Long-Tailed Learning as Multi-Objective Optimization

Weiqi Li[1], **Fan Lyu**[1,2], **Fanhua Shang**[1], **Liang Wan**[1], **Wei Feng**[1*]

[1]College of Intelligence and Computing, Tianjin University
[2]CRIPAC, MAIS, CASIA
{wicky, fhshang, lwan}@tju.edu.cn, fan.lyu@cripac.ia.ac.cn, wfeng@ieee.org

Abstract

Real-world data is extremely imbalanced and presents a long-tailed distribution, resulting in models biased towards classes with sufficient samples and performing poorly on rare classes. Recent methods propose to rebalance classes but they undertake the seesaw dilemma (what is increasing performance on tail classes may decrease that of head classes, and vice versa). In this paper, we argue that the seesaw dilemma is derived from the gradient imbalance of different classes, in which gradients of inappropriate classes are set to important for updating, thus prone to overcompensation or undercompensation on tail classes. To achieve ideal compensation, we formulate long-tailed recognition as a multi-objective optimization problem, which fairly respects the contributions of head and tail classes simultaneously. For efficiency, we propose a Gradient-Balancing Grouping (GBG) strategy to gather the classes with similar gradient directions, thus approximately making every update under a Pareto descent direction. Our GBG method drives classes with similar gradient directions to form a more representative gradient and provides ideal compensation to the tail classes. Moreover, we conduct extensive experiments on commonly used benchmarks in long-tailed learning and demonstrate the superiority of our method over existing SOTA methods. Our code is released at https://github.com/WickyLee1998/GBG_v1.

Introduction

Deep learning has significantly progressed and has been widely applied in many applications (Li et al. 2022). Most of these excellent achievements rely on large and relatively balanced datasets like ImageNet (Deng et al. 2009). However, real-world data is often extremely imbalanced, presenting a long-tailed distribution. Training on long-tailed data usually results in a serious bias towards classes with sufficient samples (head classes) and performs poorly on rare classes (tail classes), giving rise to the field of long-tailed learning.

To address the problem of learning in long-tailed distribution, recent progress in long-tailed learning can be categorized into three groups. First, the class-rebalancing methods (Kang et al. 2019) increase the importance of tail classes via resampling or reweighting directly. Second, the decoupling methods (Zhou et al. 2020) use a two-stage training scheme to balance the classifier after observing from a pre-training phase. Third, the representing methods (Cao et al. 2019) design specific loss functions to achieve inter-class sparsity and a more balanced feature distribution. To sum up, the key consensus of these methods is to improve the importance of tail classes in long-tailed training. However, the existing rebalancing (Kang et al. 2019; Sun et al. 2022)

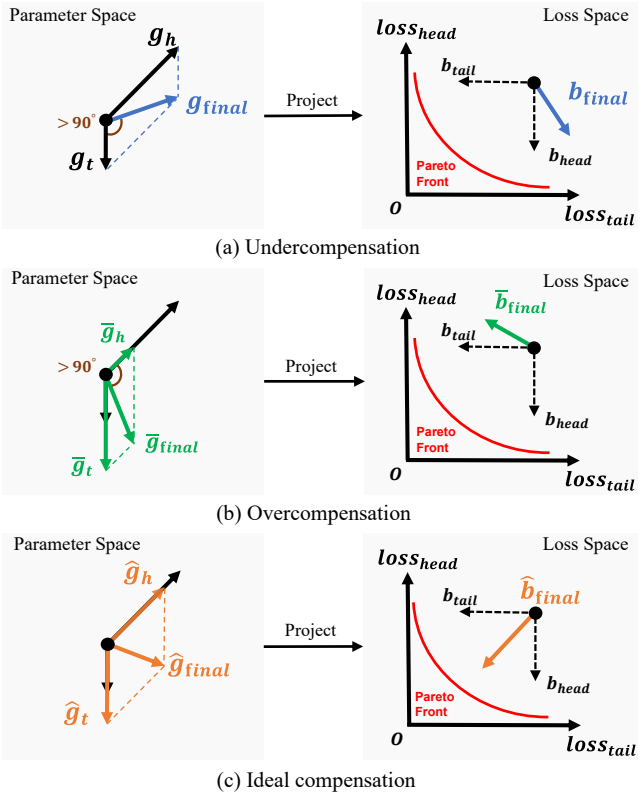

Figure 1: Three gradient compensation scenarios in reweighting for two-class imbalance training. By optimizing two classes simultaneously at each step, an ideal gradient should step towards the Pareto front without harming both classes, which means the loss descent (b_{final}) should be suited between two class-independent loss descent directions (b_{head} and b_{tail}).

*Corresponding author

methods aiming to increase the importance of tail-class gradients, may suffer from the *seesaw dilemma*. That is, to increase performance on tail classes may decrease that of head classes, and vice versa.

In this paper, we study the seesaw dilemma from the perspective of gradient imbalance in long-tailed learning, and we observe that the tail-class gradients are suppressed by those of head classes. Under this observation, an inappropriate weighting scheme may lead to the *overcompensation* or *undercompensation* on the gradient of tail classes. In general, undercompensation refers to a bias towards head class learning and overcompensation refers to the over-bias towards learning tail classes. Taking an imbalanced two-class classification as an example, we illustrate the effects of different compensations in Fig. 1. By projecting from parameter space to loss space, we find that undercompensation may result in insufficient learning (Fig. 1(a)) for tail classes, while overcompensation may hinder the learning of head classes (Fig. 1(b)). Ideally, *a feasible compensation to the gradients in a long-tailed problem should maintain a Pareto descent direction (Harada, Sakuma, and Kobayashi 2006), which should never damage any classes in the imbalanced distribution*, as shown in Fig. 1(c).

To achieve feasible compensations in the seesaw dilemma, we propose formulating the long-tailed learning into a multi-objective optimization problem (MOO), where each class holds its class-level training empirical loss. In this way, our goal is to find a compromising gradient that does not damage any of these losses at each update from a set of class-level gradients. Furthermore, it is impractical to extract gradients for every class independently in each mini-batch training because of two reasons. On one hand, more classes lead to more computation time and may also cause out-of-memory problem. On the other hand, a limited batch size can not guarantee access to each class, especially tail classes leading to incompleteness of objectives.

