Neural Networks 151 (2022) 132–142

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

# Dynamic Auxiliary Soft Labels for decoupled learning

Yan Wang [a,b], Yongshun Zhang [a,b], Furao Shen [a,c,*], Jian Zhao [d]

[a] State Key Laboratory for Novel Software Technology, Nanjing University, China
[b] School of Artificial Intelligence, Nanjing University, China
[c] Department of Computer Science and Technology, Nanjing University, China
[d] School of Electronic Science and Engineering, Nanjing University, China

ARTICLE INFO

*Article history:*
Received 16 November 2021
Received in revised form 11 February 2022
Accepted 24 March 2022
Available online 31 March 2022

*Keywords:*
Neural network
Decoupled learning
Long-tailed

ABSTRACT

The long-tailed distribution in the dataset is one of the major challenges of deep learning. Convolutional Neural Networks have poor performance in identifying classes with only a few samples. For this problem, it has been proved that separating the feature learning stage and the classifier learning stage improves the performance of models effectively, which is called decoupled learning. We use soft labels to improve the performance of the decoupled learning framework by proposing a Dynamic Auxiliary Soft Labels (DaSL) method. Specifically, we design a dedicated auxiliary network to generate auxiliary soft labels for the two different training stages. In the feature learning stage, it helps to learn features with smaller variance within the class, and in the classifier learning stage it helps to alleviate the overconfidence of the model prediction. We also introduce a feature-level distillation method for the feature learning, and improve the learning of general features through multi-scale feature fusion. We conduct extensive experiments on three long-tailed recognition benchmark datasets to demonstrate the effectiveness of our DaSL.



## 1. Introduction

With the continuous development of deep learning, Convolutional neural networks (CNNs) have made significant breakthroughs in many fields of computer vision. One of the important reasons is the construction of rich image datasets in different fields, such as ILSVRC 2012 (Russakovsky et al., 2015) for image recognition, MS COCO (Lin et al., 2014) for object detection, and Places (Zhou, Lapedriza, Khosla, Oliva, & Torralba, 2018) for scene recognition. These artificially constructed datasets usually consider the problem of class balance when collecting samples, to ensure that the number of images of different categories in the dataset is less different. However, in the real world, the number of samples in different categories is usually imbalanced. A few categories (also known as the head categories) occupy a large number of samples, and most categories (also known as the tail categories) can only occupy a small number of samples, which is called the long-tailed distribution. In recent years, the computer vision community has released more and more datasets that reflect the long-tail distribution, such as iNaturalist (Cui, Song, Sun, Howard, & Belongie, 2018) and LVIS (Gupta, Dollár, & Girshick, 2019). This kind of distribution appears in many practical applications. If we directly use general training strategies to train models, it will cause significant performance degradation. Because the model shows a bias towards the head categories, the recognition accuracy of the tail categories is affected.

Many works have tried to solve this problem. Some works use re-balancing methods to alleviate the model's bias towards head categories. Such methods are generally divided into two categories: re-sampling (Buda, Maki, & Mazurowski, 2018; Byrd & Lipton, 2019; Chawla, Bowyer, Hall, & Kegelmeyer, 2002; Japkowicz & Stephen, 2002) and re-weighting (Cao, Wei, Gaidon, Aréchiga, & Ma, 2019; Dong, Gong, & Zhu, 2017; Lin, Goyal, Girshick, He and Dollár, 2017; Zhang, Fang, Wen, Li, & Qiao, 2017). The re-sampling method builds a more balanced data distribution by oversampling the head categories or undersampling the tail class, while the re-weighting method adjusts the weight of the tail categories in the objective function to make the model pay more attention to the tail categories. However, if the re-balancing method is used directly, it may have a negative impact on the model to learn a general feature representation. Therefore, some studies (Kang et al., 2020; Zhou, Cui, Wei, & Chen, 2020) separate feature representation learning and classifier learning by decoupled learning to achieve a good performance. These methods only use the re-balancing method in the classifier learning stage, and still use the instance-sampling method in the feature learning stage.

* Corresponding author at: State Key Laboratory for Novel Software Technology, Nanjing University, China.
*E-mail addresses:* yanwang@smail.nju.edu.cn (Y. Wang), zhangys@lamda.nju.edu.cn (Y. Zhang), frshen@nju.edu.cn (F. Shen), jianzhao@nju.edu.cn (J. Zhao).

https://doi.org/10.1016/j.neunet.2022.03.027
0893-6080/

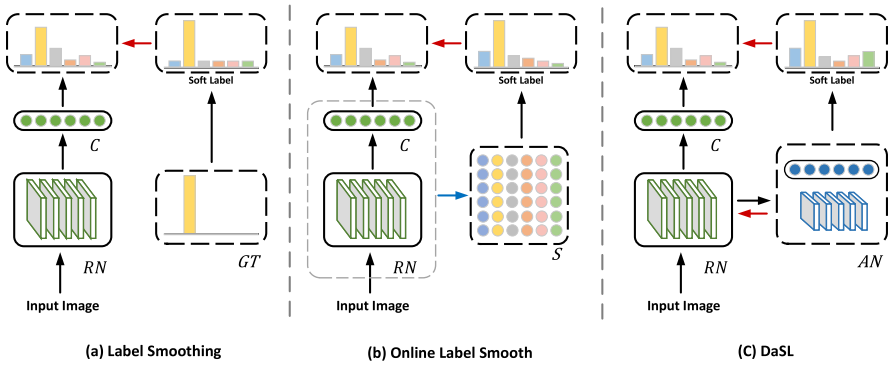


**Fig. 1.** Comparison of various soft labels generation methods. (a) Label Smoothing. (b) Online Label Smoothing. (C) Our proposed method, Dynamic Auxiliary Soft Label.

However, most of the existing decoupled methods use hard labels to supervise the two-stage learning. We argue that using hard labels is not the best way to learn feature representation (Müller, Kornblith, & Hinton, 2019), and the use of hard labels in the learning stage of the classifier will increase the over-confidence of the head categories and aggravate the bias of the model for the head categories. Some recent studies (He, Wu, & Wei, 2021) have shown that if the label distribution becomes flatter than the original label distribution, it will help the model learn the under-represented tail categories. It uses knowledge distillation (Hinton, Vinyals, & Dean, 2015) to supervise the student model through a re-balancing teacher model, and has achieved a significant improvement on the long-tail distributed dataset. But the student model still uses a single-stage training method. We try to find a suitable flat distribution to improve the model performance in both the feature representation stage and the classifier learning stage.

A simple attempt is to use Label Smoothing (Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2016) to generate soft labels in both stages to increase the proportion of the tail categories in the labels. However, the experiments have proved that it has an adverse effect on the performance of the model. One possible reason is that when Label Smoothing generates soft labels, the uniform distribution strategy exacerbates the model's bias towards head categories. Although Online Label Smoothing (Zhang et al., 2020) proved that the model itself can be used to generate labels with a more reasonable distribution. However, the prediction of the model learned in the first stage is biased, while the representations will be kept fixed in the classifier learning stage. Based on the above reasons, if the model learned in the first stage is directly used to generate the labels for the second stage, the generated soft labels will also be biased. For the long-tail distribution, due to the model's bias towards head categories, as well as the separation of feature representation learning and classifier learning in decoupled learning, the existing Label Smoothing method cannot generate effective soft labels for the two stages of decoupled learning at the same time.

In order to generate effective soft labels for decoupled learning, we propose a simple and novel method to generate soft labels dynamically, named Dynamic Auxiliary Soft Label (DaSL). The core idea is to design a general auxiliary model that provides soft labels for training of both two stages. Without changing the structure and learning strategy of the original model, the auxiliary model generates additional labels with flatter distributions for all its supervised learning stages. Specifically, our auxiliary model takes the feature maps of the original model as input to generate soft labels online. During the training process, our auxiliary model also uses instance-balanced sampling in the first stage, and re-balancing in the second stage. But in the second stage the representations are not fixed so that the auxiliary model can generate soft labels for feature representation learning and classifier unbiased learning for the two stages. We design the connection way between the original model and the auxiliary model. In order to improve the feature representation learning of the original model, we introduce multi-scale fusion features to the auxiliary network to supervise the feature learning of the original model. We demonstrate that our method not only assists the original model to learn feature representations with smaller intra-class variance in the first stage, but also alleviates its bias towards the head categories and improves the recognition accuracy of the tail categories. Fig. 1 shows the difference between our method and existing soft labels generation methods.

