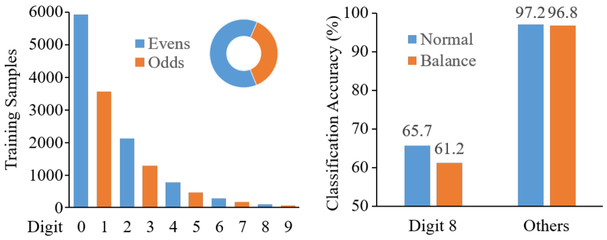
2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) | 978-1-6654-6946-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/CVPR52688.2022.00017

A Re-Balancing Strategy for Class-Imbalanced Classification Based on Instance Difficulty

Sihao Yu, Jiafeng Guo*, Ruqing Zhang, Yixing Fan, Zizhen Wang and Xueqi Cheng
University of Chinese Academy of Sciences, Beijing, China
CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology,
Chinese Academy of Sciences, Beijing, China
{yusihao,guojiafeng,zhangruqing,fanyixing,wangzizhen,cxq}@ict.ac.cn

Abstract

Real-world data often exhibits class-imbalanced distributions, where a few classes (a.k.a. majority classes) occupy most instances and lots of classes (a.k.a. minority classes) have few instances. Neural classification models usually perform poorly on minority classes when training on such imbalanced datasets. To improve the performance on minority classes, existing methods typically re-balance the data distribution at the class level, i.e., assigning higher weights to minority classes and lower weights to majority classes during the training process. However, we observe that even the majority classes contain difficult instances to learn. By reducing the weights of the majority classes, such instances would become more difficult to learn and hurt the overall performance consequently. To tackle this problem, we propose a novel instance-level re-balancing strategy, which dynamically adjusts the sampling probabilities of instances according to the instance difficulty. Here the instance difficulty is measured based on the learning speed of instance, which is inspired by the human-leaning process (i.e., easier instances will be learned faster). We theoretically prove the correctness and convergence of our re-sampling algorithm. Empirical experiments demonstrate that our method significantly outperforms state-of-the-art re-balancing methods on the class-imbalanced datasets.



(a) The Digit Distribution.

(b) Accuracy of Evens.

Figure 1. Binary Classification on the Long-Taied MNIST. Fig. 1a shows the digit and class distribution of the training data. Fig. 1b shows the validation accuracy of digits in the even class. ”Others” denotes the average accuracy of digit 0, 2, 4, 6. ”Normal” shows the results of base model. ”Balance” shows the results of the class-balance loss which reduces the weight of the majority class.

1. Introduction

Over the years, the performance of the classification has witnessed incredible progress on high-quality synthetic datasets, *e.g.*, CIFAR [40], ImageNet [36], MS-COCO [26]. However, the datasets in real-world applications often exhibit imbalanced data distributions [27, 30]. This imbalance is intuitively reflected by the sizes of different classes. On one hand, there are some classes that have a large number of instances. We call them majority classes in this paper. On the other hand, there are also some classes that have rarely few instances. We call them minority classes in this paper. Such class-imbalanced distributions pose critical challenges for neural classification models. Neural models perform with biases toward the majority classes when training on such datasets [1, 37]. Therefore, models usually perform poorly on the minority classes [7].

The class-imbalanced classification problem has attracted a lot of attention in the machine learning community [13, 27]. Researchers have introduced a variety of strategies to re-balance the data distribution when training the model. The mainstream solutions are re-sampling and re-weighting. Re-sampling methods directly adjust the training data by repeating the instances of minority classes and removing some instances of majority classes [6, 16, 41, 44]. The re-weighting methods focus on the cost(*e.g.*, loss) of different classes, specifically paying more attention to the minority classes' cost and less on the majority classes [5, 7]. In summary, existing methods typically consider and solve the imbalance problem at the class level by adjusting the observed class distribution.

[*] Corresponding author: Jiafeng Guo

978-1-6654-6946-3/22/$31.00 ©2022 IEEE
DOI 10.1109/CVPR52688.2022.00017

However, these class-level re-balancing strategies are too coarse to distinguish the difference of instances. Even within a majority class, there are some difficult instances for classification models to learn. By reducing the weights of the majority classes, such instances would be further ignored and become more difficult to learn, which hurts the models' performance. Fig. 1 gives a support example by a simulation experiment. A neural model needs to learn the binary classification of odd and even numbers on the digital pictures of 0-9(digits 0,2,4,6,8 are labeled as even, digits 1,3,5,7,9 are labeled as odd), whose distribution is shown as Fig. 1a. When testing the model on the validation set, we find that pictures of digit 8 only have a 65% probability to be right inferred as odd numbers. Compared with other even digits, digit 8 is badly learned by the model. If we adopt the class-balance loss [7] (*i.e.*, an effective class-level re-balancing method) in the training process, the weight of the majority class(*i.e.*, even) will be reduced. Compared with the 1% drop of other even digits, the accuracy of digit 8 has dropped significantly. It indicates that digit 8 should not be treated as same as other digits in the even class.

In the above case, adjusting the sub-class (*i.e.*, digit) distribution seems to be an effective solution. However, in most other cases, we do not know the labels of the sub-classes, and we even can not determine whether sub-classes exist. Moreover, even in a sub-class, the difficulty of different instances is also different. Therefore, we need to consider the weight adjustment at the instance level. At the instance level, we can not assign weights like existing class-level methods, because each instance usually appears only once. Even worse, the model's performance on the training set also can not reflect the difficulty of instances, because most instances in the training set can be correctly inferred after training.

