CUDA: CURRICULUM OF DATA AUGMENTATION FOR LONG-TAILED RECOGNITION

Anonymous authors
Paper under double-blind review

ABSTRACT

Class imbalance problems frequently occur in real-world tasks, and conventional deep learning algorithms are well known for performance degradation on imbalanced training datasets. To mitigate this problem, many approaches have aimed to balance among given classes by *re-weighting* or *re-sampling* training samples. These re-balancing methods increase the impact of minority classes and reduce the influence of majority classes on the output of models. However, the extracted representations may be of poor quality owing to the limited number of minority samples. To handle this restriction, several methods have been developed that increase the representations of minority samples by leveraging the features of the majority samples. Despite extensive recent studies, no deep analysis has been conducted on determination of classes to be augmented and strength of augmentation has been conducted. In this study, we first investigate the correlation between the degree of augmentation and class-wise performance, and find that the proper degree of augmentation must be allocated for each class to mitigate class imbalance problems. Motivated by this finding, we propose a simple and efficient novel curriculum, which is designed to find the appropriate per-class strength of data augmentation, called CUDA: **CU**rriculum of **D**ata **A**ugmentation for long-tailed recognition. CUDA can simply be integrated into existing long-tailed recognition methods. We present the results of experiments showing that CUDA effectively achieves better generalization performance compared to the state-of-the-art method on various imbalanced datasets such as CIFAR-100-LT, ImageNet-LT, and iNaturalist 2018.

1 INTRODUCTION

Deep neural networks (DNNs) have significantly improved over the past few decades on a wide range of tasks (He et al., 2017; Redmon & Farhadi, 2017; Qi et al., 2017). This effective performance is made possible by come from well-organized datasets such as MNIST (LeCun et al., 1998), CIFAR-10/100 (Krizhevsky et al., 2009), and ImageNet (Russakovsky et al., 2015). However, as Van Horn et al. (2018) indicated, gathering such balanced datasets is notoriously difficult in real-world applications. In addition, the models perform poorly when trained on an improperly organized dataset, *e.g.,* in cases with class imbalance, because minority samples can be ignored due to their small portion.

The simplest solution to the class imbalance problem is to prevent the model from ignoring minority classes. To improve generalization performance, many studies have aimed to emphasize minority classes or reduce the influence of the majority samples. Reweighting (Cao et al., 2019; Menon et al., 2021) or resampling (Buda et al., 2018; Van Hulse et al., 2007) are two representative methods that have been frequently applied to achieve this goal. (i) *Reweighting* techniques increase the weight of the training loss of the samples in the minority classes. (ii) *Resampling* techniques reconstruct a class-balanced training dataset by upsampling minority classes or downsampling majority classes.

Although these elaborate rebalancing approaches have been adopted in some applications, limited information on minority classes due to fewer samples remains problematic. To address this issue, some works have attempted to spawn minority samples by leveraging the information of the minority samples themselves. For example, Chawla et al. (2002); Ando & Huang (2017) proposed a method to generate interpolated minority samples. Recently (Kim et al., 2020; Chu et al., 2020; Park et al., 2022) suggested enriching the information of minority classes by transferring information gathered from majority classes to the minority ones. For example, Kim et al. (2020) generated a balanced

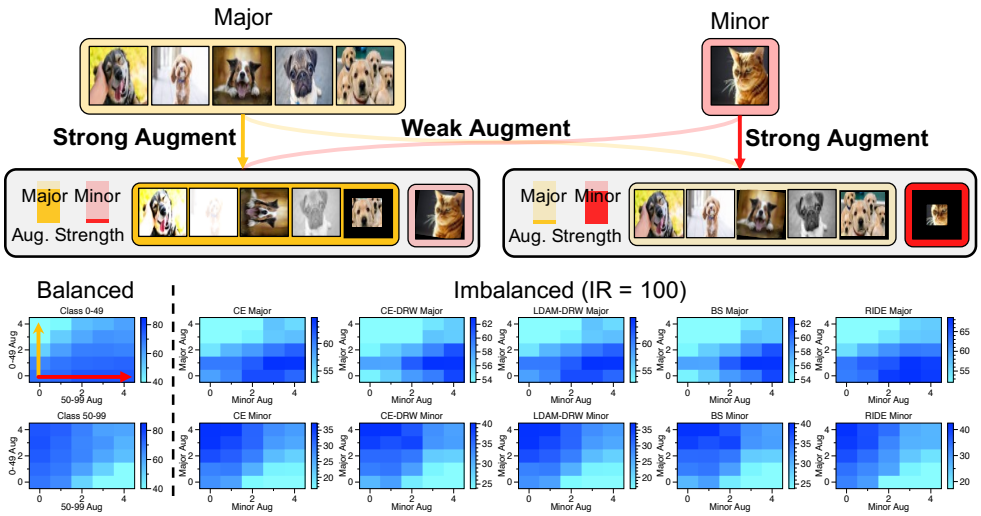


Figure 1: Motivation of CUDA. If one half of the classes (*e.g.,* class indices in 0 to 49) are strongly augmented, the performance of the other half of the classes (*i.e.,* in 50 to 99) increases. This phenomenon is also observed in the imbalanced case in which many more instances are present for the top first half (major) have more samples than the second half classes (minor). Setup and further analysis are described in Appendix A.

training dataset by creating adversarial examples from the majority class to consider them as minority classes.

Although many approaches have been proposed to utilize data augmentation methods to generate various information about minority samples, relatively few works have considered the influence of the degree of augmentation of different classes on class imbalance problems. In particular, few detailed observations have been conducted as to which classes should be augmented and how intensively.

To this end, we first consider that controlling the strength of class-wise augmentation can provide another dimension to mitigate the class imbalance problem. In this paper, we use the number of augmentation operations and their magnitude to control the extent of the augmentation, which we refer to herein as its strength, *e.g.,* a strength parameter of 2 means that two randomly sampled operations with a pre-defined magnitude index of 2 are used.