Accordingly, we develop a Gradient-Balancing Grouping (GBG) algorithm to present a batch-level gradient balance in long-tailed learning. Specifically, we first compute the gradients of all classes and obtain the gradient similarity between classes to build a similarity matrix. Then, we learn to group the classes with similar gradients according to the similarity matrix. To obtain a balanced gradient to guarantee the Pareto descent, inspired by the classic multi-gradient descent algorithm (Sener and Koltun 2018), we take the bundled gradients from each group as a min-norm optimization, which can be solved easily via quadratic programming.

Our main contributions are three-fold:

(1) To the best of our knowledge, it is the first time that we have formulated long-tailed recognition as a multi-objective optimization problem, to address the seesaw dilemma for the head and tail classes in previous methods.

(2) We propose a grouping method based on gradient similarity to solve the multi-objective optimization efficiently without compromising accuracy.

(3) Our method is validated to outperform the state-of-the-arts on the benchmarks datasets including CIFAR10/100-LT, ImageNet-LT and INaturalist2018, which demonstrate its capability in solving long-tailed problems efficiently.

Related Work

Long-tailed Learning via Class-Rebalancing. Class-Rebalancing includes resampling and reweighting. Resampling strategies aim to attain a balanced training data distribution. They use over-sampling (Buda, Maki, and Mazurowski 2018) to enlarge the instance number of tail classes or use under-sampling (He and Garcia 2009) to decrease that of head classes. However, they afford the risks of overfitting tail classes or impairing model generalization. Reweighting methods assign weights to the loss functions of each class that are negatively correlated with their sample sizes, aiming to balance the gradient contribution of different classes (Jamal et al. 2020). However, inappropriate weights used in reweighting methods may cause problems such as underfitting or overfitting to the model.

Long-tailed Learning via Grouping Strategy. Grouping strategies decompose long-tailed classification problem into multi-task problem or multi-level recognition problem by grouping the label sets according to certain rules (Yang et al. 2022) such as grouping based on instance numbers of classes (Li et al. 2020). Though current grouping strategies can avoid tail categories being suppressed to some extent, they could not solve the problem of knowledge interaction blocking between different groups (Yang et al. 2022).

Multi-Objective Optimization in Deep Learning. Multi-Objective Optimization (MOO) refers to optimizing multiple objective functions which may be conflicting in optimization problems (Lyu et al. 2021, 2023). The target of MOO is to find a set of optimal solutions that can simultaneously optimize multiple objectives. MOO can be applied to fields that require simultaneously optimizing multiple targets such as multi-task learning (Sener and Koltun 2018; Chen et al. 2023) and recommendation systems (Geng et al. 2015). In this paper, we use MOO to balance the learning of head classes and tail classes.

The Proposed Method

Gradient Imbalance Problem in LT Learning

Let $\mathcal{D}=\{(x_i, y_i), \cdots, (x_N, y_N)\}$ denote a long-tailed training set, with N samples and K classes in total. Long-tailed classification aims to learn a function $f(\boldsymbol{\theta})$ with parameters θ to predict each test sample correctly. For a data point (x_i, y_i), x_i represents the *i*-th data point in the training set and y_i represents its ground-truth label. In general, the model will be trained using an empirical risk loss as follows:

$$L(\mathbf{x}, \mathbf{y}) = \frac{1}{N}\sum_{i=1}^{N} L(x_i, y_i) = -\frac{1}{N}\sum_{i=1}^{N} \log\left(\frac{e^{z_{y_i}}}{\sum_{j=1}^{K} e^{z_j}}\right), \tag{1}$$

where z_j is the predicted logit of class j and z_{y_i} is the logit of the corresponding ground-truth class.

To explore the gradient imbalance in long-tailed learning, we split the training loss into head and tail losses as follows:

$$L(\mathbf{x}, \mathbf{y}) = \frac{1}{N}\left[\sum\nolimits_{i=1}^{N_{\text{tail}}} L(x_i, y_i) + \sum\nolimits_{j=1}^{N_{\text{head}}} L(x_i, y_i)\right], \tag{2}$$

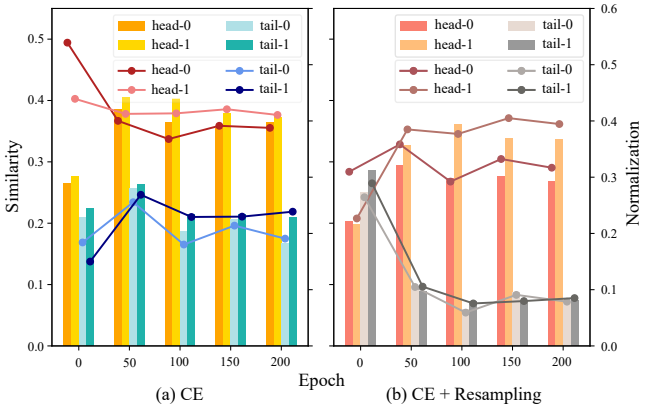

Figure 2: Gradient imbalance in long-tail learning. The bars denote the mean similarity between class-level and batch-level gradients in each batch. The dots represent the norm of mean gradients of classes in epochs. We experiment on CIFAR10-LT, where (a) uses only cross entropy loss and (b) uses a resampling strategy. We show the result of the top two head classes and the last two tail classes.

where we use N_{head} and N_{tail} to represent the numbers of head-class samples and tail-class samples in a mini-batch. Thus, the gradient of parameter θ can be denoted as:

$$\nabla_{\boldsymbol{\theta}} = \frac{\partial L\left(\boldsymbol{x}, \boldsymbol{y}\right)}{\partial \boldsymbol{\theta}} = \nabla_{\boldsymbol{\theta}}^{\text{tail}} + \nabla_{\boldsymbol{\theta}}^{\text{head}}, \tag{3}$$

where $\nabla_{\boldsymbol{\theta}}^{\text{tail}}$ and $\nabla_{\boldsymbol{\theta}}^{\text{head}}$ are the gradients of $\boldsymbol{\theta}$ generated by the tail-class and head-class instances in the mini-batch.