In this paper, we have three key contributions:

- For the problem of long-tail recognition, we propose a Dynamic Auxiliary Soft Label method, which uses an auxiliary model to generate soft labels for decoupled learning. The feature representation learning and classifier learning of the original model can be improved at the same time.
- In order to improve the general feature representation of the original model, we design a feature-level self-distillation method that uses multi-scale fusion features to supervise the feature learning of the original model.
- We demonstrate the effectiveness of our DaSL model on multiple long-tail recognition benchmark datasets, and use visualization methods to demonstrate that our method is beneficial to the general feature representation of the original model.

## 2. Related works

There are four mainstream methods for dealing with long-tail recognition problems, Re-sampling based, Cost-sensitive based, Decoupled Learning based, and Transfer-learning based methods. In this section we will briefly review the existing methods for long-tail recognition and Label Smoothing.

**Re-sampling based methods** A classic method to solve long-tailed recognition is based on the Re-sampling method, which uses a more balanced data distribution to reduce the bias of the model to the head categories. The method based on re-sampling has two main parts, which are the over-sampling of the tail categories (Buda et al., 2018; Byrd & Lipton, 2019; Shen, Lin, & Huang, 2016) and the undersampling of the head categories (He & Garcia, 2009; Japkowicz & Stephen, 2002). There are also some works (Chu, Bian, Liu, & Ling, 2020a; Liu, Sun, Han, Dou, & Li, 2020; Yin, Yu, Sohn, Liu, & Chandraker, 2019a) to implement re-sampling by data augmentation of tail categories in feature space. Recently, studies (Chou, Chang, Pan, Wei, & Juan, 2020; Liu, Li, Kang, Hua, & Vasconcelos, 2021; Zhang, Wei, Zhou and Wu,

2021) proved that using appropriate data augmentation methods can improve the model's ability to deal with long-tailed distributions, especially Mixup (Zhong, Cui, Liu, & Jia, 2021). However, the Re-sampling method may lead to overfitting of the tail categories (Cui, Jia, Lin, Song, & Belongie, 2019) and reduce the generalization of the model.

**Cost-sensitive based methods** The cost-sensitive method (also called the re-weighting method) reduces the model's recognition error rate for the tail categories by assigning more cost to the tail categories when calculating the loss function. A simple idea is to use the number of samples contained in different categories to determine the assigned weight. Cost-sensitive softmax cross entropy loss (Japkowicz & Stephen, 2002) weights the loss function by the number of real samples. Cui et al. (2019) used the number of effective samples to achieve re-weighting. Another way is to focus on difficult samples (Li, Liu, & Wang, 2019; Lin, Goyal, Girshick, He, & Dollár, 2020), and improve the performance of the model by increasing the error cost of difficult samples. This kind of methods are usually on sample-level. There are also some works (Tan et al., 2020; Wang et al., 2020) to reduce the adverse effect of the head categories on the tail categories by correcting the negative gradient in the back propagation process.

**Decoupled-Learning based methods** The decoupled learning method separates the model's feature representation learning and classifier learning. Kang et al. (2020) found that using the re-balance method in the classifier learning stage and fixing the representations can significantly improve the model performance. Zhou et al. (2020) also discovered this rule at the same time. They designed a Bilateral-Branch Network to learn feature representation and classifier respectively, and used the Cumulative Learning to smoothly merge the two branches. However, both methods used hard labels to supervise the two-stage training. Several recent studies improved the single learning stage of decoupled learning (Chu et al., 2020a; Wang et al., 2020; Zhang, Li, Yan, He and Sun, 2021). Zhong et al. (2021) proposed label-aware smoothing strategy to optimize the classifier learning stage. As a comparison, our paper attempts to use soft labels to optimize the two-stage training under the framework of decoupled learning.

**Transfer-learning based methods** In solving the problem that the tail categories has fewer samples and cannot be fully learned, an idea of transfer learning based method is to transfer the model's knowledge learned from the head categories to the tail categories (Liu et al., 2019; Wang, Ramanan, & Hebert, 2017; Xiang, Ding, & Han, 2020; Yin, Yu, Sohn, Liu, & Chandraker, 2019b). Another idea is to use multiple data groups with more balanced data distribution to train different experts, then transfer the knowledge of experts (Xiang et al., 2020). Recently, there are some studies using the meta learning to deal with long-tailed recognition (Jamal, Brown, Yang, Wang, & Gong, 2020; Lee et al., 2020; Li et al., 2021). Knowledge distillation has been proved to be effective in solving long-tail problems (He et al., 2021; Li, Wang and Wu, 2021). He et al. (2021) proposed the DiVE, using the model trained by the Virtual Examples with flat distribution as the teacher model. Similar to DiVE, our method embodies the idea of knowledge distillation, and also treats the generation of flat labels as Label Distribution Learning. The difference is that our proposed soft labels generation method is based on decoupled learning, and uses the feature map of the auxiliary model to supervise the original model, which can be regarded as a feature-level self-distillation method.

**Label Smoothing** Label Smoothing (Szegedy et al., 2016), as a simple and effective method to generate soft labels for model training, improves the generalization performance of the model on many recognition tasks. Müller et al. (2019) used a visualization method to show that Label Smoothing can help the model learn a more compact feature representation. Zhang et al. (2020) designed Online Label Smoothing to use the model of the last epoch to predict the current label distribution, replacing the uniform distribution used in general Label Smoothing. It can be regarded as a label-level self-distillation method (Kim, Ji, Yoon and Hwang, 2020). Shen et al. (2021) proved that the use of Label Smoothing in the single-stage model has an adverse effect on long-tail recognition. Our model also explores the impact of soft labels on long-tail recognition. The difference is that our model can generate auxiliary soft labels for the two-stages training in decoupled learning, and improve the feature representation learning and classifier learning of decoupled learning at the same time.

## 3. Methodology

Our model is called Dynamic Auxiliary Soft Label (DaSL). In this section, we first review the related methods of Label Smoothing, and introduce some of our attempts based on existing Label Smoothing methods in Section 3.1. Then we introduce our DaSL model in Section 3.2. Fig. 2 shows the overview of DaSL. Our model is also based on the decoupled learning. We will separately introduce the feature representation learning stage in Section 3.2.1, the classifier learning stage in Section 3.2.2 and the auxiliary network we designed in Section 3.2.3.

### 3.1. Study of label smoothing for decoupled learning

Kang et al. (2020) and Zhou et al. (2020) found that for the long-tail recognition problem, if the feature learning stage and the classifier learning stage of the model are disassembled, the model prediction accuracy will be significantly improved. Based on this discovery, Kang et al. (2020) proposed a simple and effective method called Classifier Re-training (cRT). Specifically, in the first stage, the feature learning network and classifier of the model are trained using instance-based sampling, with the goal of learning a general feature representation. In the second stage, the representations are fixed, then we use the re-sampling method to train the classifier from scratch, in order to learn a classifier with less bias towards the head categories. Both stages use hard labels to supervise the feature representation network and classifier. We argue that using hard labels in these two stages is not the best choice. The study of Müller et al. (2019) shows that soft labels can improve the model in the feature space, to make clusters of the same categories more compact and clusters of different categories more dispersed, and it can reduce the model's overconfidence in the sample prediction. Clusters of different categories in the feature space become more dispersed, which is considered as a better general feature representation, and this decrease in overconfidence helps to reduce the model's bias towards the head categories in the second stage. Based on the above two reasons, we believe that the supervised signal based on soft labels is more suitable for decoupled learning than hard labels.