However, different instances are learned at different speeds. Inspired by the process of human learning, the speed and difficulty of learning are usually strongly correlated. By the study of instance learning speed in the model's training process, we find that the instance learning process is directly affected by the data distribution. Specifically, instances have intermittent unlearning events during the learning process, which are performed as loss increments. Instances from the minority classes or minority sub-classes usually have more unlearning events in training. So these instances are learned more slowly. Therefore, identifying instances with slow learning speed as more difficult instances and increasing their weights in learning can effectively balance the data distribution.

Based on the above analyses, we design an instance difficulty model according to the learning speed and propose a novel instance-level re-balancing strategy for training classification models. Specifically, we record the predictions of each instance after each training epoch, and measure the instance difficulty based on the prediction variations. Then our method re-samples the data according to the instance difficulty model by assigning higher weights to difficult instances. In addition, we prove the correctness and convergence of our re-sampling strategy theoretically.

In this paper, we conduct some empirical experiments to show the multifaceted capabilities of our method. Specifically, the long-tailed classification experiments indicate that our strategy outperforms some strong baselines on the class-imbalanced datasets. Especially, we achieve new state-of-the-art results on the long-tailed CIFAR-10/-100 for image classification. The simulation experiments further verify the data re-balancing ability of our method, which reduces the imbalance ratio of labeled classes and unlabeled sub-classes. And the generality experiments show the generality of our methods on several different datasets.

The key contributions of this paper can be summarized as follows: (1) We demonstrate the pitfalls of popular class-level methods, and point out the importance of the instance-level distribution adjustment. (2) We theoretically propose a new difficulty definition for instances inspired by the learning speed, and we analyze the relationship of our difficulty and data distribution. (3) We propose an instance-level re-balancing strategy. It empirically performs well with theoretical proof.

2. Related Works

In this section, we briefly review two lines of related works: (1) Class re-balancing strategies. (2) Difficulty-sensitive methods.

2.1. Class Re-Balancing Strategies

From the intuitive experimental performance, the minority class performs poorly when the training data is class-imbalanced. So it is effective to directly re-balance the class contribution of the training data. To achieve this goal, there are two implementations: re-sampling [4, 30, 44] and re-weighting [5, 21, 38].

Re-sampling methods aim to achieve a more balanced data distribution by adjusting the frequency of instances during training. This adjustment is mainly at the class level, including the following three types. Over-sampling increases the instances of minority classes by repeating [6, 11] or interpolating [6, 12, 45]. Under-sampling [13, 16] removes some instances of majority classes to decrease the proportion of majority classes. And class-balance sampling methods [24, 28] not only increase the minority classes' frequency but also decrease the majority classes'.

Re-weighting methods balance the class distribution by adjusting the weights for training instances in the loss function. Specifically, allocating high weights for minority classes and low weights for majority classes. The core of this type of method is to determine the weights. One intuitive way is to use inverse class frequency as the weights of different classes [42]. Such a tough setting does not perform well in many situations, so [28,32] proposes some smoothed versions of the inverse class frequency. Inspired by the random covering problem, the class-balance loss models the effective number, replacing the class frequency, for each class [7].

Class-level adjustment is the relatively mainstream way to solve the class-imbalance problem in classification. However, the class-level approaches are too coarse, which ignores the differences between instances in the same class.

Besides, it is worthy to introduce existing work that has also explored other ways to solve the class-imbalanced classification. With the idea of transferring knowledge from majority classes to minority classes, transfer learning [2, 33, 43], two-stage training [5, 19] and dynamic curriculum learning [41] significantly increase the performance of class-imbalanced classification. Based on causal inference [34,35], removing the causal effect of bad gradient momentum can effectively relieve the bias to head class [39].

2.2. Difficulty-Sensitive Methods

Difficulty-sensitive methods adjust the data distribution to balance the difficulty. The adjustments usually are implemented by re-weighting, more difficult instances are assigned higher weights in training. So the core of these methods is to quantify the difficulty. Difficulty-sensitive methods typically focus on instance level. The most popular works quantify instances' difficulty in terms of the losses incurred by the model [8, 10, 25, 29]. Besides, meta-learning can be used to find the conditional weights for instances [15]. While our method presents a novel difficulty inspired by the learning speed. There is a strong connection between our difficulty and data distribution, so our method is more advantageous in solving the class-imbalanced problem. Moreover, when the difficulty is defined, additional controls are available beyond adjusting the data distribution. For example, curriculum learning [3] and self-paced learning [18, 23] not only adjust the distribution of the data, but focus more on the order in which the data appears from easy to difficult.

3. Our Method

In this section, we introduce our re-sampling method, which re-balances the data distribution by instance-level adjustments in training neural models for classification. Specifically, we introduce our method in 3 steps: (1) Task Formulation: we theoretically define the classification task and the optimization object of the model; (2) Re-sampling Framework: we introduce the role of our re-sampling strategy in the training framework; (3) Instance Difficulty Model: By theoretical analysis, we measure the difficulties of learning instances, which is the key of our method to assign weights for instances.

