From SMOTE to Mixup for Deep Imbalanced Classification

Wei-Chao Cheng[1], Tan-Ha Mai[1] and Hsuan-Tien Lin[1*]

[1*]Department of Computer Science and Information Engineering, National Taiwan University, Taiwan.

*Corresponding author(s). E-mail(s): htlin@csie.ntu.edu.tw; Contributing authors: wccheng3011@gmail.com; d10922024@csie.ntu.edu.tw;

Abstract

Given imbalanced data, it is hard to train a good classifier using deep learning because of the poor generalization of minority classes. Traditionally, the well-known synthetic minority oversampling technique (SMOTE) for data augmentation, a data mining approach for imbalanced learning, has been used to improve this generalization. However, it is unclear whether SMOTE also benefits deep learning. In this work, we study why the original SMOTE is insufficient for deep learning, and enhance SMOTE using soft labels. Connecting the resulting soft SMOTE with Mixup, a modern data augmentation technique, leads to a unified framework that puts traditional and modern data augmentation techniques under the same umbrella. A careful study within this framework shows that Mixup improves generalization by implicitly achieving uneven margins between majority and minority classes. We then propose a novel margin-aware Mixup technique that more explicitly achieves uneven margins. Extensive experimental results demonstrate that our proposed technique yields state-of-the-art performance on deep imbalanced classification while achieving superior performance on extremely imbalanced data. The code is open-sourced in our developed package https://github.com/ntucllab/imbalanced-DL to foster future research in this direction.

Keywords: Deep Learning, Imbalanced Classification, Margin, Mixup, Data Augmentation

1 Introduction

Imbalanced classification is an old yet practical research problem for the machine learning and artificial intelligence community. For example, fraud detection applications [1, 2] are often characterized by data imbalance, because there are far fewer fraudulent cases than normal ones. Another example is real-world image data for computer vision, which often exhibits long-tail properties, where minority classes occur less frequently [3–5].

One immediate challenge in imbalanced classification is that minority classes are under-represented in the objective function, which can result in underfitting to these minority classes. This is typically addressed via re-weighting [6, 7] or re-sampling [8, 9] techniques. Re-weighting techniques belong to the family of algorithm-oriented approaches, which directly modify the objective function and optimization steps. Re-sampling techniques, on the other hand, belong to the family of data-oriented approaches, which manipulate the data being fed to the learning model.

Among algorithm-oriented techniques, re-weighting by inverse class frequencies stands out as one of the simplest methods, as discussed in previous works [10]. Other approaches assign weights in various ways as [7, 11]. For instance, in the study by Cui et al. [7], a theoretical framework is developed to calculate the effective number of examples for each class, subsequently assigning suitable weights based on this calculated value. More sophisticated approaches in the algorithm-oriented family modify the objective function to favor minority classes. For instance, the label-distribution-aware margin (LDAM) loss proposed in [12] is based on a theoretical framework that gives minority classes a larger margin. LDAM achieves state-of-the-art performance on benchmark datasets. Nevertheless, it is harder to optimize LDAM loss across general deep learning models due to its sophisticated design.

The most basic approaches in the data-oriented family involve oversampling minority classes or downsampling majority classes [8] in an attempt to make the data distribution less skewed. Compared with re-weighting approaches, such sampling approaches tend to be less stable. Moreover, oversampling or downsampling from the original data brings no new information to the learning model. Advanced approaches in the data-oriented family are thus based on *synthetic* (or virtual) examples, such as the well-known synthetic minority oversampling technique (SMOTE) [8]. As its name suggests, SMOTE synthesizes virtual examples for minority classes to improve imbalanced classification. Its concept has inspired various follow-up studies that also synthesize virtual examples for imbalanced classification [9, 13]. SMOTE and its follow-ups are closely related to data augmentation techniques commonly used in modern deep learning [14–16]. Nevertheless, despite the practical success of SMOTE for non-deep models [13, 17], SMOTE has not been thoroughly studied for its validity when coupled with modern deep learning models.

A recent follow-up to SMOTE, designed for addressing imbalanced learning in the context of modern deep learning, is DeepSMOTE [18]. This method leverages the concept of Generative Adversarial Networks (GANs) [19] for oversampling. Effective SMOTE-based generation of synthetic examples is achieved by utilizing a deep encoder-decoder model to convert the original data into a lower-dimensional representation space. It allows DeepSMOTE to perform better on complex data than the original SMOTE. DeepSMOTE is claimed to produce high-quality synthetic examples to assist imbalanced classification. Somehow to the best of our knowledge, DeepSMOTE needs more benchmarks to demonstrate its practical potential.

Another oversampling technique is Major-to-minor Translation (M2m) [20]. M2m also addresses class imbalance by augmenting less-frequent classes through sample translation from more-frequent ones. It employs a pre-trained model to identify potential samples by introducing random noise to majority-class images; in case, the pre-trained model does not identify synthetic data, it uses existing minority samples to achieve balance. By leveraging and integrating the diversity of majority information, this approach enables the classifier to acquire more generalized features from the minority classes. Despite its benefits, M2m is computationally intensive and complex to implement due to the translation process.

In this work, we examine the SMOTE approach to understand its disadvantages when coupled with modern deep learning models. We correct these disadvantages via a soft variant of SMOTE that achieves competitive performance on benchmark datasets. We then show that the soft variant of SMOTE is coincidentally connected with Mixup [16], a modern and popular augmentation technique for deep learning, which however was not originally proposed for imbalanced classification. Although a recent workshop paper [21] proposes a variant that modifies Mixup [16] to improve deep imbalanced classification, the effectiveness and rationale of Mixup and its variants for deep imbalanced classification have not been adequately studied, to the best of our knowledge.

Inspired by LDAM [12], which successfully improves deep imbalanced classification with uneven margins, we study the effectiveness of Mixup via margin statistics analysis. We introduce a new tool called the *margin gap* between the majority and minority classes. The gap is empirically demonstrated to be loosely correlated to the accuracy in deep imbalanced classification. We find that Mixup [16] implicitly improves the margin gap, which constitutes a new piece of empirical evidence that explains its effectiveness. We further demonstrate that the gap can be more explicitly fine-tuned by making Mixup margin-aware when synthesizing the inputs and output of the virtual example. The proposed margin-aware Mixup (MAMix) approach empirically achieves state-of-the-art performance on common imbalanced classification benchmarks, and achieves significantly better performance than Mixup and LDAM for extremely imbalanced datasets. The results validate the usefulness of our study and our proposed approach.

To make deep imbalanced learning easier for researchers and real-world users, we further develop an open-sourced python package called **imbalanced-DL** for this community. From our experience, we observed that to tackle deep imbalanced classification effectively, a single model may not be sufficient, thus we provide several strategies for people to use. The package not only implements several popular deep imbalanced learning strategies, but also provides benchmark results on several image classification tasks. We hope that our research findings along with our developed software can not only help with reproducibility but also shed lights on more comprehensive research in this community in the future.

We summarize our contributions as the following: (i) We systematically design and study the variants of the SMOTE algorithm for deep learning, (ii) We are first to utilize margin statistics to analyze whether a model has learned proper representations through uneven margins for deep imbalanced classification, (iii) We determine that a direct application of the original Mixup [16] already achieves competitive results for imbalanced learning by implicitly enforcing uneven margins, (iv) We further develop a simple yet effective algorithm that guides Mixup to take margins into account more explicitly, and show that the algorithm works particularly well when the data is extremely imbalanced.

2 Related Work

In this section, we first define the imbalanced learning problem and review existing solutions. Then we discuss studies that are closely related to our approach. For a more comprehensive survey, see [22].