We try to generate suitable soft labels for the decoupled learning framework. We first consider Label Smoothing (Szegedy et al., 2016), a simple soft labels generation method for recognition problems. For an image classification task with $K$ categories, we have a training dataset with $n$ samples $D_{train} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, where $x_i$ is an image, $y_i$ is the correct categories corresponding to the image $x_i$, where $y_i \in \{1, 2, \ldots, K\}$. For each sample $(x_i, y_i)$, if we use hard labels, it means that the label $y_i$ is encoded by one-hot encoding. It converts the scalar $y_i$ into a vector $\boldsymbol{y_i}$, where $\boldsymbol{y_i} \in \mathbb{R}^K$, $\boldsymbol{y_i} = (q_i^1, q_i^2, \ldots, q_i^K)$, where $q_i^j \in \{0, 1\}$. If the label $y_i = k$, then there is

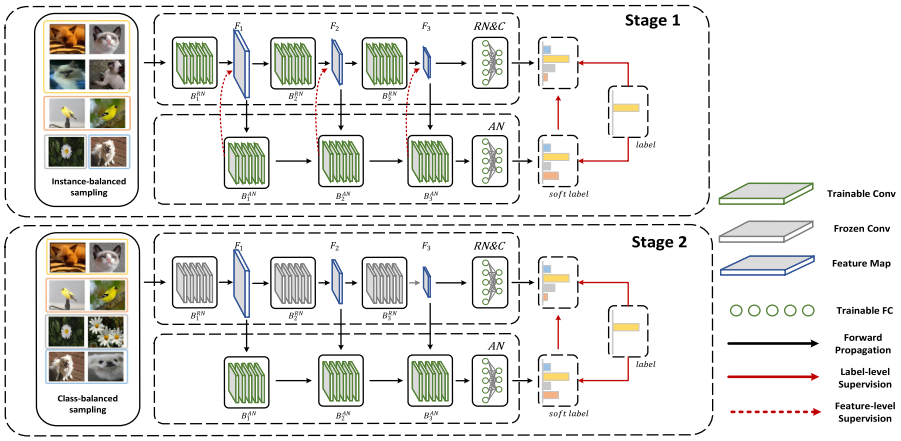


**Fig. 2.** Our framework for long-tailed recognition, Dynamic Auxiliary Soft Label (DaLS). An auxiliary network is designed to generate auxiliary soft labels for the two training stages. The stage 1 is the feature representation learning stage, and stage 2 is the classifier learning stage.

$q_i^{j=k} = 1$, and there is $q_i^{j\neq k} = 0$. Label Smoothing (Szegedy et al., 2016) changes the encoding method for the vector $\boldsymbol{y_i}$ as

$$q_i^j = \begin{cases} 1-\varepsilon & \text{if } j = k \\ \varepsilon/(K-1) & \text{otherwise} \end{cases}. \tag{1}$$

$\varepsilon$ is a small constant used to adjust the smoothing ratio of the soft labels, usually with $\varepsilon = 0.1$. We use Label Smoothing in the first stage, the second stage, and the two stages of decoupled learning. We find that Label Smoothing has an adverse effect on the model, the experimental results of which are shown in Section 4.4.5. This simple soft labels method does not have a positive impact on the model. The possible reason is that Label Smoothing will use uniform distribution when assigning labels to non-label classes. This uniform distribution intensifies the model's bias towards head categories. Therefore, we consider using a non-uniform soft labels generation method.

Zhang et al. (2020) considered the true relationship between different categories and designed Online Label Smoothing (OLS), which is a non-uniformly distributed soft labels generation method. It defines a soft labels matrix $S$, where $S \in \mathbb{R}^{K\times K}$, $S = \{S_1, S_2 \ldots S_K\}$, where one column $S_i$ in $S$ corresponds to $i$th categories of soft labels. OLS uses the model learned in the last epoch to generate the soft labels needed for this epoch. For the sample $(x_i, y_i)$, if the prediction is correct, the soft labels matrix will be updated:

$$S_{y_i,k}^t = S_{y_i,k}^t + M_{last}\left(k \mid \boldsymbol{x_i}\right). \tag{2}$$

$k$ is the true label of the sample, and $M_{last}$ is the model after the last epoch parameter update. After the current epoch is over, OLS normalizes the soft labels matrix for supervised training in the next epoch:

$$L_{\text{soft}} = -\sum_{k=1}^{K} S_{y_i,k}^{t-1} \cdot \log p\left(k \mid \boldsymbol{x_i}\right). \tag{3}$$

We will also try the compatibility between OLS and decoupled learning. We apply OLS to the first stage and the both stages in the cRT. We found that OLS had unfavorable or insignificant effects on the model. The experimental results are shown in Section 4.4.5. From the perspective of decoupled learning, the reason why OLS is not applicable, is that cRT freezes the parameters of the feature representation network in the second stage. The feature indicates that the network has a great influence on the generation of soft labels. Therefore, the general feature representation is not suitable for generating soft labels for the second stage.

Based on the above two parts of the experiment, we judge that if we want to generate a suitable soft labels for the decoupled learning framework, we need to consider two factors. First, an adaptive method should be used to generate soft labels instead of using uniformly distributed soft labels. Second, in the process of label generation, the difference between the two training processes should be considered. Based on the above two considerations, we propose a method of dynamically generating auxiliary soft labels using an auxiliary network to generate soft labels for the two-stage training.

### 3.2. Dynamic auxiliary soft label

#### 3.2.1. Representation learning stage

The Dynamic Auxiliary Soft Label model consists of three parts. In addition to the feature representation network *RN* and the classifier *C* under the decoupled learning framework, we have designed an auxiliary network *AN* for the two-stage training to generate auxiliary soft labels. Due to the different training methods of the two stages, the auxiliary network has different characteristics in the two stages, and the corresponding soft labels can be generated for the feature representation network and the classifier of the original network in the two stages. The study by Furlanello, Lipton, Tschannen, Itti, and Anandkumar (2018) showed that self-distillation can use predictions of model to find correlations between categories. Our auxiliary network is similar to a teacher network, but without the complex ensemble structure. The auxiliary network structure will be introduced in Section 3.2.3.

In the representation learning stage, we use an instance-sampling method to train the feature representation network, classifier and auxiliary network at the same time. The groundtruth of the sample uses the form of hard labels to supervise the original network and auxiliary network. The auxiliary network uses intermediate feature maps to provide feature-level distillation for the feature representation network of the original network, and uses the generated soft labels to provide label-level auxiliary supervision for the classifier. Using two levels of supervision information helps the feature representation network learn a better general feature representation.

Specifically, in the representation learning stage, for an input sample $(x_i, y_i)$, we use the feature representation network *RN* to extract the feature maps of each layer of the input image $x_i$, $RN(x_i) = [F_i^1, F_i^2, \ldots, F_i^B]$, where $B$ is the number of convolutional block layers used by *RN*. The classifier *C* uses the last layer of features $F_i^B$ to predict the result, and obtains the prediction

distribution $s_c = (s_c^1, s_c^2, \ldots, s_c^K)$. We use the cross-entropy loss function to supervise the model using one-hot encoding of real labels:

$$L_{\text{gt2c}}(\boldsymbol{y_i}, \boldsymbol{s_c}) = -\sum_{j=1}^{K} y_i^j \log s_c^j. \tag{4}$$

After that, the auxiliary network uses the feature maps extracted by the feature representation network, $[F_i^1, F_i^2, \ldots, F_i^B]$. Then we get the auxiliary features $[FA_i^1, FA_i^2, \ldots, FA_i^B]$ and the predicted distribution of the sample $s_a = (s_a^1, s_a^2, \ldots, s_a^K)$. The auxiliary feature $FA_i^b$ corresponds to the original feature $F_i^b$ layer by layer. We also use the one-hot encoding of the groundtruth to supervise the auxiliary network:

$$L_{\text{gt2a}}(\boldsymbol{y_i}, \boldsymbol{s_a}) = -\sum_{j=1}^{K} y_i^j \log s_a^j. \tag{5}$$

In order to realize the improvement of feature representation learning by soft labels, we use the auxiliary network prediction distribution as the auxiliary soft labels of the original network for supervision. We refer to the loss function in Knowledge Distillation (Hinton et al., 2015), set the temperature coefficient $T$, and transform $s_a$ and $s_c$ into $\tilde{s}_a$ and $\tilde{s}_c$, where $\tilde{s_a^i} = \exp\left(s_a^i/T\right) / \sum_j \exp\left(s_a^j/T\right)$. The transformation of $\tilde{s_c^i}$ is the same, and the loss function method of the auxiliary soft labels is as follows:

$$L_{\text{a2c}}(\boldsymbol{s_a}, \boldsymbol{s_c}) = \sum_{j=1}^{K} \tilde{s_a^j} \log \frac{\tilde{s_a^j}}{\tilde{s_c^j}}. \tag{6}$$

Since the auxiliary network distills the feature maps of each layer in the original model, similar to the feature-level knowledge distillation, we refer to the method of Zagoruyko and Komodakis (2017) to establish the supervision between the auxiliary network features and the original network features:

$$L_{\text{a2r}}(\boldsymbol{FA}, \boldsymbol{F}) = \sum_{j=1}^{D} \left\| \frac{Tr(F_j)}{\left\|Tr(F_j)\right\|_2} - \frac{Tr(FA_j)}{\left\|Tr(FA_j)\right\|_2} \right\|_2. \tag{7}$$

The transformation $Tr$ is a kind of attention mapping (Zagoruyko & Komodakis, 2017). We use $Tr(F_b) = \sum_{i=1}^{Ch} \left|F_b^i\right|^p$, where $Ch$ is the number of channels in the feature map. We can get the auxiliary network's supervision to the original network as:

$$L_{\text{a2cr}}(\boldsymbol{s_a}, \boldsymbol{s_c}, \boldsymbol{FA}, \boldsymbol{F}) = L_{\text{a2c}}(\boldsymbol{s_a}, \boldsymbol{s_c}) + \beta L_{\text{a2r}}(\boldsymbol{FA}, \boldsymbol{F}). \tag{8}$$

In the feature representation learning stage, we can construct the objective function as:

$$L_{RepL} = L_{\text{gt2c}}(\boldsymbol{y_i}, \boldsymbol{s_c}) + L_{\text{gt2a}}(\boldsymbol{y_i}, \boldsymbol{s_a}) + L_{\text{a2cr}}(\boldsymbol{s_a}, \boldsymbol{s_c}, \boldsymbol{FA}, \boldsymbol{F}). \tag{9}$$

We use $L_{RepL}$ to realize the supervision of the real label on the original network and the supervision of the auxiliary network on the original network at the same time. The latter supervision includes label-level and feature-level, to improve the feature representation network for general feature learning.