Algorithm 1 Re-Sampling

```
Input: dataset $\mathcal{S}$, network $Net$, training times T
Initialize sampling weight (probability) $\omega \leftarrow \{\frac{1}{|\mathcal{S}|}\}^{|\mathcal{S}|}$
Initialize $p_{i,0} \leftarrow \{\frac{1}{k}, \dots\}$ for each $\boldsymbol{x}_i$ in $\mathcal{S}$
for $t$ in $1$ to T do
    $\mathcal{S}^* \leftarrow$ Sample from $\mathcal{S}$ according to $\omega$
    Train $Net$ by using $\mathcal{S}^*$
    for $\boldsymbol{x}_i$ in $\mathcal{S}$ do
        $p_{i,t} \leftarrow Net(\boldsymbol{x}_i)$
        $D_{i,t} \leftarrow Difficulty(p_{i,0}, \dots, p_{i,t})$
    end for
    calculate new $\omega$ by $D$
end for
```

3.1. Task Formulation

Without loss of generality, a classification task with k classes are formed in this section. Let $\mathcal{S} := \{z_i = (\boldsymbol{x}_i, y_i) \colon 1 \leq i \leq N\}$ be the training data with N instances, where z_i denotes the i^{th} instance, $\boldsymbol{x}_i$ denotes its features and $y_i \in \{1, \dots, k\}$ denotes its class label. Then a neural model Net is adopted to fit the mapping of features to class labels. Suppose the final layer of Net is $softmax$, which normalizes the output of Net as a probability distribution for prediction. Specifically, we denote $p_i = Net(\boldsymbol{x}_i)$ as the prediction distribution of the instance z_i. $argmax(p_i) = y_i$ indicates that the instance z_i are correctly inferred by Net. To achieve higher inference accuracy, a suitable loss function $\mathcal{L}$ is adopted to help us learn the parameters θ of Net. Assume that $\mathcal{L}(\theta, \mathcal{S}) = \sum_{i=1}^{N} \mathcal{L}(\theta, z_i)$ is twice-differentiable [22]. The learning goal is to minimize the total loss $\mathcal{L}(\theta, \mathcal{S})$ by changing θ of Net.

3.2. Re-sampling Framework

The model optimizes the parameters by training on the dataset. However, even if the optimization method remains unchanged, the data distribution will greatly affect the learning of the model. Our method only adjusts the data distribution used in training to optimize the model. The overall training framework with our re-sampling method is shown in Algorithm 1.

The core of our re-sampling method is calculating the sampling weights for all instances. Different from the existing class-level methods, the probability of sampling each instance in our method can be different, even in the same class. With different difficulty models and weight calculation methods, the performance of final trained model is different. Inspired by the idea that difficult instances should be paid more attention to. In our method, we simply calculate the sampling weights as

$$w_{i,t} = \frac{D_{i,t}}{\sum_{j=1}^{N} D_{j,t}}, \tag{1}$$

where $w_{i,t}$ determines the sampling probability of the instance z_i after the t^{th} iteration. Our sampling method dynamically adjusts the sampling weight of each instance according to its current difficulty. So the difficulty model is the core of our method, which directly determines the sampling probability of each instance. Next, we will introduce it in detail.

3.3. Instance Difficulty Modeling

In this section, we introduce the instance difficulty model of our method through 3 steps: (1) Theoretical Analysis: To design the difficulty model, we make a theoretical analysis for the learning process of the instance. (2) Model Design: we design the instance difficulty model based on the analysis. (3) Characteristics: we explain our difficulty model in vector space and prove its convergence.

3.3.1 Theoretical Analysis

In this section, we analyze the reasons why an instance becomes difficult to learn. When using gradient descent to update θ, the goal of updates is to make $\mathcal{L}(\theta, \mathcal{S})$ smaller. According to the Taylor Expansion, when $\theta \to \theta_0$, $\mathcal{L}(\theta, \mathcal{S})$ can be approximated as

$$\mathcal{L}(\theta, \mathcal{S}) = \mathcal{L}(\theta_0, \mathcal{S}) + \mathcal{L}'(\theta_0, \mathcal{S})(\theta - \theta_0), \tag{2}$$

where $\mathcal{L}'(\theta_0, \mathcal{S}) = \sum_{i=1}^{N} \mathcal{L}'(\theta_0, z_i)$. To get the fastest descent speed, $\triangle\theta = (\theta - \theta_0) = -\eta\mathcal{L}'(\theta_0, \mathcal{S})$, where η denotes the learning rate.

Suppose that the parameters change from θ_0 to θ_1 after an update, and satisfies $\theta_1 = \theta_0 - \eta\mathcal{L}'(\theta_0, \mathcal{S})$. For a specific instance z, its loss will be changed after the parameter update. The variation which is $\mathcal{L}(\theta_1, z) - \mathcal{L}(\theta_0, z)$ can be estimated as