Our key finding is that class-wise augmentation improves performance in the non-augmented classes while that for the augmented classes may not be significantly improved, and in some cases, performances may even decrease. As described in Figure 1, regardless of whether a given dataset is class imbalanced, conventional class imbalance methods show similar trends: when only the major classes are strongly augmented (*e.g.,* strength 4), the performance of majority classes decreases, whereas that for the minority classes have better results. We find that diversifying features of augmented classes prohibit the growth of the norm of a linear classifier for corresponding classes, and this classifier tends to increase the logit value for the non-augmented classes. It is described in Appendix A. This result motivates us to find the proper augmentation strength for each class to improve the performance for other classes while maintaining its own performance.

Contribution. We propose a simple algorithm called **CU**rriculum of **D**ata **A**ugmentation (CUDA) to find the proper class-wise augmentation strength for long-tailed recognition. Based on our motivation, we have to increase the augmentation strength of majorities for the performance of minorities when the model successfully predicts the majorities. On the other hand, we have to lower the strength of majorities when the model makes wrong predictions about majorities. The proposed method consists of two modules, which compute a level-of-learning score for each class and leverage the score to determine the augmentation. Therefore, CUDA increases and decreases the augmentation strength of the class that was successfully and wrongly predicted by the trained model. To the best of our knowledge, this work is the first to suggest a class-wise augmentation method to find a proper augmentation strength for class imbalance problem.

We empirically examine performance of CUDA on synthetically imbalanced datasets such as CIFAR-100-LT (Cao et al., 2019), ImageNet-LT (Liu et al., 2019), and a real-world benchmark, iNaturalist 2018 (Van Horn et al., 2018). With the high compatibility of CUDA, we apply our framework to

various long-tailed recognition methods and achieve better performance compared to the existing long-tailed recognition methods. Furthermore, we conduct an extensive exploratory analysis to obtain a better understanding of CUDA. The results of these analyses verify that CUDA exhibits two effects that mitigate class imbalance, including its balanced classifier and improved feature extractor.

2 RELATED WORKS

Long-tailed Recognition (LTR). The datasets with class imbalances can lead DNNs to learn biases toward training data, and their performance may decrease significantly on the balanced test data. To improve the robustness of such models to imbalance, LTR methods have been evolving in two main directions: (1) reweighting (Cui et al., 2019; Cao et al., 2019; Park et al., 2021) methods that reweight the loss for each class by a factor inversely proportional to the number of data points, and (2) resampling methods (Kubat et al., 1997; Chawla et al., 2002; Ando & Huang, 2017) that balance the number of training samples for each class in the training set. However, studies along these lines commonly sacrifice performance on majority classes to enhance that on minority classes, because the overfitting problem occurs with limited information on minority classes as a result of increasing the weight of a small number of minority samples.

Several methods have recently been developed to alleviate the overfitting issues in various categories: (1) two-stage training (Cao et al., 2019; Kang et al., 2020; Liu et al., 2019), (2) ensemble methods (Zhou et al., 2020a; Xiang et al., 2020; Wang et al., 2021; Cai et al., 2021), and (3) contrastive learning approach (Kang et al., 2021; Cui et al., 2021; Zhu et al., 2022; Li et al., 2022a;b). To re-balance the classifier layers after achieving a good representation on the imbalanced training dataset in an early phase, Cao et al. (2019) proposed deferred resampling (DRS) and reweighting (DRW) approaches. Kang et al. (2020) decoupled the learning procedure into representation learning and training linear classifier, achieved higher performance than previous balancing methods. Wang et al. (2021) and Cai et al. (2021) suggested efficient ensemble methods using multiple experts with a routing module and a shared architecture for experts to capture various representations. Liu et al. (2022) found that self-supervised representations are more robust to class imbalance than supervised representations, and some works have developed supervised contrastive learning methods (Khosla et al., 2020) for imbalanced datasets (Cui et al., 2021; Zhu et al., 2022; Li et al., 2022b).

Another line of research has considered augmentation methods in terms of both input and feature spaces (Kim et al., 2020; Chu et al., 2020; Li et al., 2021). Recently, Park et al. (2022) mixed minority and majority images by using CutMix with different sampling strategies to enhance balancing and robustness simultaneously. These methods commonly focus on utilizing the rich context of majority samples to improve the diversity of minority samples. Moreover, these augmentation-based methods are relatively in easy to apply orthogonally with other LTR methods.

Data Augmentation (DA). DA has been studied to mitigate overfitting which may occur due to a lack of data samples. Some works have been proposed to erase random parts of images to enhance the generalization performance of neural networks (DeVries & Taylor, 2017; Zhong et al., 2020; Kumar Singh & Jae Lee, 2017; Choe & Shim, 2019). Recently, variants of MixUp (Zhang et al., 2018) have been proposed; this method combines two images with specific weights (Tokozume et al., 2018; Guo et al., 2019; Takahashi et al., 2018; DeVries & Taylor, 2017; Verma et al., 2019). By aggregating two approaches, CutMix (Yun et al., 2019) was proposed to erase and replace a small rectangular part of an image into another image. In another line of research, methods have been proposed to automatically configure augmentation operations (Cubuk et al., 2019; Lim et al., 2019; Li et al., 2020b; Hataya et al., 2020; Gudovskiy et al., 2021). In addition, Cubuk et al. (2020) randomly selected augmentation operations using the given hyperparameters of the number of sampling augmentation and their magnitudes. Recently, class-wise or per-sample auto-augmentation methods have also been proposed (Cheung & Yeung, 2021; Rommel et al., 2022).