### 3.2.2. Classifier learning stage

In the classifier learning stage, we get the representations, and use the re-balance method to train the classifier and auxiliary network. In the process of re-balance, we choose to use class-balanced sampling (Kang et al., 2020) for re-sampling, so that each class $j$ has the same probability $p_j$ of being selected.

$$p_j = \frac{n_j^q}{\sum_{i=1}^{K} n_i^q}. \tag{10}$$

$n_i$ is the number of samples in the $i$th categories, and we define $q = 0$. It can be regarded as a two-stage sampling. First, a category is uniformly selected from the class set, and then an instance is uniformly sampled from the categories.

Different from the representation learning stage, the parameters of the feature representation network have been fixed. Thus, the auxiliary network does not need to perform feature-level feature supervision on the feature representation network, but only includes the supervision of its soft labels on the classifier:

$$L_{\text{a2cr}}(\boldsymbol{s_a}, \boldsymbol{s_c}) = L_{\text{a2c}}(\boldsymbol{s_a}, \boldsymbol{s_c}). \tag{11}$$

We use one-hot encoded groundtruth to supervise the original network classifier and auxiliary network. Therefore, the objective function of the classifier learning stage can be constructed as:

$$L_{ClaL} = L_{\text{gt2c}}(\boldsymbol{y_i}, \boldsymbol{s_c}) + L_{\text{gt2a}}(\boldsymbol{y_i}, \boldsymbol{s_a}) + L_{\text{a2cr}}(\boldsymbol{s_a}, \boldsymbol{s_c}). \tag{12}$$

In the classifier learning stage, we use soft labels to reduce the overconfidence of the model prediction, so that the targets are flatter (He et al., 2021). Moreover, the model's bias is alleviated for the head categories.

### 3.2.3. Auxiliary model

Our auxiliary network is to provide suitable auxiliary supervision information for the two stages of decoupled learning, including label-level supervision and feature-level supervision. The input of the auxiliary network is the feature map of each layer extracted in the feature representation network $[F_i^1, F_i^2, \ldots, F_i^B]$, to get the predicted soft distribution $s_a = (s_a^1, s_a^2, \ldots, s_a^K)$ of the sample $x_i$. Furthermore, we use the middle feature layer to supervise the feature representation network.

In order to let the auxiliary network get more effective middle layer features, we use the multi-scale feature fusion method. We refer to the existing multi-scale feature fusion methods, FPN (Lin et al., 2017), PANet (Liu, Qi, Qin, Shi, & Jia, 2018), and BiFPN (Tan, Pang and Le, 2020). We design three kinds of auxiliary network designs, that include Single-way fusion structure, two-way fusion structure, and complex two-way fusion structure. The specific structures are shown in Fig. 3.

**Single-way fusion structure:** The Single-way fusion structure only has one path from low-level features to high-level features, which is a bottom-up path. The auxiliary network with Single-way fusion structure includes two sets of convolution $Conv_{L1}$, $Conv_{L2}$ and a classifier. $Conv_{L1}$, $Conv_{L2}$ contain $B$ convolution operations respectively, for example $Conv_{L1} = [Conv_{L1}^1, Conv_{L1}^2, \ldots, Conv_{L1}^B]$. For the features of each layer, we use first convolution $Conv_{L1}$ for processing, and then use $Conv_{L2}$ to form a bottom-up fusion path. The output of $Conv_{L2}^B$ is finally used to predict the features of the soft labels, and the features of each layer of $Conv_{L2}$ are used to distill the features of the original network.

**Two-way fusion structure:** The two-way fusion structure is based on the one-way fusion structure, with a path added from high-level features to low-level features, which is a top-down path. Specifically, we add a set of convolution operations, including $Conv_{L1}$, $Conv_{L2}$ and $Conv_{L3}$. We use $Conv_{L1}$ for preliminary processing of the original input features, and then implement a top-down fusion path on $Conv_{L2}$. Finally, bottom-up fusion is realized on $Conv_{L3}$, and each layer feature of $Conv_{L3}$ is used to supervise the original network.

**Complex two-way fusion structure:** The complex two-way fusion structure is based on the two-way fusion structure. It also uses three sets of convolution $Conv_{L1}$, $Conv_{L2}$ and $Conv_{L3}$ which increases Cross-Scale Connections (Tan, Pang et al., 2020), and weighted skip connection. For the intermediate convolutional layer, we no longer directly use the output of $Conv_{L1}$ as the input of the convolution, but use Cross-Scale Connections to achieve

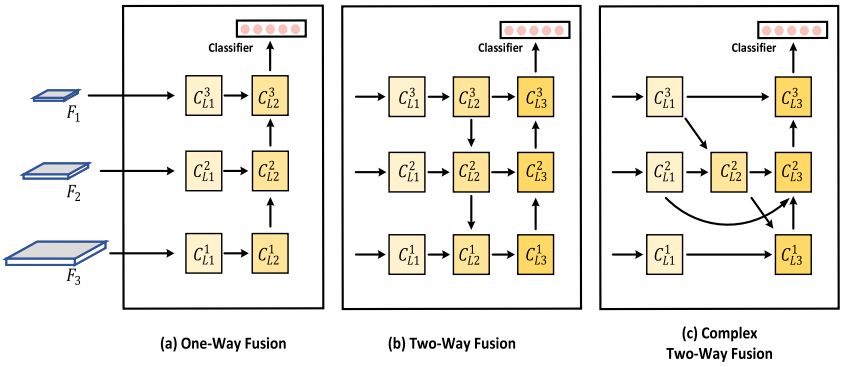


**Fig. 3.** Illustration of three auxiliary network structures considered in this paper. (a) Single-way fusion structure. (b) Two-way fusion structure. (c) Complex two-way fusion structure.

top-down fusion. The specific operations of $Conv_{L2}$ are as follows:

$$Conv_2^i = \text{Conv}(w_{i,1}^{C2} \cdot Conv_1^i + w_{i,2}^{C2} \cdot \text{ Resize } \left(Conv_2^{i+1}\right)). \tag{13}$$

At the $Conv_{L3}$ layer, we use the structure of skip connections and weight the fusion features to achieve bottom-up fusion. The specific operations of $Conv_{L3}$ are as follows:

$$\begin{aligned} Conv_{L3}^i = \text{Conv}(&w_{i,1}^{C3} \cdot Conv_{L1}^i + w_{i,2}^{C2} \cdot Conv_{L2}^i \\ &+ w_{i,3}^{C3} \cdot \text{ Resize } \left(Conv_{L3}^{i-1}\right)). \end{aligned} \tag{14}$$

We also use the features of each layer of $Conv_{L3}$ to supervise the original network. In Section 4.4.2, we introduce the comparative experimental results of the three structures in detail, which indicates the auxiliary network with the complex two-way structure has the best performance.

## 4. Experiments

We demonstrate the effectiveness of our DaSL on multiple long-tail datasets, with the datasets and implementation details in Section 4.1, our comparison method in Section 4.2, our main experimental results in Section 4.3, and a series of results of our ablation experiments in Section 4.4. Finally, we further analyze our DaSL model in Section 4.5.

### 4.1. Datasets and implementations

We evaluate our model on the following three datasets, including CIFAR-10-LT, CIFAR-100-LT and ImageNet-LT.