2.1 Problem Setup and Notations

We consider the imbalanced K-class classification problem. Let $x \in \mathbb{R}^d$ denote the input and $y \in \{1, \ldots, K\}$ denote the corresponding label. Given the training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ generated from some unknown $P(x, y)$ independently, our goal is to learn a classifier $f(x) \colon \mathbb{R}^d \to \{1, \ldots, K\}$, which predicts the correct label from a given input x. Let n_j be the size of class j. We assume the training data to be *imbalanced*. That is, the size of the largest class $\max_i n_i$ is very different from the size of the smallest class $\min_i n_i$. The larger classes are generally called the *majority*, and the smaller ones are called the *minority*. After learning $f(x)$, we follow [12] to evaluate its accuracy on a *balanced* test set generated from the same $P(x \mid y)$ for each class. The evaluation essentially equalizes the importance of each class.

In this work, we adopt two standard benchmark settings to generate controllable synthetic datasets from real-world datasets [12, 23]. Both settings first decide the target size of each class by some parameters, and randomly sample within the real-world dataset to obtain the corresponding synthetic dataset under the target sizes. Both settings are based on the parameter of *class imbalance ratio*, which is the ratio between the size of the largest (head) class and that of the smallest (tail) class, that is, $\rho = \max_i n_i \,/\, min_i n_i$. The parameter characterizes the difficulty level of the dataset.

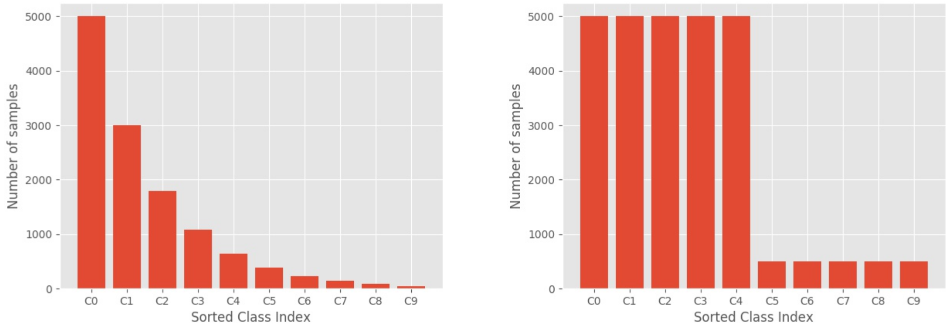
The first setting is called *step imbalance*, defined by ρ and another parameter μ. Step imbalance requires that μK of the classes be the minority, and the other $(1-\mu)K$ be the majority. All the minority classes are of the same size, and so are all the majority classes. Following the class imbalance ratio, the size of the majority classes is ρ times larger than that of the minority ones.

The second setting is called *long-tailed imbalance* [7, 12] defined by ρ, where the sizes of the classes follow an exponentially decreasing sequence with a decreasing constant of $\rho^{1/(K-1)}$. The constant ensures that the class imbalance ratio is exactly ρ. An illustrative example for long-tailed and step imbalance is in Fig. 1.

2.2 Algorithm-Oriented Approach

Traditionally, many classification approaches are designed from the principle of empirical risk minimization (ERM), which minimizes the summation of some loss function on each example. For the imbalanced classification, the ERM principle easily leads to underfitting the minority classes, as they are under-represented in the summation.

Approaches that improve ERM for the imbalanced classification problem can be roughly divided to two categories: algorithm-oriented and data-oriented. One possible algorithm-oriented approach, known as cost-sensitive learning, gives a higher cost when



(a) $\rho = 100$ (b) $\mu = 0.5, \rho = 10$

Fig. 1 Number of training samples per class in synthetically generated imbalanced CIFAR-10 datasets for (a) long-tailed imbalance with $\rho = 100$ and (b) step imbalance with $\rho = 10,\ \mu = 0.5$

mis-classifying the minority class [24]. Cost-sensitive learning can also be carried out by giving larger weights to the minority examples. For instance, the class balance (CB) loss [7] re-weights each class by calculating its effective number of examples. Re-weighting increases the importance of the minority examples in the loss function, therefore preventing underfitting the minority classes. [12] shows that learning with re-weighting from the beginning of training can result in degraded representations because of early overfitting to the minority classes, making the performance of the re-weighting even worse than ERM. To solve the overfitting issue, [12] also proposed the deferred re-weighting (DRW) technique. DRW splits the one-stage training of deep learning into two phases. In the first phase, ERM without any re-weighting is used to learn a good representation, with the hope of not overfitting to the minority classes. Then, the training continues with an annealed (smaller) learning rate on a re-weighted loss, such as CB loss, in the second phase.

With the DRW technique, some other algorithmic attempts are used to improve ERM. Label-distribution-aware margin (LDAM) [12] follows the rich literature of margin classifiers [25, 26] and proposes a loss function that encourages class-dependent margins to tackle the class imbalance issue. The ideal margin τ_i for each class is theoretically derived to be proportional to $n_i^{1/4}$. That is,

$$\tau_i = \frac{C}{n_i^{1/4}} \tag{1}$$

with some constant C. The ideal margin hints the need to enforce larger margins for the minority classes.

With the definition of τ_i, the authors of LDAM propose a margin-aware loss function that can be used in both the ERM phase and the re-weighting phase of DRW.

Combining LDAM and DRW with the CB loss in the second phase results in a state-of-the-art approach for imbalanced learning [12], which will serve as the baseline of our comparison.

2.3 Data-Oriented Approach

A common approach for imbalanced multi-class classification at the data level is undersampling for majority classes or oversampling for minority classes. One such approach is SMOTE [8], which essentially oversamples minority classes by creating artificial examples through k-nearest neighbors within the same class. In the context of deep learning, this kind of oversampling can be viewed as a type of data augmentation. Also note ADASYN [9] and LoRAS [13], SMOTE extensions that address class imbalance using machine learning approaches. In this work, we revisit SMOTE and incorporate it into a modern deep learning pipeline.

2.3.1 SMOTE

Traditional replication-based oversampling techniques are prone to overfitting. To account for this, [8] proposes oversampling by creating synthetic examples for minority classes; in this case, the synthetic examples are thus not replicated. Specifically, for those samples categorized as belonging to a minority class, they create new data points by interpolating them with their k-nearest neighbors which belong to the same categories. Note that at the time this technique was proposed, deep learning techniques were not yet widely used. Thus, we first study this technique and design two SMOTE-like techniques along with the current end-to-end deep learning training pipeline. This is described in detail in the next section. We also note DeepSMOTE [18], which was published during the course of the current study. However, since this approach requires two-stage training in which the first stage requires training an encoder-decoder framework, followed by DeepSMOTE generation, we consider it to be aligned more with GAN-based work, which is not our main focus.

2.4 Mixup-based Techniques

2.4.1 Mixup

One of the most famous regularization—or data augmentation—techniques in deep neural networks for image classification problem is Mixup [16], which constructs virtual training examples via simple linear combinations as:

$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j \tag{2}$$

$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j, \tag{3}$$

in which (x_i, y_i) and (x_j, y_j) are two examples drawn uniformly from the training data and $\lambda \in [0, 1)$. Mixup-based techniques have been shown to mitigate the memorization of corrupt labels, increase robustness to adversarial training, and improve the generalizability of deep networks, which has led to state-of-the-art performance on tasks such as image classification.