$$\triangle\mathcal{L}_z = -\eta\langle\mathcal{L}'(\theta_0, z), \mathcal{L}'(\theta_0, \mathcal{S})\rangle, \tag{3}$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product. The loss of the z will rise if $\langle\mathcal{L}'(\theta_0, z), \mathcal{L}'(\theta_0, \mathcal{S}\rangle < 0$. We call that z is unlearned in this update. In particular, we can construct two subsets of $\mathcal{S}$, which are the assistance set $\mathcal{A}_z \coloneqq \{a\colon a \in \mathcal{S}, \langle\mathcal{L}'(\theta_0, z), \mathcal{L}'(\theta_0, a)\rangle > 0\}$ and the hindrance set $\mathcal{H}_z \coloneqq \{r\colon r \in \mathcal{S}, \langle\mathcal{L}'(\theta_0, z), \mathcal{L}'(\theta_0, r)\rangle < 0\}$. The decline of instances' loss indicates the degree to which the instances are learned by the model. In the process of learning z, it is obvious that the instances in $\mathcal{A}_z$ provide the assistance but the instances in $\mathcal{H}_z$ create the hindrance. Moreover, when the weight of any instance in $\mathcal{A}_z$ is decreased or the weight of any instance in $\mathcal{H}_z$ is increased, the loss of z will become more difficult to reduce. It indicates that the difficulty of instances is affected by the data distribution.

As a naive idea, the instance difficulty for learning can be evaluated by all inner products of every two gradients. However, this naive method is too slow because of complex calculations. In fact, because of the twice-differentiable assumption, the gradient changes little when the model parameters are slightly perturbed. Between two adjacent iterations, the prediction variations of the model have similar trends when $\triangle\theta$ is slight. Therefore, prediction variations in the last iteration can be used to estimate the variations in the current iteration. Specifically, we denote Net_t as the model which has been trained t iterations, and $p_{i,t} = Net_t(\boldsymbol{x}_i)$ as the prediction distribution of Net_t for instance z_i. At the $t+1$ training iteration, we take the variation between $p_{i,t-1}$ and $p_{i,t}$ to estimate the learning results whether z_i tends to be learned or unlearned. If an instance is often unlearned, it will be difficult to be learned by the model. Obviously, such an instance will be learned more easily when its weight is increased.

3.3.2 Model Design

Following our analysis, to measure the difficulty, we estimate the prediction variations in the learning direction and the unlearning direction. And then we design the instance difficulty model. Specifically, for a given instance z_i, its difficulty after T iterations is estimated as

$$D_{i,T} = \frac{c + \sum_{t=1}^{T} du_{i,t}}{c + \sum_{t=1}^{T} dl_{i,t}}, \tag{4}$$

where c is the prior parameter of instance difficulty, $du_{i,t}$ denotes the prediction variation on the unlearning direction after t iterations, $dl_{i,t}$ denotes the prediction variation on the learning direction. In particular, all instances have the same c, which regulates the sensitivity of difficulty to prediction variations. In Eq. (4), the numerator records the accumulation of the unlearning trends, and the denominator records the accumulation of the learning trends. For any instances z_i and z_j, $D_{i,t} > D_{j,t}$ means z_i is more difficult than z_j so far, after t iterations have been updated. Consistent with the priori, all difficulties are treated as the same before the first iteration, since $D_{i,0} = 1$ for any instance z_i.

According to the different calculation methods of $du_{i,t}$ and $dl_{i,t}$, we can get different difficulty models. In this paper, we define them based on the PSI (*a.k.a.* Population Stability Index [17, 20]), which is a well defined index to measure the distance between distributions. Regardless of learning direction, the prediction variation between $p_{i,t-1}$ and $p_{i,t}$ is the distance that

$$d_{i,t} = \sum_{j=1}^{k}(p_{i,t}^{j} - p_{i,t-1}^{j})\ln(\frac{p_{i,t}^{j}}{p_{i,t-1}^{j}}), \tag{5}$$

where $p_{i,t}^{j}$ denotes the probability that Net_t predicts the class of z_i as j. Then we take into account the learning direction. Specifically, $p_{i,t}^{y_i} - p_{i,t-1}^{y_i} > 0$ indicates learning and $p_{i,t}^{y_i} - p_{i,t-1}^{y_i} < 0$ indicates unlearning. Moreover, there are similar settings in other dimensions of the probability distribution but the conclusion is opposite. Therefore, we define $du_{i,t}$ and $dl_{i,t}$ as

$$\begin{aligned} du_{i,t} = \min(p^{y_i}_{i,t} - p^{y_i}_{i,t-1}, 0) \ln(\frac{p^{y_i}_{i,t}}{p^{y_i}_{i,t-1}}) \\ + \sum_{j=1, j\neq y_i}^{k} \max(p^{j}_{i,t} - p^{j}_{i,t-1}, 0) \ln(\frac{p^{j}_{i,t}}{p^{j}_{i,t-1}}), \end{aligned} \tag{6}$$

and

$$\begin{aligned} dl_{i,t} = \max(p^{y_i}_{i,t} - p^{y_i}_{i,t-1}, 0) \ln(\frac{p^{y_i}_{i,t}}{p^{y_i}_{i,t-1}}) \\ + \sum_{j=1, j\neq y_i}^{k} \min(p^{j}_{i,t} - p^{j}_{i,t-1}, 0) \ln(\frac{p^{j}_{i,t}}{p^{j}_{i,t-1}}), \end{aligned} \tag{7}$$

which satisfy that $d_{i,t} = du_{i,t} + dl_{i,t}$.

Whenever the model is iterated after multiple batches of training such as an epoch, our method needs to infer the instances of training data once to record the prediction of the current model. By all records, *i.e.*, $(p_{i,0}, p_{i,1} \dots)$, the instance difficulty is calculated to adjust the sampling weight of instance z_i. Here $p_{i,0}$ can be the prediction before training or initialized as uniform distribution.