3 CURRICULUM OF DATA AUGMENTATION FOR LONG-TAILED RECOGNITION

The core philosophy of CUDA is to *"generate an augmented sample that becomes the most difficult sample without losing its original information."* In this section, we describe design of CUDA in terms

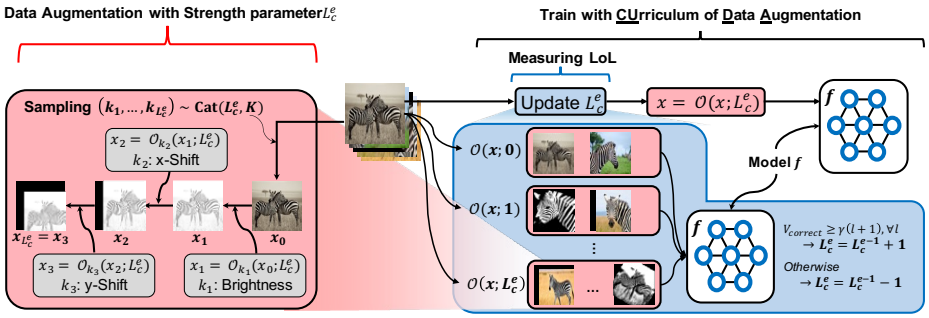


Figure 2: Algorithm overview. CUDA is composed of two main parts: (1) strength-based augmentation and (2) Level-of-Learning (LoL) score. To control the difficulty of augmented images, strength-based augmentation utilizes two values, the number of augmentations and their magnitudes. We use the strength-based augmentation module to score the LoL. Based on the measured LoL score, CUDA generates adequately augmented images for LTR algorithms.

of two parts: (1) a method to generate the augmented samples based on the given strength parameter, and (2) a method to measure a Level-of-Learning (LoL) score for each class.

3.1 Problem Formulation of Long-tailed Recognition

Suppose that the training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ is composed of images with size d, $x_i \in \mathbb{R}^d$, and their corresponding labels $y_i \in \{1, ..., C\}$. $\mathcal{D}_c \subset \mathcal{D}$ is a set of class c, *i.e.,* $\mathcal{D}_c = \{(x, y) | y = c, (x, y) \in \mathcal{D}\}$. Without loss of generality, we assume $|\mathcal{D}_1| \geq |\mathcal{D}_2| \geq \cdots \geq |\mathcal{D}_C|$, where $|\mathcal{D}|$ denotes the cardinality of the set $\mathcal{D}$. We denote the $N_{\text{max}} := |\mathcal{D}_1|$ and $N_{\text{min}} := |\mathcal{D}_C|$. LTR algorithms, $\mathcal{A}_{\text{LTR}}(f_\theta, \mathcal{D})$, mainly focus on training the model f_θ with parameter θ when the class distribution of training dataset $\mathcal{P}_{\text{train}}(y)$ and test dataset $\mathcal{P}_{\text{test}}(y)$ are not identical. More precisely, $\mathcal{P}_{\text{train}}(y)$ is highly imbalanced while $\mathcal{P}_{\text{test}}(y)$ is balanced, *i.e.,* uniform distribution.

3.2 Curriculum of Data Augmentation

In this section, we describe our proposed DA with strength parameter, and the methods used to measured the LoL score. Then, we integrate the two methods in a single framework to propose CUDA.

DA with a strength parameter. Let us assume that there exist pre-defined K augmentation operations. We utilize visual augmentation operations which is indexed as $k \in \{1, \cdots, K\}$, *e.g.,* Gaussian blur, Rotation, Horizontal flip. Each augmentation operation $\mathcal{O}_k^{m_k(s)} : \mathbb{R}^d \rightarrow \mathbb{R}^d$ has its own pre-defined augmentation magnitude function $m_k(s)$ where the strength parameter $s \in \{0, ..., S\}$. These operations are described in detail along with each magnitude functions in Appendix D.

Given an augmentation strength parameter s and an input image x, we model a sequence of augmentation operations $\mathcal{O}(x; s)$ as follows:

$$\mathcal{O}(x; s) = \mathcal{O}_{k_s}^{m_{k_s}(s)} \circ \mathcal{O}_{k_{s-1}}^{m_{k_{s-1}}(s)} \circ \cdots \circ \mathcal{O}_{k_1}^{m_{k_1}(s)}(x), \quad k_i \sim \text{Cat}(K, \mathcal{U}(K)) \quad \forall i = \{1, \dots, s\},$$

where, $\text{Cat}(\cdot)$ and $\mathcal{U}(\cdot)$ denote categorical and discrete uniform distributions, respectively. The sequential augmentation operation $\mathcal{O}(x; s)$ samples s operations from the categorical distribution when the probability of seeing the operations follows uniform distribution. As depicted on the left side Figure 2, suppose that the random sampled augmentations k_1, k_2, and k_3 are brightness, X-shift, and Y-shift, respectively. Then, $\mathcal{O}(x; 3)$ outputs an image in which bright is raised by $m_{\text{bright}}(3)$ and moved by $m_{\text{x-shift}}(3)$ on the x-axis and shifted by $m_{\text{y-shift}}(3)$ on the y-axis.

Level-of-Learning (LoL). To control the strength of augmentation properly, we check whether the model can correctly predict augmented versions without losing the original information. To enable this, we define the LoL for each class c at epoch e, *i.e.,* L_c^e, which is adaptively updated as the training continues as follows:

$$L_c^e = V_{\text{LoL}}(\mathcal{D}_c, L_c^{e-1}, f_\theta, \gamma, T),$$

Algorithm 1: **CU**rriculum of **D**ata **A**ugmentation

Input: LTR algorithm $\mathcal{A}_{\text{LTR}}(f, \mathcal{D})$, training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, train epochs E, aug. probability p_{aug}, threshold γ, number of sample coefficient T.
Output: trained model f_θ
Initialize: $L_c^0 = 0 \; \forall c \in \{1, ..., C\}$
for $e \leq E$ **do**
  Update $L_c^e = V_{\text{LoL}}(\mathcal{D}_c, L_c^{e-1}, f_\theta, \gamma, T) \quad \forall c$ `// Alg. 2`
  Generate $\mathcal{D}_{\text{CUDA}} = \{(\bar{x}_i, y_i) | (x_i, y_i) \in \mathcal{D}\}$ where

$$\bar{x}_i = \begin{cases} \mathcal{O}(x_i, L_{y_i}^e) & \text{with prob. } p_{\text{aug}} \\ x_i & \text{otherwise.} \end{cases}$$

  Run LTR algorithm using $\mathcal{D}_{\text{CUDA}}$, *i.e.,* $\mathcal{A}_{\text{LTR}}(f_\theta, \mathcal{D}_{\text{CUDA}})$.
end