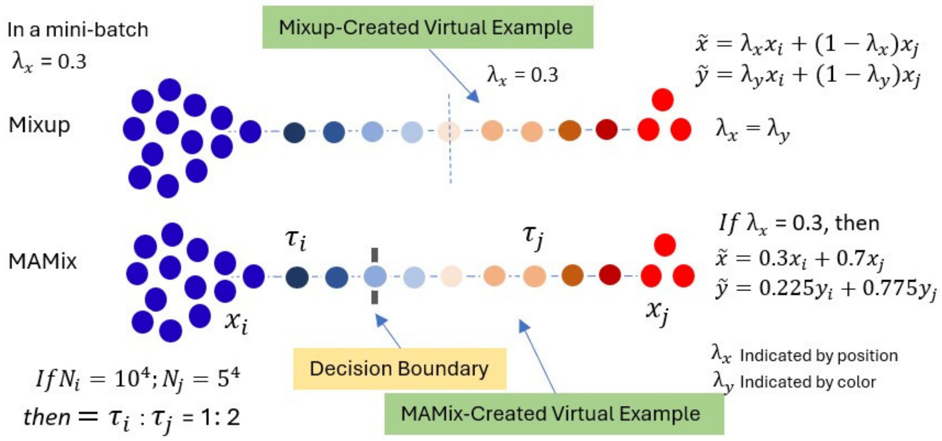


Fig. 2 Mixup Framework Illustration

3 Main Approach

We observe that Mixup [16] can generalize to a general framework, in the sense that they both train with similar fashion. We term this *Mixup framework* (Fig. 2), and describe the training algorithm for Mixup framework in Algorithm 1.

Algorithm 1: Mixup Framework Training Algorithm

Required Dataset $D = \{(x_i, y_i)\}_{i=1}^{n}$, model with parameter θ
Initialize;
while *training* **do**
  Sample $\{(x_i, y_i), (x_j, y_j)\}_{m=1}^{M}$ from D;
  Sample $\lambda_x \sim Beta(\alpha, \alpha)$;
  for $m = 1$ *to* M **do**
    (a) Obtain mixed input $\tilde{x}$;
    (b) Obtain λ_y ;
    (c) Obtain mixed label $\tilde{y}$;
  end
  $\mathcal{L}(\theta) \leftarrow \frac{1}{M} \sum_{(\tilde{x},\tilde{y})} L((\tilde{x}, \tilde{y});\ \theta)$;
  $\theta \leftarrow \theta - \delta \bigtriangledown_\theta \mathcal{L}(\theta)$;
end

Specifically, within this Mixup Framework, the main difference between each method lies in three steps during mini-batch training, that is, (a) How to obtain mixed input (b) How to obtain label mixing factor λ_y and (c) How to obtain mixed label.

With this Mixup Framework, we design new methods through two perspectives. First, we design two SMOTE-like techniques—SMOTE-Mix and Neighbor-Mix—within this framework to examine the effectiveness of SMOTE in modern deep learning from input mixing perspective, and this is described in the following *Approach 1*. Secondly, we propose to incorporate the idea of *uneven margin* into this Mixup framework to better tackle deep imbalanced learning, which will be illustrated in *Approach 2*. Our proposed Approach 2 can be viewed from non-uniform label mixing perspective.

3.1 Approach 1: SMOTE-like Techniques

We introduce two SMOTE-like techniques from input mixing perspective in SMOTE-Mix and Neighbor-Mix. First, we perform SMOTE-like input mixing under Mixup framework and term this *SMOTE-Mix*. Recall that SMOTE performs linear interpolation with their same-class samples on input only. Formally, with SMOTE-Mix, we create synthetic examples from two training samples $(x_i, y_i), (x_j, y_j)$ with the following equations:

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j \tag{4}$$

$$x_j = \text{same-class nearest neighbor of } x_i$$

$$\tilde{y} = y_i. \tag{5}$$

Following Algorithm 1, SMOTE-Mix obtains mixed input by (4), mixed label by (5), and $\lambda_y = \lambda_x$. Note that in SMOTE-Mix, the mix pair for creating synthetic examples is sampled from its same-class nearest neighbors. Thus for each pair, the label is the same $(y_i = y_j)$; that is, they are hard labels.

We then further relax the above idea by not restricting x_j to be the same class as x_i; that is, we still create synthetic samples through the nearest neighbors, but due to the fact that data are in a high dimensional space, its nearest neighbors may not belong to the same categories. We term this relaxed version *Neighbor-Mix*, and formulate it as:

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j \tag{6}$$

$$x_j = \text{nearest neighbor of } x_i$$

$$\tilde{y} = \lambda y_i + (1-\lambda)y_{x_j}. \tag{7}$$

Following Algorithm 1, Neighbor-Mix obtains mixed input by (6), mixed label by (7), and $\lambda_y = \lambda_x$. Note that for $\tilde{y}$, Neighbor-Mix is soft-label, as x_j may belong to other categories.

We discuss the empirical results of SMOTE-Mix and Neighbor-Mix on modern long-tailed image datasets in Table 1 to verify the effectiveness of SMOTE in deep learning. Now we further propose our main strategy within the Mixup framework to address deep imbalanced classification.

3.2 Approach 2: Margin-Aware Mixup (MAMix)

Inspired by the attempt to achieve uneven margins through a well-designed LDAM loss [12], we propose incorporating the concept of uneven margins into Mixup-based data augmentation techniques. We adopt the common definition and define the margin of an example (x, y) as:

$$\gamma(x,y) = f(x)_y - \max_{j \neq y} f(x)_j. \tag{8}$$

The margin for class j is defined as the average margin of all examples in the class:

$$\overline{\gamma_j} = \frac{1}{n_j} \sum_{i:y_i=j} \gamma(x_j, y_j), \tag{9}$$

Recall that the optimal class-distribution-aware margin trade-off follows (1) [12]. Suppose that (x_i, y_i) and (x_j, y_j) are two samples of different classes. Define η_i as the distance from x_i to the decision boundary between class i and j, and define η_j similarly. Motivated by (1), we set:

$$\eta_i = 1 \,/\, n_i^{\omega};\ \eta_j = 1 \,/\, n_j^{\omega}. \tag{10}$$

We tune the hyper-parameter ω to strike the best trade-off in the proposed margin-aware Mixup. The sensitivity of this hyper-parameter is discussed in detail in the Appendix A.

The proposed margin-aware Mixup (MAMix) is formulated as:

$$\tilde{x}^{MAM} = \lambda_x x_i + (1 - \lambda_x) x_j \tag{11}$$

$$\tilde{y}^{MAM} = \lambda_y y_i + (1 - \lambda_y) y_j. \tag{12}$$

Note that here, λ_x and the Mixup-selected pair (x_i, y_i) and (x_j, y_j) are obtained as in the original Mixup, whereas we compute λ_y for each Mixup-selected pair based on the following formula, where $\lambda_y \in [0, 1]$:

$$\lambda_y = \begin{cases} 1 - \dfrac{(1-\lambda_x) \times 0.5}{\eta_i \,/\, (\eta_i + \eta_j)}\ , & \text{if } \lambda_x \geq \eta_j \,/\, (\eta_i + \eta_j) \\ \dfrac{(0.5) \times (\lambda_x)}{\eta_j \,/\, (\eta_i + \eta_j)}\ , & \text{if } \lambda_x < \eta_j \,/\, (\eta_i + \eta_j). \end{cases} \tag{13}$$

Therefore, with Algorithm 1, our proposed MAMix obtains mixed input by (11), mixed label by (12), and λ_y through (13). Essentially, we obtain the optimal mixing factor by $\eta_j \,/\, (\eta_i + \eta_j)$; note that η_i, η_j are obtained via (10). If the mixing factor λ_x is exactly the same as $\eta_j \,/\, (\eta_i + \eta_j)$, the probability to output this synthetic example should be exactly 50% for class i and 50% for class j. Also, if the mixing factor λ_x is larger or smaller than $\eta_j \,/\, (\eta_i + \eta_j)$, we compute its corresponding λ_y using (13), where the core idea is to use arithmetic progression to ensure that the classifier favors minority classes and therefore achieves uneven margins.