3.3.3 Model Characteristics

In this section, we discuss the characteristics of our instance difficulty model(*i.e.*, Eq. (4)), which directly controls the sampling weights. Our instance difficulty has a intuitive explanation in the vector space. $D_{i,T}$ is the slope of the difficulty vector $\vec{\boldsymbol{D}_{i,T}} = \vec{\boldsymbol{c}} + \sum_{t=1}^{T} \vec{\boldsymbol{d}_{i,t}}$, where $\vec{\boldsymbol{c}} = (c, c)$ and $\vec{\boldsymbol{d}_{i,t}} = (du_{i,t}, dl_{i,t})$. When our method tries to update the sampling weight, a new difficulty vector will be calculated for each instance. As illustrated in Fig. 2, the difficulty space is composed of the unlearning trend and the learning trend. If the model tends to unlearn an instance, the direction of its difficulty vector will closely point to the unlearning trend. Similarly, if the model tends to learn an instance, the direction of its difficulty vector will closely point to the learning trend.

In particular, for a single difficulty vector of an instance, its direction may be deviated due to the error of trend prediction. In our method, the overall trend summation makes the trend estimation more accurate. Generally, such accumulation can reduce the error in the direction of a single vector. In addition, the final results of our difficulty will be converged with the convergence of models. In an ideal situation, we prove that $||\vec{\boldsymbol{D}_{i,t-1}}|| = ||\vec{\boldsymbol{D}_{i,t}}||$ when $t \to \infty$. The specific proof are presented in the Appendix.

4. Experiments

In this section, we show the ability of our method by experiments, comparing with baselines introduced in Sec. 4.1. Then experiments are divided into three parts, according to different purposes: (1) Long-tail classification experiments test the performance of our method under different imbalance ratios. (2) Simulation experiments illustrate the rebalancing ability of our method. (3) Generality experiments demonstrate the generality of our method.

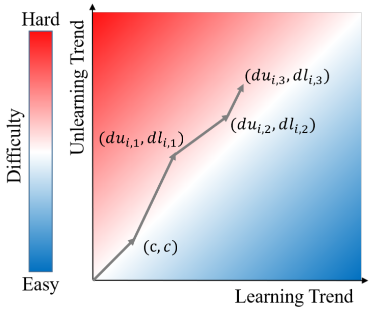


Figure 2. Instance Difficulty in Vector Space. Instance difficulty defined in Eq. (4) is presented by colors. The final difficulty is accumulated by period sliced difficulty vectors(grey solid arrow) calculated in different iterations.

4.1. Baselines

This section introduces the baseline methods used in the experiments.

- **Class-Balance Loss**: A class-level re-balancing method which adjusts the weights of classes based on effective numbers of classes. [7]. The effective number is related to the sample size of the category and regulated by the hyperparameter of the method.
- **Focal Loss**: An instance-level difficulty-sensitive method which assigns higher weights to instances in terms of losses. Every instance may have different weights [25].
- **TDE**: A state-of-the-art method that removes the accumulated preference of the majority class according to the causal effect in inference [39]. In particular, TDE does not adjust the data distribution during training.

Especially, we adopt different base models for different experiments. The baselines do not change the base model itself. They are a kind of additional adjustment method to optimize the learning of the given base model. Specifically, we adopt the ResNet [14] with different layers, the Multilayer Perceptron [9], and the logistic regression [31] as the base models in experiments.

4.2. Long-Tailed Classification Experiments

The intention of our method is to solve the class-imbalanced problem in classification. This section presents the performance of our method to solve the long-tailed classification problem with different imbalance ratios.

Table 1. Accuracy % on Long-Tailed CIFAR-10/-100 with Different Imbalance Ratios. All methods use the same network structure (the ResNet-32 backbone and a multi-head decision classifier).

Dataset Name	Long-tailed CIFAR 10				Long-tailed CIFAR 100			
Imbalance Ratio	1	20	50	100	1	20	50	100
Base Model	92.1	83.9	78.3	72.2	70.6	53.0	45.0	40.6
Focal Loss	92.2	83.8	78.2	72.6	70.8	53.1	45.6	41.0
Class-Balance Loss	92.1	84.1	79.1	74.3	70.6	54.9	46.1	41.1
Our Method	**93.8**	85.5	80.2	75.0	**71.5**	54.5	48.0	42.3
TDE	91.1	84.7	82.1	79.1	67.8	54.5	48.5	43.5
TDE + Our Method	93.5	**87.2**	**84.5**	**79.6**	70.5	**55.9**	**50.3**	**44.9**

4.2.1 Experimental Settings

To verify the superiority of our instance-level strategy, we conduct extensive studies on long-tailed CIFAR datasets [7] with various imbalance ratios. Specifically, the training instances of each class are reduced, according to an exponential function $n = n_i \mu^i$, where i is the class index, n_i is the original number of class i and $\mu \in (0, 1)$. And the overall imbalance ratio is denoted as $n_{\max}/n_{\min}$.

