep learning frameworks), because we keep training and validation settings consistent with references.

Trick gallery

We divide the long-tail related tricks into four families: re-weighting, re-sampling, mixup training, and two-stage training. We take mixup training as a long-tail related trick, because we find that mixup training (Zhang et al. 2018; Verma et al. 2019) delivers good recognition accuracy in long-tailed visual recognition, especially when combined with re-sampling. In each trick family, we introduce commonly used tricks and compare their accuracy.

In addition, we propose a simple yet effective data augmentation approach tailored for two-stage training. The proposed approach is based on the class activation maps (CAM) (Zhou et al. 2016), which can be friendly combined with re-sampling and termed as "CAM-based sampling".

Re-weighting methods

Cost-sensitive re-weighting methods are commonly adopted methods in the long-tailed literature. These methods guide the network to pay more attention on minority categories by assigning different weights to different classes.

Formally, for each image with label $c \in \{1, 2, \ldots, C\}$, we set the predicted outputs as $\mathbf{z} = [z_1, z_2, \ldots, z_C]^\top$, where C

is the total number of classes. We define n_c as the number of training images in class c and n_{min} as the number of training images in the smallest class. Softmax cross-entropy loss (CE) is used as the baseline, which is defined as

$$\mathcal{L}_{\text{CE}}(\mathbf{z}, c) = -\log\left(\frac{\exp(z_c)}{\sum_{i=1}^{C}\exp(z_i)}\right). \quad (1)$$

Existing re-weighting methods We review commonly used re-weighting methods, including cost-sensitive softmax cross-entropy loss (Japkowicz and Stephen 2002), focal loss (Lin et al. 2017), and the recently proposed class-balanced loss (Cui et al. 2019).

• Cost-sensitive softmax cross-entropy loss (CS_CE) (Japkowicz and Stephen 2002) is defined as

$$\mathcal{L}_{\text{CS\_CE}}(\mathbf{z}, c) = -\frac{n_{min}}{n_c}\log\left(\frac{\exp(z_c)}{\sum_{i=1}^{C}\exp(z_i)}\right). \quad (2)$$

• Focal loss (Lin et al. 2017) adds an adjusting factor to the sigmoid cross-entropy loss to focus training on difficult samples. We denote $p_i = \text{sigmoid}(z_i) = \frac{1}{1+\exp(-z_i)}$ and define p_i^t as

$$p_i^t = \begin{cases} p_i, & i = c \\ 1 - p_i, & i \neq c \end{cases}, \quad (3)$$

and then the focal loss can be written as

$$\mathcal{L}_{\text{Focal}}(\mathbf{z}, c) = -\sum_{i=1}^{C}\left(1-p_i^t\right)^{\gamma}\log\left(p_i^t\right), \quad (4)$$

where γ is a hyper-parameter to control the importances of different samples.

• Class-balanced loss (Cui et al. 2019) considers the real volumes of different classes, named effective numbers, rather than the nominal numbers of images provided by datasets. With the theory of effective numbers, the class-balanced focal loss (CB_Focal) and class-balanced softmax cross-entropy loss (CB_CE) are defined as

$$\mathcal{L}_{\text{CB\_Focal}}(\mathbf{z}, c) = -\frac{1-\beta}{1-\beta^{n_c}}\sum_{i=1}^{C}\left(1-p_i^t\right)^{\gamma}\log\left(p_i^t\right), \quad (5)$$

$$\mathcal{L}_{\text{CB\_CE}}(\mathbf{z}, c) = -\frac{1-\beta}{1-\beta^{n_c}}\log\left(\frac{\exp(z_c)}{\sum_{i=1}^{C}\exp(z_i)}\right), \quad (6)$$

where γ and β are two hyper-parameters. We set γ and β on different long-tailed datasets according to (Cui et al. 2019).

Experimental results We evaluate re-weighting methods on long-tailed CIFAR datasets. As shown in Table 3, we discover that re-weighting delivers lower error rates on CIFAR-10-LT, but obtains worse results on CIFAR-100-LT compared with vanilla ResNet-32. This indicates that applying re-weighting directly in the training procedure is not a proper choice, especially when the number of categories increases and data becomes more imbalanced.

