Pattern Recognition 145 (2024) 109842

Contents lists available at ScienceDirect

Pattern Recognition

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

Hierarchical long-tailed classification based on multi-granularity knowledge transfer driven by multi-scale feature fusion

Wei Zhao, Hong Zhao *

School of Computer Science, Minnan Normal University, Zhangzhou, Fujian, 363000, China
Key Laboratory of Data Science and Intelligence Application, Fujian Province University, Zhangzhou, Fujian, 363000, China

ARTICLE INFO

Keywords:
Long-tailed learning
Hierarchical classification
Multi-granularity
Multi-scale feature fusion
Knowledge transfer

ABSTRACT

Long-tailed learning is attracting increasing attention due to the unbalanced distributions of real-world data. The aim is to train well-performing depth models. Traditional knowledge transfer methods for long-tailed learning are classified into feature-based horizontal knowledge transfer (HKT) and class-based vertical knowledge transfer (VKT). HKT transfers head-to-tail feature knowledge from different classes to improve classification performance when there are few tail classes. However, HKT easily leads to invalid transfer due to the deviation caused by the difference between the knowledge of head and tail classes. Fortunately, the class space has a multi-grained relationship and can form a multi-granularity knowledge graph (MGKG), which can be recast as coarse-grained and fine-grained losses to guide VKT. In this paper, we propose a hierarchical long-tailed classification method based on multi-granularity knowledge transfer (MGKT), which vertically transfers knowledge from coarse- to fine-grained classes. First, we exploit the semantic information of classes to construct an MGKG, which forms an affiliation of fine- and coarse-grained classes. Fine-grained knowledge can inherit coarse-grained knowledge to reduce transfer bias with the help of MGKG. We then propose a multi-scale feature fusion network, which aims to fully mine the rich information of the features to drive MGKT. Experiments show that the proposed model outperforms several state-of-the-art models in classifying long-tailed data. For example, our model performed 4.46% better than the next-best model on the SUN-LT dataset.

1. Introduction

Long-tailed distribution learning is a long-standing research problem in machine learning, which aims to solve model deviation caused by data imbalance. Real-world data are generally unbalanced and follow a long-tailed distribution [1] (as shown in Fig. 1); i.e., a few classes (head classes) contain a large number of samples, while most classes (tail classes) have very few samples. Most existing models perform well in the head classes but not in the tail classes. Such skewed distributions challenge the generalizability of machine learning models [2]. In 2004, Chris Anderson proposed the long-tailed theory. Since then, many people have devoted themselves to studying long-tailed distribution learning. Recently, long-tailed data distribution models have been widely used in various research fields [3], such as facial recognition, species classification, medical image diagnosis, and urban scene understanding.

Most existing approaches concentrate on strengthening the impact of tail classes in training to prevent the model from becoming dominated by the head classes. Typical methods include class re-balancing and transfer learning. Class re-balancing includes class re-sampling and class re-weighting. The re-sampling method involves under-sampling the head classes and over-sampling the tail classes to obtain a balanced data distribution. Under-sampling causes information loss in some samples and affects the generalizability of deep neural networks [4], while over-sampling leads to the over-fitting of tail classes [5]. On the contrary, class re-weighting pays attention to tail classes by assigning high weights to the losses of tail classes [6,7]. These methods improve the accuracy of classifying long-tailed data.

Another common solution is transfer learning based on a continuous and iterative assumption that the processing mechanism of deep neural networks is like that of the human brain, as both identify new things based on existing knowledge [8]. Traditional knowledge transfer methods in long-tailed learning are grouped into feature-based horizontal knowledge transfer (HKT) and class-based vertical knowledge transfer (VKT). The HKT methods transfer the knowledge learned from the massive data of the head classes to the tail classes. For instance, Yin et al. [9] presented a feature transfer learning method that leverages knowledge of the intra-class variance of the head class to guide the enhancement of tail class features so that the tail class has a higher

* Corresponding author at: School of Computer Science, Minnan Normal University, Zhangzhou, Fujian, 363000, China
E-mail address: hongzhaocn@163.com (H. Zhao).

https://doi.org/10.1016/j.patcog.2023.109842
Received 25 May 2022; Received in revised form 31 May 2023; Accepted 26 July 2023
Available online 3 August 2023

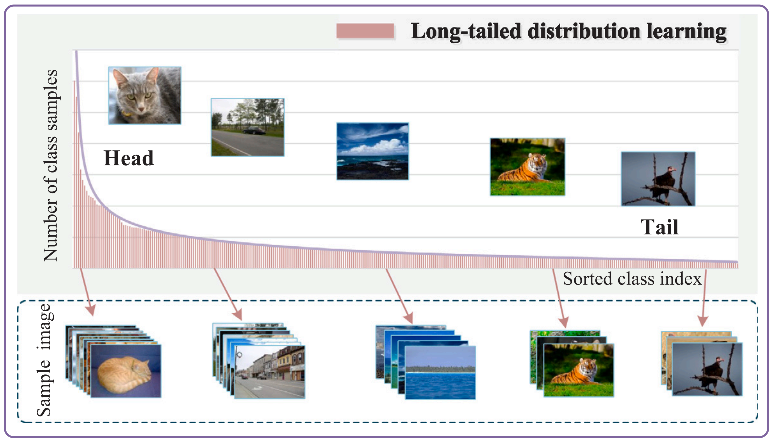


Fig. 1. Structural representation of long-tailed data.

intra-class variance. The method may cause erroneous movement of the classification decision boundary. Chu et al. [10] provided an online feature augmentation method, which aims to combine the class-specific features of tail class samples with the class-agnostic features of head class samples to expand the overall number of features. Subsequently, Liu et al. [11] constructed a "feature cloud" for each class and transferred the head class "feature cloud" knowledge to tail classes, thereby enhancing the intra-class diversity of tail classes. Wang et al. [12] proposed utilizing a special attention mechanism and visual memory blocks to associate and transfer features between head and tail classes.

The abovementioned methods alleviate the imbalance problem of long-tailed data to a certain extent. However, most feature-based HKT learning methods have a very limited impact on knowledge transfer when there is a large difference between the features of the head and tail classes. These obvious differences may lead to an invalid transfer; for instance, head class knowledge will transfer to the tail class in the case of head class = *animal* and tail class = *plant*. The differentiated HKT ineffectively improves tail class performance.

Fortunately, the class space has a multi-grained relationship [13] (i.e., a relationship between coarse- and fine-grained classes). The multi-granularity relationship, as an important auxiliary relationship, forms a multi-granularity knowledge graph (MGKG). Fig. 2 shows a visualization of an MGKG constructed from the semantic dependencies between classes in WordNet [14]. For instance, the classes *Plane*, *Motorbike*, and *Car* are fine-grained classes and belong to the coarse-grained class *Vehicle*. With the help of MGKG, similar samples of fine-grained tail classes and fine-grained head classes are combined to form numerous samples of coarse-grained classes. This can alleviate the class imbalance problem [13]. Drawing on the human "coarse to fine" cognitive mechanism [15], the knowledge gained from learning coarse-grained classes is transferred vertically to the model learned from fine-grained classes by a "top-down" MGKG. In addition, we propose a multi-scale feature fusion network to fuse different scale features, where the obtained features are distinguishable and highly semantic. Consequently, we adopt MGKG as an external knowledge to drive coarse- to fine-grained knowledge transfer by fully mining sample feature information.

In this paper, we propose a hierarchical long-tailed classification method based on multi-granularity knowledge transfer driven by multi-scale feature fusion (MGKT-MFF). This method drives vertical knowledge transfer from coarse- to fine-grained by fully mining channel and spatial multi-scale feature information. First, we design a channel and spatial multi-scale feature fusion network, which fuses the global and local features of inter-layer channels and then fuses the features obtained in intra-layer space to obtain the final rich features. This fusion fully mines sample feature information and strengthens the representation ability of semantic information. Then, we leverage the rich fusion features to carry out coarse- to fine-grained vertical knowledge transfer through an MGKG. Specifically, MGKG can be recast into coarse- and fine-grained losses to guide class-based vertical knowledge transfer. Finally, we establish a multi-granularity loss function to balance coarse- and fine-grained losses. We can efficiently grasp the vertical transfer of coarse- and fine-grained knowledge by dynamically adjusting the multi-granularity loss function.

Experiments were performed on four long-tailed datasets to compare the proposed MGKT-MFF method with several other advanced methods. The experimental results show that our method significantly improves the accuracy of classifying long-tailed data. Our method is 2.45% and 4.46% more accurate than the optimal method applied to the Cifar-100-LT (with an unbalanced rate of 10) and the SUN-LT datasets, respectively.

The main contributions of the paper can be summarized as follows:

(1) We propose a multi-scale feature fusion network by fusing inter-layer channel features and intra-layer spatial features. The obtained features have strong semantic information and focus on "where" the most informative part is.

(2) We investigate a multi-granularity relationship of the class space and gain an MGKG through semantic knowledge between classes in WordNet.

(3) We explore a vertical transfer of coarse- to fine-grained knowledge and establish a multi-granularity loss function to weigh the coarse- and fine-grained losses. We can effectively control the vertical transfer of coarse-grained knowledge to fine-grained knowledge by dynamically adjusting the multi-granularity loss function.

(4) We perform many experiments on the long-tailed datasets and evaluate our model with three metrics. We demonstrate the superiority of our model in terms of efficiency and accuracy.

The remainder of this paper is organized as follows. In Section 2, related research articles are reviewed. In Section 3, we describe the details of our proposed approach. We present the experimental settings used to test our approach in Section 4. The experimental results and analysis are summarized in Section 5, then the paper concludes in Section 6.

2. Related work

In this section, we review the literature on class re-balancing and information augmentation. Class re-balancing includes re-sampling and re-weighting, while information augmentation includes conventional data augmentation and transfer learning.

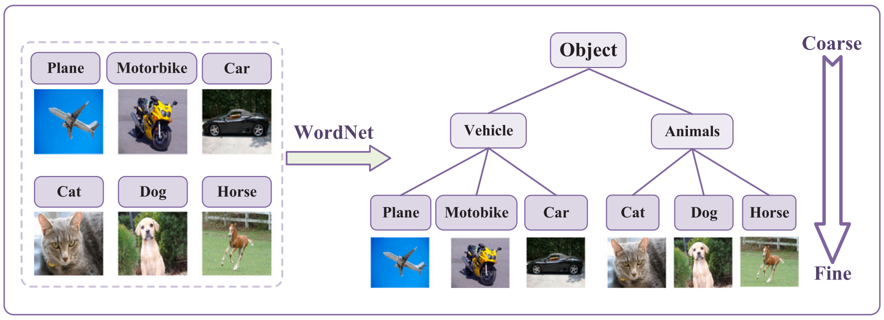


Fig. 2. Diagram of MGKG constructed by class semantics.

2.1. Class re-balancing

Class re-balancing is the mainstream long-tailed learning method. The re-sampling methods used to achieve a balanced data distribution include an over-sampling strategy and an under-sampling strategy. Although the re-sampling method is effective, over-sampling may overfit tail classes, while under-sampling may weaken feature learning for most classes due to the loss of valuable samples. Most sampling methods are based on over- or under-sampling methods.

In addition to the above re-sampling-based methods, other re-weighting methods can reduce the impact of data imbalance by modifying the loss function [16]. The basic idea is to adjust the loss of different classes in the loss function according to the statistical sample data of each class [2]. Existing solutions mainly focus on how to define weights for different classes effectively. This is inspired by the predictive accuracy of tail classes being lower than that of head classes in class imbalance. The simple method is a weighted softmax loss function, where the weight allocated to each class is inversely proportional to the number of samples. Lin et al. [6] proposed a focus loss function that uses predictive probability to perform reverse re-weighting. Subsequently, Cui et al. [17] proposed that the class balance loss approximates the expected number of samples in each class based on the number of valid samples in each class, which can be used to redefine the weights. This is a different approach to exploring re-weighting at the class level. Cao et al. [7] designed a label-distribution-aware loss function that encourages the model to obtain the optimal trade-off between per-class margins. Similarly, Deng et al. [18] proposed a progressive marginal loss function that uses two margin terms to adjust the class-wise margins for long-tailed learning.