Recall that in original Mixup, the mixing factor λ_x is the same for synthetic x and y, that is, $\lambda_x = \lambda_y$. The core idea for Mixup to better account for class-imbalanced learning is by making *y not uniform*. Our proposed method achieves this by incorporating margin-aware concepts.

4 Experiments

4.1 Experiment Setup

We follow [12] in creating synthetic datasets for CIFAR-10 [27], CIFAR-100, and Tiny ImageNet. Additionally, our approach is aligned with the guidelines presented in [21] to ensure comprehensive coverage across CINIC-10 and SVHN datasets. To further enhance the depth of our study, we also examine CIFAR-10 and CINIC-10 datasets with extreme imbalance ratios to simulate extremely imbalanced scenarios. Furthermore, we adopt the protocol detailed in [12] with a fixed $\mu = 0.5$ for step imbalance.

Table 1 Top-1 validation accuracy (mean ± std) on long-tailed imbalanced CIFAR-10 with ratio ρ = 100 with ResNet32 using SMOTE and its two variants

Method	Accuracy
ERM	71.23 ± 0.51
SMOTE	72.68 ± 1.41
DRW	75.08 ± 0.61
M2m	76.15 ± 0.72
DeepSMOTE	76.66 ± 0.57
SMOTE-Mix–DRW	77.46 ± 0.64
Neighbor-Mix–DRW	80.44 ± 0.32
Mixup–DRW	82.11 ± 0.57

For a more comprehensive description about the dataset preparation, please refer to the Appendix B.

4.2 Compared methods

We compared our method with the baseline training methods: (1) Empirical risk minimization (ERM) loss, where we use standard cross-entropy loss with all examples sharing the same weights. (2) Deferred re-weighting (DRW), proposed by [12], where we train with standard ERM in the first stage and then apply re-weighting in the second stage with the final learning rate decay. (3) The margin-based state-of-the-art work of LDAM-DRW [12]. (4) The recent Mixup-based Remix [21]. Note that following the notation of [12], when two methods are combined, we abbreviate their acronyms with a dash. Our main proposed method is margin-aware Mixup (MAMix). For all experiments, we report the mean and standard deviation over 5 runs with different seeds. We computed the margin gap γ_{gap} (introduced in the Appendix A) on the validation sets and our proposed method was developed using PyTorch.

5 Results and Analysis

In this section, we discuss SMOTE-like techniques—SMOTE-Mix and Neighbor-Mix—for imbalanced deep learning.

When directly using SMOTE for oversampling, the performance gain from around 71% to 72% is not competitive enough (Table 1). Previous studies [12, 28] show that training with re-weighting or re-sampling based approaches is harmful for representation learning with deep models. Therefore, direct incorporation of SMOTE into deep learning achieves only limited performance improvements. However, SMOTE-Mix and Neighbor-Mix are effective when coupled with DRW (Table 1). Neighbor-Mix coupled with DRW achieves a greater performance improvement over SMOTE-Mix, whereas the performance of Neighbor-Mix is still inferior to that of Mixup, as demonstrated in Table 1, in which the performance difference lies in how to select the Mixup pair during training.

Motivated by the competitive results of SMOTE-Mix and Neighbor-Mix, we further relaxed Neighbor-Mix back to the original form of Mixup to examine the effectiveness of this approach on imbalanced data. Mixup is a modern data augmentation technique that is widely recognized to be effective in the deep image classification literature. However, the datasets are usually balanced; the effect of Mixup for imbalanced datasets

Table 2 Top-1 validation accuracy (mean ± std) on extremely long-tailed imbalanced CIFAR-10 using ResNet32

Imbalance ratio	200	250	300
Mixup–DRW	77.02 ± 0.53	76.33 ± 0.78	73.39 ± 0.47
Remix–DRW	77.23 ± 0.61	75.39 ± 0.72	73.79 ± 0.29
MAMix–DRW	**78.08 ± 0.23**	**76.34 ± 0.71**	**74.85 ± 0.29**
MAMix-Remix–DRW	78.01 ± 0.23	76.25 ± 0.63	74.87 ± 0.56

Table 3 Top-1 validation accuracy (mean ± std) on extremely imbalanced CINIC-10 using ResNet18

Dataset	Long-tailed	Step
Imbalance ratio	200	200
ERM	56.22 ± 1.46	52.01 ± 0.52
DRW	58.97 ± 0.30	57.87 ± 1.01
LDAM–DRW	63.09 ± 0.54	65.47 ± 0.63
Mixup–DRW	66.86 ± 0.50	65.61 ± 0.59
Remix–DRW	66.46 ± 0.51	66.61 ± 0.27
MAMix–DRW	**67.59 ± 0.37**	**67.34 ± 0.32**

has not been widely studied. Therefore, by simply applying Mixup on imbalanced learning settings, we expect to see improvement over a non-Mixup counterpart. For example, in long-tailed imbalanced CIFAR-10 with an imbalance ratio of $\rho = 100$, we can see that the top-1 validation accuracy improves from 72% to around 74% (Table 4) when applying Mixup, which is expected. However, when Mixup is deployed with DRW, the performance boosts from 72% to around 82% (Table 4) under the same setting, which exceeds the previous state-of-the-art result on imbalanced learning of LDAM-DRW [12]. The comprehensive results for imbalanced CIFAR-10 and CIFAR-100 are given in Tables 4 and 5; those for imbalanced CINIC-10 are given in Table 6. The detailed results for imbalanced SVHN and imbalanced Tiny-ImageNet, please refer to the Appendix B.

Note that Mixup-based methods demonstrate their highest efficacy when combined with DRW. Traditional re-weighting or re-sampling approaches have been shown to harm feature extraction when learning with imbalanced data [12, 28]. As a result, DRW provides a training scheme which first learns a good representation and further accounts for minority classes by re-weighting at later training stages.

In general imbalanced settings where the imbalance ratios are not extreme (e.g., $\rho < 200$), the original Mixup coupled with DRW already achieves competitive results, with the results among different Mixup-based approaches comparable to each other. However, our proposed MAMix outperforms the original Mixup and Remix in extremely imbalanced cases (e.g., $\rho \geq 200$), as demonstrated in Tables 2, and 3. When the imbalance ratio is extreme, our method consistently achieves results superior to those of Mixup and Remix, demonstrating the effectiveness of our method as well as the necessity of our algorithm in extremely imbalanced scenarios. Moreover, MAMix also serves as a general technique used to improve over Mixup or Remix; when deploying MAMix on top of Remix (MAMix–Remix–DRW in Table 2), there is also

Table 4 Top-1 validation accuracy (mean ± std) on imbalanced CIFAR-10 using ResNet32