In the later section of "Two-stage training procedures", we will describe the two-stage training strategy for long-tailed visual recognition, which demonstrates an effective strategy to apply re-weighting.

Table 3: Top-1 error rates of re-weighting methods. It shows directly applying re-weighting is inappropriate, especially when the number of classes increases.

Datasets	CIFAR-10-LT		CIFAR-100-LT	
Imbalance factor	100	50	100	50
CE	30.18	24.78	**61.73**	57.90
CB_CE	28.26	22.76	66.40	63.48
CS_CE	29.07	23.74	70.92	63.78
Focal loss	29.62	24.75	61.90	57.56
CB_Focal	**27.02**	**22.03**	62.36	**57.24**

Table 4: Top-1 error rates of re-sampling methods. It demonstrates directly applying re-sampling methods brings slight improvements.

Datasets	CIFAR-10-LT		CIFAR-100-LT	
Imbalance factor	100	50	100	50
Baseline (Vanilla ResNet)	30.18	24.78	61.73	57.90
Random under-sampling	34.14	26.91	67.23	60.98
Random over-sampling	33.24	26.53	67.00	61.11
Class-balanced sampling	**30.44**	**23.97**	67.34	61.48
Square-root sampling	31.36	24.84	64.47	59.82
Progressively-balanced sampling	32.91	25.03	**61.41**	**57.09**

Re-sampling methods

Re-sampling is popular used for dealing with long-tailed problems, which attempts to sample the data to get an evenly-distributed dataset.

Existing re-sampling methods We review existing simple and commonly used re-sampling methods as follows.

• Random over-sampling (Buda, Maki, and Mazurowski 2018) is one of the representative re-sampling methods, which replicates randomly sampled training images from minority classes. Random over-sampling is effective, but might lead to overfitting (Sarafianos, Xu, and Kakadiaris 2018).

• Random under-sampling (More 2016) randomly removes training images of majority classes until all classes become balanced. Drummond and Holte (2003) show that under-sampling can be preferable to over-sampling in some situations.

• Class-balanced sampling (Kang et al. 2020) makes each class to have an equal probability of being selected. The probability p_j^{CB} of each class j is given by the following Eq. (7) with $q = 0$. Specifically, class-balance sampling firstly samples a class uniformly and then an instance from the chosen class is uniformly sampled:

$$p_j = \frac{n_j^q}{\sum_{i=1}^{C} n_i^q}, \quad (7)$$

where j is the current class, and n_i is the number of samples in class i with $q \in [0, 1]$. C is the number of total classes.

• Square-root sampling (Kang et al. 2020) sets q to $\frac{1}{2}$ in Eq. (7), which aims to return a lighter imbalanced dataset.

• Progressively-balanced sampling (Kang et al. 2020) progressively changes the sampling probabilities of classes from vanilla imbalanced sampling to class-balanced sampling. The

corresponding sampling probability p_j of class j can be calculated by Eq. (8) for the current epoch t:

$$p_j^{PB} = (1 - \frac{t}{T})\frac{n_j}{\Sigma_{i=1}^{C} n_i} + \frac{t}{T}\frac{1}{C}\,, \tag{8}$$

where T is the total epochs.

Furthermore, there are also other sampling methods that create artificial samples or sample based on gradients and features (Yan et al. 2019; Chawla et al. 2002; Shen, Lin, and Huang 2018; Han, Wang, and Mao 2005; Perez-Ortiz et al. 2019; Yu and Lam 2019). However, these methods are usually complicated and likely to introduce noisy data (Yu and Lam 2019). Therefore, we have not considered these methods in this paper which targets on simple tricks.

Experimental results Table 4 shows the error rates of different re-sampling methods on long-tailed CIFAR datasets. It can be observed that directly applying re-sampling to the training procedure gets slight improvements.