The above methods improve the classification performance of the tail class by designing an algorithm that adjusts the loss weight. However, these methods affect the classification of the head class when performing loss weighting. The proposed method leverages the vertical transfer from coarse-grained knowledge to fine-grained knowledge to reduce the impact on head class classification. The loss function of our method combines coarse- and fine-grained losses. The advantage is that the knowledge obtained from coarse-grained classes helps in classifying fine-grained classes.

2.2. Information augmentation

Information augmentation-based methods introduce additional information into model training to improve a model’s performance in long-tailed distribution learning. Information augmentation generally includes conventional data augmentation and transfer learning methods. In existing conventional data augmentation methods, Zhang et al. [19] proposed a general data augmentation algorithm that expands the training distribution by combining the prior knowledge that the linear interpolation of feature vectors leads to the linear interpolation of related targets. Similarly, Verma et al. [20] proposed applying a hybrid operation to random pairs of samples in manifold feature space for reinforcement. Unlike the data augmentation methods above, Zhong et al. [21] used data mixing in a decoupling scheme to enhance feature representation learning. Chou et al. [22] proposed a new regularization technique that relaxes the mix-up formulation to disentangle the mixing factors of features and labels.

Other transfer learning-based methods transfer the knowledge learned from head-to-tail classes to alleviate the sparse features of the tail classes. For instance, Yin et al. [9] transferred intra-class variance from head to tail classes. This method may cause erroneous movement of the classification decision boundary. Chu et al. [10] presented an online feature augmentation method, which decouples sample features into class-specific and class-agnostic features by using class activation maps. It then combines the two features to expand the tail classes to fine-tune the model classifier. Subsequently, Liu et al. [11] proposed the construction of a “feature cloud" for each class and transfer of the knowledge from the “feature cloud" of the head class to the tail classes to enhance their intra-class diversity. Wang et al. [12] leveraged a special attention mechanism and visual memory block to associate and transfer features between head and tail classes.

Conventional data augmentation methods are not beneficial for improving long-tailed classification. The reason is that many head class samples and comparatively large enhancements further expand the imbalance. Existing transfer learning methods transfer knowledge from head to tail classes to improve tail-class performance. However, the effect of knowledge transfer is limited when the head and tail features are very different. Therefore, our method enhances feature information by fusing the channel and spatial features in the network and transferring knowledge vertically from coarse- to fine-grained classes. The vertical transfer of knowledge can improve the transfer effectiveness.

3. Hierarchical long-tailed classification based on multi-granularity knowledge transfer driven by multi-scale feature fusion

In this section, we introduce the proposed framework and detail a hierarchical long-tailed classification method based on multi-granular knowledge transfer driven by a multi-scale feature fusion training process (MGKT-MFF).

3.1. Basic framework

This section provides an overview of the MGKT-MFF method. The framework of our model is illustrated in Fig. 3. The MGKT-MFF process mainly consists of three stages: First, we extract multi-scale fusion features through inter-layer channel features and intra-layer spatial features. Second, we utilize the multi-scale fusion features obtained in the previous stage to drive coarse- to fine-grained knowledge transfer through a multi-grained knowledge graph. Finally, we exploit coarse-grained knowledge to assist the classification of fine-grained classes and establish a multi-granularity loss function.

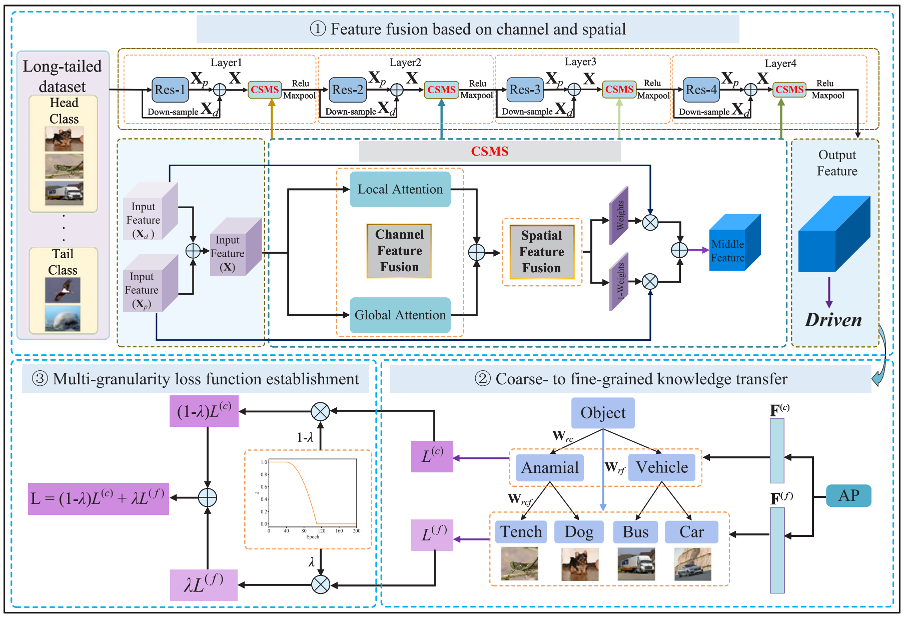


Fig. 3. Framework of the MGKT-MFF model. $\mathbf{W}_{rc}$ is coarse-grained knowledge, $\mathbf{W}_{rcf}$ is coarse- and fine-grained knowledge, and $\mathbf{W}_{rf}$ is fine-grained knowledge. AP represents average pooling.

3.2. *Multi-scale feature fusion based on channel and spatial attentions*

Feature extraction is the fundamental processing step in machine learning [23]. The quality of features has a significant impact on the generalization and classification ability of a model. We propose a feature fusion network that combines inter-layer and intra-layer multi-scale features to obtain high-quality features of different scales.

3.2.1. *Multi-scale feature fusion*

Let $D = \{(x_m, y_m^{(f)}, y_m^{(c)})\}_{m=1}^{N}$ represent a long-tailed dataset, where x_m is a sample image with a corresponding fine-grained label $y_m^{(f)} \in \{1, 2, \ldots, U\}$, and U is the number of fine-grained classes. $y_m^{(c)} \in \{1, 2, \ldots, C\}$ denotes the corresponding coarse-grained label and C is the number of coarse-grained classes. N is the number of samples in the dataset. Inspired by multi-scale feature fusion in the channel [24], we propose a multi-scale feature fusion network based on channel and spatial attention. Immediately afterwards, we randomly selected a sample $(x_m, y_m^{(f)}, y_m^{(c)})$ from dataset D for feature extraction to describe the process of multi-scale feature fusion in more detail. There are four levels in CSMS-ResNet, all of which contain CSMS modules. Since it performs level-by-level optimization from a low-to-high level, we randomly choose a level as an example.

Take two feature maps $\mathbf{X}_d$, $\mathbf{X}_p \in \mathbb{R}^{C\times H\times W}$ from a layer in the network, where C, H, and W represent the number of channels, spatial height, and width of the feature map, respectively. We assume that $\mathbf{X}_p$ is a feature graph with a large receptive field. More specifically, the difference between $\mathbf{X}_d$ and $\mathbf{X}_p$: (1) $\mathbf{X}_d$ is obtained by down-sampling the input features of each level; i.e., without passing through the residual blocks in the network. (2) $\mathbf{X}_p$ is the result of the residual blocks that enter the input features of each level into the network. The residual blocks have operations such as convolution and pooling. $\mathbf{X}_p$ has a large receptive field. On this basis, we first obtain the following fusion features:

$$\mathbf{X} = \mathbf{X}_d + \mathbf{X}_p, \tag{1}$$

where $\mathbf{X}$ is a simple fusion of low-level features and comparatively high-level features, and is the input feature before entering the CSMS module. We feed feature map $\mathbf{X}$ into the CSMS module for multi-scale feature fusion to aggregate the context information of different scales. The multi-scale feature fusion is represented as follows:

$$\mathbf{Z} = \mathrm{CSMS}(\mathbf{X}) \otimes \mathbf{X}_d + (1 - \mathrm{CSMS}(\mathbf{X})) \otimes \mathbf{X}_p, \tag{2}$$

where $\mathbf{Z} \in \mathbb{R}^{C\times H\times W}$ is a multi-scale fusion feature based on channel and spatial attention. $\otimes$ denotes the element-wise multiplication. Next, we analyze the CSMS module in detail. This module includes channel attention feature fusion and spatial attention feature fusion. The detailed implementation process is shown in Fig. 4. The following sections describe the details of each attention module.

3.2.2. *Channel feature fusion module*

We utilize the inter-layer channel feature relations to aggregate more feature information. The channel feature fusion module includes *local feature attention* and *global feature attention* components. The features obtained through local feature attention can retain and highlight the fine details of the underlying features. The *global feature attention* component captures features that aggregate spatial information and make features more meaningful for each channel dimension. More notably, *global feature attention* component can reduce the parameters of the model and prevent overfitting.

For *local feature attention* component, we select a point-wise convolution (PW^{Conv}) as the local channel context aggregator, which only utilizes point-by-channel interactions at each spatial location. To illustrate the calculation process of the *local feature attention* component, local features $L(\mathbf{X}) \in \mathbb{R}^{C\times H\times W}$ are calculated by it and can be expressed as:

$$L(\mathbf{X}) = B(PW_2^{Conv}(\delta(B(PW_1^{Conv}(\mathbf{X}))))), \tag{3}$$

where the kernel sizes of PW_1^{Conv} and PW_2^{Conv} are $\frac{C}{R} \times C \times 1 \times 1$ and $C \times C \times 1 \times 1$, respectively. R is the channel reduction ratio. δ refers to the ReLU function, which is an activation function, and $B(\cdot)$ is a batch normalization function that enables the distribution of data to have a mean of 0 and variance of 1. $L(\mathbf{X})$ has the same shape as input feature $\mathbf{X}$, which can aggregate more feature information; i.e., it can retain and highlight the details of low-level features.

Similarly, for *global feature attention* component, it can aggregate the spatial information and decrease the model parameters. We compress the spatial dimension of the input features. Currently, global average pooling methods are commonly employed. A channel-wise statistic $\mathbf{g}$

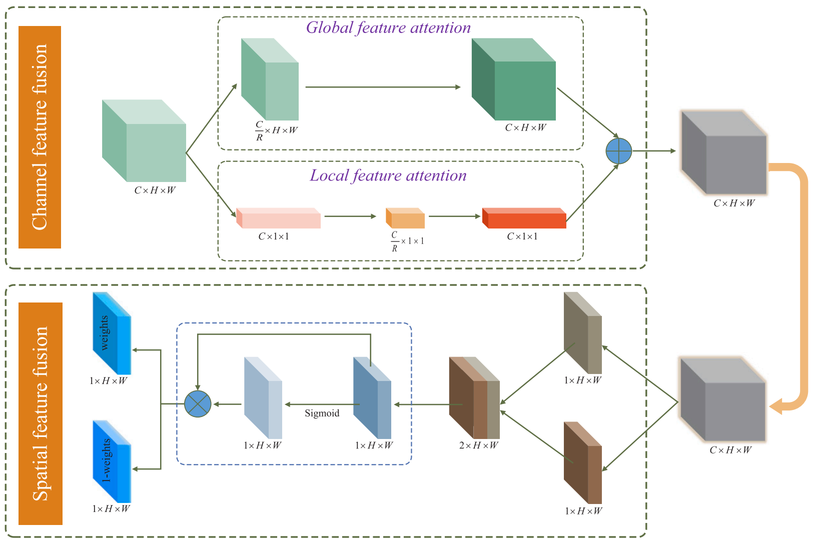


Fig. 4. Description of the channel and spatial multi-scale feature (CSMS) fusion module.