Dataset	Long-tailed			Step		
Imbalance ratio	100	50	10	100	50	10
ERM	71.23 ± 0.51	77.33 ± 0.74	86.72 ± 0.36	65.64 ± 0.82	71.41 ± 1.21	85.02 ± 0.33
Mixup	74.03 ± 0.96	78.79 ± 0.16	87.79 ± 0.42	66.91 ± 0.74	72.84 ± 0.60	85.50 ± 0.37
Remix	75.18 ± 0.26	80.21 ± 0.26	88.36 ± 0.36	69.26 ± 0.48	74.50 ± 1.16	86.68 ± 0.38
MAMix	74.74 ± 0.76	80.00 ± 0.24	88.17 ± 0.15	68.24 ± 0.43	73.88 ± 0.35	85.91 ± 0.33
LDAM	74.01 ± 0.68	78.71 ± 0.38	86.43 ± 0.32	65.64 ± 0.52	72.37 ± 0.61	84.74 ± 0.26
DRW	75.08 ± 0.61	80.11 ± 0.67	87.52 ± 0.25	72.02 ± 0.59	78.17 ± 0.27	87.73 ± 0.15
M2m	76.15 ± 0.72	80.71 ± 0.17	88.01 ± 0.24	72.91 ± 0.90	79.12 ± 0.21	87.85 ± 0.11
DeepSMOTE	76.66 ± 0.57	80.60 ± 0.38	87.60 ± 0.25	72.47 ± 0.64	77.52 ± 0.42	87.33 ± 0.07
LDAM–DRW	77.75 ± 0.39	81.70 ± 0.22	87.67 ± 0.39	77.99 ± 0.65	81.80 ± 0.39	87.68 ± 0.38
Mixup–DRW	82.11 ± 0.57	85.15 ± 0.27	89.28 ± 0.23	79.22 ± 0.98	83.28 ± 0.50	89.24 ± 0.15
Remix–DRW	81.82 ± 0.14	84.73 ± 0.23	89.33 ± 0.36	80.31 ± 0.70	83.61 ± 0.24	89.10 ± 0.15
MAMix–DRW	**82.29 ± 0.60**	85.11 ± 0.32	89.30 ± 0.14	80.02 ± 0.27	83.47 ± 0.19	**89.29 ± 0.29**

Table 5 Top-1 validation accuracy (mean ± std) on imbalanced CIFAR-100 using ResNet32

Dataset	Long-tailed			Step		
Imbalance ratio	100	50	10	100	50	10
ERM	38.46 ± 0.36	43.51 ± 0.55	56.90 ± 0.13	39.56 ± 0.31	42.81 ± 0.21	55.09 ± 0.21
Mixup	40.69 ± 0.39	46.07 ± 0.60	59.63 ± 0.32	39.89 ± 0.10	41.09 ± 0.16	55.79 ± 0.35
Remix	42.46 ± 0.51	47.81 ± 0.48	60.71 ± 0.41	40.27 ± 0.18	42.97 ± 0.24	58.77 ± 0.23
MAMix	42.59 ± 0.22	47.89 ± 0.87	60.86 ± 0.55	40.02 ± 0.19	41.85 ± 0.44	57.39 ± 0.40
LDAM	40.49 ± 0.62	44.69 ± 0.37	56.06 ± 0.44	40.56 ± 0.29	43.11 ± 0.09	54.29 ± 0.41
DRW	40.40 ± 0.80	45.19 ± 0.49	57.23 ± 0.33	42.97 ± 0.24	46.78 ± 0.38	56.82 ± 0.38
M2m	41.92 ± 1.01	46.25 ± 0.15	58.34 ± 0.07	45.66 ± 0.02	49.54 ± 0.06	59.08 ± 0.22
DeepSMOTE	38.87 ± 0.19	44.70 ± 0.34	56.97 ± 0.25	42.27 ± 0.16	46.22 ± 0.39	55.45 ± 0.20
LDAM–DRW	41.28 ± 0.43	45.61 ± 0.41	56.42 ± 0.38	43.51 ± 0.61	46.81 ± 0.29	56.07 ± 0.30
Mixup–DRW	46.91 ± 0.46	51.75 ± 0.20	62.18 ± 0.24	47.56 ± 0.34	53.50 ± 0.47	62.91 ± 0.53
Remix–DRW	46.00 ± 0.48	51.16 ± 0.23	61.63 ± 0.25	48.91 ± 0.29	53.75 ± 0.26	62.47 ± 0.35
MAMix–DRW	**46.93 ± 0.24**	**51.92 ± 0.20**	**62.30 ± 0.33**	48.87 ± 0.36	**53.87 ± 0.62**	**62.84 ± 0.18**

improvement (Table 2). However, simple deployment of MAMix already yields superior results. We also discuss Mixup-based approaches [16], [21] and their effects on margin statistics compared with margin-based state-of-the-art work in LDAM [12]. A comprehensive analysis of it has been provided in detail in the Appendix A.

Additionally, we also compared our method with two additional methods: (1) Major-to-minor translation (M2m) [20]. (2) Fusing deep learning and SMOTE for imbalance data (DeepSMOTE) [18] to have a variety of comparisons. The results reveal that our proposed method performs better on all five datasets CIFAR10, CIFAR100, CINIC10 (Tables 4, 5, 6), SVHN, Tiny-ImageNet (in the Appendix B).

6 Conclusion

In this work, we are first to utilize margin statistics to analyze whether the model has learned a proper representation under a class-imbalanced learning setting from a margin perspective. We propose achieving uneven margins via Mixup-based techniques. We first show that coupled with DRW training, the original Mixup implicitly

Table 6 Top-1 validation accuracy (mean ± std) on imbalanced CINIC-10 using ResNet18

Dataset	Long-tailed			Step		
Imbalance ratio	100	50	10	100	50	10
ERM	61.08 ± 0.55	66.17 ± 0.37	77.64 ± 0.08	57.29 ± 0.73	62.26 ± 0.42	75.39 ± 0.30
DRW	63.75 ± 0.22	69.35 ± 0.35	78.66 ± 0.10	64.34 ± 0.25	68.73 ± 0.27	78.24 ± 0.21
M2m	64.20 ± 0.22	69.84 ± 0.41	78.67 ± 0.11	63.99 ± 1.25	69.82 ± 0.20	78.66 ± 0.03
LDAM–DRW	68.15 ± 0.22	72.34 ± 0.42	79.03 ± 0.17	70.09 ± 0.32	73.16 ± 0.48	79.07 ± 0.10
Mixup–DRW	71.40 ± 0.25	75.02 ± 0.16	81.36 ± 0.09	71.33 ± 0.23	74.74 ± 0.20	81.37 ± 0.18
Remix–DRW	71.15 ± 0.24	74.68 ± 0.09	81.27 ± 0.13	71.48 ± 0.50	74.91 ± 0.21	81.26 ± 0.08
MAMix–DRW	**71.76 ± 0.29**	**75.27 ± 0.17**	**81.46 ± 0.08**	**71.91 ± 0.23**	**75.26 ± 0.08**	**81.39 ± 0.08**

achieves uneven margins in general imbalanced multi-class classification. However, in the case of extreme data imbalance (for example, CINIC-10 with an imbalance ratio $\rho \geq 200$), the proposed margin-aware Mixup outperforms Mixup by explicitly controlling the degree of uneven margins, and also outperforms the proposed Remix [21]. Therefore, in practice, we suggest using the original Mixup for good results on general imbalanced tasks; for extremely imbalanced tasks, we offer the proposed method to better account for such data imbalance. In sum, our study connects SMOTE to Mixup in deep imbalanced classification, while shedding light on a novel framework that combines both traditional [8] and modern [16, 21] data augmentation techniques under the same umbrella. Future work is needed to examine the theoretical aspects of these Mixup-based approaches. With this method and our developed software, we hope that our work can serve as a starting point for future research in the community.

Declarations

- Funding: The work is mainly supported by the MOST of Taiwan under 107-2628-E-002-008-MY3.
- Conflicts of interest/Competing interests: n/a
- Ethics approval: n/a
- Consent to participate: n/a
- Consent for publication: n/a
- Availability of data and material: experiments are based on public benchmark data
- Code availability: released at open-source at https://github.com/ntucllab/imbalanced-DL
- Authors' contributions: Cheng contributes to detailed literature survey, the initial idea of studying Mixup for deep imbalanced classification, experimental comparison, code implementation and release and initial manuscript writing; Ha rigorously reviewed the code implementation, addressed issues and bugs, and expanded the scope of experimental comparison by incorporating additional methods like DeepSMOTE and M2m; Lin contributes to the bigger picture of linking SMOTE and Mixup, the initial idea of the margin-aware extension, and suggestions on the research methodology.