4.2.2 Performance of Different Imbalance Ratios

To show the performance of our method for solving class-imbalanced classification, we adopt experiments on long-tailed CIFAR-10/-100 with different imbalance ratios. As shown in Tab. 1, we can see that: (1) Training with different methods can differently improve the accuracy of the base model. (2) The improvement of the focal loss is not obvious, which indicates difficulty in terms of losses can not effectively solve the class-imbalanced classification. (3) Under the uniform class distribution (the imbalance ratio is 1), Class-Balance Loss performs the same as Base Model. Because it does not adjust any weight of class when the class distribution has been balanced. Under the imbalanced class distribution, performances of Class-Balance Loss are improved, because the distribution is re-balanced on class level. (4) Focus on the Focal Loss, Class-Balance Loss and our method, in the case of ensuring the model is the same, the final accuracy can reflect the performance of the different distribution adjustments. Our method outperforms the focal loss and class-balance loss at most situations, which shows the effectiveness of our strategy for re-balancing the distribution. (5) TDE only works on class-imbalanced situations, because TDE is based on causal analysis under the class-imbalanced assumption. Moreover, the effect of TDE is more obvious when the imbalance ratio is higher. Under a higher imbalance ratio, the model usually has a more obvious preference for majority classes. So the correction by TDE is larger and more accurate. (6) Compared with TDE, our method works better when the imbalance ratios are low. However, since TDE does not modify the data distribution, our method can be integrated with TDE. After fusion, our method can further improve the performance of TDE.

Table 2. Accuracy % on Long-Tailed CIFAR100 with Imbalance Ratio 100. The network is the same as that in Tab. 1. Class-Balance(More Minority) is another instance of Class-Balance Loss, which assigns much more weights for minority classes.

Methods	Majority	Minority	Overall
Base Model	54.1	9.0	40.6
Focal Loss	54.7	9.1	41.0
Class-Balance Loss	53.5	11.0	41.1
Class-Balance(More Minority)	49.3	**12.2**	38.2
Our Method	**56.2**	9.9	**42.3**

4.2.3 Performance on Majority and Minority Classes

To study the model performance in detail, we observe the performance on majority and minority classes in the experiment of long-tailed CIFAR-100 whose imbalance ratio is 100. Specifically, following the previous study [39], the 30 classes with the least number of instances are defined as the minority classes. They only accounted for 2.9% of all data in training. Then we denote the rest as the majority classes.

As shown in Tab. 2, different strategies have their own characteristics when improving the base model. We can see that: (1) Focal Loss mainly improves the performance on majority classes, since Focal Loss is not forced to assign higher weights to minority classes. This indicates that only adopting the values of losses to determine the difficulties of samples is hard for the model to perceive the minority classes. (2) Class-Balance Loss improves the performance on minority classes but slightly deteriorates the performance on majority classes. As a typical class-level method, it directly adjusts the weights of classes. As expected, the performance on the majority classes deteriorates because their weights are reduced. On the contrary, the weights of the minority classes are increased, so the performance on the minority classes is improved. Since

Table 3. Accuracy % on Long-Tailed MNIST with Imbalance Ratio 100 for Simulation Binary Classification. All methods use the same network structure. The Base Model is the Multilayer Perceptron. Two ratio columns present the class imbalance ratio(*i.e.*, "Class Ratio") and the sub-class imbalance ratio(*i.e.* "Sub-Class Ratio"). Especially, the values in these two columns are the imbalance ratios that calculated after re-balancing. The "Major" and "Minor" represent the majority sub-classes and the minority sub-classes within a class.

Methods	Class Ratio	Sub-Class Ratio	Overall	Even (Majority Class)			Odd (Minority Class)		
				Overall	Major	Minor	Overall	Major	Minor
Base Model	1.7	100.0	86.04	90.95	98.88	85.54	81.28	96.92	70.09
Class-Balance Loss	**1.2**	69.6	86.19	89.77	98.73	83.66	82.72	97.11	72.44
Our Approach	1.4	**37.3**	**87.99**	**92.33**	**99.08**	**87.73**	**83.78**	**97.41**	**77.43**

the performance improvement on minority classes is greater than the performance loss of majority ones, the overall performance is improved. (3) Class Balance(More Minority) performs better on minority classes, since it assigns higher weights for minority classes. However, lower weights for majority classes result in severe performance degradation on majority classes, thus the final performance is even less than the base model which does not adjust the data distribution. (4) Our method can effectively improve both majority and minority classes. Because our method pays more attention to the instances that are really hard to learn. Difficult instances in the majority classes are strengthened, so the performance on majority classes is improved. Although the improvement in the minority classes is not as good as Class-Balance Loss, our approach has a better performance overall. Compared with Focal Loss, our method can better perceive the importance of minority classes and is more effective in improving the majority classes.

4.3. Simulation Experiments

In this section, we show the performance of our method for unlabeled imbalanced sub-classes. Moreover, we verify the re-balancing ability of our method and analyze why our method can re-balance the distribution of classes and unlabeled sub-classes.

4.3.1 Experimental Settings

The simulation experiments are carried out by the task of the binary classification on Long-Tailed MNIST. The two classes are even and odd. Original MNIST is a popular dataset of handwritten digit recognition. We constructed a long-tailed version of MNIST by under-sampling according to the construction method of Long-tailed CIFAR [7]. From the perspective of the class level, the distribution is long-tailed, since the even class is much larger than the odd class. From the perspective of the sub-class (digit) level, the sub-class distribution in each class obeys the long-tailed either. In the training process, the model knows the class labels, but not the sub-class labels. For a more specific introduction and settings, please refer to the Appendix.