#### 4.1.1. Datasets

**CIFAR-10-LT & CIFAR-100-LT** CIFAR-10 and CIFAR-100 are both typical datasets for image classification tasks containing 10 classes and 100 classes respectively. They both have 50 000 samples for training and 10 000 samples for verification. We use the long-tailed version of two datasets. The sampling method follows the previous work (Cao et al., 2019), by setting the imbalanced factor $IF$ to adjust the imbalance of the training samples in the dataset. $IF = N_{min}/N_{max}$, where $N_{min}$ and $N_{max}$ represent the number of samples with the smallest and largest sampling rate, respectively. For the two datasets, we set $IF \in \{10, 50, 100\}$.

**ImageNet-LT** ImageNet-LT is a long-tail version of ImageNet. The sampling method follows the previous work (Liu et al., 2019), using the Pareto distribution with the power value $\alpha = 6$ for sampling. ImageNet-LT contains 115 846 images for training, with a total of 1000 categories. Among them, the maximum sample size of the same categories is 1280, and the minimum is 5, so the imbalanced factor is 256.

**iNaturalist 2018** iNaturalist 2018 is a large-scale real-world dataset for species classification, which suffers from extremely imbalanced label distribution. It is composed of 437 513 images from 8142 categories. Besides the severe long-tail problems, iNaturalist 2018 also faces the fine-grained problem.

#### 4.1.2. Implementation details

For CIFAR-10-LT and CIFAR-100-LT, we use ResNet-32 as the backbone, set the batch size to 128, use SGD with 0.9 moment to train the model, set the initial learning rate to 0.2, and set weight decay to 2e–4. A total of 200 epochs are trained, including 160 epochs for the first stage training and 40 epochs for the second stage training. For the first stage, we use warmup in the first 5 epochs, and use 0.1 attenuation at the 140th and the 150th epoch. For ImageNet-LT, we use ResNet-10 as the backbone, set the batch size to 512, and also use SGD with 0.9 moment to train the model, set the initial learning rate to 0.2. We train a total of 100 epochs, including the first stage training for 90 epochs, the second stage training for 10 epochs. For the first stage, we use 0.1 attenuation at the 60th and the 80th epoch, and do not use warmup. For the two datasets, when the auxiliary network supervises the original network, we both set $\beta = 100$. Our training for backbone is done from scratch, and after backbone, we use GAP structure to reduce the feature dimension. For fair comparisons, we also use ResNeXt-50 as the backbone. We train our model for 90 epochs with batch size 256. For iNaturalist 2018, we use ResNet-50 as the backbone, use SGD to train the model, set the initial learning rate to 0.2. We train a total of 90 epochs, including the first stage training for 80 epochs, the second stage training for 10 epochs. For the first stage, we use 0.1 attenuation at the 30th and the 60th epoch, and also do not use warmup.

### 4.2. Comparison methods

In this section, we define baseline methods and briefly introduce the model for comparison.

**Baseline methods** This paper uses the decoupled learning model as the baseline. In the first stage it uses instance-based sampling for training, and in the second stage it uses the class-balance-based balanced sampling method for training. Both stages use cross entropy as the loss function.

**Competing methods** We also compare our model with the latest models of different kinds, including re-weight based methods: Focal (Lin et al., 2020), Class-Balanced Loss (Cui et al., 2019), LDAM-DRW (Cao et al., 2019), BSCE, transfer learning based methods: LDAM-M2 m (Kim, Jeong et al., 2020), Meta-learning (Jamal et al., 2020), decoupled learning based methods: BBN (Zhou et al., 2020), LWS (Kang et al., 2020), and some other methods: TDE (Tang, Huang, & Zhang, 2020) and FSA (Chu, Bian, Liu, & Ling, 2020b).

**Table 1**
Top-1 accuracy (%) for ResNet-32 based models trained on CIFAR-10-LT.

| CIFAR-10-LT | | | |
|---|---|---|---|
| Models | IF = 10 | IF = 50 | IF = 100 |
| Focal (Lin et al., 2020) | 86.66 | 76.71 | 70.38 |
| Class-balanced loss (Cui et al., 2019) | 86.40 | 77.73 | 72.11 |
| LDAM-DRW (Cao et al., 2019) | 88.16 | 81.27 | 77.03 |
| MW-Net (Shu et al., 2019) | 87.84 | 80.06 | 75.21 |
| LDAM-M2 m (Kim, Jeong and Shin, 2020) | 87.5 | – | 79.1 |
| BBN (Zhou et al., 2020) | 88.32 | 82.18 | 79.82 |
| Meta-learning (Jamal et al., 2020) | 88.37 | 82.88 | 78.90 |
| Baseline + mixup | 88.01 | 83.52 | 78.90 |
| DaLS | **89.21** | **84.43** | **79.91** |

**Table 2**
Top-1 accuracy (%) for ResNet-32 based models trained on CIFAR-100-LT.

| CIFAR-100-LT | | | |
|---|---|---|---|
| Models | IF = 10 | IF = 50 | IF = 100 |
| Focal (Lin et al., 2020) | 56.51 | 42.41 | 38.35 |
| Class-balanced loss (Cui et al., 2019) | 54.87 | 38.57 | 32.65 |
| LDAM-DRW (Cao et al., 2019) | 58.71 | 46.62 | 42.04 |
| BSCE | 58.38 | 47.60 | 42.39 |
| BBN (Zhou et al., 2020) | 59.12 | 47.02 | 42.60 |
| TDE (Tang et al., 2020) | 59.60 | 50.30 | 44.10 |
| DiVE (He et al., 2021) | 62.00 | 51.13 | 45.35 |
| SSD (Li, Wang et al., 2021) | **62.30** | 50.50 | 46.00 |
| Baseline + mixup | 60.42 | 50.72 | 45.80 |
| DaLS | 61.34 | **51.90** | **47.68** |

**Table 3**
Top-1 accuracy (%) for ResNet-10/ResNeXt-50 based models trained on ImageNet-LT.

| ImageNet-LT | | |
|---|---|---|
| Models | ResNet-10 | ResNeXt-50 |
| OLTR (Liu et al., 2019) | 35.60 | 46.70 |
| LWS (Kang et al., 2020) | 41.40 | 49.90 |
| LFME+OLTR (Xiang et al., 2020) | 38.80 | 47.00 |
| FSA (Chu et al., 2020a) | 35.20 | – |
| BALMS (Ren et al., 2020) | 41.80 | – |
| MiSLAS (Zhong et al., 2021) | – | 51.40 |
| Baseline + mixup | 39.01 | 48.82 |
| DaLS | **42.43** | **51.71** |

**Table 4**
Top-1 accuracy (%) for ResNet-50 based models trained on iNaturalist 2018.

| iNaturalist 2018 | |
|---|---|
| Models | Top1 Acc |
| CB-Focal (Cui et al., 2019) | 61.12 |
| LWS (Kang et al., 2020) | 65.90 |
| FSA (Chu et al., 2020a) | 65.91 |
| Meta-learning (Jamal et al., 2020) | 67.55 |
| BBN (Zhou et al., 2020) | 66.29 |
| BBN(2× ) (Zhou et al., 2020) | **69.62** |
| DaLS | 67.83 |

### 4.3. Main results

**Results on CIFAR-10-LT & CIFAR-100-LT** For the two datasets CIFAR-10-LT and CIFAR-100-LT, we carried out extensive experiments under different imbalanced factor settings, with $IF = 10, 50, 100$. The experimental results of CIFAR-10-LT are summarized in Table 1, and the experimental results of CIFAR-100-LT are summarized in Table 2. The experimental results show that our proposed DaSL has better performance on the CIFAR-LT dataset compared with existing methods under different imbalanced factors. (see Fig. 4).

**Table 5**
Top-1 accuracy (%) on CIFAR-100-LT. We use DaSL in different stages of decoupled learning.

| CIFAR-100-LT | | | |
|---|---|---|---|
| Models | IF = 10 | IF = 50 | IF = 100 |
| Baseline + mixup | 60.42 | 50.72 | 45.80 |
| Only 1-sta | 60.80 | 51.42 | 46.92 |
| Only 2-sta | 60.93 | 51.60 | 47.21 |
| 1-sta & 2-sta | **61.34** | **51.90** | **47.68** |

**Results on ImageNet-LT** We further verify the effectiveness of our proposed model on a large dataset ImageNet-LT, and the experimental results are summarized in Table 3. We use both ResNet-10 and ResNeXt-50 as the backbone like the comparison method. Based on the same experimental design, our model has better performance on large datasets.