In Fig. 2, we measure the mean gradient similarity between classes and the whole gradient in each batch in different epochs. The similarity measures the contribution of gradients from different classes to the gradient descent process. A larger similarity means a larger contribution. It is easy to observe that *the gradients of head and tail classes, which are presented as dots in the figure, are significantly imbalanced*, where we have $\|\nabla_{\boldsymbol{\theta}}^{\text{tail}}\| < \|\nabla_{\boldsymbol{\theta}}^{\text{head}}\|$ in every epoch. The reason of the gradient imbalance in long-tailed distribution, we argue, is the head-class samples make up the majority of most batches, resulting in the gradient domination of head classes in magnitude and direction over tail classes, which can be represented as $\nabla_{\boldsymbol{\theta}}^{\top}\nabla_{\boldsymbol{\theta}}^{\text{head}} > \nabla_{\boldsymbol{\theta}}^{\top}\nabla_{\boldsymbol{\theta}}^{\text{tail}}$. Finally, the model cannot obtain enough knowledge from tail-class data. At the same time, it raises confusion that the imbalance in Fig. 2(a) is caused by the imbalanced data distribution in each batch. We conduct a similar experiment with a resampling strategy. In Fig. 2(b), the magnitude of the tail gradient decreases after resampling because resampling means frequently sample duplicated. That is, the model quickly overfits to the tailed categories, leading to a rapid loss decrease in them, which indicates the gradient imbalance is not caused by an imbalanced distribution within each batch but the imbalanced distribution of the dataset.

To solve this problem, previous methods compensate tail-class gradients by raising the weight of tail-class loss by rebalancing strategies (Lin et al. 2017; Wu et al. 2020). However, as shown in Fig. 1, intuitive rebalancing methods encounter the *seesaw dilemma*, where the solutions may suffer from either overcompensation or undercompensation. Overcompensation refers to the tail classes being overemphasized, while the head classes are underestimated, resulting in the learning of the head classes being excessively inhibited. Undercompensation is equivalent to no compensation.

In this paper, we seek to find an ideal compensation at each training iteration in long-tail learning, where the update should not damage any class in a long-tailed distribution. To achieve this, for the first time to the best of our knowledge, we formulate the long-tailed learning into a multi-objective optimization problem as illustrated in the next subsection.

LT Problem as Multi-Objective Optimization

Multi-Objective Optimization (MOO) means optimizing multiple objectives simultaneously. Given T different objectives, a deep model with MOO yields the following multi-objective empirical risk minimization formulation:

$$\min_{\boldsymbol{\theta}} \ \left\{L_i\left(\mathcal{D}_i\right), \cdots, L_T\left(\mathcal{D}_T\right)\right\}, \tag{4}$$

where $\mathcal{D}_i$ is the data of objective i. Because of the conflict among objectives, the goal of MOO is to achieve Pareto optimality via training.

Definition 1 (Pareto Optimality).
*(1) (*Pareto Dominate*) Let $\boldsymbol{\theta}_a$, $\boldsymbol{\theta}_b$ be two solutions for Problem* (4)*, $\boldsymbol{\theta}_a$ is said to dominate $\boldsymbol{\theta}_b$ ($\boldsymbol{\theta}_a \prec \boldsymbol{\theta}_b$) if and only if $L_i(\boldsymbol{\theta}_a) \leq L_i(\boldsymbol{\theta}_b)$, $\forall i \in \{1, 2, \cdots, T\}$ and $L_i(\boldsymbol{\theta}_a) < L_i(\boldsymbol{\theta}_b)$, $\exists i \in \{1, 2, \cdots, T\}$.*
*(2) (*Pareto Critical*) $\boldsymbol{\theta}$ is called Pareto critical if no other solution in its neighborhood can have better values in all objective functions.*
*(3) (*Pareto Descent Direction*) If $\boldsymbol{\theta}_1$ is not Pareto critical and can be updated to $\boldsymbol{\theta}_2$ by gradient* $\mathbf{g}$*. If $\boldsymbol{\theta}_2 \prec \boldsymbol{\theta}_1$, say* $\mathbf{g}$ *is a Pareto descent direction.*

A MOO problem may have multiple solutions, consisting of a Pareto set, whose projection in loss space is called *Pareto Front*. To approach the Pareto front in the loss space for all classes in a long-tailed distribution, we need to make each update under a Pareto descent direction that does not damage any class's performance. To this end, we convert the single objective loss function in Eq. (1) into a multi-objective optimization problem, which yields a gather of loss functions for each category

$$\mathcal{L}\left(\boldsymbol{\theta}; \mathcal{D}\right) = \left\{L_1\left(\boldsymbol{\theta}; \mathcal{D}_1\right), \cdots, L_K\left(\boldsymbol{\theta}; \mathcal{D}_K\right)\right\}, \tag{5}$$

where $L_k\left(\boldsymbol{\theta}; \mathcal{D}_k\right)$ and $\mathcal{D}_k \subseteq \mathcal{D}$ represent loss function of class k and the samples from class k respectively.