Algorithm 2: V_{LoL}: Update LoL score

Input: $\mathcal{D}_c$, L, f_θ, γ, T
Output: updated L
Initialize: check $= 1$
for $l \leq L$ **do**
  /* $V_{\text{correct}}(\mathcal{D}_c, l, f_\theta, T)$ */
  Sample $\mathcal{D}'_c \subset \mathcal{D}_c$ s.t. $|\mathcal{D}'_c| = T(l+1)$
  Compute $v = \sum_{x \in \mathcal{D}'_c} \mathbb{1}_{\{f(\mathcal{O}(x;l)=c\}}$
  if $v \leq \gamma T(l+1)$ **then**
    check $\leftarrow 0$; **break**
  end
end
if check $= 1$ **then** $L \leftarrow L + 1$
else $L \leftarrow L - 1$

where

$$V_{\text{LoL}}(\mathcal{D}_c, L_c^{e-1}, f_\theta, \gamma, T) = \begin{cases} L_c^{e-1} + 1 & \text{if } V_{\text{Correct}}(\mathcal{D}_c, l, f_\theta, T) \geq \gamma T(l+1) \quad \forall l \in \{0, ..., L_c^{e-1}\} \\ L_c^{e-1} - 1 & \text{otherwise} \end{cases} .$$

Here, $\gamma \in [0, 1]$ is threshold hyperparameter, T is coefficient of the number of samples used to updating LoL. V_{correct} is a function which outputs the number of correctly predicted examples by the model f_θ among $l + 1$ randomly augmented samples with strength l. V_{correct} is defined as:

$$V_{\text{Correct}}(\mathcal{D}_c, l, f_\theta, T) = \sum_{x \in \mathcal{D}'_c} \mathbb{1}_{\{f_\theta(\mathcal{O}(x;l))=c\}} \quad \text{where } \mathcal{D}'_c \subset \mathcal{D}_c.$$

Note that $\mathcal{D}'_c$ is a randomly sampled subset of $\mathcal{D}_c$ with replacement and its size is $T(l + 1)$.

The key philosophy of this criterion is two fold. (1) If samples in the class c are trained sufficiently with an augmentation strength of L_c^e, the model is ready to learn a more difficult version with augmentation strength of $L_c^{e+1} \leftarrow L_c^e + 1$. In contrast, if the model predicts incorrectly, it should re-learn the easier sample with an augmentation strength of $L_c^{e+1} \leftarrow L_c^e - 1$. (2) As the strength parameter increases, the number of candidates for the sequential augmentation operation $\mathcal{O}(x; L)$ increases exponentially. For example, the amount of increment is $N^L(N - 1)$ when L is increases to $L + 1$. To control the LoL in a large sequential augmentation operation space, we take more random samples to check as the strength parameter gets bigger. In our experiments, linearly increasing the number of samples to evaluate corresponding to the strength with a small additional computation time was sufficient. V_{LoL} is described in Figure 2 and Algorithm 2.

Curriculum of DA. By combining two components, including DA with a strength parameter and LoL, our CUDA provides class-wise adaptive augmentation to enhance the performance of the others without losing its own information. As shown in Figure 2 and Algorithm 1, we measure the LoL score L_c for all classes in the training dataset to determine the augmentation strength for every epoch. Based on L_c, we generate the augmented version $\mathcal{O}(x; L_c)$ for $x \in \mathcal{D}_c$ and train the model with the augmented samples. Additionally, we randomly use the original sample instead of the augmented sample with probability p_{aug} so that the trained models do not forget the original information. In our experiments, this operation improved performance robustly on a wide range of p_{aug} values. The results are provided in Section 4.3.

Advantage of CUDA design. Our proposed approach mainly has three advantages. (1) CUDA adaptively finds proper augmentation strengths for each class without need for a validation set. (2) Following the spirits of existing curriculum learning methods (Hacohen & Weinshall, 2019; Zhou et al., 2020b; Wu et al., 2021), CUDA enables modeling by first presenting easier examples earlier during training to improve generalization. This encourages the model to learn difficult samples (*i.e.,* within high augmentation strength) better. (3) Moreover, owing to the universality of data augmentation, CUDA is easily compatible with other LTR algorithms, such as (Cao et al., 2019; Ren et al., 2020; Wang et al., 2021).

4 Experiments

In this section, we present empirical evaluation, the results of which demonstrate the superior performance of our proposed algorithm for class imbalance. We first describe the long-tailed classification

Table 1: Validation accuracy on CIFAR-100-LT dataset. † are from Park et al. (2022) and ‡, ⋆ are from the original papers (Kim et al., 2020; Zhu et al., 2022). Other results are from our implementation. We format the first and second best results as **bold** and underline . We report the average results of three random trials.

Algorithm	Imbalance Ratio (IR)			Statistics (IR 100)		
	100	50	10	Many	Med	Few
CE	38.7	43.4	56.5	66.2	37.3	8.2
CE + CMO (Park et al., 2022)	42.0	47.0	60.0	69.1	41.2	11.3
CE + CUDA	42.7	47.2	59.6	**71.6**	42.3	9.4
CE + CMO + CUDA	43.5	48.7	60.0	70.0	43.4	12.7
CE-DRW (Cao et al., 2019)	41.4	45.5	57.8	62.8	41.7	16.1
CE-DRW + Remix (Chou et al., 2020)†	45.8	49.5	59.2	-	-	-
CE-DRW + CUDA	47.7	52.4	61.6	64.3	49.2	26.7
LDAM-DRW (Cao et al., 2019)	42.5	47.4	57.6	62.8	42.3	19.0
LDAM + M2m (Kim et al., 2020)‡	43.5	-	57.6	-	-	-
LDAM-DRW + CUDA	47.6	51.1	58.4	67.3	50.4	21.4
BS (Ren et al., 2020)	43.3	46.9	58.3	61.6	42.3	23.0
BS + CUDA	47.7	52.1	61.7	63.3	48.4	28.7
RIDE (3 experts) (Wang et al., 2021)†	48.6	51.4	59.8	-	-	-
RIDE (3 experts)	49.7	52.7	60.2	67.7	51.5	26.7
RIDE + CMO (Park et al., 2022)†	50.0	53.0	60.2	-	-	-
RIDE + CMO	49.9	53.0	58.9	67.3	51.3	28.1
RIDE (3 experts) + CUDA	50.7	53.7	60.2	69.2	52.8	27.3
BCL (Zhu et al., 2022)⋆	51.0	54.9	64.4	67.2	53.1	32.9
BCL + CUDA	**52.3**	**56.2**	**64.6**	66.4	**54.2**	**33.9**

benchmarks and implementations in detail (Section 4.1). Then, we describe the experimental results on several synthetic (CIFAR-100-LT, ImageNet-LT) and real-world (iNaturalist 2018) long-tailed benchmark datasets in Section 4.2. Moreover, we conduct additional experiments to obtain a better understanding of CUDA, and this analysis is provided in Section 4.3.