**Results on iNaturalist 2018** We further verify the effectiveness of DaSL on iNaturalist 2018, and the experimental results are summarized in Table 4. We use ResNet-50 as the backbone and train 90 epochs like the comparison method. The results show that our model has good performance on large-scale real-world datasets.

### 4.4. Ablation experiments

#### 4.4.1. Assisting the label network to supervise experiments in different stages of supervision

Our model uses the auxiliary network to generate soft labels for the two stages of decoupled learning for auxiliary supervision. We hope that the soft labels classifier learning stage will reduce the model's bias towards the head categories by reducing the overconfidence of the model prediction. In the first stage, through a more reasonable auxiliary distribution, the feature representation network can learn more aggregated in-class features. In order to verify that our model has a positive effect on the two training stages at the same time, we use the auxiliary network

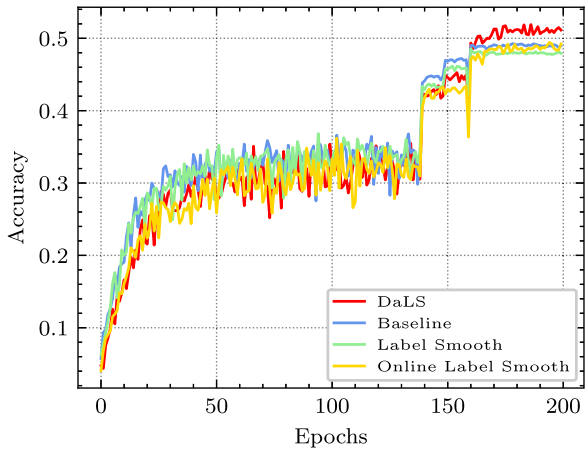


**Fig. 4.** Compare the training process of different three models, illustration of the accuracy curves by Label Smoothing, Online Label Smoothing and DaSL.

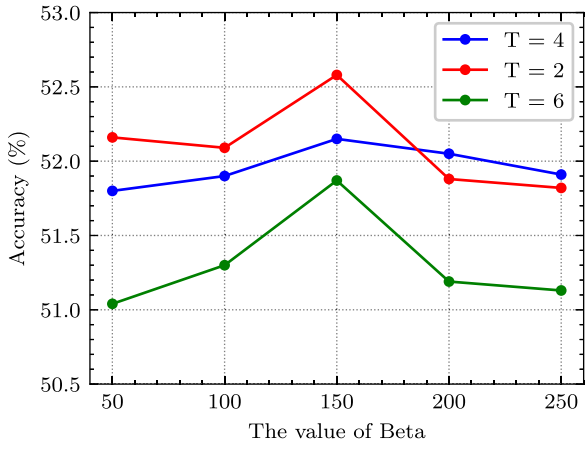


**Fig. 5.** Impact of $\beta$ and $T$ in DaLS, performance is reported on CIFAR-100-LT.

**Table 6**
Top-1 accuracy (%) on CIFAR-100-LT. We compare three different auxiliary models.

| CIFAR-100-LT | |
|---|---|
| Models | IF = 50 |
| Baseline + mixup | 50.72 |
| Single-way fusion | 51.52 |
| Two-way fusion | 51.76 |
| Complex two-way fusion | **51.90** |

to generate soft labels only for the first stage, only for the second stage, and for both stages. We conduct control experiments under the three imbalanced factors of CIFAR-100-LT ($IF = 10, 50, 100$). The experimental results are shown in Table 5. Through the experimental results, we find that using soft labels only in a single stage can have a positive effect compared to the baseline. Moreover, when the soft labels are used as auxiliary supervision in both stages, the model has the best performance, which can prove that our auxiliary soft labels are useful for the original model in both stages.

*4.4.2. Auxiliary network model's impact on performance*

For the auxiliary model, we design three structures, which are distributed as single-way fusion structure, two-way fusion structure and complex two-way fusion structure. In Table 6, we report the experimental results of the three structures on the CIFAR-100-LT dataset with $IF = 50$, and the parameter statistics

**Table 7**
Top-1 accuracy (%) on CIFAR-100-LT. We compare our DaLS with Label Smoothing (LS) and Online Label Smoothing (OLS).

| CIFAR-100-LT | | | |
|---|---|---|---|
| Models | IF = 10 | IF = 50 | IF = 100 |
| Baseline + mixup | 60.42 | 50.72 | 45.80 |
| LS 1-sta | 60.10 | 50.44 | 45.35 |
| LS 2-sta | 60.02 | 50.65 | 45.68 |
| LS 1&2-sta | 59.93 | 50.38 | 45.32 |
| OLS 1-sta | 60.51 | 50.76 | 45.78 |
| OLS 1&2-sta | 60.36 | 50.58 | 45.71 |
| DaSL | **61.34** | **51.90** | **47.68** |

of the model. Through the experimental results, it can be found that even using the simplest one-way fusion mechanism as the auxiliary network can still help the original network to achieve the improvement of the model performance. Using complex two-way fusion as the auxiliary network can further improve the performance of the model.

*4.4.3. Experiments comparing our method with LS and DLS*

In order to illustrate the effectiveness of the soft labels we designed compared to the existing soft labels method, we conduct a comparative experiment on the CIFAR-100-LT dataset, and compare them under three imbalanced factors ($IF \in \{10, 50, 100\}$). We conduct detailed experiments to mainly compare the comparative effects of DaLS and two soft labels generation methods: Label Smoothing (Szegedy et al., 2016) and Online Label Smoothing (Zhang et al., 2020). As shown in Section 3.1, we use Label Smoothing in the first, second and both stages of decoupled learning, and use Online Label smoothing in the first and both stages respectively. The experimental results are summarized in Table 7. Through experiments, we can find that no matter how Label Smoothing is used, it cannot have a positive impact on the original model. We find that if Online Label Smoothing is only used in the first stage, it can have a small positive effect on the original model. But if Online Label smoothing is used at both stages, it will have an adverse effect on the original model. This also validates our idea that in the second stage, if a biased model is used to generate soft labels for the original network, it will not help the classifier to relieve the bias.

*4.4.4. The influence of loss fusion ratio*

In Section 3.2.1, when using auxiliary network to perform label-level and feature-level supervision on the original network features, we introduce the hyperparameter $\beta$ and $T$. In this part, we explore the impact of hyperparameter $\beta$, $T$ on the model. We use ResNet-32 as the backbone to conduct experiments on the CIFAR-100-LT dataset with $IF = 50$, $\beta \in \{50, 100, 150, 200, 250\}$, $T \in \{2, 4, 6\}$ and other parameters unchanged. The experimental results are shown in Fig. 5. It is found through experiments that different $\beta$ and $T$ affect the performance of the model to a certain extent. Furthermore, for different datasets, the trend of the impact of $\beta$ on the model performance is not the same.

*4.4.5. The influence of decoupled learning*

We use a decoupled learning framework as the learning strategy for the DaSL, which separates the representation learning stage from the classifier learning stage. In order to verify the adaptability of coupled learning and DaSL, we compare the performance of model using decoupled learning with that using only a single kind of sampling strategy. In Table 8, we report the experimental results on the CIFAR-100-LT dataset, which compares our model with that using instance-balanced sampling or class-balanced sampling in both two stages. Through the experimental