References

[1] Awoyemi, J.O., Adetunmbi, A.O., Oluwadare, S.A.: Credit card fraud detection

using machine learning techniques: A comparative analysis. In: 2017 (ICCNI), pp. 1–9 (2017). https://doi.org/10.1109/ICCNI.2017.8123782

[2] Roy, A., Sun, J., Mahoney, R., Alonzi, L., Adams, S., Beling, P.: Deep learning detecting fraud in credit card transactions. In: 2018 SIEDS, pp. 129–134 (2018). https://doi.org/10.1109/SIEDS.2018.8374722

[3] Horn, G.V., Perona, P.: The devil is in the tails: Fine-grained classification in the wild. CoRR **abs/1709.01450** (2017) https://arxiv.org/abs/1709.01450

[4] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014, pp. 740–755. Springer, Cham (2014)

[5] Zhong, Y., Deng, W., Wang, M., Hu, J., Peng, J., Tao, X., Huang, Y.: Unequal-training for deep face recognition with long-tailed noisy data. In: 2019 CVPR, pp. 7804–7813 (2019). https://doi.org/10.1109/CVPR.2019.00800

[6] Huang, C., Li, Y., Loy, C.C., Tang, X.: Deep imbalanced learning for face recognition and attribute prediction. TPAMI **42**(11), 2781–2794 (2020). https://doi.org/10.1109/TPAMI.2019.2914680

[7] Cui, Y., Jia, M., Lin, T.-Y., Song, Y., Belongie, S.: Class-balanced loss based on effective number of samples. In: CVPR (2019)

[8] Chawla, N., Bowyer, K., Hall, L.O., Kegelmeyer, W.: SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research **16**(1), 321–357 (2002)

[9] Haibo He, Yang Bai, Garcia, E.A., Shutao Li: ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: 2008 IEEE International Joint Conference on Neural Networks, pp. 1322–1328 (2008). https://doi.org/10.1109/IJCNN.2008.4633969

[10] Huang, C., Li, Y., Loy, C.C., Tang, X.: Learning deep representation for imbalanced classification. In: 2016 CVPR, pp. 5375–5384 (2016). https://doi.org/10.1109/CVPR.2016.580

[11] Liu, X., Zhou, Z.: The influence of class imbalance on cost-sensitive learning: An empirical study. In: ICDM'06, pp. 970–974 (2006). https://doi.org/10.1109/ICDM.2006.158

[12] Cao, K., Wei, C., Gaidon, A., Arechiga, N., Ma, T.: Learning imbalanced datasets with label-distribution-aware margin loss. In: NeurIPS (2019)

[13] Bej, S., Davtyan, N., Wolfien, M., Nassar, M., Wolkenhauer, O.: LoRAS: An oversampling approach for imbalanced datasets. CoRR **abs/1908.08346** (2019) https://arxiv.org/abs/1908.08346

[14] DeVries, T., Taylor, G.W.: Dataset augmentation in feature space. In: ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings (2017). https://openreview.net/forum?id=HyaF53XYx

[15] Inoue, H.: Data Augmentation by Pairing Samples for Images Classification (2018). https://openreview.net/forum?id=SJn0sLgRb

[16] Zhang, H., Cissé, M., Dauphin, Y.N., Lopez-Paz, D.: Mixup: Beyond empirical risk minimization. In: ICLR (2018). https://openreview.net/forum?id=r1Ddp1-Rb

[17] Mathew, J., Luo, M., Pang, C.K., Chan, H.L.: Kernel-based SMOTE for SVM classification of imbalanced datasets. In: IECON 2015, pp. 001127–001132 (2015). https://doi.org/10.1109/IECON.2015.7392251

[18] Dablain, D., Krawczyk, B., Chawla, N.: Deepsmote: Fusing deep learning and smote for imbalanced data. IEEE Transactions on Neural Networks and Learning Systems **PP**, 1–15 (2022). https://doi.org/10.1109/TNNLS.2021.3136503

[19] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Networks (2014). Accessed 2017-01-08

[20] Kim, J., Jeong, J., Shin, J.: M2m: Imbalanced classification via major-to-minor translation, pp. 13893–13902 (2020). https://doi.org/10.1109/CVPR42600.2020.01391

[21] Chou, H.-P., Chang, S.-C., Pan, J.-Y., Wei, W., Juan, D.-C.: Remix: Rebalanced mixup. In: arXiv (2020). 2007.03943

[22] Johnson, J., Khoshgoftaar, T.: Survey on deep learning with class imbalance. Journal of Big Data **6**, 1–54 (2019)

[23] Buda, M., Maki, A., Mazurowski, M.A.: A systematic study of the class imbalance problem in convolutional neural networks. Neural Networks **106**, 249–259 (2018). https://doi.org/10.1016/j.neunet.2018.07.011

[24] Khan, S.H., Hayat, M., Bennamoun, M., Sohel, F.A., Togneri, R.: Cost-sensitive learning of deep feature representations from imbalanced data. IEEE Transactions on Neural Networks and Learning Systems **29**(8), 3573–3587 (2018). https://doi.org/10.1109/TNNLS.2017.2732482

[25] Liu, W., Wen, Y., Yu, Z., Yang, M.: Large-margin softmax loss for convolutional neural networks. In: ICML, pp. 507–516 (2016)

[26] Wang, F., Cheng, J., Liu, W., Liu, H.: Additive margin softmax for face verification. IEEE Signal Processing Letters **25**(7), 926–930 (2018). https://doi.org/10.1109/LSP.2018.2822810

[27] Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images.

Master's thesis, Department of Computer Science, University of Toronto (2009)

[28] Ye, H.-J., Chen, H.-Y., Zhan, D.-C., Chao, W.-L.: Identifying and compensating for feature deviation in imbalanced deep learning. In: arXiv (2020). 2001.01385

[29] Reyzin, L., Schapire, R.E.: How boosting the margin can also boost classifier complexity. In: Proceedings of the 23rd International Conference on Machine Learning. ICML '06, pp. 753–760 (2006). https://doi.org/10.1145/1143844.1143939. https://doi.org/10.1145/1143844.1143939

[30] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 CPVR, pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90

[31] Yang, Y., Xu, Z.: Rethinking the value of labels for improving class-imbalanced learning. In: NeurIPS (2020)

Appendix A Margin Statistics Analysis

We discuss Mixup-based approaches [16], [21] and their effects on margin statistics compared with margin-based state-of-the-art work in LDAM [12].

A.1 Margin Perspectives

To better analyze and quantify the effect of different learning algorithms on the majority- and minority-class margins, we define the margin gap metric γ_{gap} as:

$$\gamma_{gap} = \frac{\sum_i n_i \cdot \overline{\gamma_i}}{\sum_i n_i} - \frac{\sum_j n_j \cdot \overline{\gamma_j}}{\sum_j n_j}, \tag{A1}$$

where i, j belong to majority and minority classes, respectively. To decide which class belongs to a majority class, and which belongs a minority class, we set a threshold: if the class sample numbers exceed $1\ /\ K$ of the total training samples, we categorize them as majority classes; the others are viewed as minority classes.

Hence a large margin gap corresponds to majority classes with larger margins and minority classes with smaller margins, and hence poor generalizability for the minority classes. We hope to achieve a smaller margin gap when given unbalanced classes. Note that this metric can be negative, as the margins for minority classes are larger than those of majority classes. To better determine whether this is a good indicator of the correlation between the margin gap and top-1 validation accuracy, we further evaluate with Spearman's rank order correlation ρ in Fig. A1.