4.1 Experimental setup

Datasets. We evaluate CUDA on the most commonly used long-tailed image classification tasks: CIFAR-100-LT (Cao et al., 2019), ImageNet-LT (Liu et al., 2019), and iNaturalist 2018 (Van Horn et al., 2018). CIFAR-100-LT and ImageNet-LT are provided with imbalanced classes by synthetically sampling the training samples. CIFAR-100-LT is examined with various imbalance ratios $\{100, 50, 10\}$, where an imbalance ratio is defined as $N_{\text{max}}/N_{\text{min}}$. iNaturalist 2018 is a large-scale real-world dataset includes natural long-tailed imbalance. We utilize the officially provided datasets.

Baselines. We compare CUDA with previous long-tailed learning algorithms , including cross-entropy loss (CE), two-stage approaches: CE-DRW (Cao et al., 2019) and cRT (Kang et al., 2020), balanced loss approaches: LDAM-DRW (Cao et al., 2019) and Balanced Softmax (BS; Ren et al. 2020), the ensemble method: RIDE with three experts (Wang et al., 2021), resampling algorithms: Remix (Chou et al., 2020) and CMO (Park et al., 2022), and contrastive learning-based approach: BCL (Zhu et al., 2022). We integrate CUDA with CE, CE-DRW, LDAM-DRW, BS, RIDE, and BCL algorithms. For longer epochs, we compare CUDA with PaCo (Cui et al., 2021), BCL, and NCL (Li et al., 2022a), by combining CUDA with BCL and NCL. For a fair comparison of the computational cost, we train the network with the official one-stage implementation of RIDE (*i.e.,* without distillation and routing).

Implementation. For CIFAR-100-LT dataset, almost all implementations follow the general setting from Cao et al. (2019), whereas cRT (Kang et al., 2020), BCL, NCL and RIDE follow the settings used in their original implementation. Following Cao et al. (2019), we use ResNet-32 (He et al., 2016) as a backbone network for CIFAR-100-LT. The network is trained on SGD with a momentum of 0.9 and a weight decay of 2×10^{-4}. The initial learning rate is 0.1 and a linear learning rate warm-up is used in the first 5 epochs to reach the initial learning rate. During training over 200 epochs, the learning rate is decayed at the 160th and 180th epochs by 0.01. For the ImageNet-LT and iNaturalist, the ResNet-50 is used as a backbone network and is trained for 100 epochs. The learning rate is decayed at the 60th and 80th epochs by 0.1. As with CIFAR, for cRT, RIDE, and BCL, we follow the original experimental settings of the official released code. For the hyperparameter values of CUDA, we apply a p_{aug} of 0.5 and T of 10 for all experiments. For γ, we set the values as 0.6 for

Table 2: Validation accuracy on ImageNet-LT and iNaturalist 2018 datasets. † indicates reported results from the Park et al. (2022) and ‡ indicates those from the original paper (Kang et al., 2020). ⋆ means we train the network with the official code in an one-stage RIDE.

Algorithm	ImageNet-LT				iNaturalist 2018			
	Many	Med	Few	All	Many	Med	Few	All
CE†	64.0	33.8	5.8	41.6	73.9	63.5	55.5	61.0
CE + CUDA	**67.1**	47.1	13.4	47.2	**74.6**	65.0	57.2	62.5
CE-DRW (Cao et al., 2019)	61.7	47.3	28.8	50.1	68.2	67.3	66.4	67.0
CE-DRW + CUDA	61.7	48.4	30.5	51.1	68.8	67.9	66.5	67.4
LWS (Kang et al., 2020)‡	57.1	45.2	29.3	47.7	65.0	66.3	65.5	65.9
cRT (Kang et al., 2020)‡	58.8	44.0	26.1	47.3	69.0	66.0	63.2	65.2
cRT + CUDA	62.3	47.2	28.1	50.2	68.2	67.9	66.4	67.3
LDAM-DRW (Cao et al., 2019)†	60.4	46.9	30.7	49.8	-	-	-	66.1
LDAM-DRW + CUDA	63.2	48.2	31.2	51.5	68.0	67.5	66.8	67.3
BS (Ren et al., 2020)	60.9	48.8	32.1	51.0	65.7	67.4	67.5	67.3
BS + CUDA	61.8	49.1	31.8	51.5	67.6	68.2	68.3	68.2
RIDE (3 experts) (Wang et al., 2021)⋆	64.9	50.4	34.4	53.6	70.4	71.8	71.8	71.6
RIDE + CMO (Park et al., 2022)⋆	65.6	50.6	34.8	54.0	68.0	70.6	72.0	70.9
RIDE (3 experts) + CUDA⋆	66.0	51.7	34.7	54.7	70.6	72.6	**72.7**	**72.4**
BCL (Zhu et al., 2022)	65.3	53.5	36.3	55.6	69.4	72.4	71.8	71.8
BCL + CUDA	66.8	**53.9**	**36.6**	**56.3**	70.8	**72.7**	72.0	72.2

Table 3: Comparison for CIFAR-LT-100 performance on ResNet-32 with 400 epochs.

Algorithm	Imbalance Ratio	
	100	50
PaCo	52.0	56.0
BCL	52.6	57.2
NCL	54.2	58.2
BCL + CUDA	53.5	57.4
NCL + CUDA	**54.8**	**59.6**

Table 4: Augmentation analysis on CIFAR-100-LT with IR 100. AA (Cubuk et al., 2019), FAA (Lim et al., 2019), DADA (Li et al., 2020b), and RA (Cubuk et al., 2020) with $n = 1, m = 2$ policies are used. C, S, I represent CIFAR, SVHN, and ImageNet policy.