Let us review the seesaw dilemma under the MOO setting. After splitting training loss, we have K different losses w.r.t. K classes. Subsequently, we obtain task-specific gradients $\{\nabla_1, \cdots, \nabla_K\}$ via derivation, where $\nabla_i = \nabla_{\boldsymbol{\theta}}^{L_i}$ and update only once. That is, we need to aggregate all gradients into one. A simple aggregation way is to set weights and sum class-level gradients. At the iteration n, the problem can be reformulated as follows:

$$\min_{\{\alpha_1, \cdots, \alpha_K\}} \ \left\{L_i\left(\boldsymbol{\theta}^{(n-1)} - \tau \sum\nolimits_{i=1}^{K} \alpha_i \nabla_i \ ; \ \mathcal{D}_i\right) \middle| \forall i\right\}. \tag{6}$$

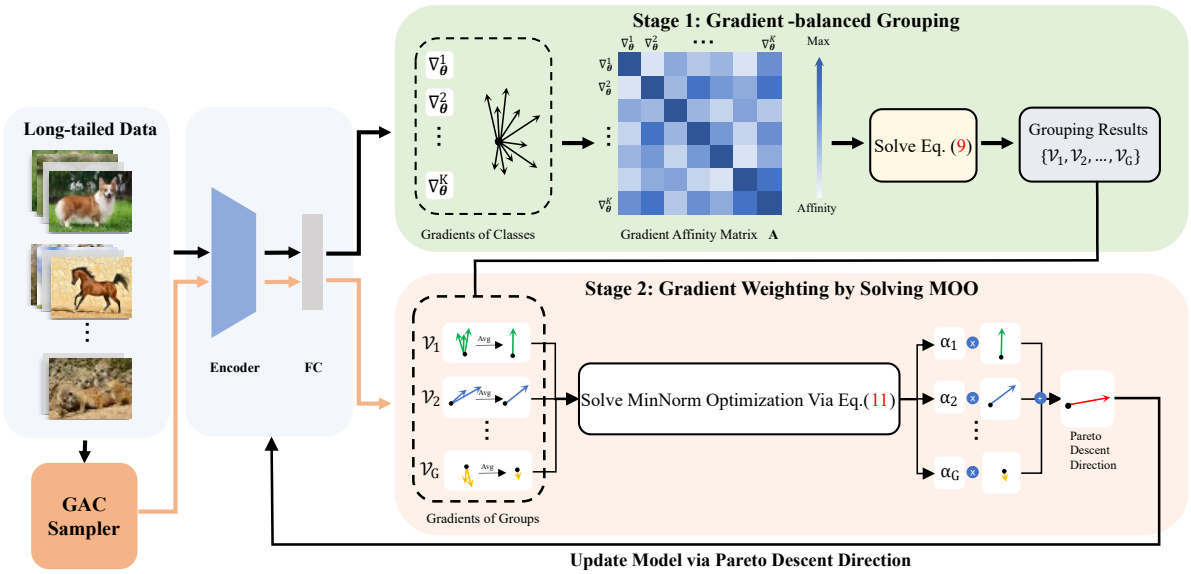

Figure 3: Illustration of our proposed method. In the first stage, we use GBG to gather the classes with high gradient similarity together. In the second stage, we use an averaging strategy to merge the gradients of the categories in the same group, and then solve a MOO problem to obtain an approximate Pareto descent direction in each iteration.

To avoid the damage for all classes, we prefer to have $L_i\left(\boldsymbol{\theta}^{(n)};\mathcal{D}_i\right) \leq L_i\left(\boldsymbol{\theta}^{(n-1)};\mathcal{D}_i\right)$ for any $i \in [1,K]$. *However, the multiple gradients may have large conflicts in terms of magnitude and direction.* Inappropriate weighting may result in overcompensation and undercompensation that some classes may decrease. In contrast, the goal of multi-objective optimization is to achieve Pareto descent direction in each step, which will damage no class.

Intuitively, using the losses of each category as optimization objectives can achieve better performance. However, more objectives do not necessarily mean better performance. Multi-objective optimization problems can pose significant challenges due to the increase in the dimensionality of search space and the complexity of Pareto fronts as the number of objectives increases. Therefore, it is impractical to directly solve Problem (6) to achieve accurate Pareto descent direction, especially when the label space has a large dimension. Furthermore, the batch size is limited by the restricted size of hardware memory so it is difficult to cover all classes and store the gradients from all classes into the memory. In the next subsection, we propose a simple yet effective gradient-balancing grouping strategy to obtain an approximate Pareto descent direction.

Gradient-based Class Grouping

Class grouping (Li et al. 2020) is one of the effective solutions in long-tailed learning. However, most of them rely on heuristic ideas and they can not guarantee a good compensation. In this paper, we propose a Gradient-Balancing Grouping (GBG) strategy to solve the gradient conflict and obtain an approximate Pareto descent direction. GBG assigns classes with similar gradient descent direction into a group to make their gradients form a resultant force, which represents the approximate no-conflict direction to those of all corresponding classes in the group. Specifically, given class gradient in a batch $\{\nabla_1, \cdots, \nabla_c\}$, where c denotes the contained class numbers of the batch. Let the category set be $\mathcal{C} = \{1, \cdots, K\}$. We first compute a similarity matrix $\mathbf{A}$ to measure the similarity between any two gradients, and the element $\boldsymbol{A}_{i,j} = \frac{\nabla_{\boldsymbol{\theta}}^i \nabla_{\boldsymbol{\theta}}^j}{\|\nabla_{\boldsymbol{\theta}}^i\| \|\nabla_{\boldsymbol{\theta}}^j\|}$.

According to the similarity matrix $\boldsymbol{A}$, we then build a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. $\mathcal{V}$ denotes the set of nodes in the graph, and each node represents a class. $\mathcal{E} = \{A_{i,j}\}, i \leq j \leq K$ denotes the set of edges, where each edge represents the gradient similarity between class i and class j. Our target is to find a way of grouping categories so that categories with high similarity in the direction of gradient descent are placed in the same group. Then, we define the affinity between groups as follows:

$$a\left(\mathcal{V}_m, \mathcal{V}_n\right) = \sum\nolimits_{i \in \mathcal{V}_m, j \in \mathcal{V}_n} \boldsymbol{A}_{i,j}, \tag{7}$$

where $\mathcal{V}_m, \mathcal{V}_n \subset \mathcal{V}$ represent two different groups and $\mathcal{V}_m \cap \mathcal{V}_n = \emptyset, \forall m \neq n$. Inspired by spectrum-clustering (Ng, Jordan, and Weiss 2001), our target is equivalent to finding a graph cutting $\mathcal{P} = \{\mathcal{V}_1, \mathcal{V}_2, \cdots, \mathcal{V}_G\}$ that minimizes the summation of affinity between groups, where G is the number of groups. We formulate the problem as follows:

$$\min_{\mathcal{P} \in \mathbb{P}} \quad \sum\nolimits_{\mathcal{V}' \in \mathcal{P}} a\left(\mathcal{V}', \mathcal{V}\right) - a\left(\mathcal{V}', \mathcal{V}'\right), \tag{8}$$

where $\mathbb{P}$ is the searching space for possible grouping strategy. Then, we use NCut (Shi and Malik 2000) to transform the optimization problem into the form of minimizing the Rayleigh entropy to obtain the partitioning result $\mathcal{P}$.