Also, we will show in the section of "Two-stage training procedures" that combining re-sampling methods with two-stage training obtains significant improvements.

Mixup training

Mixup training can be viewed as a data augmentation trick, which aims to regularize CNNs. We find mixup training delivers good accuracy in long-tailed visual recognition, especially when combined with re-sampling.

Existing mixup methods We introduce two mixup methods in this section: input mixup (Zhang et al. 2018) and manifold mixup (Verma et al. 2019).

• Input mixup has been proved effective to alleviate adversarial perturbations in CNNs (He et al. 2019a; Zhang et al. 2019). In details, each new example is formed with two randomly sampled examples (x_i, y_i) and (x_j, y_j), by a weighted linear interpolation as follows

$$\widehat{x} = \lambda x_i + (1 - \lambda)x_j\,, \tag{9}$$

$$\widehat{y} = \lambda y_i + (1 - \lambda)y_j\,, \tag{10}$$

where λ is randomly sampled from a Beta distribution. We only use $(\widehat{x}, \widehat{y})$ when training with input mixup.

• Manifold mixup encourages neural networks to predict less confidently on interpolations of hidden representations, which leverages semantic interpolations as additional training signals. The mixed example is produced by

$$\widehat{g_k} = \lambda g_k(x_i) + (1 - \lambda)g_k(x_j)\,, \tag{11}$$

$$\widehat{y} = \lambda y_i + (1 - \lambda)y_j\,, \tag{12}$$

where $(g_k(x_i), y_i)$ and $(g_k(x_j), y_j)$ are intermediate outputs of two randomly sampled examples (x_i, y_i) and (x_j, y_j) after layer k, and λ is the mixing coefficient sampled from a Beta distribution. We apply manifold mixup on only one layer in our experiments.

Fine-tuning after mixup training He et al. (2019b) show that the results of models trained by mixup can be further improved if we remove the mixup in last several epochs. In our experiments, we use the mixup training firstly, and then fine-tune the models trained by mixup for several epochs in order to obtain further improvements, which is named "fine-tuning after mixup training".

Table 5: Top-1 error rates of mixup methods. α is the hyper-parameter of the Beta distribution. "FC" represents "fully-connected". We can see that input mixup and manifold mixup are comparable. But in the section of "Removing controversial tricks in each trick family", we will show that input mixup delivers better results.

Datasets	CIFAR-10-LT		CIFAR-100-LT	
Imbalance factor	100	50	100	50
Baseline (Vanilla ResNet)	30.18	24.78	61.73	57.90
Input mixup ($\alpha = 2$)	28.65	24.90	60.38	56.39
Input mixup ($\alpha = 1$)	26.99	22.91	**59.66**	**55.75**
Manifold mixup on layer3 ($\alpha = 2$)	27.14	22.31	60.73	56.52
Manifold mixup on layer3 ($\alpha = 1$)	27.30	22.41	60.81	56.68
Manifold mixup on FC layer ($\alpha = 2$)	27.79	21.87	60.21	57.09
Manifold mixup on FC layer ($\alpha = 1$)	26.64	22.55	60.20	56.72
Manifold mixup on pooling layer ($\alpha = 2$)	27.67	22.02	60.45	56.45
Manifold mixup on pooling layer ($\alpha = 1$)	**26.61**	**21.50**	60.14	56.44

Table 6: Top-1 error rates of fine-tuning after mixup training. Fine-tuning the models trained with input mixup obtains further improvements. "ft." represents "fine-tuning"

Datasets	CIFAR-10-LT		CIFAR-100-LT	
Imbalance factor	100	50	100	50
Baseline (Vanilla ResNet)	30.18	24.78	61.73	57.90
Input mixup ($\alpha = 1$)	26.99	22.91	59.66	55.75
Input mixup ($\alpha = 1$) + ft. after mixup training	**26.27**	**20.32**	**58.21**	**53.97**
Manifold mixup on pooling layer ($\alpha = 1$)	26.61	21.50	60.14	56.44
Manifold mixup on pooling layer ($\alpha = 1$) + ft. after mixup training	28.88	22.59	61.16	57.43

Experimental results Experiments of mixup methods are shown in Table 5. Especially, we do not try all possible values of hyper-parameter α for the Beta distribution, which is not the main purpose of our work. We can discover from Table 5 that 1) both input mixup and manifold mixup deliver better results over baseline, and 2) when α is 1 and mixing up location is set to the pooling layer, input mixup and manifold mixup achieve comparable results, which need to conduct more experiments with other tricks.