$\in \mathbb{R}^{C\times 1\times 1}$ is generated by shrinking the spatial dimensions $H \times W$ of each channel in $\mathbf{X}$. Therefore, the calculation formula of $\mathbf{g}$ is as follows:

$$\mathbf{g} = \mathcal{F}_{gap}(\mathbf{X}) = \frac{1}{H\times W}\sum_{i=1}^{H}\sum_{j=1}^{W}\mathbf{X}(:,i,j), \tag{4}$$

where $\mathcal{F}_{gap}(\cdot)$ is the global average pooling function. Immediately afterwards, the global features $G(\mathbf{X}) \in \mathbb{R}^{C\times 1\times 1}$ are calculated by *global feature attention*, which can be expressed as:

$$\begin{aligned} G(\mathbf{X}) &= B(PW_2^{Conv}(\delta(B(PW_1^{Conv}(\mathcal{F}_{gap}(\mathbf{X})))))) \\ &= B(PW_2^{Conv}(\delta(B(PW_1^{Conv}(\mathbf{g}))))). \end{aligned} \tag{5}$$

To make it as lightweight as possible, we only need to fuse features of different scales within the *channel feature fusion module*; i.e., the fusion of local and global features. We fuse the global and local features of the inter-layer channels as follows:

$$\mathbf{X}_q = L(\mathbf{X}) \oplus G(\mathbf{X}), \tag{6}$$

where $\mathbf{X}_q \in \mathbb{R}^{C\times H\times W}$ is the feature of inter-layer channel fusion. $\oplus$ indicates the broadcasting addition. Nevertheless, the approach based on the channel feature fusion module ignores the spatial relationship between intra-layer features. With the obtained multi-scale channel fusion features, we focus on the fusion of spatial features.

3.2.3. Spatial feature fusion module

We generate spatially attentional features using spatial relationships between intra-layer features. Unlike the channel feature fusion module, the spatial feature fusion module focuses on “where” as the information-rich part, which complements the channel feature fusion module. To calculate spatial attention weights, we first perform average pooling and max pooling along the channel dimensions and connect them to generate an effective feature map. Such operations can effectively aggregate and highlight spatial information areas. Subsequently, we apply the connected feature map to a 1×1 convolution operation to generate the spatial feature map $M^s(\mathbf{X}_q) \in \mathbb{R}^{1\times H\times W}$, where M^s is spatial feature fusion module. The calculation process is described in detail as follows.

We generate two feature graphs $\mathbf{X}^s_{ap} \in \mathbb{R}^{1\times H\times W}$ and $\mathbf{X}^s_{mp} \in \mathbb{R}^{1\times H\times W}$ through two pooling operations. They are then spliced along their channel dimensions and convolved, which can be expressed as follows:

$$\begin{aligned} M^s(\mathbf{X}_q) &= PW_3^{Conv}([\mathcal{F}_{ap}(\mathbf{X}_q); \mathcal{F}_{mp}(\mathbf{X}_q)]) \\ &= PW_3^{Conv}([\mathbf{X}^s_{ap}; \mathbf{X}^s_{mp}]), \end{aligned} \tag{7}$$

where the convolution kernel of PW_3^{Conv} is $1 \times 2 \times 1 \times 1$. This convolution is used to fuse the spliced features to create a larger receptive field. $\mathcal{F}_{ap}(\cdot)$ and $\mathcal{F}_{mp}(\cdot)$ are the average pooling and max pooling functions, respectively. Then, we compute the weight of attention after spatial feature fusion. We use a sigmoid function to scale $M^s(\mathbf{X}_q)$ to $0 \sim 1$ and multiply $M^s(\mathbf{X}_q)$ accordingly to obtain the spatial weight. The weight obtained by this method aims to emphasize the more effective information and reduce the more redundant information in the intra-layers of spatial feature information. Finally, the channel feature fusion is followed by spatial feature fusion to obtain the optimized multi-scale fusion feature, where the weight $\mathbf{W}^{cs}$ is obtained by the operation and is calculated as follows:

$$\mathbf{W}^{cs} = W^s(M^s(\mathbf{X}_q)) = sigmoid(M^s(\mathbf{X}_q)) \otimes M^s(\mathbf{X}_q), \tag{8}$$

where $\mathbf{W}^{cs} \in \mathbb{R}^{1\times H\times W}$ and W^s is the calculation weight module. It should be noted that the resulting fusion weight $\mathbf{W}^{cs}$, as well as 1 - $\mathbf{W}^{cs}$, enable the network to perform soft selection or calculate the weighted average between $\mathbf{X}_d$ and $\mathbf{X}_p$.

Based on the discussion above, Eq. (2) can also be expressed as follows:

$$\mathbf{Z} = \mathbf{W}^{cs} \otimes \mathbf{X}_d + (1 - \mathbf{W}^{cs}) \otimes \mathbf{X}_p. \tag{9}$$

Consequently, we perform optimization from the low-level to the high-level through the single-level multi-scale feature fusion process described above, which can aggregate the high-level semantic and low-level spatial features. Ultimately, we can obtain multi-scale fusion features that retain and highlight features related to classification and have low redundancy and rich high-level semantic information. In terms of the classifier, multi-scale fusion features can increase its classification performance.

3.3. Coarse- to fine-grained knowledge transfer

Traditional flat structures ignore the structural information between different classes. To make up for its deficiency, we consider the information within different levels of detail in the hierarchy of classes and find a multi-granularity relationship between these classes [13]. Accordingly, an MGKG is constructed through the semantic knowledge between classes in WordNet [14]. The transfer method used in long-tailed learning is head-to-tail knowledge transfer. It is worth noting that this method of knowledge transfer can easily lead to inconsistency in the transfer targets; i.e., the knowledge transfer learned by the head class is different from that of the tail class target. Therefore, we utilize the MGKG as prior knowledge to achieve vertical knowledge transfer from coarse-grained to fine-grained.

To learn coarse-grained knowledge and transform it into fine-grained knowledge, we design a transformation strategy in CSMS-ResNet. The strategy involves sharing the representation learning layers and using the corresponding granularity of knowledge-specific learning layers. As a platform, the shared representation learning layers provide stored and fused knowledge for knowledge-specific learning layers of different granularity to promote knowledge transfer, hence assisting long-tailed classification learning more effectively.

Shared representation learning layers: The shared representation learning layer in CSMS-ResNet mostly consists of convolution, pooling, activation, and batch normalization operations. For a sample x_m, each convolution kernel is convolved at the width and height of x_m, i.e., the dot product between the convolution kernel and x_m. The shared representation learning layers usually stack multiple convolutional layers and pooling layers. It is worth mentioning that the weight $\mathbf{W}_r$ of the shared representation learning layer is usually applicable to the convolution layer and can be expressed as follows:

$$\mathbf{W}_r = [\mathbf{W}_1; \mathbf{W}_2; \cdots; \mathbf{W}_l], \tag{10}$$

where l is the number of convolutional layers and $\mathbf{W}_k$ is the weight matrix of the kth convolutional layer.

Granular knowledge-specific learning layers: Based on the learning of the shared representation learning layer, we transfer coarse-grained knowledge to fine-grained knowledge in the granular knowledge-specific learning layers by using the multi-grained knowledge graph. We set two thresholds E_c and E_f to adapt to knowledge-specific learning level knowledge learning of different granularity during the training process. The threshold is set in the experimental setting and the specific transfer process is as follows:

$$\begin{cases} \mathbf{W}_{rc} = H_{bp}((x_m, y_m^{(f)}, y_m^{(c)}), \mathbf{W}_r), & \text{if } t < E_c; \\ \mathbf{W}_{rcf} = H_{bp}((x_m, y_m^{(f)}, y_m^{(c)}), \mathbf{W}_{rc}), & \text{if } E_c \le t < E_{\max} - E_f; \\ \mathbf{W}_{rf} = H_{bp}((x_m, y_m^{(f)}, y_m^{(c)}), \mathbf{W}_{rcf}), & \text{if } t \ge E_{\max} - E_f, \end{cases} \tag{11}$$

where E_c and E_f represent coarse-grained and fine-grained training epochs, respectively. Here, t and $E_{\max}$ represent the tth training epoch and the total training epoch, respectively. $H_{bp}()$ indicates the parameter update after back propagation. In terms of the above, there are two knowledge-specific learning layers to learn, being coarse-grained and fine-grained. Each learning layer is attached to a specific FC layer that learns coarse- or fine-grained knowledge, being denoted as $\mathbf{W}_{rc}$ and $\mathbf{W}_{rf}$, respectively. It is worth noting that the transfer of coarse-grained knowledge to fine-grained knowledge is carried out simultaneously by combining two specific layers and is expressed as $\mathbf{W}_{rcf}$.

3.4. Multi-granularity loss function establishment

Our method of establishing a multi-grained loss function focuses on coarse-grained knowledge assistance and fine-grained knowledge learning. This improves the transfer capability of the model. The multi-granularity loss function $\mathcal{L}$ in our model is expressed as follows:

$$\mathcal{L} = \lambda \mathcal{L}^{(c)} + (1-\lambda)\mathcal{L}^{(f)}, \tag{12}$$

where λ is a weight that balances the learning attention of the two specific learning layers. The λ-value is dynamically adjusted within the range of 0–1. Recall that according to Eq. (11), there are three main stages in our training process: (1) coarse-grained knowledge-specific learning layers are trained to master coarse-grained knowledge; (2) training two knowledge-specific layers at the same time can benefit the two knowledge-specific layers not only mutually but also gradually transfer the coarse-grained knowledge to fine-grained knowledge; (3) separately, training the fine-grained knowledge-specific learning layer can effectively improve the classification ability and generalizability of the model. Subsequently, we describe the establishment of the multi-granularity loss function in detail. After obtaining the final output feature $\mathbf{Z}$, we conduct an average pooling operation and then reconstruct the feature dimension to form a two-dimensional feature matrix. We then obtain the feature matrix $\mathbf{F}_s \in \mathbb{R}^{1\times S}$. The feature matrix $\mathbf{F}_s$ is sent to classifiers $\mathbf{W}^{(c)} \in \mathbb{R}^{C\times S}$ and $\mathbf{W}^{(f)} \in \mathbb{R}^{U\times S}$. The specific expression is as follows:

$$\begin{cases} \mathbf{F}^{(c)} = \mathbf{F}_s(\mathbf{W}^{(c)})^\top + \mathbf{b}^{(c)}; \\ \mathbf{F}^{(f)} = \mathbf{F}_s(\mathbf{W}^{(f)})^\top + \mathbf{b}^{(f)}, \end{cases} \tag{13}$$

where $\mathbf{F}^{(c)}$ and $\mathbf{F}^{(f)}$ are the predicted output for coarse- and fine-grained knowledge-specific learning layers, respectively. The symbol $\top$ represents a transpose. Parameters $\mathbf{b}^{(c)}$ and $\mathbf{b}^{(f)}$ are the bias vectors for the coarse-grained and fine-grained knowledge-specific learning layers, respectively.

Next, we use the softmax cross-entropy loss function to calculate the losses of two specific layers. For the loss of coarse-grained knowledge-specific learning layers, we calculate the softmax cross-entropy loss using the predicted output $\mathbf{F}^{(c)} = [F_1^{(c)}, F_2^{(c)}, \ldots, F_C^{(c)}]^\top$. The loss can be expressed as follows:

$$\mathcal{L}^{(c)} = -\log\left(\frac{\exp(F_c^{(c)})}{\sum_{\hat{j}=1}^{C}\exp(F_{\hat{j}}^{(c)})}\right), \tag{14}$$

where $c \in \{1, 2, \ldots, C\}$ is the coarse-grained label value corresponding to x_m. $\mathcal{L}^{(c)}$ is formed by mapping $\mathbf{F}^{(c)}$ to the coarse-grained labels in MGKG.

For the loss of the fine-grained knowledge-specific learning layers, we calculate the softmax cross-entropy loss using the predicted output $\mathbf{F}^{(f)} = [F_1^{(f)}, F_2^{(f)}, \ldots, F_U^{(f)}]^\top$. The loss can be expressed as follows:

$$\mathcal{L}^{(f)} = -\log\left(\frac{\exp(F_u^{(f)})}{\sum_{\hat{j}=1}^{U}\exp(F_{\hat{j}}^{(f)})}\right), \tag{15}$$

where $u \in \{1, 2, \ldots, U\}$ is the fine-grained label value corresponding to x_m. $\mathcal{L}^{(f)}$ is formed by mapping $\mathbf{F}^{(f)}$ to the coarse-grained labels in MGKG. Ultimately, we obtain the multi-granularity loss function. It is worth noting that it is the training process in terms of the above. The MGKT-MFF method is based on the above content; i.e., multi-scale fusion features drive coarse-grained to fine-grained knowledge transfer. We consider the trained MGKT-MFF parameters and load them into the multi-scale fusion network to obtain the feature map of the last layer. As shown in Fig. 5, we visualize the activation of the random channel feature map and the entire feature map. We can infer the meaningful semantic regions of the original image (highlighted by warm colors). In the testing process, we directly select the predicted output value of the fine-grained knowledge-specific layer and calculate the accuracy. The whole model is end-to-end trainable.