The grouping result $\mathcal{P}$ obtained through the above method ensures that the categories in the same group have similar update habits. Then, during the model training process,

Algorithm 1: Gradient-balanced grouping

Input: Training set $\mathcal{D}$, Model parameters $\boldsymbol{\theta}$.

1: **for** $k = 1 \rightarrow K$ **do**
2: Calculate average gradient $\nabla_{\boldsymbol{\theta}}^{k}$ of class k;
3: **end for**
// Compute gradient similarity matrix $\boldsymbol{A}$
4: $\nabla \leftarrow [\nabla_{\boldsymbol{\theta}}^{\mathbf{1}}, \cdots, \nabla_{\boldsymbol{\theta}}^{\mathbf{K}}]^{\top}$;
5: $\boldsymbol{A} \leftarrow \left(\boldsymbol{\nabla} \cdot \boldsymbol{\nabla}^{\top}\right) / \{\|\boldsymbol{\nabla}\| \cdot \|\boldsymbol{\nabla}\|^{\top}\}$;
6: Build the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ according to $\boldsymbol{A}$;
// Generate groups by solving Eq. (8)
7: $\mathcal{P}^{*} \leftarrow \underset{\mathcal{P} \in \mathbb{P}}{\arg\min} \sum_{\mathcal{V}' \in \mathcal{P}} a\left(\mathcal{V}', \mathcal{V}\right) - a\left(\mathcal{V}', \mathcal{V}'\right)$;

Output: Group partitioning result $\mathcal{P}^{*}$.

the gradients of each group we obtained in each batch are equivalent to the average of the gradients of the categories in their corresponding group, as shown in Fig. 3 (Stage 2). In other words, the gradients obtained from each group are as similar as possible to the gradients of any category within the group, which enables the gradients within each group to work together and implicitly increase the contribution of the tail class during training. Our grouping strategy is designed to improve both head and tailed classes. 1) We group gradient-similar classes instead of semantic-similar classes. 2) By ideal compensation, both head and tailed classes can be improved. Because in a group, similar gradients as tailed classes will not be ignored.

The group memberships are calculated one time only at the start of training. The overall class grouping procedure of GBG in the LT problem is summarized in Algorithm 1. First, we fix the parameters of the initialized or pre-trained model $f\left(\boldsymbol{\theta}\right)$. Next, we calculate the average gradients of each class and obtain the gradient similarity between each class through cosine similarity to build a symmetrical gradient similarity matrix $\boldsymbol{A}$. Eventually, we solve the graph-cutting problem Eq. (8) and get the final grouping result $\mathcal{P}$. However, the gradient conflicts between each group still exist. In the following, we show how to solve the group-level MOO problem.

Solving Group-level MOO Problem

Following previous studies (Zhou et al. 2020), our method is split into 2 stages as shown in Fig. 3. In the first stage, we calculate the average gradients of each class and form a gradient similarity matrix. Then we divide categories into G groups according to their gradient similarity. In the second stage, we bundle the gradients of each group in each batch and form an MOO problem. Then, we solve the MOO problem to approximate a Pareto descent direction, achieving optimization of all groups at each iteration. Based on the class grouping $\mathcal{P}^{*} = \{\mathcal{V}_1, \mathcal{V}_2, \cdots, \mathcal{V}_G\}$, the optimization goal of the long-tail problem can be converted into the training loss of G groups

$$\min_{\boldsymbol{\theta}} \left(L_1\left(\boldsymbol{\theta}\right), \cdots, L_G\left(\boldsymbol{\theta}\right)\right). \tag{9}$$

To efficiently solve the MOO problem, we adopt Multi Gradient Descent Algorithm (MGDA) (Sener and Koltun 2018)

Algorithm 2: Update model by grouping MOO

Input: Training set $\mathcal{D}$, Group results $\mathcal{P}^{*}$, Model parameters $\boldsymbol{\theta}$, Learning rate η.

1: Sample a mini-batch $\mathcal{B} = \{\mathcal{B}_1, \cdots, \mathcal{B}_G\}$ from the training set $\mathcal{D}$;
2: **for** $i = 1 \rightarrow G$ **do**
3: Compute group-level loss $L_i = L\left(\boldsymbol{\theta}; \mathcal{B}_i\right)$;
4: Back-propagation and compute gradients $\nabla_{\boldsymbol{\theta}} L_i$;
5: **end for**
6: $\alpha_1^{*}, \cdots, \alpha_G^{*} \leftarrow$ Solve Eq. (10);
7: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \sum_{i=1}^{G} \alpha_i \nabla_{\boldsymbol{\theta}} L_i$;

Output: Updated parameters $\boldsymbol{\theta}$.

that leverages the Karush-Kuhn-Tucker (KKT) conditions and transforms the MOO problem into a min-norm single objective optimization as follows:

$$\begin{aligned} \min_{\alpha_1, \cdots, \alpha_G} \quad & \left\| \sum\nolimits_{i=1}^{G} \alpha_i \nabla_{\boldsymbol{\theta}} L_i\left(\boldsymbol{\theta}\right) \right\|^2, \\ \text{s.t.} \quad & \sum\nolimits_{i=1}^{G} \alpha_i = 1 \text{ and } \alpha_i \geq 0, \forall i. \end{aligned} \tag{10}$$