The results of fine-tuning after mixup training are shown in Table 6. We can find that fine-tuning after input mixup training can obtain further improvements, but fine-tuning after manifold mixup gets worse results.

Two-stage training procedures

Two-stage training consists of imbalanced training and balanced fine-tuning. In this section, we focus on exploring different methods of balanced fine-tuning. We firstly describe existing fine-tuning methods and then present our CAM-based sampling approach.

Balanced fine-tuning after imbalanced training CNNs trained on imbalanced datasets without any re-weighting or re-sampling method learn good feature representations but suffer poor recognition accuracy on under-represented tail categories. Cui et al. (2018) fine-tune these networks

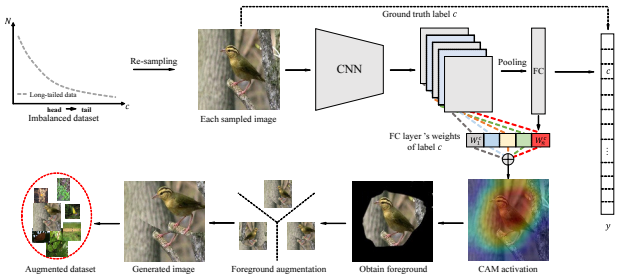

Figure 1: Overview of our proposed CAM-based sampling. For each image sampled by re-sampling, CAM is firstly generated based on feature maps and FC weights of ground truth label c. We separate the foreground and background based on the average of its CAM values, and subsequently we transform foreground while keeping background unchanged to get the generated informative sampled dataset.

on balanced subsets to make the learned features from imbalanced datasets be transferred and re-balanced among all categories. These fine-tuning methods (Cao et al. 2019) can be divided into two sections: deferred re-balancing by re-sampling (DRS) and by re-weighting (DRW).

• DRS uses the vanilla training schedule firstly, and then applies re-sampling for balanced fine-tuning. In order to get a balanced subset for fine-tuning, re-sampling methods introduced in the section of "Re-sampling methods" will be applied. Furthermore, we propose a sample yet effective generative sampling method termed "CAM-based sampling".

• DRW uses the vanilla training schedule in the first stage, and then applies re-weighting methods in the second stage. Re-weighting methods introduced in the section of "Re-weighting methods" will be applied in the second stage.

The proposed CAM-based sampling for DRS Existing re-sampling methods used in DRS only replicate or remove randomly selected samples from the original dataset to generate balanced subsets, which deliver limited improvements during balanced fine-tuning. In order to generate discriminative information, inspired by class activation maps (CAM) (Zhou et al. 2016), we propose CAM-based sampling, which shows a significant accuracy improvement over existing methods with a marginal extra cost.

As illustrated in Figure 1, we firstly apply re-sampling to get balanced sampled images. For each sampled image, we use the parameterized model trained in the first training stage to generate CAM based on its ground truth label and corresponding fully-connected layer's weights. The foreground and background are separated based on the average value of its CAM, where the foreground contains pixels larger than the average and the background contains the rest (Wei et al. 2017). Finally, we apply transformations to the foreground while keeping the background unchanged. The transformation includes horizontal flipping, translation, rotating and scaling, and we randomly choose only one transformation for each image.

In concretely, we combine CAM with random over-sampling, random under-sampling, class-balanced sampling, square-root sampling, and progressively-balanced sampling, which are named "CAM-based over-sampling", "CAM-based under-sampling", "CAM-based balance-sampling", "CAM-based square-sampling", and "CAM-based progressive-sampling", respectively.