4.3.2 Performance on Imbalanced Sub-Classes

To observe the performance of our method on unlabeled sub-classes, we conduct the simulation experiment, whose class distribution and unlabeled sub-class distribution are imbalanced. The overall results are summarized in Tab. 3. As an illustration, the majority sub-classes of the even class are composed of the digit 0 and 2, while the minority sub-classes are composed of the numbers 4,6,8. The majority sub-classes of the odd class are composed of 1 and 3, while the minority sub-classes are composed of 5,7,9.

As shown in Tab. 3, we can see that: (1) On the class level, the performance is consistent with the results in Tab. 1, which leads to a similar conclusion. (2) The performance on minority sub-classes is much smaller than that on majority sub-classes, which indicates that the imbalanced issue also exists in unlabeled sub-classes. (3) The class-balance Loss performs more poorly on minority sub-classes of the majority class(*i.e.*, Even). Because re-balancing the distribution on the class level leads to lower weights for such minority sub-classes. (4) Our method has a more significant improvement on minority sub-classes, compared to the class-level adjustment, which indicates the superiority of instance-level adjustments. The difficulty defined by our method can perceive the problems caused by the imbalanced distribution of unlabeled sub-classes.

In addition, we recalculated the current imbalance ratio based on the weighted result of instances. The results (two ratio columns in Tab. 3) show that our method effectively reduces the imbalance ratio for both labeled classes and unlabeled sub-classes.

4.3.3 Analysis for Re-Balancing the Class Distribution

Because of the connection between our difficulty model and the data distribution, our method can re-balance the distribution of the classes or even the unlabeled sub-classes. In this section, we verify this conclusion by simulation experiments. Specifically, we record the unlearning frequency of different classes and sub-classes, and visualize the relationship between unlearning frequency and difficulty.

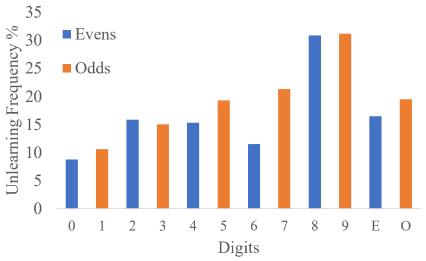
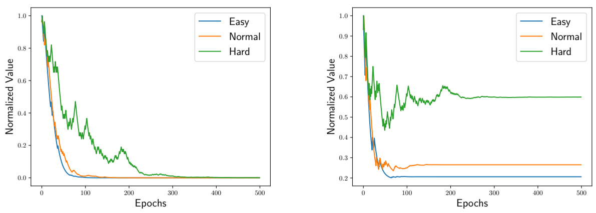


Figure 3. Unlearning Frequency of Classes and Sub-Classes. E denotes the even class. O denotes the odd class.



(a) Loss

(b) Difficulty

Figure 4. Loss and Difficulty of Instances in Training. ”Easy” is unlearned with 10% probability, ”Normal” is unlearned with 20% probability, ”Hard” is unlearned with 40% probability.

As the analysis in Sec. 3.3, the instance sometimes is unlearned during the learning process. Fig. 3 presents the unlearning frequency of different classes and sub-classes. We can see that rarer classes or sub-classes typically have a higher unlearning frequency. It indicates there is a strong correlation between the data distribution and the unlearning frequency. Then we conduct simulations for instances with different unlearning probabilities (see the Appendix for specific settings). Fig. 4 presents the variation of loss and difficulty for three types of instances which are unlearned with different probabilities. We can see that our instance difficulty is consistent with the unlearning frequency. Instances with higher unlearning frequency will receive higher difficulties and higher weights. Therefore, our method can re-balance the distribution of classes or sub-classes.

4.4. Generality Experiments

To demonstrate the generality of our method, we perform our method on various classification datasets. Specifically, the datasets include ten tiny classification datasets in the UCI machine learning repository, whose results are shown in Tab. 4. The results show that our method can steadily improve the performance of the base models on tiny datasets. Compared with the class-level method, our instance-level method is more effective. The detailed information of datasets is introduced in the Appendix. Moreover, we also evaluate our method on a large-scale dataset named iNaturalist 2019, our method also is effective to improve the performance of models. Specifically, our method improves the accuracy of the 50-layer ResNet from 70.19% to 71.08% and the accuracy of the 101-layer ResNet from 72.84% to 73.32%. The completed experimental results are shown in the Appendix. In conclusion, these results illustrate the generality of our method.

Table 4. Accuracy % on 10 Datasets. All methods use the same network structure. The ”Base” here is the Logistic Regression. ”CB” denotes the class re-balance loss [7].

Methods	Base	CB	Ours
Sonar	83.3	83.3	**85.7**
Balance	91.2	92.0	**92.8**
CMC	59.0	60.0	**62.4**
Ecoli	82.4	**85.3**	**85.3**
Glass	34.9	44.2	**53.5**
Heart	**72.2**	**72.2**	**72.2**
Iris	93.3	93.3	**96.7**
Robot	93.5	94.0	**94.1**
Seeds	**97.6**	**97.6**	**97.6**
Wine	**41.7**	**41.7**	**41.7**
Average	74.9	76.4	**78.2**

5. Conclusions and Future Works

In this paper, we studied the class-imbalanced classification problem from a more general instance-level view. Inspired by the idea that learning speed reflects the learning difficulty, we designed an instance difficulty model and presented a novel instance-level re-sampling strategy. Our method can re-balance the distribution of classes and unlabeled sub-classes. Moreover, our method achieved state-of-the-art results on the long-tailed benchmarks. In particular, this paper analyzed the relationship between the instance difficulty and the data distribution. Following our analysis, variant methods can be designed. The method of estimating the difficulty in this paper needs more computation, and the performance would be destroyed when there are wrong labeled instances. For such limitations, we presented more progress of our method and ideas for future works in the Appendix. In future works, we hope to design more efficient and robust variants for class-imbalanced classification.