Algorithm 1 describes the pseudo-code for the model training process. In particular, we extract multi-scale fusion features based on inter-layer and intra-layer in lines 4∼6. We construct the loss function of the coarse-grained knowledge-specific layer in the first E_c epochs and carry out loss back propagation and parameter update in lines 7∼9. Then, we transfer the coarse-grained knowledge to the fine-grained knowledge in $E_{\max} - E_c - E_f$ epochs in lines 10∼12. Next, we construct the loss function of the fine-grained knowledge-specific layer in the last E_f epochs and perform loss back propagation and parameter update in lines 13∼15.

4. Experimental settings

In this section, our experimental setting includes the following four aspects: (1) the dataset descriptions; (2) the implementation details; (3) the evaluation measures; (4) the comparison methods.

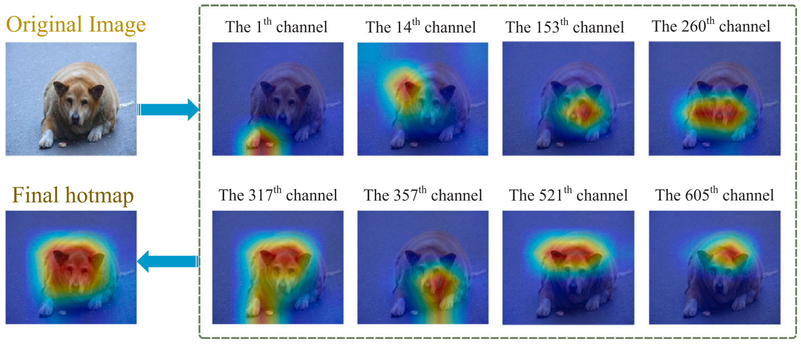


Fig. 5. Visualization of MGKT-MFF method and partial channel feature map.

Algorithm 1: The training procedure of the proposed approach

Input: The training sample $(x_m, y_m^{(f)}, y_m^{(c)})$, where x_m is a sample image with corresponding fine-grained label $y_m^{(f)}$ and $y_m^{(c)}$ denotes the corresponding coarse-grained label. The number of total training epochs is $E_{\max}$. E_c and E_f represent coarse-grained and fine-grained knowledge-specific layers training epochs, respectively.
Output: The parameters $\mathbf{W}_r$, $\mathbf{W}^{(c)}$, and $\mathbf{W}^{(f)}$.

```
1:  Initialize parameters $\mathbf{W}_r$, $\mathbf{W}^{(c)}$, and $\mathbf{W}^{(f)}$;
2:  for $epoch = 1 : E_{\max}$ do
3:      Extract the multi-scale fusion features by CSMS module;
4:      Obtain the inter-layer channel fusion features $\mathbf{X}_q$ through Eq. (6);
5:      Obtain the intra-layer spatial fusion features $M^s(\mathbf{X}_q)$ through Eq. (7);
6:      Obtain multi-scale fusion feature $\mathbf{Z}$ according to Eq. (9);
7:      if $t < E_c$ then
8:          Calculate the coarse-grained class loss with $\lambda = 1$ by Eqs. (12) and (14);
9:          Update the model $\mathbf{W}_r$ and $\mathbf{W}^{(c)}$ by the coarse-grained class loss back propagation according to Eq. (11);
10:     else if $E_c \le t < E_{\max} - E_f$ then
11:         Calculate the multi-granularity loss with $0 < \lambda < 1$ by Eqs. (12), (14), and (15);
12:         Update the model $\mathbf{W}_r$, $\mathbf{W}^{(c)}$, and $\mathbf{W}^{(f)}$ by the multi-granularity loss back propagation according to Eq. (11);
13:     else if $t \ge E_{\max} - E_f$ then
14:         Calculate the fine-grained class loss with $\lambda = 0$ by Eqs. (12) and (15);
15:         Update the model $\mathbf{W}_r$ and $\mathbf{W}^{(f)}$ by the fine-grained class loss back propagation according to Eq. (11);
16:     end if
17: end for
```

4.1. Dataset descriptions

We evaluate our method on various class-imbalanced classification tasks, which use four synthetic imbalanced datasets consisting of Cifar-100-LT, VOC-LT, SUN-LT, and tiredImagenet-LT. Fig. 6 shows a histogram of the number of samples for coarse-grained classes. Fig. 7 demonstrates the histogram of the class-wise sample distributions for the datasets considered in our experiments.

Cifar-100-LT. The long-tailed Cifar-100 is built on the original Cifar-100 dataset [25] by reducing training samples per class. The original Cifar-100 dataset contains 60,000 samples, and the number of classes is 100, including 50,000 training samples and 10,000 test samples. For a fair comparison, we use a long-tailed version of the Cifar-100 dataset with the same settings as used in [7]. We construct five training sets by changing the imbalance factor $\rho \in \{200,100,50,20,10\}$, where ρ represents the image quantity ratio between the most frequent class and the least frequent class. Let N_{max} be the number of samples in the most frequent fine-grained class and N_{min} the number of samples in the least frequent fine-grained class. We define $\rho = \frac{N_{max}}{N_{min}}$. For the test sets, we use the original balanced test sets.

Table 1
Descriptions of the experimental datasets.
(a)

Dataset	Cifar-100-LT				
Imbalance ratio	200	100	50	20	10
Coarse-grained class	20	20	20	20	20
Fine-grained class	100	100	100	100	100
Training size	9,502	10,847	12,607	15,907	19,572
Test size	10,000	10,000	10,000	10,000	10,000
Type	Image	Image	Image	Image	Image

(b)

Dataset	VOC-LT	SUN-LT	tiredImagenet-LT
Imbalance ratio	57	27.9	203
Coarse-grained class	4	15	34
Fine-grained class	20	324	608
Training size	3,437	85,147	69,545
Test size	3,539	5,184	74,176
Type	Image	Image	Image

VOC-LT. VOC [26] is a benchmark visual object classification and recognition dataset from the challenge of PASCAL visual object classes. VOC-LT extracts from the 2012 train-val set of VOC. Therefore, the VOC-LT dataset contains 6,976 samples and 20 classes, including 3,437 training samples and 3,539 test samples. The resulting VOC-LT is characterized by long-tailed data distribution.

SUN-LT. The Scene UNderstanding image dataset [27] contains 397 classes. Since the experiment considers single-label data, it is necessary to delete the multi-label classes, which obtain 324 classes. The SUN-LT dataset contains 90,311 images, 85,147 for training, 5,184 for the test, and the number of classes is 324. Therefore, the imbalance rate is 27.9.

tiredImagenet-LT. tiredImagenet [28] is a subset of Imagenet2012. The 608 classes divide into 34 coarse-grained classes by the Imagenet hierarchy [14]. Each class contains at least 732 images of 84 × 84 size. We randomly select 610 images per class as training images and 122 images per class as test images, which creates tiredImagenet. We create a tiredImagenet-LT dataset containing 143,721 images, 69,545 for training, and 74,176 for the test.

Based on the long-tailed datasets mentioned above, it is worth mentioning that the number of classes is constant when building hierarchical datasets semantically.

Table 1 describes the hierarchical long-tailed datasets in detail.

4.2. Implementation details

For all the datasets in the experiment, we follow the data augmentation strategy proposed in [29]. During training, each side of each

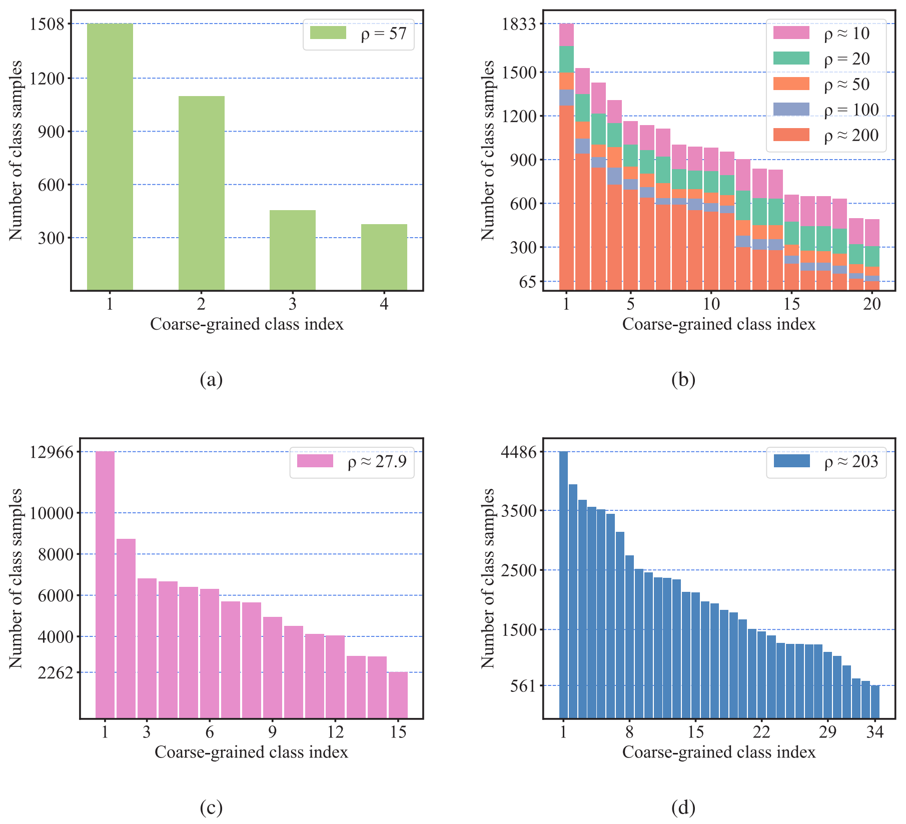


Fig. 6. Histogram of the number of samples for coarse-grained classes on different datasets. (a) VOC-LT; (b) Cifar-100-LT; (c) SUN-LT; (d) tiredImagenet-LT.

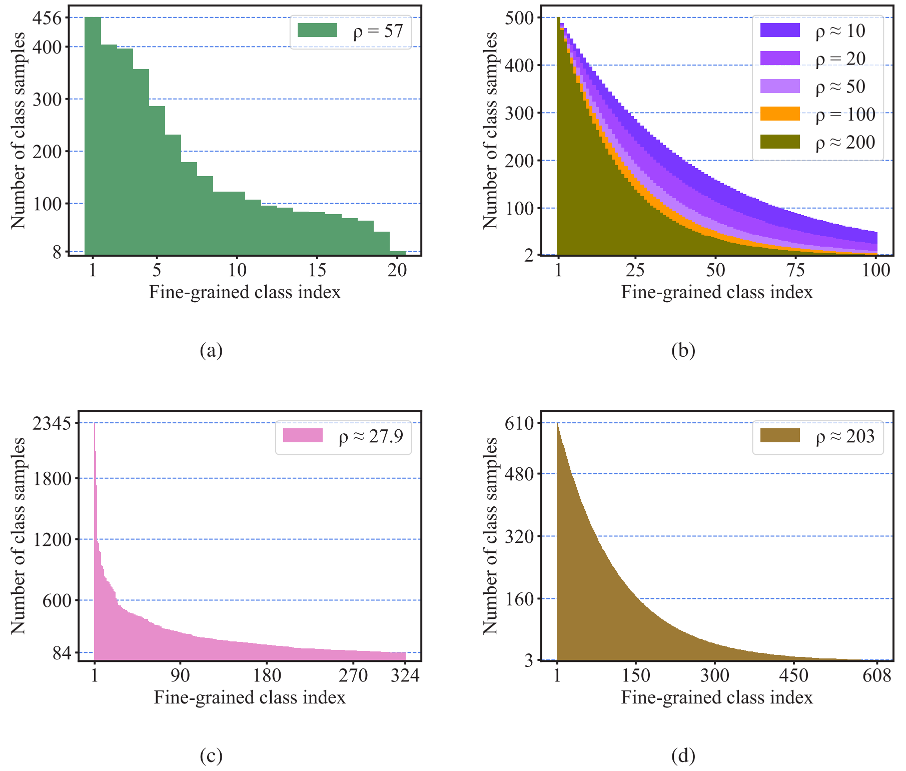


Fig. 7. Histograms of the number of fine-grained class samples on different datasets. (a) VOC-LT; (b) Cifar-100-LT; (c) SUN-LT; (d) tiredImagenet-LT.

image was filled with 4 pixels and randomly cropped out to a size of 32 × 32. The clipped image is flipped horizontally with a probability of 0.5 and normalized by the mean and standard deviation of each color before training. Furthermore, to better check the performance changes of different numbers of classes during training, we follow the [30] setting method. These classes are divided into three groups

Table 2
Experimental setup of comparison methods and the proposed method. (▲ represents the four sides of the image filled with 4 pixels and randomly cut into 32 × 32 size. The cropped images were flipped horizontally with a probability of 0.5 and normalized with the mean and standard deviation of each color before training.).