Table 7: Top-1 error rates of different re-sampling methods used in DRS. The proposed CAM-based sampling delivers better results. In particular, CAM-based balance-sampling obtains the best results. Compared with using re-sampling directly in Table 4, DRS provides an effective way to apply re-sampling methods.

First stage	Second stage of DRS	CIFAR-10-LT		CIFAR-100-LT	
		Imbalance factor			
		100	50	100	50
CE	Baseline without re-sampling	30.18	24.78	61.73	57.90
	Random under-sampling	28.18	21.33	60.21	55.97
	Random over-sampling	28.88	21.52	59.76	55.90
	Class-balanced sampling	29.04	21.34	59.56	55.67
	Square-root sampling	31.31	22.21	61.02	57.05
	Progressively-balanced sampling	33.48	24.58	61.35	56.93
	CAM-based under-sampling	24.98	19.15	58.99	54.17
	CAM-based over-sampling	24.87	18.82	58.45	54.36
	CAM-based balance-sampling	**24.63**	**18.60**	**58.27**	**54.05**
	CAM-based square-sampling	28.14	20.69	60.07	55.61
	CAM-based progressive-sampling	27.39	19.46	59.67	55.28
	ImageTrans balance-sampling	28.10	21.60	59.28	55.05

Table 8: Top-1 error rates of different re-weighting methods used in DRW. CS_CE obtains the best results in DRW training schedule. Compared with using re-weighting directly in Table 3, applying re-weighting in second training stage is more effective.

First stage	Second stage of DRW	CIFAR-10-LT		CIFAR-100-LT	
		Imbalance factor			
		100	50	100	50
CE	CE	30.18	24.78	61.73	57.90
	Focal loss	29.71	23.77	61.74	57.32
	CB_Focal	25.62	21.25	61.99	55.54
	CS_CE	**25.31**	**20.81**	**58.92**	**54.57**

Experimental results The results of re-sampling methods in DRS are shown in Table 7. We add a sampling method named image transferring balance-sampling (ImageTrans balance-sampling) to prove the effectiveness of our CAM-based balance-sampling. Its pipeline is the same as CAM-based balance-sampling, but without using CAM to separate the foreground and background.

From the results in Table 7, we have the following observations: 1) Compared with applying re-sampling directly in Table 4, applying re-sampling in DRS delivers better results. 2) Our proposed CAM-based sampling obtains substantially large gains. 3) In CAM-based sampling, CAM-based balance-sampling delivers the best results. 4) The results of ImageTrans balance-sampling prove the effectiveness of CAM used in our CAM-based balance-sampling.

Table 8 shows the results of different re-weighting methods in DRW. From the results, we observe that: 1) compared

Table 9: Top-1 error rates of different strategies to apply DRW and DRS. Apply DRS (CAM-based balance-sampling) only shows the best result. "CAM-BS" represents "CAM-based balance-sampling".

First stage	Second stage	CIFAR-10-LT		CIFAR-100-LT	
		Imbalance factor			
		100	50	100	50
CE	CE	30.18	24.78	61.73	57.90
	CS_CE	25.31	20.81	58.92	54.57
	CAM-BS	**24.63**	**18.60**	**58.27**	**54.05**
	CS_CE + CAM-BS	24.82	18.96	58.36	54.09

Table 10: Top-1 error rates of combining mixup methods with other best tricks. We can easily find that input mixup obtains larger gains over manifold mixup. "CAM-BS" represents "CAM-based balance-sampling". In mixup, α is 1 and mainifold mixup's location is set to pooling layer.

Training scheduler	Mixup training	CIFAR-10-LT		CIFAR-100-LT	
		Imbalance factor			
		100	50	100	50
DRS with CAM-BS	Manifold mixup	22.65	19.17	57.20	56.94
	Input mixup	**21.88**	**17.94**	**53.94**	**50.04**

with apply re-weighting directly in Table 3, combining re-weighting with DRW delivers better results, and 2) DRW with CS_CE obtains the best results.