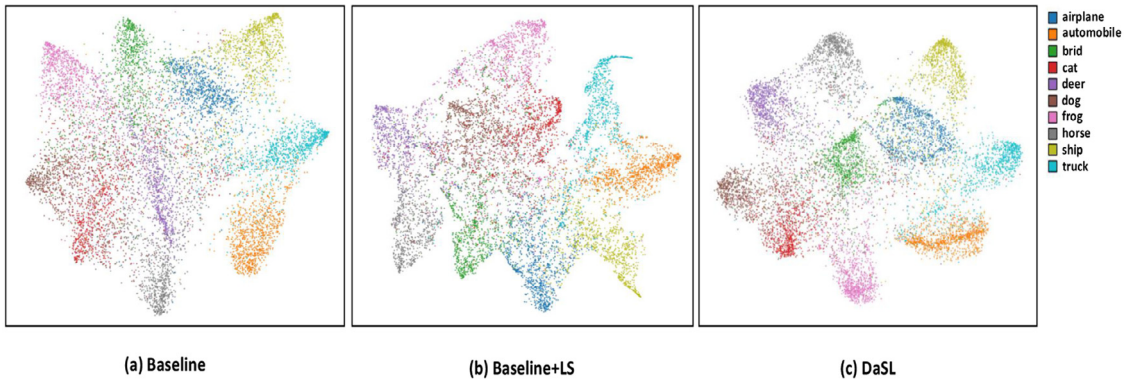


**Fig. 6.** Use t-SNE to visualize the feature distribution of the various methods.

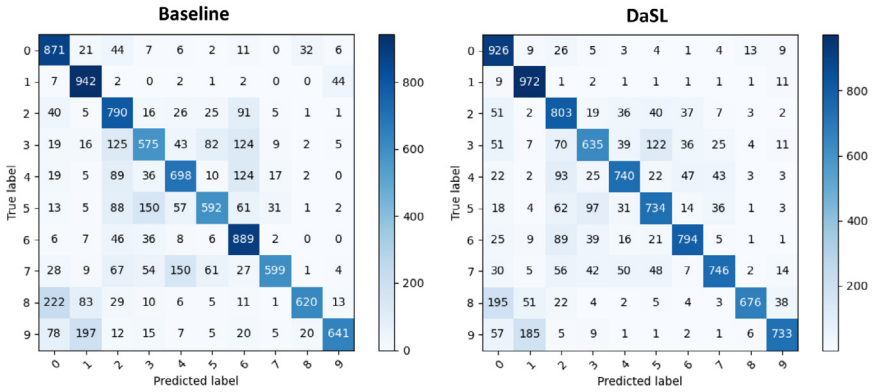


**Fig. 7.** Illustration of the confusion matrices by baseline method and DaSL, performance is reported on CIFAR-10-LT.

**Table 8**
Top-1 accuracy (%) on CIFAR-100-LT. We compare our DaLS with model using single kind of sampling strategy.

| CIFAR-100-LT | | | |
|---|---|---|---|
| Models | IF = 10 | IF = 50 | IF = 100 |
| Only instance-balanced sampling | 59.12 | 44.91 | 41.17 |
| Only class-balanced sampling | 60.46 | 48.39 | 45.35 |
| Decoupled learning | **61.34** | **51.90** | **47.68** |

results, it can be found that using decoupled learning can help the model solve the long-tailed distribution problem better than using the same sampling strategy in the two stages. Moreover, when the data becomes more imbalanced ($IF \in \{10, 50\} \rightarrow IF \in \{100\}$), decoupled learning brings larger improvements than other settings, which further reflects the effectiveness of the decoupled learning strategy for long-tailed problems.

### 4.5. Further analyses on our method

In order to further illustrate the effectiveness of our DaLS, we use visualization methods to analyze it. We use t-SNE diagrams to analyze the influence of DaSL on feature distribution, and use confusion matrix to analyze the degree of DaSL's attention to different categories.

Fig. 6 shows the feature distribution obtained by using t-SNE to compare different training models. We conduct experiments on the CIFAR10-LT ($\beta = 100$) dataset, and use t-SNE to show the feature distribution of 10 categories of validation samples. The distribution includes two training stages using hard labels, both using Label Smoothing, and both using the DaSL method. Through the visual display, we can find that using the soft labels generated by DaSL to assist in training the model helps the model generate a more aggregated feature distribution with a smaller variance for similar samples in the stage of feature representation learning.

Fig. 7 shows that we use the confusion matrix to show the predictions of the two models for different categories. We also conduct experiments on the CIFAR10-LT ($\beta = 100$) dataset. Through comparison, we can intuitively find that our model can help the original model to pay more attention to the tail categories and improve the prediction accuracy of several tail categories.

## 5. Conclusion

In this work, we explored how to design an effective soft labels generation method for the decoupled learning framework for long tail recognition. We designed an auxiliary network to generate auxiliary soft labels for the two training stages at the same time. Moreover, in order to help feature representation learning and classifier learning at the same time, we used the intermediate features of the auxiliary network to perform feature-level distillation of the original network in the first stage. At the same time, we used visualization methods to prove that our model extracts more aggregated in-class features, and pays more attention to the tail categories. Experimental results on multiple long-tail benchmarks show that our DaSL has the best performance.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgment

This work is supported in part by the National Natural Science Foundation of China under Grant Nos. (61876076).

## References


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

Byrd, J., & Lipton, Z. C. (2019). What is the effect of importance weighting in deep learning? In K. Chaudhuri, & R. Salakhutdinov (Eds.), *Proceedings of machine learning research*: *Vol. 97*, *Icml 2019, 9-15 June 2019, Long Beach, California, USA* (pp. 872–881). PMLR.

Cao, K., Wei, C., Gaidon, A., Aréchiga, N., & Ma, T. (2019). Learning imbalanced datasets with label-distribution-aware margin loss. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, & R. Garnett (Eds.), *Neurips 2019, December 8-14, 2019, Vancouver, BC, Canada* (pp. 1565–1576).

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal for Artificial Intelligence Research*, 321–357.

Chou, H., Chang, S., Pan, J., Wei, W., & Juan, D. (2020). Remix: Rebalanced mixup. In A. Bartoli, & A. Fusiello (Eds.), *Lecture notes in computer science*: *Vol. 12540*, *Computer vision - ECCV 2020 workshops - Glasgow, UK, August 23-28, 2020, proceedings, part VI* (pp. 95–110). Springer, http://dx.doi.org/10.1007/978-3-030-65414-6_9.

Chu, P., Bian, X., Liu, S., & Ling, H. (2020a). Feature space augmentation for long-tailed data. In A. Vedaldi, H. Bischof, T. Brox, & J. Frahm (Eds.), *Lecture notes in computer science*: *Vol. 12374*, *Eccv 2020, Glasgow, UK, August 23-28, 2020, proceedings, part XXIX* (pp. 694–710). Springer.

Chu, P., Bian, X., Liu, S., & Ling, H. (2020b). Feature space augmentation for long-tailed data. In A. Vedaldi, H. Bischof, T. Brox, & J. Frahm (Eds.), *Lecture notes in computer science*: *Vol. 12374*, *Eccv 2020, Glasgow, UK, August 23-28, 2020, proceedings, part XXIX* (pp. 694–710). Springer.

Cui, Y., Jia, M., Lin, T., Song, Y., & Belongie, S. J. (2019). Class-balanced loss based on effective number of samples. In *IEEE conference on computer vision and pattern recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019* (pp. 9268–9277). Computer Vision Foundation.

Cui, Y., Song, Y., Sun, C., Howard, A., & Belongie, S. J. (2018). Large scale fine-grained categorization and domain-specific transfer learning. In *CVPR 2018, salt lake city, ut, usa, june 18-22, 2018* (pp. 4109–4118). IEEE Computer Society.

Dong, Q., Gong, S., & Zhu, X. (2017). Class rectification hard mining for imbalanced deep learning. In *Iccv 2017, Venice, Italy, October 22-29, 2017* (pp. 1869–1878). IEEE Computer Society.

Furlanello, T., Lipton, Z. C., Tschannen, M., Itti, L., & Anandkumar, A. (2018). Born-again neural networks. In J. G. Dy, & A. Krause (Eds.), *Proceedings of machine learning research*: *Vol. 80*, *Icml 2018, StockholmsmäSsan, Stockholm, Sweden, July 10-15, 2018* (pp. 1602–1611). PMLR.

Gupta, A., Dollár, P., & Girshick, R. B. (2019). LVIS: a dataset for large vocabulary instance segmentation. In *IEEE conference on computer vision and pattern recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019* (pp. 5356–5364). Computer Vision Foundation.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(9), 1263–1284.

He, Y., Wu, J., & Wei, X. (2021). Distilling virtual examples for long-tailed recognition. CoRR 2103.15042 arXiv:2103.15042.

Hinton, G. E., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. CoRR 1503.02531 arXiv:1503.02531.

Jamal, M. A., Brown, M., Yang, M., Wang, L., & Gong, B. (2020). Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective. In *Cvpr 2020, Seattle, WA, USA, June 13-19, 2020* (pp. 7607–7616). IEEE.

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

Kang, B., Xie, S., Rohrbach, M., Yan, Z., Gordo, A., Feng, J., et al. (2020). Decoupling representation and classifier for long-tailed recognition. In *Iclr 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Kim, J., Jeong, J., & Shin, J. (2020). M2m: Imbalanced classification via major-to-minor translation. In *Cvpr 2020, Seattle, WA, USA, June 13-19, 2020* (pp. 13893–13902). IEEE.

Kim, K., Ji, B., Yoon, D., & Hwang, S. (2020). Self-knowledge distillation: A simple way for better generalization. CoRR 2006.12000 arXiv:2006.12000.