Type	Network	Total epoch	Coarse-grained epoch	Fine-grained epoch	Batch size	Optimizer	Learning rate	Image cropping
Comparison method	ResNet32	200	–	–	128	SGD	0.1	▲
MGKT-MFF	CSMS-ResNet	200	40	90	128	SGD	0.1	▲

based on the number of images each class appears in the training set: the head classes (over 100 images), the middle classes (20–100 images), and the tail classes (1–20 images). In all comparative experiments, we use ResNet32 [29] as the backbone network. For our method, we use our proposed CSMS-ResNet for experiments. All experiments train each model by stochastic gradient descent (SGD) with a momentum of 0.9 and weight decay of 2×10^4. The number of training epochs is 200, and the batch size is 128. The initial learning rate is 0.1, and the first five epochs are trained using a linear warm-up learning rate schedule [31]. In addition, the learning rate decays by 0.01 at the 160th and 180th epoch. For our method, we set coarse-grained knowledge task learning in the first 40 epochs in the training stage, multi-grained knowledge transfer in the 40th to 110th epoch, and fine-grained task fine-tuning in the post-90 epochs.

We experimentally determined the threshold by setting the set of E_c and E_f. The basic principle has two aspects: The first aspect: $E_c < E_f$. The reason is that coarse-grained classes are expected to participate in model training to help initialize the model. Therefore, the coarse-grained training epochs should not be too many, accounting for 1/5 of the total. The second aspect: the ultimate goal of model training is to classify fine-grained categories. Therefore, the training epochs for fine-grained categories account for a large proportion, nearly half. Consequently, based on the above two principles and through experiments, we retain the best experimental settings. E_c and E_f are 40 and 90, respectively.

Table 2 provides the differences between the experimental settings.

All experiments were performed on a Ubuntu20.04 desktop computer using NVIDIA 270 GTX3090 with 24.0 GB video memory and a 2.40 GHz × 24 Intel Xeon Silver 4214R CPU. The code is available at https://github.com/fhqxa/MGKT-MFF.

4.3. Evaluation measures

In this section, we introduce three evaluation measures that are proposed in [32], including *ACC*, *TIE*, and F_H.

Classification accuracy (*ACC*): The *ACC* is a flat evaluation metric widely used in various classification tasks without considering hierarchical structure.

Tree induced error (*TIE*): The *TIE* measures the distance between the predicted class of the sample and the correct class in the hierarchy structure. The *TIE* evaluation reflects the degree of sample prediction error in the hierarchical structure. Its detailed formula is: $TIE(y_t, y_p) = |E_h(y_t, y_p)|$, where y_t is the true label of a sample x_t and y_p is the predicted label of x_t. $E_h(y_t, y_p)$ represents the edges from y_t to y_p, and $|\cdot|$ is expressed as the number of elements in the set.

Hierarchical F1-measure evaluation (F_H): The F_H is an evaluation standard of the hierarchical structure, which uses the joint calculation of hierarchical precision and recall rate. Hierarchical precision (P_H) and hierarchical recall rate (R_H) are defined as $P_H = \frac{|y_{aug} \cap \hat{y}_{aug}|}{|\hat{y}_{aug}|}$ and $R_H = \frac{|y_{aug} \cap \hat{y}_{aug}|}{|y_{aug}|}$, respectively, where y_{aug} represents the augmented set of the predicted class including its ancestor nodes and $\hat{y}_{aug}$ indicates the augmented set of the true class. $|\cdot|$ is an operator to calculate the number of elements. The F_H is defined as follows: $F_H = \frac{2 \cdot R_H \cdot P_H}{R_H + P_H}$.

It is worth noting that the *ACC* and F_H indicate the higher the value, the better the performance. The *TIE* indicates that the lower the value, the better the performance.

4.4. Comparison methods

In this section, we consider a wide range of baseline methods and advanced long-tailed methods, and the details of them are followed as:

Baseline methods. Loss functions: (a) cross-entropy loss (CE): standard training in CE loss without any re-balancing; (b) focal loss (Focal) [6]: reconstructing standard cross-entropy loss to focus on and improve the weight of relatively difficult classes; (c) label-distribution-aware margin loss (LDAM) [7]: a loss function for label assignment to encourage minority classes to obtain large margins. Strategies: (a) re-sampling (RS) [33]: form a balanced dataset from different sampling probabilities of each sample; (b) re-weighting (RW) [2]: the inverse of the total samples of each class is used as the weight of the loss function; (c) class-balanced re-weighting (CB-RW) [17]: (RW variant), instead of inverse class frequencies, the samples are re-weighted based inverse of the effective number for each class, defined as $(1 - \beta^{N_k}) / (1 - \beta)$, i.e., $\beta = 0.9999$; (d) deferred re-sampling (DRS) [7], (e) deferred re-weighting (DRW) [7], and (f) deferred class-balanced re-weighting (DCB) [17]: RS, RW, and CB is deferred until the later stage of training of a model, respectively.

The loss function combines the relevant strategies to form the algorithm of the original paper.

Advanced long-tailed methods.

(1) Mixup [19] proposes a neural network based on convex combinations of pairs of instances and their labels.

(2) L2RW [34] proposes a novel meta-learning algorithm that learns to assign weights to training examples based on their gradient directions.

(3) Class-balanced fine-tuning [35] presents a measure to estimate domain similarity by Earth Mover's Distance.

(4) Meta-weight net [36] presents an adaptive sample weighting strategy that can automatically learn explicit weighting functions from data.

(5) Meta-class-weight with CE [37] proposes to enhance classical class balancing learning by explicitly estimating the difference between class condition distributions.

(6) BBN [25] proposes a unified Bilateral-Branch Network to take care of both representation learning and classifier learning simultaneously.

(7) Hybrid-PSC [30] proposes a novel hybrid network structure that consists of a supervised contrast loss learning image representation and a cross-entropy loss learning classifier.

(8) Bag of Tricks [38] assembles existing techniques in long-tailed visual recognition to obtain an effective combination of these techniques and proposes a novel data augmentation method based on class activation maps.

(9) MetaSAug with CE [39] proposes a novel approach to learning transformed semantic directions with meta-learning automatically.

(10) MiSLAS [21] proposes label-aware smoothing to deal with different degrees of over-confidence for classes and improve classifier learning.

5. Experimental results and analysis

In this section, we discuss the experimental results from six aspects to verify the validity of our model: (1) the effectiveness of the channel and spatial multi-scale feature fusion ResNet model; (2) the performance of the multi-granularity loss function; (3) ablation experiments; (4) visualization of our model; (5) statistical tests of different models; and (6) efficiency comparison with other models.

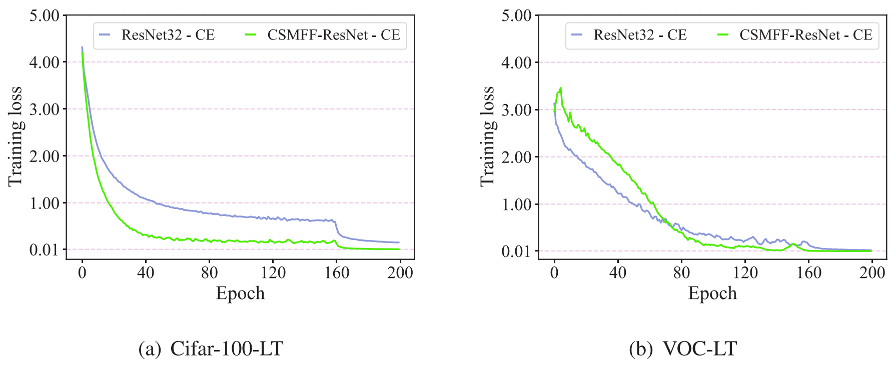


Fig. 8. Comparison of the training losses of different networks on the Cifar-100-LT (ρ=10) and VOC-LT datasets.

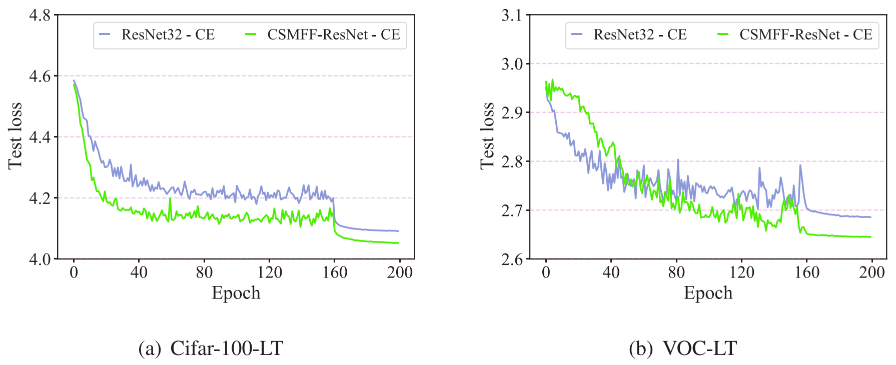


Fig. 9. Comparison of the test losses of different networks on the Cifar-100-LT (ρ=10) and VOC-LT datasets.

5.1. Effectiveness of the channel and spatial multi-scale feature fusion network

In this section, we explore the proposed effectiveness of the channel and spatial multi-scale feature fusion model called ResNet (CSMS-ResNet). Classification with CSMS-ResNet is characterized by the fusion of inter-layer channels and intra-layer spatial features. We compare the baseline network (i.e., ResNet32) with CSMS-ResNet in terms of training loss on the Cifar-100-LT and VOC-LT datasets. We set the loss function as cross-entropy (CE) loss to better compare the performance of the two networks. Fig. 8 shows the effect of CSMS-ResNet on experimental training loss on two long-tailed datasets. The following observations are obtained:

(1) The loss convergence of CSMS-ResNet is faster than that of ResNet32 on both long-tailed datasets. This indicates that CSMS-ResNet can accelerate the loss convergence and reach the optimal global solution.

(2) The loss gradient of CSMS-ResNet is more stable than that of ResNet32 in the post-160 epochs stage on both long-tailed datasets. This proves the effectiveness of CSMS-ResNet, which can retain the detailed inter- and intra-layer information and enhance the semantic information.

In addition, Fig. 9 shows the test losses for different networks on the two long-tailed datasets. Based on this, we can obtain the following conclusions:

(1) The overall trend in training loss is decreasing, as is that in test loss, which means that the model is learning without over- or under-fitting.

(2) When comparing the training and test losses, we can find that the model eventually converges as the number of epochs increases.

Further, we continue to prove the effectiveness of CSMS-ResNet. Hence, we list the results of all long-tailed datasets to evaluate the performance of our proposed network. The results of all datasets are shown in Tables 3 and 4. The best results are marked in bold. The symbol “-" indicates that no strategy was used during the training phase, and “DRW" indicates that the DRW strategy was used during the training phase. We obtained the following observations:

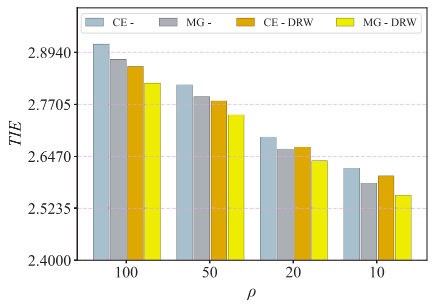
(1) CSMS-ResNet had better *ACC* and F_H scores than ResNet32 under all long-tailed datasets without using the strategy. For instance, the *ACC* values of CSMS- ResNet are 60.03% and 42.19%, respectively, which are about 4.01% and 4.05% higher than those of ResNet32 on the Cifar-100-LT (ρ = 10) and SUN-LT datasets. This indicates that CSMS-ResNet can fully mine the feature information of image samples, which improves classification accuracy.