Trick combinations

In this section, we first review the conflictual tricks in each trick family, which obtain comparable results. We combine these conflictual tricks with other best tricks across trick families, in order to find the best trick combination. Furthermore, we apply the best trick combination incrementally to show the negligible conflicts between these tricks.

Removing conflictual tricks in each trick family

Experiments in the section of "Two stage training procedures" have shown the best training schedule of two-stage training is DRS with CAM-based over-sampling and DRW with CS_CE, but DRS and DRW are both two-stage training tricks, we need more experiments to explore the best strategy to apply them. Moreover, in mixup training, input mixup and manifold mixup achieve comparable results, as shown in Table 5. Thus, we conduct more experiments to compare their results when they are combined with other tricks.

Results in Table 9 show that the best strategy of applying two-stage training is DRS with CAM-based balance-sampling. We can also find that combining CS_CE and CAM-based balance-sampling together cannot further improve the accuracy, since both of them try to enlarge the influence of tail classes and the joint use of the two could cause an accuracy drop due to the overfitting problem. Furthermore, from Table 10, we observe that input mixup obtains substantially larger gains over manifold mixup when combined with other best tricks.

Table 11: Reductions of top-1 error rates with incremental tricks. Our bag of tricks shows a steady trend of accuracy improvement, which proves the effectiveness of our tricks on both small and large scale real-world datasets. "iNat 18" represents "iNaturalist 2018" and "IM" represents "input mixup". α is 1 in input mixup.

Datasets	CIFAR-10		CIFAR-100		iNat 18	ImageNet-LT
	Imbalance factor					
	100	50	100	50		
Baseline (Vanilla ResNet)	30.18	24.78	61.73	57.90	39.89	65.99
+ IM & DRS with CAM-BS	21.88	17.94	53.94	50.04	29.72	58.13
+ ft. after mixup training	**19.97**	**16.41**	**52.17**	**48.31**	**29.13**	**56.87**

From experiments in each trick family and trick combinations, we find the optimal trick combination is input mixup, DRS with CAM-based balance-sampling, and fine-tuning after mixup training, which we name as *bag of tricks*.

Applying the best tricks incrementally

In order to demonstrate the performances and negligible conflicts of our bag of tricks, we apply these tricks incrementally on long-tailed datasets, including large scale real-world datasets iNaturalist 2018 and ImageNet-LT. By considering that we use CAM-based balance-sampling in DRS with input mixup, in fine-tuning after mixup training, we also adopt class-balanced sampling to maintain the learned features.

The results are shown in Table 11. From the results, we have the following observations: 1) By stacking input mixup, DRS with CAM-based balance-sampling, fine-tuning after mixup training, the results are steadily improved. 2) The results on iNaturalist 2018 and ImageNet-LT demonstrate the effectiveness of our bag of tricks on real-world large scale datasets clearly. 3) With all of our tricks, we reduce about 10% error rates on all long-tailed datasets, which demonstrates significant improvements compared with existing state-of-the-art methods.

Conclusions

In this paper, we systematically explored existing simple yet effective long-tail related tricks and provided a scientific experimental guideline for long-tailed visual recognition. Furthermore, we found that existing simple sampling methods are lack of discriminative information. Motivated by this, we proposed a novel data augmentation approach based on the class activation maps and combined it with existing re-sampling methods. By conducting extensive experiments, we obtain the optimal trick combination, which consists of input mixup, DRS with CAM-based balanced sampling, and fine-tuning after mixup training. The optimal trick combination, *i.e.*, *bag of tricks*, contained negligible conflicts and achieved the best results on long-tailed benchmarks without introducing extra FLOPs. We also release our source codes as a scientific and practical toolbox, which could benefit future researches of long-tailed visual recognition. In the future, we attempt to explore bag of tricks in other challenging long-tailed tasks, *e.g.*, detection and segmentation.

References

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

Cao, K.; Wei, C.; Gaidon, A.; Arechiga, N.; and Ma, T. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. In *NeurIPS 32*, 1565–1576.

Chawla, N. V.; Bowyer, K. W.; Hall, L. O.; and Kegelmeyer, W. P. 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* 16: 321–357.