A.1.1 Spearman's Rank Order Correlation

We demonstrate the results of analysis using Spearman's rank order correlation in Fig. A1. We note a negative rank order correlation between validation accuracy and margin gap γ_{gap}, as our definition of margin gap reflects the trend in which the better the model generalizes to the minority class, the lower the margin gap is. That is, better models produce smaller margin gaps between majority and minority classes. As seen in Fig. A1, Spearman's rank order correlation is -0.820, showing that although it is sometimes noisy, in general γ_{gap} is a good indicator for top-1 validation accuracy. Note that we will discuss the noisy part later in the next subsection.

A.1.2 Uneven Margin

Given the superior empirical performance of Mixup-based methods, we further analyzed this from a margin perspective to demonstrate the effectiveness of our method. First, we establish our baseline margin gap when the model is trained using ERM. Then, we examine the margin-based LDAM work in which larger margins are enforced for minority classes [12]. As seen in Table A1, the margin gap for ERM is the highest; that is, for deep models trained using ERM, majority classes tend to have higher margins than minority classes, resulting in poor generalizability for minority classes. LDAM-DRW [12] demonstrates its ability to shrink the margin gap, reducing the generalization error for the minority class through margin-based softmax training. Moreover, we observe that in long-tailed imbalance, the original Mixup alone yields competitive results, as the margin gaps are similar between the original Mixup, Remix,

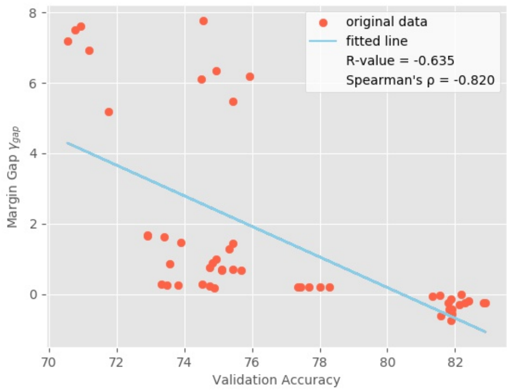


Fig. A1 Relationship between margin gap and validation accuracy for long-tailed imbalanced CIFAR-10 with imbalance ratio $\rho = 100$ using ResNet32

Table A1 Margin gap on imbalanced CIFAR-10 with $\rho = 100$ using ResNet32

Dataset	Long-tailed	Step
Imbalance ratio	100	100
ERM	7.645	8.515
DRW	6.089	7.086
LDAM–DRW	0.171	0.056
Mixup–DRW	-0.978	-0.481
Remix–DRW	-1.598	-1.870
MAMix–DRW	-1.136	-1.798

Table A2 Margin gap for extremely imbalanced CIFAR-10 with $\rho = 300$ using ResNet32

Method	Margin gap
Remix–DRW	-0.101
MAMix–DRW	-0.487

and our proposed method. This observation is consistent with Remix, for which similar performance is reported in a long-tailed imbalance setting. However, in a step imbalance setting, the superiority of our method is evident, as it not only achieves better performance but also shrinks the margin gap more than the original Mixup.

Note that in Table A1, we see that for the long-tailed scenario, the margin gap of Remix-DRW is -1.598 and that of MAMix-DRW is -1.136. However, as shown in Table 4, their respective validation accuracies are 81.82 and 82.29. This is an example of the noisy part that is mentioned in the previous context. Here Remix-DRW yields a smaller margin gap than that of MAMix-DRW but poorer validation accuracy, because Remix tends to enforce excessive margins in minority classes, whereas our method strikes a better trade-off.

Table A3 Margin decomposition on long-tailed imbalanced CIFAR-10 with $\rho = 100$ using ResNet32 (Majority: Class 0 to Class 2; Minority: Class 3 to Class 9)

Average Margin	$\gamma < 0$	$\gamma \geq 0$
Remix–DRW Majority	-1.587	2.371
MAMix–DRW Majority	-1.523	2.308
Remix–DRW Minority	-1.933	**4.891**
MAMix–DRW Minority	-1.875	**4.213**

To further study why excessive margins in minority classes do not help with validation accuracy, we first decompose the margins into two parts: $\gamma \geq 0$ and $\gamma < 0$ part, where validation accuracy is decided by the $\gamma < 0$ part ($\gamma < 0$ determines the validation error). The detailed decomposition result is in Table A3, where we take all $\gamma < 0$ margins and report the average among majority classes and minority classes for each method, and we compute $\gamma \geq 0$ part the same way. From our observation, $\gamma < 0$ part is generally similar between Remix and our MAMix, thus there is only slight accuracy difference, however, the $\gamma \geq 0$ part is generally higher for Remix, as we can see from Table A3. Therefore, the reason why in this case Remix has lower margin gap lies in the fact that it enforces more margins in $\gamma \geq 0$ part of minority classes, as we can see the $\gamma \geq 0$ part is 4.891 for Remix minority classes, and 4.213 for that of MAMix counterpart. From this observation, we identify that there seems to be *excessive margins* in minority classes for Remix, but—Do these excessive margins help or not ?—Previous research [29] has indicated that overly optimizing the margin may be an over-kill, in which the performance may be worse. We further answer this question by examining the difference between theoretical and practical margin distribution.

Recall that LDAM [12] derives a theoretically optimal ratio (1) for per class margin distribution, where such a ratio hints the need to *not over-push* the margin of minority classes. To further analyze how close the practical per class margin distribution of different methods are than that of theoretical margin distribution, we fit theoretical margin by practical margin, and since there is a constant multiplier C in theoretical margin, as in the form of (1), we choose to use linear regression without bias. We set $C = 1$ and compare the fitting (L_2) error in Table A4. As we can see from Table A4, our proposed MAMix shows the smallest L_2 error, hinting that the per class margin distribution produced by our method is the *closest* to the theoretical margin distribution derived by [12], while the per class margin distribution produced by Remix [21] is slightly inferior than ours in terms of L_2 error between theoretical and practical margin, which is due to the excessive margins in minority classes as shown in *Remix-DRW Minority* $\gamma \geq 0$ part in Table A3. Moreover, from Table A4 and Table 4, we observe that the closer practical margin is to theoretical margin, the higher the validation accuracy. Therefore, from the above evidence, we argue that we not only need to enforce larger margin for minority classes, but also need to not over-push minority margins, indicating the need for our method to strike for the better trade-off.

Note that in Table A2—the extremely imbalanced setting—our method brings the margin gap closer than Remix, verifying that our method consistently outperforms Remix.

Therefore, from a margin perspective, we first establish the baseline: when trained with ERM for imbalanced learning, the margins for majority classes are significantly larger than those for minority classes. Second, the recently proposed LDAM loss indeed

Table A4 L_2 error on long-tailed imbalanced CIFAR-10 with $\rho = 100$ using ResNet32

Method	L_2 Error
ERM	0.435
LDAM–DRW	0.195
Mixup–DRW	0.0133
Remix–DRW	0.0179
MAMix–DRW	**0.0126**

Table A5 Per Class Accuracy in long-tailed imbalanced CIFAR-10 with $\rho = 100$ using ResNet32

Method	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9
ERM	0.94	0.97	0.83	0.71	0.76	0.61	0.72	0.61	0.46	0.48
LDAM–DRW	0.95	0.97	0.79	0.73	0.82	0.69	0.78	0.70	0.63	0.66
MAMix–DRW	0.89	0.94	0.79	0.71	0.82	0.76	0.85	**0.81**	**0.79**	**0.82**

shrinks the margin gap significantly, suggesting that their approach is effective. To answer the original question—Can we achieve uneven margins for class-imbalanced learning through data augmentation?—the answer is positive, as we observe that applying the original Mixup implicitly closes the gap from a margin perspective, achieving comparable results. We further achieve uneven margins explicitly through the proposed MAMix.