As a min-norm single objective optimization, this problem can be easily solved by quadratic programming. With the solution to this optimization problem, we obtain the final gradient for the long-tailed learning

$$\boldsymbol{d}^{*} = \sum\nolimits_{i=1}^{G} \alpha_i \nabla_{\boldsymbol{\theta}} L_i. \tag{11}$$

According to (Sener and Koltun 2018), vector $\boldsymbol{d}^{*}$ is either zero or a feasible Pareto descent direction for all groups. We show the multi-objective optimization-based gradient descent steps in Algorithm 2. Specifically, in every iteration, we compute the loss of each group and conduct backward over the model parameters for each loss to get $\nabla_{\boldsymbol{\theta}} L_i\left(\boldsymbol{\theta}\right)$. Then we acquire the weights $\{\alpha_1, \cdots, \alpha_G\}$ by solving 10 and use it carrying out weighted summation $\sum_{i=1}^{G} \alpha_i \nabla_{\boldsymbol{\theta}} L_i\left(\boldsymbol{\theta}\right)$ to get the final gradients which are used to update model parameters. Moreover, we propose a simple but effective resampling method Group-Aware Completion (GAC) Sampler to guarantee that each batch contains samples from all groups, *i.e.*, we have data from all groups in every mini-batch. For each iteration, if the data of a group is missing in the mini-batch, we sample from the missing training data of the missing group with probability based on the class-balanced term (Cui et al. 2019), so that the number of samples reaches 1/10 of the batch size. This can ensure that each batch contains samples from all groups and to some extent improve the contribution of tail-class samples.

Compared with the original optimization problem containing a large number of objectives, our method greatly reduces the number of optimization objectives and can complete the training relatively efficiently.

Experiment

Datasets

CIFAR10/100-LT. CIFAR10/100-LT are the long-tailed version of CIFAR10/100. Specifically, they are generated by

Method	CIFAR10-LT		CIFAR100-LT	
Imbalance Factor	100	50	100	50
CE	70.36	74.81	38.32	43.85
CB (Cui et al. 2019)	74.57	79.27	39.60	45.17
LDAM (Cao et al. 2019)	77.03	-	42.04	-
BBN (Zhou et al. 2020)	79.82	82.18	42.56	47.02
PaCo (Cui et al. 2021)	-	-	50.00	56.00
RISDA (Chen et al. 2022)	-	-	50.16	53.84
xERM (Zhu et al. 2022a)	-	-	46.90	52.80
FDC (Ma et al. 2023)	83.40	86.50	50.40	54.10
DisVar (Tian et al. 2023)	78.87	83.69	48.07	52.35
BCL† (Zhu et al. 2022b)	84.07	86.98	51.52	56.23
GBG (Ours)	**85.05**	**87.73**	**52.31**	**57.18**

Table 1: Top-1 accuracy comparison on the CIFAR10/100-LT . † indicates the results reproduced by ourselves. The best results are presented in **bold**. The indicates are the same in other tables.

downsampling CIFAR10/100 with different Imbalance Factor (IF) $\beta = N_{\text{max}}/N_{\text{min}}$ where N_{max} and N_{min} are the instance size of most frequent and least frequent classes in the training set (Cui et al. 2019; Cao et al. 2019).

ImageNet-LT. ImageNet-LT is sampled from vanilla ImageNet following a Pareto distribution with the power value $\alpha = 6$. It contains 115.8K training images of 1,000 categories with $N_{max} = 1,280$ and $N_{min} = 5$. We use the balanced validation set of vanilla ImageNet which contains 50 images per class.

iNaturalist 2018. iNaturailist 2018(iNat) is a large-scale real-world dataset that naturally presents a long-tailed distribution. It consists of 437.5K images from 8,142 classes with $\beta = 512$. The validation set contains 24.4K images with 3 images per class to test our method.

Experiment Setting

We perform experiments on CIFAR10/100-LT (IF = 100,50), ImageNet-LT and iNaturalist. We use ResNet-32 as the backbone for CIFAR, ResNet-50, ResNeXT-50 for ImageNet-LT, and ResNet-50 for iNaturalist 2018. We use SGD for all datasets. For CIFAR and ImageNet-LT, weight decay (wd) is 5e-4 and momentum (m) is 0.9. For iNat, wd is 1e-4. We set batch size as 256 for all datasets. We use a fully-connected layer as classifier for all models. We train all the above models on NVIDIA GeForce RTX 3090 GPU.

Main Results

We compare our method with state-of-the-art methods. We use top-1 accuracy as metric in all experiments. The comparison results are shown in Tables 1, 2 and 3.

Our method achieves the best performance compared with recent SOTA methods on three benchmarks. On ImageNet-LT, our method gets 57.6% in top-1 accuracy, which is 1.2% over BCL with ResNet-50 and achieves 58.7% which surpasses the second-best method for 1.8%. Our method obtains 0.4% over BCL on iNat. Because GBG gathers classes with similar gradient directions, it makes the gradients of the groups represent those of classes within the group which implicitly increases the contribution of tail-class gradients.

Method	ImageNet-LT	
Backbone	ResNet-50	ResNext-50
CE	41.6	44.4
Decouple (Kang et al. 2019)	-	49.9
RIDE (Wang et al. 2020)	54.4	55.9
LADE (Hong et al. 2021)	-	51.9
LA (Menon et al. 2020)	51.1	-
RISDA (Chen et al. 2022)	50.7	-
WD (Alshammari et al. 2022)	-	53.9
xERM (Zhu et al. 2022a)	-	54.1
FDC (Ma et al. 2023)	-	55.3
DisVar (Tian et al. 2023)	49.4	-
BCL† (Zhu et al. 2022b)	56.4	56.9
GBG (Ours)	**57.6**	**58.7**

Table 2: Top-1 accuracy comparison on ImageNet-LT.

Methods	iNatualist 2018
CE	63.8
LDAM (Cao et al. 2019)	68.0
BBN (Zhou et al. 2020)	69.6
RIDE (Wang et al. 2020)	71.4
LADE (Hong et al. 2021)	70.0
LA (Menon et al. 2020)	66.4
RISDA (Chen et al. 2022)	69.2
WD (Alshammari et al. 2022)	70.2
DisVar (Tian et al. 2023)	69.3
BCL† (Zhu et al. 2022b)	71.5
GBG (Ours)	**71.9**

Table 3: Top-1 accuracy comparison on iNatutalist 2018.