Chu, P.; Bian, X.; Liu, S.; and Ling, H. 2020. Feature space augmentation for long-tailed data. In *ECCV*.

Cui, Y.; Jia, M.; Lin, T.-Y.; Song, Y.; and Belongie, S. 2019. Class-balanced loss based on effective number of samples. In *CVPR*, 9268–9277.

Cui, Y.; Song, Y.; Sun, C.; Howard, A.; and Belongie, S. 2018. Large scale fine-grained categorization and domain-specific transfer learning. In *CVPR*, 4109–4118.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *CVPR*, 248–255.

Drummond, C.; and Holte, R. C. 2003. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, volume 11, 1–8.

Goyal, P.; Dollár, P.; Girshick, R.; Noordhuis, P.; Wesolowski, L.; Kyrola, A.; Tulloch, A.; Jia, Y.; and He, K. 2017. Accurate, large minibatch SGD: Training ImageNet in 1 hour. *arXiv preprint arXiv:1706.02677* .

Gupta, A.; Dollar, P.; and Girshick, R. 2019. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, 5356–5364.

Han, H.; Wang, W.-Y.; and Mao, B.-H. 2005. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *International Conference on Intelligent Computing*, 878–887.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 1026–1034.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

He, T.; Zhang, Z.; Zhang, H.; Zhang, Z.; Xie, J.; and Li, M. 2019a. Bag of tricks for image classification with convolutional neural networks. In *CVPR*, 558–567.

He, Z.; Xie, L.; Chen, X.; Zhang, Y.; Wang, Y.; and Tian, Q. 2019b. Data augmentation revisited: Rethinking the distribution gap between clean and augmented data. *arXiv preprint arXiv:1909.09148* .

Jamal, M. A.; Brown, M.; Yang, M.-H.; Wang, L.; and Gong, B. 2020. Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective. In *CVPR*, 7610–7619.

Japkowicz, N.; and Stephen, S. 2002. The class imbalance problem: A systematic study. *Intelligent Data Analysis* 6(5): 429–449.

Kang, B.; Xie, S.; Rohrbach, M.; Yan, Z.; Gordo, A.; Feng, J.; and Kalantidis, Y. 2020. Decoupling representation and classifier for long-tailed recognition. In *ICLR*, 1–16.

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. Technical report, University of Toronto.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *ICCV*, 2980–2988.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft COCO: Common objects in context. In *ECCV*, volume 8693 of *LNCS*, 740–755. Springer.

Liu, B.; Deng, W.; Zhong, Y.; Wang, M.; Hu, J.; Tao, X.; and Huang, Y. 2019a. Fair Loss: Margin-aware reinforcement learning for deep face recognition. In *ICCV*, 10052–10061.

Liu, Z.; Miao, Z.; Zhan, X.; Wang, J.; Gong, B.; and Yu, S. X. 2019b. Large-scale long-tailed recognition in an open world. In *CVPR*, 2537–2546.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS 26*, 3111–3119.

More, A. 2016. Survey of resampling techniques for improving classification performance in unbalanced datasets. *arXiv preprint arXiv:1608.06048* .

Ouyang, W.; Wang, X.; Zhang, C.; and Yang, X. 2016. Factors in finetuning deep model for object detection with long-tail distribution. In *CVPR*, 864–873.

Peng, M.; Zhang, Q.; Xing, X.; Gui, T.; Huang, X.; Jiang, Y.-G.; Ding, K.; and Chen, Z. 2019. Trainable undersampling for class-imbalance learning. In *AAAI*, 4707–4714.

Perez-Ortiz, M.; Tiňo, P.; Mantiuk, R.; and Hervás-Martínez, C. 2019. Exploiting synthetically generated data with semi-supervised learning for small and imbalanced datasets. In *AAAI*, 4715–4722.

Sarafianos, N.; Xu, X.; and Kakadiaris, I. A. 2018. Deep imbalanced attribute classification using visual attention aggregation. In *ECCV*, volume 11215 of *LNCS*, 708–725. Springer.