(2) CSMS-ResNet has better *ACC* and F_H than ResNet32 under the Cifar-100-LT, SUN-LT, and tiredImagenet-LT datasets using the strategy. For example, the *ACC* values of CSMS-ResNet are 49.54% and 19.04%, which are about 3.14% and 0.85% higher than those of ResNet32 on the Cifar-100-LT (ρ = 50) and tiredImagenet-LT datasets, respectively. This shows that using the strategy can improve classification accuracy. When all models use this strategy to enable a fair comparison, CSMS-ResNet still performs well. This shows that the features obtained by CSMS-ResNet have stronger semantic information than the other models.

(3) It is worth noting that the use of the strategy did not improve the *ACC* and F_H scores on the VOC-LT dataset. However, CSMS-ResNet still had higher *ACC* and F_H scores than ResNet32 without using the strategy. This demonstrates the effectiveness of CSMS-ResNet.

5.2. Effectiveness of multi-granularity loss function

In this section, we explore the effectiveness of the proposed multi-granularity (MG) loss function. We construct a multi-granularity loss function through an MGKG. The coarse-grained and fine-grained losses consist of the multi-granularity loss function, which uses coarse-grained knowledge to assist fine-grained knowledge learning. We use three evaluation methods to estimate the experimental results.


(a) ResNet32

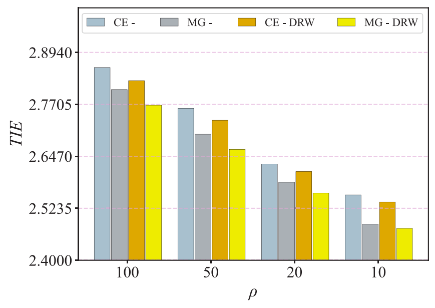

(b) CSMS-ResNet

Fig. 10. Effectiveness of *TIE* with multi-granularity loss under different networks on the Cifar-100-LT dataset.

Table 3
Performance (%) comparison of different networks on the Cifar-100-LT datasets with different ρ.

Network	Strategy	Cifar-100-LT									
		$\rho = 200$		$\rho = 100$		$\rho = 50$		$\rho = 20$		$\rho = 10$	
		ACC	F_H	*ACC*	F_H	*ACC*	F_H	*ACC*	F_H	*ACC*	F_H
ResNet32	–	35.70	62.19	39.04	64.45	44.00	67.72	50.56	71.97	56.02	75.02
CSMS-ResNet	–	38.42	64.08	42.72	66.60	47.61	69.85	55.56	74.71	60.03	77.42
ResNet32	DRW	38.15	63.94	41.81	66.26	46.40	69.16	52.31	72.95	57.50	75.82
CSMS-ResNet	DRW	**40.34**	**65.37**	**44.57**	**67.74**	**49.54**	**70.99**	**57.12**	**75.53**	**61.64**	**78.07**

Table 4
Performance (%) comparison of different networks on the VOC-LT, SUN-LT, and tiredImagenet-LT datasets.

Network	Strategy	VOC-LT		SUN-LT		tiredImagenet-LT	
		$\rho = 57$		$\rho = 27.9$		$\rho = 203$	
		ACC	F_H	*ACC*	F_H	*ACC*	F_H
ResNet32	–	39.62	68.76	38.14	67.29	15.53	51.13
CSMS-ResNet	–	**44.59**	**71.71**	42.19	69.87	17.75	52.16
ResNet32	DRW	39.25	68.44	41.69	69.27	18.19	52.52
CSMS-ResNet	DRW	44.45	71.68	**44.04**	**70.70**	**19.04**	**52.87**

Table 5 and Fig. 10 show the impact of MG on the Cifar-100-LT dataset for the three evaluation methods. We can obtain the following observations:

(1) The multi-granularity loss function can improve the classification accuracy of long-tailed datasets, whether using the strategy or not. For example, the *ACC* and F_H of MG are 44.83% and 68.46% higher than those of CE by 0.83% and 0.74%, respectively, without the strategy. In the case of strategy use, the *ACC* and F_H of MG are 47.46% and 70.06% higher than those of CE by 1.06% and 0.90%, respectively, on the Cifar-100-LT (ρ=50) dataset. This indicates that MG can transfer knowledge of the coarse-grained classes vertically to the fine-grained classes. Hence, the MG is advantageous in hierarchical classification. Further, with the CSMS-ResNet network, the *ACC* and F_H of MG are 50.06% and 71.69% higher than those of CE by 2.45% and 1.84%, respectively, on the Cifar-100-LT (ρ=50) dataset. This indicates that coarse-grained knowledge can transfer well to fine-grained knowledge learning and improve model performance. In addition, we observe that enriched fusion features can better drive coarse- to fine-grained vertical knowledge transfer.

(2) The *TIE* value of MG is less than that of CE in the variable of control strategy, as shown in Fig. 10. The *TIE* value reflects the misclassification degree of a model. The experimental results show that MG reduces the degree of misjudgment by our model.

Additionally, we explore the effectiveness of MG on other long-tailed datasets using three evaluation methods. The experimental results are shown in Table 6 and Figs. 11 and 12. We can obtain the following observations:

(1) With and without the strategy, the *ACC* and F_H of MG are lower than those of CE on the SUN-LT and tiredImagenet-LT datasets (Table 6(a)). The reasons are that the number of classes is too high and the mined feature information is not rich enough, which leads to misclassification.

(2) Table 6(b) shows that, with or without the strategy, the *ACC* and F_H of MG are better than those of CE on the SUN-LT and tiredImagenet-LT datasets. The values are 43.90% and 70.72%, which are 1.71% and 0.85% higher, respectively, than those of CE on the SUN-LT dataset. This shows that the fused features have stronger semantic information and better detail perception, which promote the transfer of coarse-grained knowledge to fine-grained knowledge.

(3) Fig. 12 shows that the *TIE* of MG is lower than that of CE without the strategy. For example, the *TIE* of MG is significantly reduced on the VOC-LT dataset. Likewise, the *TIE* of MG is, again, substantially reduced when utilizing the strategy. This shows that the degree of model misclassification in MGKG is low and that it is necessary to establish a multi-granularity loss function.

To intuitively explain the multi-granularity loss function, we use the t-SNE diagram for visualization. For a fair comparison, we use ResNet32 as the base network and the CE loss function as the base loss function. We select six classes from the VOC-LT test set. Classes 0, 1, and 2 form one coarse-grained class, while classes 3, 4, and 5 form another. The t-SNE visualization on the VOC-LT dataset is shown in Fig. 13.

By observing Fig. 13, we can obtain the following conclusions:

(1) Fig. 13(a) is the t-SNE graph obtained under random parameters, with no separation between coarse-grained classes. Fig. 13(b) is the t-SNE diagram obtained under the parameters of the trained model (ResNet32 with CE). We discover that there is no evidence of separation of the two coarse-grained classes.

(2) The model obtained by multi-granularity loss function training affects feature learning (i.e., the change feature distribution). From Fig. 13(c), we observe that three fine-grained classes belonging to the same coarse-grained class are clustered and push away three fine-grained classes belonging to another coarse-grained class.

5.3. Ablation study

In this section, we describe ablation experiments performed to explore the importance of each component. Firstly, we explore the classification accuracy of local classes on the long-tailed dataset to

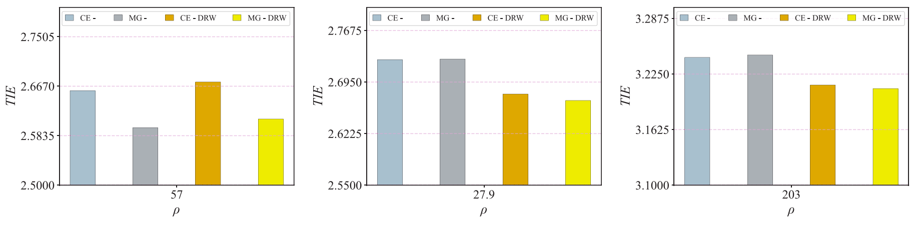
Table 5
Performance (%) comparison of different loss functions on different networks.

(a) ResNet32

Loss	Strategy	Cifar-100-LT							
		$\rho = 100$		$\rho = 50$		$\rho = 20$		$\rho = 10$	
		ACC	F_H	*ACC*	F_H	*ACC*	F_H	*ACC*	F_H
CE	–	39.04	64.45	44.00	67.72	50.56	71.97	56.02	75.02
MG	–	39.50	65.21	44.83	68.46	51.25	72.67	57.23	76.01
CE	DRW	41.81	66.26	46.40	69.16	52.31	72.95	57.50	75.82
MG	DRW	**42.60**	**67.18**	**47.46**	**70.06**	**53.76**	**73.96**	**59.03**	**77.10**

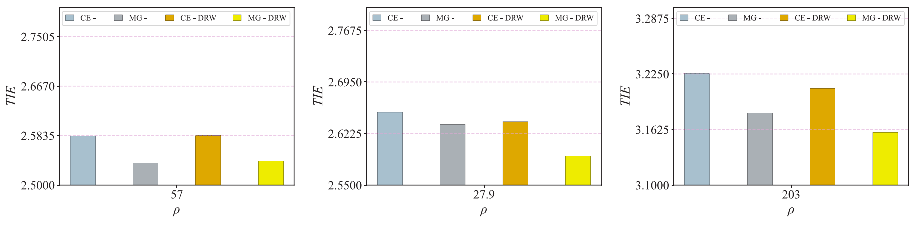
(b) CSMS-ResNet

Loss	Strategy	Cifar-100-LT							
		$\rho = 100$		$\rho = 50$		$\rho = 20$		$\rho = 10$	
		ACC	F_H	*ACC*	F_H	*ACC*	F_H	*ACC*	F_H
CE	–	42.72	66.60	47.61	69.85	55.56	74.71	60.03	77.42
MG	–	43.79	67.83	50.06	71.69	57.37	76.03	63.43	79.71
CE	DRW	44.57	67.74	49.54	70.99	57.12	75.53	61.64	78.07
MG	DRW	**46.36**	**69.31**	**52.34**	**73.05**	**59.37**	**77.12**	**64.18**	**80.13**



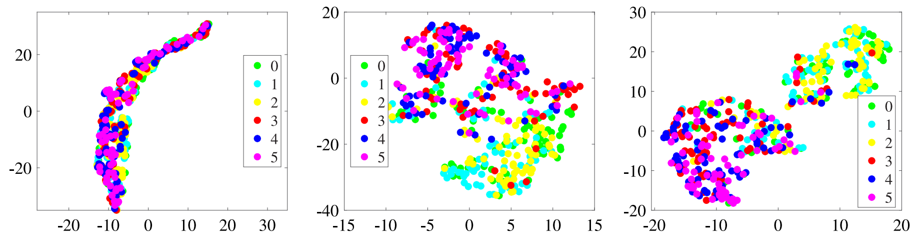
(a) VOC-LT (b) SUN-LT (c) tiredImagenet-LT

Fig. 11. *TIE* of (a) VOC-LT, (b) SUN-LT, and (c) tiredImagenet-LT based on ResNet32.



(a) VOC-LT (b) SUN-LT (c) tiredImagenet-LT

Fig. 12. *TIE* of (a) VOC-LT, (b) SUN-LT, and (c) tiredImagenet-LT based on CSMS-ResNet.



(a) ResNet32 (Random) (b) ResNet32 (c) ResNet32 (MG)

Fig. 13. t-SNE visualization on the VOC-LT dataset.

illustrate the validity of the components. Then, we observe the global classification accuracy by adding different components to obtain our model.