To gain deeper insights into the impact of our methods on various categories, we have partitioned the ImageNet-LT classes into three distinct subsets, following (Zhu et al. 2022b). These subsets are characterized as Many (>100 images), Medium (20∼100 images) and Few (<20 images) according to the instance number of classes The results presented in Table 4 unequivocally demonstrate notable performance enhancements across all three subsets. This observation underscores the effectiveness of our approach in elevating the performance of both tail and head categories. The MOO strategy employed in our method effectively addresses the challenge of imbalanced datasets, allowing for a harmonious training dynamic between the head and tail categories.

Grouping Strategy Comparison

In Table 5, we compare our strategy with another two grouping rules, *i.e.*, random grouping strategy and instance-numbers-based grouping strategy. Random grouping strategy partitions the category set into 4 groups randomly. We use several random seeds to get different random grouping results and show the average test results for a fair comparison. For instance-numbers-based grouping strategy, we follow (Li et al. 2020) to divide all categories into 4 groups according to their instance numbers, which means classes with similar instance numbers are in the same group. The results show that our grouping strategy achieves the best performance among all grouping strategies, which proves the effectiveness of grouping via gradients.

Methods (ResNext-50)	Many	Medium	Few	All
τ-norm (Kang et al. 2019)	59.1	46.9	30.7	49.4
BS (Ren et al. 2020)	62.2	48.8	29.8	51.4
Dec (Kang et al. 2019)	60.2	47.2	30.3	49.9
RIDE (Wang et al. 2020)	68.2	53.8	36.0	56.8
LADE (Hong et al. 2021)	62.3	49.3	31.2	51.9
Dis (Zhang et al. 2021)	62.7	48.8	31.6	51.8
FDC (Ma et al. 2023)	65.5	51.9	37.8	55.3
BCL† (Zhu et al. 2022b)	66.9	54.3	37.6	56.9
GBG (Ours)	**69.6**	**55.8**	**38.1**	**58.7**

Table 4: Comparison on three subsets of ImageNet-LT.

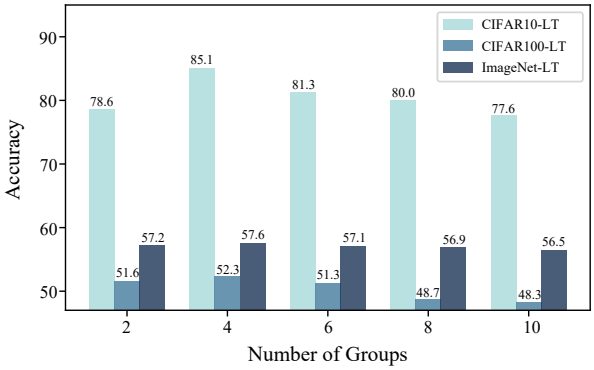

Figure 4: Comparison results of different group numbers.

Analysis on Different Group Numbers

In Fig. 4, we present the impact of different group numbers within our gradient-based class grouping mechanism. We conduct experiments across CIFAR10-LT, CIFAR100-LT, and ImageNet-LT datasets, varying the number of groups to determine the optimal configuration. The outcomes of these experiments reveal that the use of four groups yields the most favorable results. This finding suggests that employing an excess number of objectives does not necessarily enhance performance in the MOO problems. Moreover, assigning each class as an individual objective in CIFAR10-LT significantly undermines performance. This outcome is likely attributed to the heightened complexity of the optimization space when confronted with numerous objectives. The resulting increased likelihood of encountering local optima could lead to performance degradation.

Ablation Study

We perform several experiments to demonstrate the validity of each component. We split our methods into two parts including gradient-balanced grouping strategy (GBG) and multi-objective optimization (MOO) in Table. 6. We choose CIFAR10-LT (IF=100) for experiment and use ResNet-32 as backbone. We set the number of groups as four, and use the average strategy to merge the gradients of groups. To validate the compatibility of the proposed method, we add our method to the naive model, Cross-Entropy (CE). The results show a significant improvement when our method is applied to CE, indicating its compatibility.

Strategy	ImageNet-LT	
Backbone	ResNet-50	ResNext-50
Random	56.37±0.41	56.96±0.24
Instance number	56.61±0.27	57.44±0.38
GBG	57.49±0.29	58.61±0.37

Table 5: Grouping strategy comparison on ImageNet-LT (Avg±Std over 5 fixed seeds).

Method	Accuracy
CE	71.36
CE+GBG+MOO	75.53
BCL	84.07
BCL+GBG	84.20
BCL+MOO	74.58
BCL+GBG+MOO	**85.05**

Table 6: Ablation study for our proposed method with BCL as the baseline on CIFAR10-LT (IF = 100).

By utilizing GBG, we implicitly enhance the impact of tail-class gradients. Therefore, GBG achieves a modest accuracy improvement of 0.13% when adding GBG to BCL. On the other hand, directly employing the MOO method sets the loss of each category as an optimization objective. However, due to the limited batch size, data from tail classes are often absent in batches, resulting in the absence of certain optimization objectives in each iteration. Consequently, solely relying on MOO significantly decreases performance, which is only 74.58%. In other words, the optimization objectives for the MOO problem are not fixed, considerably influencing the resolution of the MOO problem.

Our method combines grouping and MOO. Through GBG, we put categories with high gradient similarity into a group, fixing the optimization objectives for MOO problem. We further solved the gradient conflicts between different groups through MOO, which enabled us to obtain approximate Pareto descent directions in every descent step. Building on this, we achieve an improvement of 0.85% compared with only using Grouping.