6. Acknowledgement

This work was funded by the National Natural Science Foundation of China (NSFC) under Grants No. 62006218, 61902381, and 61872338, the Youth Innovation Promotion Association CAS under Grants No. 20144310, and 2021100, the Lenovo-CAS Joint Lab Youth Scientist Project, and the Foundation and Frontier Research Key Program of Chongqing Science and Technology Commission (No. cstc2017jcyjBX0059).

References

[1] K. Ahuja, J. Wang, A. Dhurandhar, K. Shanmugam, and K. R. Varshney. Empirical or invariant risk minimization? a sample complexity perspective. 2020. 1

[2] Samy Bengio. Sharing representations for long tail computer vision problems. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 1–1, 2015. 3

[3] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009. 3

[4] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning*, pages 872–881. PMLR, 2019. 2

[5] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems*, 2019. 1, 2, 3

[6] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority oversampling technique. *Journal of Artificial Intelligence Research*, 16(1):321–357, 2002. 1, 2

[7] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, 2019. 1, 2, 3, 5, 6, 7, 8

[8] Q. Dong, S. Gong, and X. Zhu. Class rectification hard mining for imbalanced deep learning. *IEEE*, 2017. 3

[9] Mwgr Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric Environment*, 1998. 5

[10] Yoav Freund and Robert Elias Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 1997. 3

[11] Hui Han, Wen Yuan Wang, and Bing Huan Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *Proceedings of the 2005 international conference on Advances in Intelligent Computing - Volume Part I*, 2005. 2

[12] Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, 2008. 2

[13] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009. 1, 2

[14] K. He, X. Zhang, S. Ren, and S. Jian. Identity mappings in deep residual networks. *Springer, Cham*, 2016. 5

[15] M. A. Jamal, M. Brown, M. H. Yang, L. Wang, and B. Gong. Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[16] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002. 1, 2

[17] Harold Jeffreys. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London*, 186(1007):453, 1946. 4

[18] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. 3

[19] 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 (ICLR)*, 2020. 3

[20] Grigoris Karakoulas. Empirical validation of retail credit-scoring models. *Rma Journal*, (Sept), 2004. 4

[21] Salman H Khan, Munawar Hayat, Mohammed Bennamoun, Ferdous Sohel, and Roberto Togneri. Cost sensitive learning of deep feature representations from imbalanced data. *IEEE Transactions on Neural Networks & Learning Systems*, 29(8):3573–3587, 2018. 2

[22] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894. PMLR, 2017. 3

[23] M Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. *Advances in neural information processing systems*, 23, 2010. 3

[24] Shen Li, Zhouchen Lin, and Qingming Huang. Relay backpropagation for effective learning of deep convolutional neural networks. In *European Conference on Computer Vision*, 2016. 2

[25] Tsung Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, 2017. 3, 5

[26] Tsung Yi Lin, Michael Maire, Serge Belongie, James Hays, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 1

[27] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X. Yu. Large-scale long-tailed recognition in an open world. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[28] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018. 2, 3

[29] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *International Conference on Computer Vision*, 2011. 3

[30] Buda Mateusz, Maki Atsuto, and Maciej A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, pages S0893608018302107–, 2017. 1, 2

[31] S. Menard. Logistic regression. *American Statistician*, 58(4):364, 2004. 5

[32] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*, 2013. 3

[33] Wanli Ouyang, Xiaogang Wang, Cong Zhang, and Xiaokang Yang. Factors in finetuning deep model for object detection with long-tail distribution. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3

[34] Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. *Causal inference in statistics: A primer*. John Wiley & Sons, 2016. 3

[35] Judea Pearl and Dana Mackenzie. The book of why : the new science of cause and effect. *Science*, 361(6405):855.2–855, 2018. 3

[36] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 1

[37] S. Sinha, H. Ohashi, and K. Nakamura. Class-wise difficulty-balanced loss for solving class-imbalance. In *Computer Vision – ACCV 2020*, 2021. 1

[38] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11662–11671, 2020. 2

[39] Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. Long-tailed classification by keeping the good and removing the bad momentum causal effect. In *NeurIPS*, 2020. 3, 5, 6

[40] A Torralba, R Fergus, and W. T Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 30(11):1958–1970, 2008. 1

[41] Yiru Wang, Weihao Gan, Jie Yang, Wei Wu, and Junjie Yan. Dynamic curriculum learning for imbalanced data classification. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2020. 1, 3

[42] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 7032–7042, 2017. 3

[43] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Feature transfer learning for deep face recognition with under-represented data. *arXiv preprint arXiv:1803.09014*, 2018. 3

[44] B. Zhou, Q. Cui, X. S. Wei, and Z. M. Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2

[45] Yang Zou, Zhiding Yu, BVK Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European conference on computer vision (ECCV)*, pages 289–305, 2018. 2