Table 7 shows the local classification accuracy used to validate component performance. We can obtain the following observations:

(1) From Table 7(a), the *ACC*s of the head, middle, and tail classes are all improved when ResNet32 and MG are combined, and when CSMS-ResNet and CE are combined. Further, the proposed MGKT-MFF method is formed by the combination of CSMS-ResNet and MG. The *ACC*s of the head, middle, and tail classes are 71.22%, 43.40%, and 18.42%, respectively, which are 6.95%, 6.25%, and 3.74% higher than the baseline, respectively. This demonstrates that each component is valid. Our model improves the accuracy of the tail class and enhances the performance of the head class.

(2) From Table 7(b), the *ACC*s of the head and middle classes in the MGKT-MFF method are 50.45% and 29.31%, which are 7.13% and 5.64% better than the baseline, respectively. However, the *ACC* of the tail classes is not improved because their number and sample numbers

Table 6
Performance (%) comparison of different loss functions on the VOC-LT, SUN-LT, and tiredImagenet-LT dataset.

(a) *ResNet32 comparison loss*

Loss	Strategy	VOC-LT		SUN-LT		tiredImagenet-LT	
		$\rho = 57$		$\rho = 28$		$\rho = 203$	
		ACC	F_H	*ACC*	F_H	*ACC*	F_H
CE	–	39.62	68.76	38.14	67.29	15.53	51.13
MG	–	**41.48**	**70.45**	37.50	67.06	15.08	50.92
CE	DRW	39.25	68.44	**41.69**	**69.27**	**18.19**	52.52
MG	DRW	40.24	69.80	41.05	69.21	18.02	**52.53**

(b) *CSMS-ResNet comparsion loss*

Loss	Strategy	VOC-LT		SUN-LT		tiredImagenet-LT	
		$\rho = 57$		$\rho = 28$		$\rho = 203$	
		ACC	F_H	*ACC*	F_H	*ACC*	F_H
CE	–	44.59	71.71	42.19	69.87	17.75	52.16
MG	–	**46.45**	**73.05**	43.90	70.72	18.67	53.21
CE	DRW	44.45	71.68	44.04	70.70	19.04	52.87
MG	DRW	45.97	72.84	**46.26**	**72.24**	**20.26**	**54.10**

Table 7
The *ACC* (%) of local class on the Cifar-100-LT and VOC-LT datasets. (Best results are highlighted in bold. The values in () represent the number of classes.).

(a) Cifar-100-LT

Network	Loss	Strategy	$\rho = 50$			
			Head (41)	Middle (40)	Tail (19)	All (100)
ResNet32	CE	–	64.27	37.15	14.68	44.00
ResNet32	MG	–	65.89	37.28	15.32	44.83
CSMS-ResNet	CE	–	68.12	40.10	17.89	47.61
CSMS-ResNet	MG	–	**71.22**	**43.40**	**18.42**	**50.06**

(b) VOC-LT

Network	Loss	Strategy	$\rho = 57$			
			Head (11)	Middle (8)	Tail (1)	All (20)
ResNet32	CE	–	43.32	23.67	0	39.62
ResNet32	MG	–	45.43	24.45	0	41.48
CSMS-ResNet	CE	–	48.58	27.43	0	44.59
CSMS-ResNet	MG	–	**50.45**	**29.31**	0	**46.65**

are very few in the training process. However, this fails to affect the effectiveness of the MGKT-MFF method.

Further, to better illustrate the performance of MGKT-MFF, we compare the MGKT-MFF method with the baseline method in terms of accuracy for each class. A comparison of the *ACC* results is presented in Table 8.

We find improved classification accuracy with more classes. It is worth noting that some classes have fewer samples, but their accuracy is much improved. The classification accuracy of classes *Bicycle* and *Chair* are 14.28% and 10.59% higher than that of the baseline method. It shows that driving of the transfer of coarse-grained to fine-grained knowledge by using multi-scale fusion features is critical and effective.

To further verify the superiority of our model, we performed a global classification accuracy ablation experiment on all long-tailed datasets. The results are shown in Table 9. The following observations can be obtained:

(1) The *ACC* obtained using CSMS-ResNet alone is higher than that using the baseline method on all datasets. For example, the *ACC* of 42.19% obtained utilizing only CSMS-ResNet is 4.05% higher than that of the baseline on SUN-LT. Nevertheless, the *ACC* obtained using MG components exclusively is lower than that of the baseline on SUN-LT and tiredImagenet-LT.

The main reason is that there are 15 and 34 coarse-grained classes in SUN-LT and tiredImagenet-LT, accounting for about 1/21 and 1/17 of the fine-grained classes, respectively. The proportion of coarse-grained classes in SUN-LT and tiredImagenet-LT is smaller than the proportion of coarse-grained classes in VOC-LT; i.e., the coarse-grained classes account for 1/5 of the fine-grained classes in VOC-LT. Therefore, when MGKG is used for multi-granularity knowledge transfer, the model performs well on the VOC-LT dataset, which has a high proportion of coarse-grained classes. The SUN-LT and tiredImagenet-LT datasets, which have a low proportion of coarse-grained classes, do not perform well.

It is worth mentioning that when using both CSMS-ResNet and MG components simultaneously, the *ACC* is improved compared to when only using CSMS-ResNet. For example, the *ACC*s on SUN-LT and tiredImagenet-LT are 43.90% and 18.67%, which are 1.71% and 0.92% higher, respectively, than when using CSMS-ResNet alone. This demonstrates the validity of the MG component and that the multi-scale fusion feature obtained by CSMS-ResNet can drive multi-granularity knowledge transfer.

(2) The *ACC*s obtained using CSMS-ResNet or MG alone are higher than that of the baseline method on the VOC-LT dataset. Further, the accuracy is even better when both components are used simultaneously. However, the accuracy decreases slightly when the three components are used simultaneously. The reason is that the distribution of the VOC-LT dataset is too dispersed. Nonetheless, this does not affect the advantages of the DRW strategy on other datasets.

5.4. Visualization

To demonstrate the interpretability of the model, we visualize the validity of the MGKT-MFF model. We use the Grad-CAM [40] method to show the attention scope of our method in the form of a heat map. The main results are shown in Fig. 14. We can obtain the following conclusions:

(1) Compared with the baseline heat maps, the heat maps of MGKT-MFF indicate that the extracted features are relevant and rich in classification information. The features of the sample *cat* extracted from the baseline model contain clutter and are not rich enough. However, the features extracted using MGKT-MFF focus more on the areas related to classification and are rich in features.

(2) The features obtained by MGKT-MFF through multi-scale feature fusion of channel and spatial information weaken the background influence and focus on some major regions close to image labels. The sample-related information is preserved and noise is removed, making the heat maps clearer and more cohesive.

5.5. Statistical tests of different models

We performed statistical tests on the proposed method and other methods for comparison of their performance. In general, given K compared methods and M datasets, r_i^j denotes the rank of the jth method on the ith dataset. We calculate the average rank among all datasets according to the different methods. The formula for the average rank is as follows:

$$R_i = \frac{1}{M}\sum_{i=1}^{M} r_i^j. \tag{16}$$

The usual scheme is to rank the experimental results of all methods on the same dataset. Then, the average rank of each method over all datasets is calculated based on the ranking values.

Table 10 shows the classification accuracy ranking of the different methods.

From the results in Table 10, we can conclude that the proposed method outperforms all comparison methods in terms of average rank. Its average rank is 0.6 higher than the second-ranked method and 4.4 higher than the lowest-ranked one. Therefore, the proposed method is highly competitive compared with the other methods.

Table 8
Different classification methods on the VOC-LT dataset for each class accuracy (%). The best results are highlighted in bold. The value of () represents the number of samples of the class.

(a) Class 1–10

Method	Person(456)	Cat(404)	Dog(396)	Bird(357)	Aeroplane(286)	Car(231)	Train(179)	Boat(152)	Horse(122)	Sheep(122)
Baseline	58.06	43.84	32.89	38.71	62.17	38.43	47.18	36.23	26.15	26.92
MGKT-MFF	**67.77**	**55.21**	**41.58**	**44.28**	**66.67**	**43.06**	**54.87**	**41.30**	26.15	**36.15**

(b) Class 11–20

Method	Cow(107)	Bicycle(95)	Bottle(92)	Tv(84)	Chair(83)	Motorbike(79)	Bus(71)	Pottedplant(67)	Sofa(46)	Table(8)
Baseline	**26.02**	16.33	**9.18**	37.89	9.41	41.56	**43.66**	16.67	16.67	0.00
MGKT-MFF	23.58	**30.61**	7.14	**43.16**	**20.00**	**46.75**	39.44	**27.27**	**20.83**	0.00

Table 9
The contributions (*ACC*%) of different components in our model on all datasets. The best results are highlighted in bold.

(a)

CSMS-ResNet	MG	DRW	Cifar-100-LT				
			$\rho = 200$	$\rho = 100$	$\rho = 50$	$\rho = 20$	$\rho = 10$
			35.70	39.04	44.00	50.56	56.02
	✓		35.52	39.50	44.83	51.25	57.23
		✓	38.15	41.81	46.4	52.31	57.50
	✓	✓	37.88	42.60	47.46	53.76	59.03
✓			38.42	42.72	47.61	55.56	60.03
✓	✓		38.64	43.79	50.06	57.37	63.43
✓		✓	40.34	44.57	49.54	57.12	61.64
✓	✓	✓	**40.87**	**46.36**	**52.34**	**59.37**	**64.18**

(b)

CSMS-ResNet	MG	DRW	VOC-LT	SUN-LT	tiredImagenet-LT
			$\rho = 57$	$\rho = 27.9$	$\rho = 203$
			39.62	38.14	15.53
	✓		41.48	37.50	15.08
		✓	39.25	41.69	18.19
	✓	✓	40.24	41.05	18.02
✓			44.59	42.19	17.75
✓	✓		**46.45**	43.90	18.67
✓		✓	44.45	44.04	19.04
✓	✓	✓	45.97	**46.26**	**20.26**

Table 10
Average ranks of the accuracy of different methods on different datasets.

	CE	L2RW	Class-balanced CE	Meta-weight net	MetaSAug with CE	Ours
Cifar-100-LT($\rho = 200$)	34.70(5)	33.00(6)	35.56(4)	36.62(3)	39.84(2)	**40.87(1)**
Cifar-100-LT($\rho = 100$)	38.46(6)	38.90(4)	38.77(5)	41.61(3)	**46.87(1)**	46.36(2)
Cifar-100-LT($\rho = 50$)	44.02(5)	43.17(6)	44.79(4)	45.66(3)	51.90(2)	**52.34(1)**
Cifar-100-LT($\rho = 20$)	51.06(5)	50.75(6)	51.94 (4)	53.04(3)	57.85(2)	**59.37(1)**
Cifar-100-LT($\rho = 10$)	55.73(5)	52.12(6)	57.57(4)	58.91(3)	61.73(2)	**64.18(1)**
Avg. rank	5.2	5.6	4.2	3	1.8	**1.2**

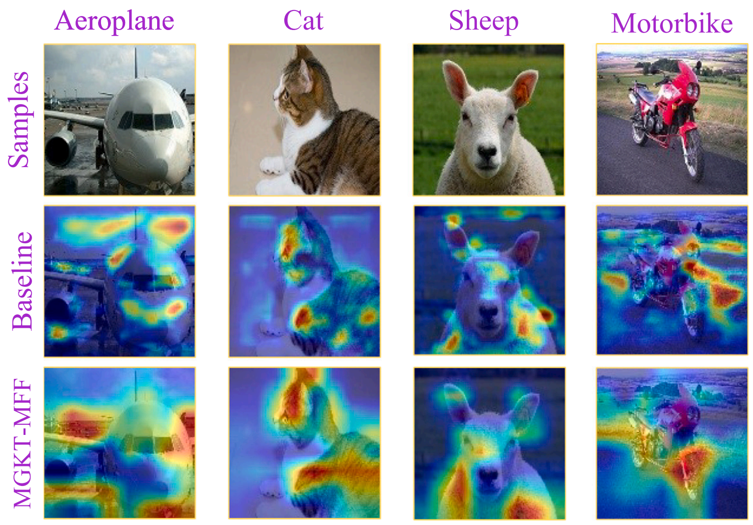


Fig. 14. Heat maps visualization of MGKT-MFF.

Table 11
The *ACC* (%) of on the Cifar-100-LT dataset under different imbalance settings.