A.1.3 Per Class Accuracy Evaluation

To further demonstrate the effectiveness of our proposed method, we can see from Table A5 for detailed per class accuracy evaluation. As we can see from Table A5, with ERM, the minority classes (i.e, C7,C8,C9), the accuracy for those classes are low, with C8 and C9 to be 0.46 and 0.48 respectively. And we can see that previous state-of-the-art in LDAM–DRW improved those two minority classes to 0.63 and 0.66. However, our proposed MAMix–DRW further elevated the per class accuracy of C8 and C9 and **0.79** and **0.82** respectively, without sacrificing the performance of the majority classes, which can be another evidence that shows the effectiveness of our algorithm.

A.1.4 Hyper-parameter ω in Margin-aware Mixup

As seen in Table A6, in the proposed MAMix, we can simply set ω to to 0.25, which is consistent with that suggested for LDAM [12]; however, the performance changes little when using different settings for ω, demonstrating that the proposed method is easy to tune.

Appendix B Implementation Details

B.1 Implementation Details for CIFAR

We followed [12] for CIFAR-10 and CIFAR-100. We also followed [12] to perform simple data augmentation described in [30] for training, where we first padded 4 pixels

Table A6 Sensitivity of ω in long-tailed extremely imbalanced CIFAR-10 with $\rho = 300$ using ResNet32

Method	Accuracy
MAMix–DRW ($\omega = 0.125$)	74.64 ± 0.17
MAMix–DRW ($\omega = 0.25$)	74.85 ± 0.28
MAMix–DRW ($\omega = 0.5$)	74.7 ± 0.75
MAMix–DRW ($\omega = 1.0$)	74.66 ± 0.36
MAMix–DRW ($\omega = 2.0$)	74.21 ± 0.56
MAMix–DRW ($\omega = 4.0$)	74.05 ± 0.50
MAMix–DRW ($\omega = 8.0$)	73.52 ± 0.52

on each side, then a 32 x 32 crop was randomly sampled from the padded image, or its horizontal flip. We also used ResNet-32 [30] as our base network. We trained the model with a batch size of 128 for 200 epochs. We use an initial learning rate of 0.1, then decay by 0.01 at the 160 and 180th epoch. We also use linear warm-up learning rate schedule for the first 5 epochs for fair comparison.

B.2 Implementation Details fo CINIC

We followed [21] for CINIC-10 where we used ResNet-18 [30] as our base network. As the training scheme provided by [21] we also trained the model for 200 epochs, with a batch size of 128, and initial learning rate of 0.1, followed by decaying the learning rate by 0.01 at the 160 and 180th epochs. We also use linear warm-up learning rate schedule. When DRW was deployed, it was deployed at the 160th epoch. When LDAM was used, we enforced the largest margin to be 0.5.

B.3 Implementation Details for SVHN

We followed [31] for SVHN. We adoped ResNet-32 [30] as our base network. We trained the model for 200 epochs, with initial learning rate of 0.1 and batch size of 128. We used linear warm-up schedule, and decay the learning rate by 0.1 at the 160th, and 180th epochs. When DRW was deployed, it was deployed at the 160th epoch. When LDAM was used, we enforced the largest margin to be 0.5.

The detailed results for imbalanced SVHN is given in Table B7.

B.4 Implementation Details for Tiny ImageNet

We followed [12] for Tiny ImageNet with 200 classes. For basic data augmentation in training, we first performed simple horizontal flips, followed by taking random crops of size 64 x 64 from images padded by 8 pixels on each side. We adopted ResNet-18 [30] as our base networks, and used stochastic gradient descent with momentum of 0.9, weight decay of $2 \cdot 10^{-4}$. We trained the model for 300 epochs, with initial learning rate of 0.1 and batch size of 128. We used linear warm-up rate schedule, and decay the learning rate by 0.1 at the 150th epoch and 0.01 at the 250th epoch. When DRW was deployed, it was deployed at the 240th epoch. When LDAM was used, we follow the original paper to enforce largest margin to be 0.5. Note that we cannot reproduce the numbers reported in [12].

Table B7 Top-1 validation accuracy (mean ± std) on imbalanced SVHN using ResNet32

Dataset	Long-tailed			Step		
Imbalance ratio	100	50	10	100	50	10
ERM	79.91 ± 0.67	83.42 ± 0.15	88.43 ± 0.22	76.38 ± 0.93	81.33 ± 1.11	87.89 ± 0.31
Mixup	81.57 ± 0.68	85.16 ± 0.48	90.75 ± 0.28	76.62 ± 1.03	82.88 ± 1.06	89.79 ± 0.61
Remix	82.37 ± 0.67	86.27 ± 0.41	91.07 ± 0.21	78.89 ± 1.30	83.57 ± 0.63	90.20 ± 0.45
Ours	82.39 ± 0.45	86.75 ± 0.37	91.09 ± 0.25	77.83 ± 1.87	83.91 ± 0.97	90.68 ± 0.32
LDAM	81.96 ± 0.69	85.31 ± 0.29	89.40 ± 0.36	77.93 ± 1.00	83.84 ± 0.62	89.45 ± 0.37
DRW	80.68 ± 0.32	83.66 ± 0.49	88.64 ± 0.26	76.33 ± 2.00	82.29 ± 1.17	88.18 ± 0.45
M2m	77.68 ± 0.45	82.25 ± 0.36	88.39 ± 0.38	76.10 ± 0.83	80.46 ± 1.96	87.84 ± 0.77
DeepSMOTE	81.12 ± 0.58	83.62 ± 0.55	88.06 ± 0.49	78.67 ± 0.88	82.08 ± 0.52	87.73 ± 0.19
LDAM–DRW	83.48 ± 1.11	86.17 ± 0.54	89.85 ± 0.26	79.24 ± 1.19	84.79 ± 0.65	90.11 ± 0.41
Mixup–DRW	85.19 ± 0.32	87.43 ± 0.63	90.14 ± 0.23	80.73 ± 1.72	87.32 ± 0.87	90.84 ± 0.24
Remix–DRW	84.52 ± 0.62	87.27 ± 0.37	90.11 ± 0.53	80.90 ± 1.96	87.09 ± 0.85	90.80 ± 0.23
MAMix–DRW	**85.41 ± 0.56**	**87.79 ± 0.45**	**90.59 ± 0.52**	**81.71 ± 1.28**	**87.62 ± 0.36**	90.57 ± 0.23

Table B8 Top-1 validation accuracy (mean ± std) on imbalanced Tiny-ImageNet using ResNet18

Dataset	Long-tailed		Step	
Imbalance ratio	100	10	100	10
ERM	32.86 ± 0.22	48.90 ± 0.43	35.44 ± 0.25	48.23 ± 0.13
DRW	33.81 ± 0.49	49.99 ± 0.27	37.79 ± 0.11	50.13 ± 0.30
M2m	34.33 ± 0.42	49.39 ± 0.63	37.02 ± 0.68	50.11 ± 0.24
LDAM	31.13 ± 0.36	46.90 ± 0.19	35.88 ± 0.09	47.91 ± 0.19
LDAM–DRW	31.90 ± 0.13	47.15 ± 0.31	36.75 ± 0.19	48.17 ± 0.16
Mixup–DRW	37.97 ± 0.38	52.51 ± 0.40	40.45 ± 0.21	54.46 ± 0.29
Remix–DRW	36.89 ± 0.61	52.13 ± 0.23	41.07 ± 0.37	53.58 ± 0.23
MAMix–DRW	37.73 ± 0.18	52.53 ± 0.34	**41.46 ± 0.38**	54.37 ± 0.29

The detailed results for imbalanced Tiny-ImageNet is given in Table B8.