	Vanilla	AA			FAA		DADA		RA	CUDA
		C	S	I	C	I	C	I		
CE	38.7	41.7	40.7	40.1	42.3	40.8	41.0	41.2	40.5	**42.7**
CE-DRW	41.4	46.5	44.7	45.5	46.3	44.8	45.6	45.7	45.8	**47.4**
LDAM-DRW	42.5	47.0	44.7	44.9	46.6	45.6	45.9	46.5	44.0	**47.2**
BS	43.3	47.0	46.1	45.5	46.5	45.0	45.0	46.9	45.2	**47.7**
RIDE (3 experts)	49.7	49.5	47.3	45.5	49.8	50.6	50.4	50.5	47.9	**50.7**

CIFAR-100-LT and 0.4 for ImageNet-LT and iNaturalist 2018. The detailed implementation for baselines are in Appendix B.

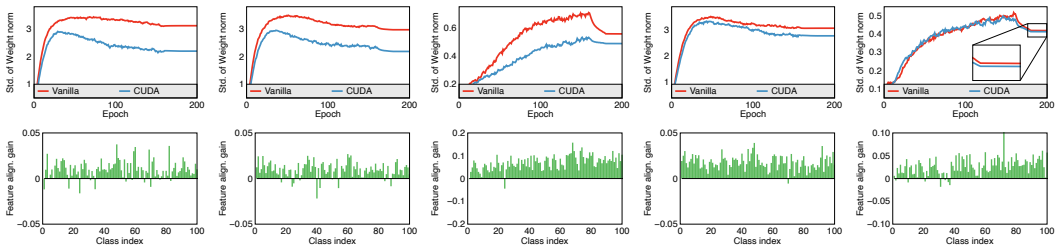
4.2 Experimental Results

In this section, we report the performances of the methods compared on the CIFAR-100-LT, ImageNet-LT, and iNaturalist 2018. We include four different categories of accuracy: all, many, med(ium), and few. Each represents the average accuracy of all samples, classes containing more than 100 samples, 20 to 100 samples, and under 20 samples, respectively.

CIFAR-100-LT. In Table 1, we report the performance when CUDA is applied to the various algorithms: CE, CE-DRW (Cao et al., 2019), LDAM-DRW (Cao et al., 2019), BS (Ren et al., 2020), RIDE (Wang et al., 2021) with 3 experts, RIDE+CMO (Park et al., 2022), and BCL (Zhu et al., 2022). Compared to the cases without CUDA, balanced validation performance is increased when we apply the proposed approach.

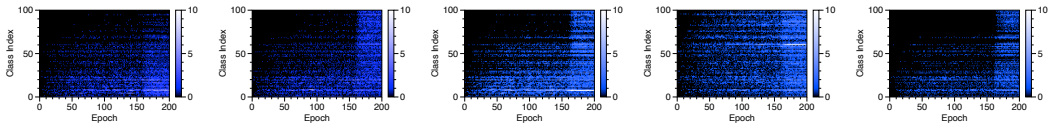
Recently, some works (Cui et al., 2021; Alshammari et al., 2022; Zhu et al., 2022; Li et al., 2022a) have shown impressive performances with diverse augmentation strategies and longer training epochs. For a fair comparison with these methods, we examine CUDA using the same experimental setups from PaCo (Cui et al. 2021; 400 epochs with batch size of 64). Table 3 shows that augmented images using CUDA can enhance LTR performance compared to the other baselines. In particular, CUDA with NCL obtains the best performance over 400 epochs. As noted by Li et al. (2022a), the NCL algorithm utilizes six times as much memory compared to the vanilla architecture with three experts. Hereinafter in large-scale benchmarks, we focus on the cases with similar network size.

ImageNet-LT and iNaturalist 2018. To evaluate the performance of CUDA on larger datasets, we conduct experiments on ImageNet-LT (Liu et al., 2019) and iNaturalist 2018 (Van Horn et al., 2018). Table 2 summarizes the performance of various LTR methods and the performance gain when integrated with CUDA. Our proposed method consistently improves performance regardless of the



(a) CE (b) CE-DRW (c) LDAM-DRW (d) BS (e) RIDE

Figure 3: Analysis of how CUDA improves long-tailed recognition performance, classifier weight norm (top row) and feature alignment (bottom row) of the CIFAR-100-LT validation set. Notably that weight norm and feature alignment represent class-wise weight magnitude of classifier and ability of feature extractor, respectively. The detailed analysis is described in Section 4.3.



(a) CE (b) CE-DRW (c) LDAM-DRW (d) BS (e) RIDE

Figure 4: Evolution of LoL score on various algorithms, CE, CE-DRW, LDAM-DRW, BS, and RIDE.

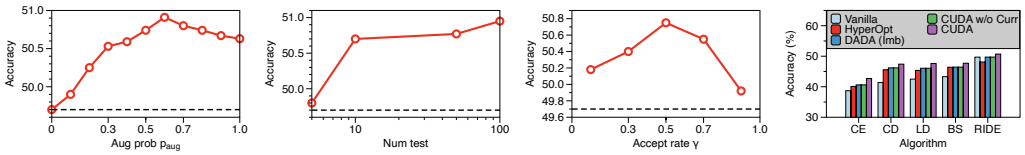
LTR method and target dataset by simply adding class-wise data augmentation without complicated methodological modification. Additionally, to evaluate the performance gain of CUDA on other architectures, we experiment with CUDA on ImageNet-LT with ResNet-10 (Liu et al., 2019) and ResNeXt-50 (Xie et al., 2017), as reported in Appendix C.

4.3 Analysis

We design our analyses to answer the following questions. (1) How does CUDA perform? (2) Does CUDA perform better than other augmentation methods? (3) How does LoL score change over training epochs when combined with various LTR methods? (4) Which part of CUDA is important to improved performance? These analyses provide additional explanations to understand CUDA. All experiments are conducted on CIFAR-100-LT with imbalance ratio of 100.