Conclusion

In this paper, we identified gradient imbalance as a key issue in long-tailed learning. We thoroughly analyzed the inappropriate compensation on the gradients of different classes resulting in the seesaw dilemma of previous methods. We addressed this problem by formulating the long-tailed recognition as a MOO problem and introducing the GBG algorithm to balance the gradient contributions of head and tail classes. Then, GBG makes classes with similar gradient directions form more representative gradients. With GBG, we approximately made every update of model parameters under a Pareto descent direction and provided ideal compensation to the tail classes. GBG outperformed existing methods on commonly used benchmarks which demonstrated the superiority of our methods. Future work will focus on refining the grouping strategy for more efficient hyperparameter tuning, enabling more efficiently applied to different datasets.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (No. 2020YFC1522701), the National Natural Science Foundation of China (Nos. 6227071567, 62276182 and 62072334), and the National Postdoctor Funding Program of China (No. GZC20232993).

References

Alshammari, S.; Wang, Y.-X.; Ramanan, D.; and Kong, S. 2022. Long-tailed recognition via weight balancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6897–6907.

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

Cao, K.; Wei, C.; Gaidon, A.; Arechiga, N.; and Ma, T. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in Neural Information Processing Systems*, 32.

Chen, H.; Li, L.; Hu, F.; Lyu, F.; Zhao, L.; Huang, K.; Feng, W.; and Xia, Z. 2023. Multi-semantic hypergraph neural network for effective few-shot learning. *Pattern Recognition*, 142: 109677.

Chen, X.; Zhou, Y.; Wu, D.; Zhang, W.; Zhou, Y.; Li, B.; and Wang, W. 2022. Imagine by reasoning: A reasoning-based implicit semantic data augmentation for long-tailed classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 356–364.

Cui, J.; Zhong, Z.; Liu, S.; Yu, B.; and Jia, J. 2021. Parametric contrastive learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 715–724.

Cui, Y.; Jia, M.; Lin, T.-Y.; Song, Y.; and Belongie, S. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9268–9277.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 248–255.

Geng, B.; Li, L.; Jiao, L.; Gong, M.; Cai, Q.; and Wu, Y. 2015. NNIA-RS: A multi-objective optimization based recommender system. *Physica A: Statistical Mechanics and its Applications*, 424: 383–397.

Harada, K.; Sakuma, J.; and Kobayashi, S. 2006. Local search for multiobjective function optimization: Pareto descent method. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, 659–666.

He, H.; and Garcia, E. A. 2009. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9): 1263–1284.

Hong, Y.; Han, S.; Choi, K.; Seo, S.; Kim, B.; and Chang, B. 2021. Disentangling label distribution for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6626–6636.

Jamal, M. A.; Brown, M.; Yang, M.-H.; Wang, L.; and Gong, B. 2020. Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7610–7619.

Kang, B.; Xie, S.; Rohrbach, M.; Yan, Z.; Gordo, A.; Feng, J.; and Kalantidis, Y. 2019. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*.

Li, J.; Tan, Z.; Wan, J.; Lei, Z.; and Guo, G. 2022. Nested collaborative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6949–6958.

Li, Y.; Wang, T.; Kang, B.; Tang, S.; Wang, C.; Li, J.; and Feng, J. 2020. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10991–11000.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2980–2988.

Lyu, F.; Sun, Q.; Shang, F.; Wan, L.; and Feng, W. 2023. Measuring Asymmetric Gradient Discrepancy in Parallel Continual Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 11411–11420.

Lyu, F.; Wang, S.; Feng, W.; Ye, Z.; Hu, F.; and Wang, S. 2021. Multi-domain multi-task rehearsal for lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 8819–8827.

Ma, Y.; Jiao, L.; Liu, F.; Yang, S.; Liu, X.; and Chen, P. 2023. Feature distribution representation learning based on knowledge transfer for long-tailed classification. *IEEE Transactions on Multimedia*.

Menon, A. K.; Jayasumana, S.; Rawat, A. S.; Jain, H.; Veit, A.; and Kumar, S. 2020. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314*.

Ng, A.; Jordan, M.; and Weiss, Y. 2001. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 14.

Ren, J.; Yu, C.; Ma, X.; Zhao, H.; Yi, S.; et al. 2020. Balanced meta-softmax for long-tailed visual recognition. *Advances in Neural Information Processing Systems*, 33: 4175–4186.

Sener, O.; and Koltun, V. 2018. Multi-task learning as multi-objective optimization. *Advances in Neural Information Processing Systems*, 31.

Shi, J.; and Malik, J. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8): 888–905.

Sun, Q.; Lyu, F.; Shang, F.; Feng, W.; and Wan, L. 2022. Exploring example influence in continual learning. *Advances in Neural Information Processing Systems*, 35: 27075–27086.

Tian, Y.; Gao, W.; Zhang, Q.; Sun, P.; and Xu, D. 2023. Improving long-tailed classification by disentangled variance transfer. *Internet of Things*, 21: 100687.

Wang, X.; Lian, L.; Miao, Z.; Liu, Z.; and Yu, S. X. 2020. Long-tailed recognition by routing diverse distribution-aware experts. *arXiv preprint arXiv:2010.01809*.

Wu, T.; Huang, Q.; Liu, Z.; Wang, Y.; and Lin, D. 2020. Distribution-balanced loss for multi-label classification in long-tailed datasets. In *Proceedings of the European Conference on Computer Vision*, 162–178. Springer.

Yang, L.; Jiang, H.; Song, Q.; and Guo, J. 2022. A survey on long-tailed visual recognition. *International Journal of Computer Vision*, 130(7): 1837–1872.

Zhang, S.; Li, Z.; Yan, S.; He, X.; and Sun, J. 2021. Distribution alignment: A unified framework for long-tail visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2361–2370.

Zhou, B.; Cui, Q.; Wei, X.-S.; and Chen, Z.-M. 2020. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9719–9728.

Zhu, B.; Niu, Y.; Hua, X.-S.; and Zhang, H. 2022a. Cross-domain empirical risk minimization for unbiased long-tailed classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 3589–3597.

Zhu, J.; Wang, Z.; Chen, J.; Chen, Y.-P. P.; and Jiang, Y.-G. 2022b. Balanced contrastive learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6908–6917.