Shen, L.; Lin, Z.; and Huang, Q. 2018. Relay backpropagation for effective learning of deep convolutional neural networks. In *ECCV*, volume 9911 of *LNCS*, 185–201. Springer.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *CVPR*, 1–9.

Van Horn, G.; Mac Aodha, O.; Song, Y.; Cui, Y.; Sun, C.; Shepard, A.; Adam, H.; Perona, P.; and Belongie, S. 2018. The iNaturalist species classification and detection dataset. In *CVPR*, 8769–8778.

Verma, V.; Lamb, A.; Beckham, C.; Najafi, A.; Mitliagkas, I.; Lopez-Paz, D.; and Bengio, Y. 2019. Manifold Mixup: Better representations by interpolating hidden states. In *ICML*, 6438–6447.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The Caltech-UCSD Birds-200-2011 dataset. *CNS-TR-2011-001* 1–8.

Wang, H.; Wang, Y.; Zhou, Z.; Ji, X.; Gong, D.; Zhou, J.; Li, Z.; and Liu, W. 2018. CosFace: Large margin cosine loss for deep face recognition. In *CVPR*, 5265–5274.

Wang, J.; Min, W.; Hou, S.; Ma, S.; Zheng, Y.; Wang, H.; and Jiang, S. 2020. Logo-2K+: A large-scale logo dataset for scalable logo classification. In *AAAI*, 6194–6201.

Wang, T.; Li, Y.; Kang, B.; Li, J.; Liew, J. H.; Tang, S.; Hoi, S.; and Feng, J. 2019. Classification calibration for long-tail instance segmentation. *arXiv preprint arXiv:1910.13081* .

Wang, W.; Wang, M.; and Wang, S. 2020. One-shot learning for long-tail visual relation detection. In *AAAI*, 12225–12232.

Wang, Y.-X.; Ramanan, D.; and Hebert, M. 2017. Learning to model the tail. In *NIPS 30*, 7029–7039.

Wei, X.-S.; Luo, J.-H.; Wu, J.; and Zhou, Z.-H. 2017. Selective convolutional descriptor aggregation for fine-grained image retrieval. *IEEE Transactions on Image Processing* 26(6): 2868–2881.

Wei, X.-S.; Wu, J.; and Cui, Q. 2019. Deep learning for fine-grained image analysis: A survey. *arXiv preprint arXiv:1907.03069* .

Xiang, L.; Ding, G.; and Han, J. 2020. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *ECCV*.

Yan, Y.; Tan, M.; Xu, Y.; Cao, J.; Ng, M.; Min, H.; and Wu, Q. 2019. Oversampling for imbalanced data via optimal transport. In *AAAI*, 5605–5612.

Yu, Q.; and Lam, W. 2019. Data augmentation based on adversarial autoencoder handling imbalance for learning to rank. In *AAAI*, 411–418.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018. mixup: Beyond empirical risk minimization. In *ICLR*, 1–13.

Zhang, X.; Fang, Z.; Wen, Y.; Li, Z.; and Qiao, Y. 2017. Range loss for deep face recognition with long-tailed training data. In *ICCV*, 5409–5418.

Zhang, Z.; He, T.; Zhang, H.; Zhang, Z.; Xie, J.; and Li, M. 2019. Bag of freebies for training object detection neural networks. *arXiv preprint arXiv:1902.04103* .

Zhong, Y.; Deng, W.; Wang, M.; Hu, J.; Peng, J.; Tao, X.; and Huang, Y. 2019. Unequal-training for deep face recognition with long-tailed noisy data. In *CVPR*, 7812–7821.

Zhou, B.; Cui, Q.; Wei, X.-S.; and Chen, Z.-M. 2020. BBN: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *CVPR*, 9719–9728.

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2016. Learning deep features for discriminative localization. In *CVPR*, 2921–2929.

Zhou, B.; Lapedriza, A.; Khosla, A.; Oliva, A.; and Torralba, A. 2017. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40(6): 1452–1464.