How does CUDA mitigate the class imbalance problem? To deeply understand CUDA, we observe two types of metrics: (1) variance of weight L1-Norm of linear classifier between each class (2) feature alignment gain for each class (*i.e.,* cosine similarity with and without CUDA) on validation dataset. The classifier weight norm is usually used to measure how balanced the model consider the input from a class-wise perspective (Kang et al., 2020; Alshammari et al., 2022). Feature alignment, especially feature cosine similarity amongst samples belonging to the same class, is a measure of the extent to which the extracted features are aligned (Oh et al., 2021). As shown in Figure 3, CUDA has two forces for alleviating imbalance. For all cases, CUDA reduces the variance of the weight norm (*i.e.,* balance the weight norm), and thus the trained model consider the minority classes in a balanced manner. Note that because LDAM-DRW and RIDE utilize a cosine classifier (*i.e.,* utilizing L2 normalized linear weight), their standard deviation scale is quite different from those other methods. Because LDAM-DRW, BS, and RIDE include balancing logic in their loss function, they exhibit lower variance reduction compared to the CE and CE-DRW. Second, as shown in the bottom row in Figure 3, CUDA obtains feature alignment gains for almost all classes. This shows that CUDA facilitates a network to learn to extract meaningful features.

Compared with other augmentations. To verify the impact of CUDA, we examine the other augmentation methods as follows. We compare five augmentation methods, including AutoAugment (AA, Cubuk et al. 2019), Fast AutoAugment (FAA, Lim et al. 2019), DADA (Li et al., 2020b), RandAugment (RA, Cubuk et al. 2020), and the proposed method CUDA. Because AA, FAA, and DADA provide their policies searched by using CIFAR, SVHN (for AA), and ImageNet, we leverage their results. Furthermore, RA suggests using their parameter $(n, m) = (1, 2)$ for CIFAR, and we follow their guidelines. As shown in Table 4, even though the augmentation policies use additional computation resources to search, CUDA outperforms the other pre-searched augmentations. This shows that CUDA is computationally efficient.



(a) Aug prob. p_{aug}. (b) Num test T. (c) Accept rate γ. (d) Curriculum.

Figure 5: Additional analysis of CUDA. (a) sensitivity of augmentation probability p_{aug}, (b) sensitivity analysis of number of sample coefficient T, (c) sensitivity of acceptance threshold γ, and (d) impact of curriculum. The dotted lines in (a), (b) and (c) represents the performance of CE.

Dynamics of LoL score. We evaluate how LoL scores vary with algorithms: CE, CE-DRW, LDAM-DRW, BS, and RIDE. Note that we set a lower class index (*i.e.,* 0) as the most common class (*i.e.,* the number of samples is 500), while an index of 100 represents the rarest class (*i.e.,* with five samples). As described in Figure 4, as training progressed, the LoL score of all algorithms increase. After learning rate decay (*i.e.,* 160 epoch) all algorithms are able to learn to classify minority classes more easily than before. In particular, except for BS, the majority classes of most algorithms show a steep increment. The reason that BS exhibit a similar increasing speed for majority and minority classes is that it includes a module to balance the impact of majority and minority samples. Moreover, because RIDE can promote when all experts satisfying promotion criterion, its increasing speed is moderate.

Parameter sensitivity. For further analysis, we conduct a sensitivity analysis of hyperparameters in CUDA. More precisely, we study three kinds of parameters, including augmentation probability p_{aug} (Figure 5a), number of tests T (Figure 5b), and LoL update threshold γ (Figure 5c). We examine each hyperparameter sensitivity on a CUDA case with RIDE and the remainder of the hyperparameters are fixed to the default values in Section 4.1. All results show that the performance gains of CUDA decreases if the parameters are adjusted to make the augmentation too strong or weak. For example, the augmentation strength of all classes steeply increases when γ becomes small. The strength cannot increase when γ becomes large, and thus it cannot improve the performance of the model. Moreover, as shown in Figure 5b, the performance of CUDA increases as T increases. However, larger T spends computational overhead, we set T as 10 and obtained cost-effective performance gain.

Impact of curriculum. In addition to studying the impact of CUDA, we examine its performance component-wise. In particular, we test the case where class-wise augmentation strength is searched based on the hyperparameter optimization algorithm. We check five cases overall: vanilla algorithm, hyperparameter optimization (HO), re-searched DADA for CIFAR-100-LT, CUDA without curriculum, (*i.e.,* re-training utilizing the final augmentation strength of CUDA), and CUDA. As described in Figure 5d, CUDA finds better augmentation strengths compare to the hyperparameter search case. This means that CUDA exhibits not only a lower searching time but also obtains better augmentation strength. Moreover, by comparing the performance of with or without curriculum, the curriculum also can provide additional advance to the model to achieve better generalization. Additionally, as Figure 4, lower augmentation strength at the beginning of training is more effective than static higher augmentation strength. These results are consistent with the results of previous studies on curriculum learning methods (Hacohen & Weinshall, 2019; Zhou et al., 2020b).

5 Conclusion

In this study, we proposed CUDA to address the class imbalance problem. The proposed approach is also compatible with existing methods. To design a proper augmentation for LTR, we first studied the impact of augmentation strength for LTR. We found that the strength of augmentation for a specific type of class (*e.g.,* major class) could affect the performance of the other type (*e.g.,* minor class). From this finding, we designed CUDA to adaptively find an appropriate augmentation strength without any further searching phase by measuring the LoL score for each epoch and determining the augmentation accordingly. To verify the superior performance of proposed approach, we examined each performance with various methods and obtained the best performance among the methods compared, including synthetically generated and real-world benchmarks. Furthermore, from our analyses, we validated that our CUDA enhanced balance and feature extraction ability, which can consistently improve performance for majority and minority classes.

REFERENCES

Shaden Alshammari, Yu-Xiong Wang, Deva Ramanan, and Shu Kong. Long-tailed recognition via weight balancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6897–6907, 2022.

Shin Ando and Chun Yuan Huang. Deep over-sampling framework for classifying imbalanced data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 770–785. Springer, 2017.

James Bergstra, Daniel Yamins, and David Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International conference on machine learning*, pp. 115–123. PMLR, 2013.