Lee, H., Lee, H., Na, D., Kim, S., Park, M., Yang, E., et al. (2020). Learning to balance: Bayesian meta-learning for imbalanced and out-of-distribution tasks. In *Iclr 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Li, S., Gong, K., Liu, C. H., Wang, Y., Qiao, F., & Cheng, X. (2021). Metasaug: Meta semantic augmentation for long-tailed visual recognition. CoRR 2103.12579 arXiv:2103.12579.

Li, B., Liu, Y., & Wang, X. (2019). Gradient harmonized single-stage detector. In *Aaai 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019* (pp. 8577–8584). AAAI Press.

Li, T., Wang, L., & Wu, G. (2021). Self supervision to distillation for long-tailed visual recognition. CoRR abs/2109.0407 arXiv:2109.04075.

Lin, T., Dollár, P., Girshick, R. B., He, K., Hariharan, B., & Belongie, S. J. (2017). Feature pyramid networks for object detection. In *Cvpr 2017, Honolulu, HI, USA, July 21-26, 2017* (pp. 936–944). IEEE Computer Society.

Lin, T., Goyal, P., Girshick, R. B., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Iccv 2017, Venice, Italy, October 22-29, 2017* (pp. 2999–3007). IEEE Computer Society.

Lin, T.-Y., Goyal, P., Girshick, R. B., He, K., & Dollár, P. (2020). Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(2), 318–327.

Lin, T., Maire, M., Belongie, S. J., Hays, J., Perona, P., Ramanan, D., et al. (2014). Microsoft COCO: common objects in context. In D. J. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), *Lecture notes in computer science*: *Vol. 8693*, *ECCV 2014, zurich, switzerland, september 6-12, 2014, proceedings, part V* (pp. 740–755). Springer.

Liu, B., Li, H., Kang, H., Hua, G., & Vasconcelos, N. (2021). Gistnet: a geometric structure transfer network for long-tailed recognition. CoRR abs/2105.00131 arXiv:2105.00131.

Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., & Yu, S. X. (2019). Large-scale long-tailed recognition in an open world. In *Cvpr 2019, Long Beach, CA, USA, June 16-20, 2019* (pp. 2537–2546). Computer Vision Foundation / IEEE.

Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018). Path aggregation network for instance segmentation. In *Cvpr 2018, Salt Lake City, UT, USA, June 18-22, 2018* (pp. 8759–8768). IEEE Computer Society.

Liu, J., Sun, Y., Han, C., Dou, Z., & Li, W. (2020). Deep representation learning on long-tailed data: A learnable embedding augmentation perspective. In *Cvpr 2020, Seattle, WA, USA, June 13-19, 2020* (pp. 2967–2976). IEEE.

Müller, R., Kornblith, S., & Hinton, G. E. (2019). When does label smoothing help? In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, & R. Garnett (Eds.), *Neurips 2019, December 8-14, 2019, Vancouver, BC, Canada* (pp. 4696–4705).

Ren, J., Yu, C., Sheng, S., Ma, X., Zhao, H., Yi, S., et al. (2020). Balanced meta-softmax for long-tailed visual recognition. In *Neurips 2020, December 6-12, 2020, virtual*.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet large scale visual recognition challenge. *115*, (3), (pp. 211–252).

Shen, L., Lin, Z., & Huang, Q. (2016). Relay backpropagation for effective learning of deep convolutional neural networks. In B. Leibe, J. Matas, N. Sebe, & M. Welling (Eds.), *Lecture notes in computer science*: *Vol. 9911*, *Eccv 2016, Amsterdam, the Netherlands, October 11-14, 2016, proceedings, part VII* (pp. 467–482). Springer.

Shen, Z., Liu, Z., Xu, D., Chen, Z., Cheng, K., & Savvides, M. (2021). Is label smoothing truly incompatible with knowledge distillation: An empirical study. CoRR 2104.00676 arXiv:2104.00676.

Shu, J., Xie, Q., Yi, L., Zhao, Q., Zhou, S., Xu, Z., et al. (2019). Meta-weight-net: Learning an explicit mapping for sample weighting. In *Neurips 2019, December 8-14, 2019, Vancouver, BC, Canada* (pp. 1917–1928).

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016* (pp. 2818–2826). IEEE Computer Society.

Tan, M., Pang, R., & Le, Q. V. (2020). Efficientdet: Scalable and efficient object detection. In *Cvpr 2020, Seattle, WA, USA, June 13-19, 2020* (pp. 10778–10787). IEEE.

Tan, J., Wang, C., Li, B., Li, Q., Ouyang, W., Yin, C., et al. (2020). Equalization loss for long-tailed object recognition. In *Cvpr 2020, Seattle, WA, USA, June 13-19, 2020* (pp. 11659–11668). IEEE.

Tang, K., Huang, J., & Zhang, H. (2020). Long-tailed classification by keeping the good and removing the bad momentum causal effect. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Neurips 2020, December 6-12, 2020, virtual*.

Wang, T., Li, Y., Kang, B., Li, J., Liew, J. H., Tang, S., et al. (2020). In A. Vedaldi, H. Bischof, T. Brox, J. Frahm (Eds.), *Lecture notes in computer science*: *Vol. 12359*, *The devil is in classification: A simple framework for long-tail instance segmentation* (pp. 728–744). Springer.

Wang, Y., Ramanan, D., & Hebert, M. (2017). Learning to model the tail. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, & R. Garnett (Eds.), *Neurips 2017, December 4-9, 2017, Long Beach, CA, USA* (pp. 7029–7039).

Wang, J., Zhang, W., Zang, Y., Cao, Y., Pang, J., Gong, T., et al. (2020). Seesaw loss for long-tailed instance segmentation. CoRR abs/2008.10032 arXiv:2008.10032.

Xiang, L., Ding, G., & Han, J. (2020). Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In A. Vedaldi, H. Bischof, T. Brox, & J. Frahm (Eds.), *Lecture notes in computer science*: *Vol. 12350*, *Eccv 2020, Glasgow, UK, August 23-28, 2020, proceedings, part V* (pp. 247–263). Springer.

Yin, X., Yu, X., Sohn, K., Liu, X., & Chandraker, M. (2019a). Feature transfer learning for face recognition with under-represented data. In *Cvpr 2019, Long Beach, CA, USA, June 16-20, 2019* (pp. 5704–5713). Computer Vision Foundation.

Yin, X., Yu, X., Sohn, K., Liu, X., & Chandraker, M. (2019b). Feature transfer learning for face recognition with under-represented data. In *Cvpr 2019, Long Beach, CA, USA, June 16-20, 2019* (pp. 5704–5713). Computer Vision Foundation.

Zagoruyko, S., & Komodakis, N. (2017). Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *Iclr 2017, Toulon, France, April 24-26, 2017, conference track proceedings*. OpenReview.net.

Zhang, X., Fang, Z., Wen, Y., Li, Z., & Qiao, Y. (2017). Range loss for deep face recognition with long-tailed training data. In *Iccv 2017, Venice, Italy, October 22-29, 2017* (pp. 5419–5428). IEEE Computer Society.

Zhang, C., Jiang, P., Hou, Q., Wei, Y., Han, Q., Li, Z., et al. (2020). Delving deep into label smoothing. CoRR 2011.12562 arXiv:2011.12562.

Zhang, S., Li, Z., Yan, S., He, X., & Sun, J. (2021). Distribution alignment: A unified framework for long-tail visual recognition. In *IEEE conference on computer vision and pattern recognition, CVPR 2021, virtual, June 19-25, 2021* (pp. 2361–2370). Computer Vision Foundation / IEEE.

Zhang, Y., Wei, X., Zhou, B., & Wu, J. (2021). Bag of tricks for long-tailed visual recognition with deep convolutional neural networks. In *Aaai*.

Zhong, Z., Cui, J., Liu, S., & Jia, J. (2021). Improving calibration for long-tailed recognition. In *IEEE conference on computer vision and pattern recognition, CVPR 2021, virtual, June 19-25, 2021* (pp. 16489–16498). Computer Vision Foundation / IEEE, URL https://openaccess.thecvf.com/content/CVPR2021/html/Zhong_Improving_Calibration_for_Long-Tailed_Recognition_CVPR_2021_paper.html.

Zhou, B., Cui, Q., Wei, X., & Chen, Z. (2020). BBN: bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Cvpr 2020, Seattle, WA, USA, June 13-19, 2020* (pp. 9716–9725). IEEE.

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., & Torralba, A. (2018). Places: A 10 million image database for scene recognition. *40*, In *IEEE Transactions on Pattern Analysis and Machine Intelligence* (6), (pp. 1452–1464).