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.

Jiarui Cai, Yizhou Wang, and Jenq-Neng Hwang. Ace: Ally complementary experts for solving long-tailed recognition in one-shot. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 112–121, 2021.

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Tsz-Him Cheung and Dit-Yan Yeung. Adaaug: Learning class-and instance-adaptive data augmentation policies. In *International Conference on Learning Representations*, 2021.

Junsuk Choe and Hyunjung Shim. Attention-based dropout layer for weakly supervised object localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2219–2228, 2019.

Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: rebalanced mixup. In *European Conference on Computer Vision*, pp. 95–110. Springer, 2020.

Peng Chu, Xiao Bian, Shaopeng Liu, and Haibin Ling. Feature space augmentation for long-tailed data. In *European Conference on Computer Vision*, pp. 694–710. Springer, 2020.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 113–123, 2019.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 702–703, 2020.

Jiequan Cui, Zhisheng Zhong, Shu Liu, Bei Yu, and Jiaya Jia. Parametric contrastive learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 715–724, 2021.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9268–9277, 2019.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Denis Gudovskiy, Luca Rigazio, Shun Ishizaka, Kazuki Kozuka, and Sotaro Tsukizawa. Autodo: Robust autoaugment for biased data with label noise via scalable probabilistic implicit differentiation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16601–16610, 2021.

Hongyu Guo, Yongyi Mao, and Richong Zhang. Mixup as locally linear out-of-manifold regularization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3714–3722, 2019.

Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. In *International Conference on Machine Learning*, pp. 2535–2544. PMLR, 2019.

Ryuichiro Hataya, Jan Zdenek, Kazuki Yoshizoe, and Hideki Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation. In *European Conference on Computer Vision*, pp. 1–16. Springer, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1gRTCVFvB.

Bingyi Kang, Yu Li, Sa Xie, Zehuan Yuan, and Jiashi Feng. Exploring balanced feature spaces for representation learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=OqtLIabPTit.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020.

Jaehyung Kim, Jongheon Jeong, and Jinwoo Shin. M2m: Imbalanced classification via major-to-minor translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13896–13905, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, pp. 179. Citeseer, 1997.

Krishna Kumar Singh and Yong Jae Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3524–3533, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Jun Li, Zichang Tan, Jun Wan, Zhen Lei, and Guodong Guo. Nested collaborative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6949–6958, 2022a.

Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-Tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. A system for massively parallel hyperparameter tuning. *Proceedings of Machine Learning and Systems*, 2:230–246, 2020a.

Shuang Li, Kaixiong Gong, Chi Harold Liu, Yulin Wang, Feng Qiao, and Xinjing Cheng. Metasaug: Meta semantic augmentation for long-tailed visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5212–5221, 2021.

Tianhong Li, Peng Cao, Yuan Yuan, Lijie Fan, Yuzhe Yang, Rogerio S Feris, Piotr Indyk, and Dina Katabi. Targeted supervised contrastive learning for long-tailed recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6918–6928, 2022b.

Yonggang Li, Guosheng Hu, Yongtao Wang, Timothy Hospedales, Neil M Robertson, and Yongxin Yang. Differentiable automatic data augmentation. In *European Conference on Computer Vision*, pp. 580–595. Springer, 2020b.

Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*, 2018.

Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *Advances in Neural Information Processing Systems*, 32, 2019.

Hong Liu, Jeff Z. HaoChen, Adrien Gaidon, and Tengyu Ma. Self-supervised learning is more robust to dataset imbalance. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=4AZz9osqrar.

Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2537–2546, 2019.

Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=37nvvqkCo5.

Jaehoon Oh, Hyungjun Yoo, ChangHwan Kim, and Se-Young Yun. {BOIL}: Towards representation change for few-shot learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=umIdUL8rMH.

Seulki Park, Jongin Lim, Younghan Jeon, and Jin Young Choi. Influence-balanced loss for imbalanced visual classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 735–744, 2021.

Seulki Park, Youngkyu Hong, Byeongho Heo, Sangdoo Yun, and Jin Young Choi. The majority can help the minority: Context-rich minority oversampling for long-tailed classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6887–6896, 2022.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

Jiawei Ren, Cunjun Yu, Xiao Ma, Haiyu Zhao, Shuai Yi, et al. Balanced meta-softmax for long-tailed visual recognition. *Advances in neural information processing systems*, 33:4175–4186, 2020.

Cédric Rommel, Thomas Moreau, Joseph Paillard, and Alexandre Gramfort. CADDA: Class-wise automatic differentiable data augmentation for EEG signals. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=6IYp-35L-xJ.

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. *International journal of computer vision*, 115(3):211–252, 2015.

Ryo Takahashi, Takashi Matsubara, and Kuniaki Uehara. Ricap: Random image cropping and patching data augmentation for deep cnns. In *Asian conference on machine learning*, pp. 786–798. PMLR, 2018.

Yuji Tokozume, Yoshitaka Ushiku, and Tatsuya Harada. Between-class learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5486–5494, 2018.

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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8769–8778, 2018.

Jason Van Hulse, Taghi M Khoshgoftaar, and Amri Napolitano. Experimental perspectives on learning from imbalanced data. In *Proceedings of the 24th international conference on Machine learning*, pp. 935–942, 2007.

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 *International Conference on Machine Learning*, pp. 6438–6447. PMLR, 2019.

Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella Yu. Long-tailed recognition by routing diverse distribution-aware experts. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=D9I3drBz4UC.

Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=tW4QEInpni.

Liuyu Xiang, Guiguang Ding, and Jungong Han. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *European Conference on Computer Vision*, pp. 247–263. Springer, 2020.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017.

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 *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1Ddp1-Rb.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 13001–13008, 2020.

Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9719–9728, 2020a.

Tianyi Zhou, Shengjie Wang, and Jeffrey Bilmes. Curriculum learning by dynamic instance hardness. *Advances in Neural Information Processing Systems*, 33:8602–8613, 2020b.

Jianggang Zhu, Zheng Wang, Jingjing Chen, Yi-Ping Phoebe Chen, and Yu-Gang Jiang. Balanced contrastive learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6908–6917, 2022.