Imbalance factor (ρ)	200	100	50	20	10
CE[a]	34.70	38.46	44.02	51.06	55.73
Mixup[a] [19]	–	39.54	44.99	–	58.02
L2RW[a] [34]	33.00	38.90	43.17	50.75	52.12
Class-balanced CE[a] [17]	35.56	38.77	44.79	51.94	57.57
Class-balanced fine-tuning[a] [35]	38.66	41.50	46.22	52.30	57.57
Meta-weight net[a] [36]	36.62	41.61	45.66	53.04	58.91
Meta-class-weight with CE[a] [37]	39.31	43.35	48.53	55.62	59.58
LDAM-DRW[b] [7]	–	42.04	47.62	–	58.71
BBN[b] [25]	–	42.56	47.02	–	59.12
Hybrid-PSC[b] [30]	–	44.97	48.93	–	62.37
MiSLAS[b] [21]	–	47.00	52.30	–	63.20
Bag of Tricks[b] [38]	–	**47.83**	51.69	–	–
MetaSAug with CE[b] [39]	39.84	46.87	51.90	57.85	61.73
Ours	**40.87**	46.36	**52.34**	**59.37**	**64.18**

[a]Indicates results reported in [39].

[b]Represents a result copied from an original paper.

5.6. Efficiency comparison with other models

In this section, we compare the efficiency of our model with those of other models. We constructed four long-tailed datasets for use in comparative experiments. We further utilized these datasets to demonstrate the applicability and feasibility of our model. The main results are shown in Table 11, with the best results marked in bold. The conclusions are as follows:

(1) Compared with the CE (baseline) model, our model achieves efficiency scores of 40.87% and 64.18% on the two Cifar-100-LT datasets (ρ = 200 and ρ = 10), which are 6.17% and 8.45% better than those of the baseline model, respectively. This illustrates the feasibility of the proposed model.

(2) Compared with the MetaSAug with CE (optimal) model (i.e., the optimal model on the Cifar-100-LT (ρ = 200 and ρ = 20) datasets), our model achieves scores of 40.87% and 59.37% on both Cifar-100-LT datasets, which are 1.03% and 1.52% higher, respectively. Our model still performs well when the Cifar-100-LT (ρ = 200) dataset is extremely unbalanced. Hence, our model has high practicability even in an extremely unbalanced dataset.

Table 12 shows the results of the experiments performed on different datasets. We can obtain the following conclusions:

(1) The *ACC* and F_H of MGKT-MFF are 2.57% and 2.16% higher than those of the LDAM loss (optimal) model without the DRW strategy on the VOC-LT dataset. However, these values *ACC* are lower but are still better than those of the LDAM loss model when the MGKT-MFF and DRW strategies are combined. The reason is that the unbalanced distribution of the VOC-LT dataset is too messy, which results in an unstable model when using the DRW strategy.

(2) The *ACC* and F_H values of MGKT-MFF are 0.48% and 0.69% higher than those of the CE-DRW (optimal) model without using the DRW strategy on the tiredImagenet-LT dataset. Furthermore, the *ACC* scores are 20.26% and 54.10%, which are 2.07% and 1.58% better than those of the CE-DRW model when MGKT-MFF is combined with the DRW strategy.

(3) The *TIE* of our model on the SUN-LT dataset is 2.5910, which is 0.0841 lower than that of the CE-DRS (optimal) model. Similarly, on the tiredImagenet-LT dataset, the *TIE* of our model is 3.1592, which is 0.0535 lower than that of the CE-DRW (optimal) model. The experimental results show that our model can reduce the degree of misclassification.

6. Conclusions and future work

In this paper, we proposed a hierarchical long-tailed classification method based on multi-granularity knowledge transfer driven by multi-scale fusion features. The method fully exploits rich multi-scale fusion features to drive the vertical transfer of coarse- to fine-grained knowledge. First, we presented a multi-scale feature fusion network that obtains rich and distinguishable features. Subsequently, we leveraged an MGKG to vertically transfer knowledge from coarse- to fine-grained classes to reduce the ineffective transfer of transfer learning. The features with richness and discriminability drive multi-granularity knowledge transfer to improve long-tailed classification performance. Through extensive experiments on several long-tailed benchmark datasets, we demonstrated the effectiveness of our method and its superiority in comparison with similar methods. The current method considers hierarchical knowledge transfer from coarse- to fine-grained classes; i.e., vertical transfer.

In this work, we only used MGKG as auxiliary knowledge for vertical transfer to enhance the learning of tail classes. In future work, we will consider using randomly designed graphs as auxiliary knowledge for vertical transfer. In addition, we will focus on combining HKT (i.e., knowledge transfer from the head to tail classes) with vertical transfer to solve the current long-tailed classification problem. Furthermore, we will continue to explore the impacts of feature fusion on the classification results.

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.

Data availability

Data will be made available on request.

Table 12
Performance comparison among the different methods on the long-tailed datasets VOC-LT, SUN-LT, and tiredImagenet-LT.

Dataset	VOC-LT			SUN-LT			tiredImagenet-LT		
Imbalance ratio	ρ = 57			ρ = 27.9			ρ = 203		
Evaluation measures	*ACC* (%)	F_H (%)	*TIE*	*ACC* (%)	F_H (%)	*TIE*	*ACC* (%)	F_H (%)	*TIE*
CE	39.62	68.76	2.6595	38.14	67.29	2.7265	15.17	50.95	3.2464
CE-RS [33]	39.67	69.01	2.6481	39.43	68.03	2.7073	11.74	47.37	3.3927
CE-RW [2]	36.56	67.18	2.6934	40.12	68.36	2.7009	10.24	45.85	3.4534
CE-CB [17]	33.94	65.61	2.7357	40.28	68.53	2.6943	10.71	46.31	3.4357
CE-DRS [7]	39.62	68.77	2.6614	41.80	69.36	2.6751	18.14	52.51	3.2123
CE-DRW [7]	39.25	68.44	2.6742	41.69	69.27	2.6782	18.19	52.52	3.2127
Focal loss [6]	38.57	68.29	2.6664	38.41	67.54	2.7167	15.4	51.07	3.2436
Focal-CB [17]	34.42	65.49	2.7515	39.31	67.91	2.7117	5.76	42.31	3.5762
LDAM loss [7]	43.88	70.89	2.6180	32.41	63.74	2.8253	14.29	50.05	3.2833
LDAM-DRS [7]	43.01	70.55	2.6222	38.19	66.73	2.7614	16.68	51.15	3.2649
LDAM-DRW [7]	42.84	70.32	2.6314	37.98	66.75	2.7558	16.91	51.3	3.2599
MGKT-MFF (Ours)	**46.45**	**73.05**	**2.5372**	**43.90**	**70.72**	**2.6353**	**18.67**	**53.21**	**3.1812**
(MGKT-MFF) - DRW (Ours)	**45.97**	**72.84**	**2.5405**	**46.26**	**72.24**	**2.5910**	**20.26**	**54.10**	**3.1592**

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant Nos. 62376114 and 62141602, and the Natural Science Foundation of Fujian Province under Grant No. 2021J011003.

References

[1] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, S.X. Yu, Large-scale long-tailed recognition in an open world, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2537–2546.

[2] C. Huang, Y. Li, C.C. Loy, X. Tang, Learning deep representation for imbalanced classification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5375–5384.

[3] G. Tian, J. Liu, H. Zhao, W. Yang, Small object detection via dual inspection mechanism for uav visual images, Appl. Intell. 52 (4) (2022) 4244–4257.

[4] M. Koziarski, Radial-based undersampling for imbalanced data classification, Pattern Recognit. 102 (2020) 107262.

[5] Y. Zhou, Q. Hu, Y. Wang, Deep super-class learning for long-tail distributed image classification, Pattern Recognit. 80 (2018) 118–128.

[6] T. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.

[7] K. Cao, C. Wei, A. Gaidon, N. Arechiga, T. Ma, Learning imbalanced datasets with label-distribution-aware margin loss, in: International Conference on Neural Information Processing Systems, 2019, pp. 1567–1578.

[8] J. Jiang, Z. He, S. Zhang, X. Zhao, J. Tan, Learning to transfer focus of graph neural network for scene graph parsing, Pattern Recognit. 112 (2021) 107707.

[9] X. Yin, X. Yu, K. Sohn, X. Liu, M. Chandraker, Feature transfer learning for face recognition with under-represented data, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 5704–5713.

[10] P. Chu, X. Bian, S. Liu, H. Ling, Feature space augmentation for long-tailed data, in: European Conference on Computer Vision, 2020, pp. 694–710.

[11] J. Liu, Y. Sun, C. Han, Z. Dou, W. Li, Deep representation learning on long-tailed data: A learnable embedding augmentation perspective, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 2970–2979.

[12] X. Wang, J. Xu, T. Zeng, L. Jing, Local distribution-based adaptive minority oversampling for imbalanced data classification, Neurocomputing 422 (2021) 200–213.

[13] G. Wang, DGCC: data-driven granular cognitive computing, Granul. Comput. 2 (4) (2017) 343–355.

[14] J. Deng, W. Dong, R. Socher, L. Li, K. Li, L. FeiFei, Imagenet: A large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

[15] S. Guo, H. Zhao, Hierarchical classification with multi-path selection based on granular computing, Artif. Intell. Rev. 54 (3) (2021) 2067–2089.

[16] C. Santiago, C. Barata, M. Sasdelli, G. Carneiro, J.C. Nascimento, LOW: Training deep neural networks by learning optimal sample weights, Pattern Recognit. 110 (2021) 107585.

[17] Y. Cui, M. Jia, T. Lin, Y. Song, S. Belongie, Class-balanced loss based on effective number of samples, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9268–9277.

[18] Z. Deng, H. Liu, Y. Wang, C. Wang, Z. Yu, X. Sun, PML: Progressive margin loss for long-tailed age classification, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 10503–10512.

[19] H. Zhang, M. Cisse, Y.N. Dauphin, D.L. Paz, mixup: Beyond empirical risk minimization, in: IEEE International Conference on Learning Representations, 2018, pp. 1–13.

[20] V. Verma, A. Lamb, C. Beckham, A. Najafi, I. Mitliagkas, D. Lopez-Paz, Y. Bengio, Manifold mixup: Better representations by interpolating hidden states, in: International Conference on Machine Learning, 2019, pp. 6438–6447.

[21] Z. Zhong, J. Cui, S. Liu, J. Jia, Improving calibration for long-tailed recognition, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 16489–16498.

[22] H. Chou, S. Chang, J. Pan, W. Wei, D. Juan, Remix: Rebalanced mixup, in: European Conference on Computer Vision, 2020, pp. 95–110.

[23] H. Zhao, S. Yu, Cost-sensitive feature selection via the $l_{2,1}$-norm, Internat. J. Approx. Reason. 104 (2019) 25–37.

[24] Y. Dai, F. Gieseke, S. Oehmcke, Y. Wu, K. Barnard, Attentional feature fusion, in: IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 3560–3569.

[25] B. Zhou, Q. Cui, X. Wei, Z. Chen, BBN: Bilateral-branch network with cumulative learning for long-tailed visual recognition, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9719–9728.

[26] M. Everingham, L. Van Gool, C.K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (VOC) challenge, Int. J. Comput. Vis. 88 (2) (2010) 303–338.

[27] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, A. Oliva, Places: An image database for deep scene understanding, 2016, arXiv preprint arXiv:1610.02055.

[28] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J.B. Tenenbaum, H. Larochelle, R.S. Zemel, Meta-learning for semi-supervised few-shot classification, in: International Conference on Learning Representations, 2018.

[29] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[30] P. Wang, K. Han, X. Wei, L. Zhang, L. Wang, Contrastive learning based hybrid networks for long-tailed image classification, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 943–952.

[31] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, K. He, Accurate, large minibatch SGD: Training imagenet in 1 hour, 2017, arXiv preprint arXiv:1706.02677.

[32] A. Kosmopoulos, I. Partalas, E. Gaussier, G. Paliouras, I. Androutsopoulos, Evaluation measures for hierarchical classification: a unified view and novel approaches, Data Min. Knowl. Discov. 29 (3) (2015) 820–865.

[33] N. Japkowicz, The class imbalance problem: Significance and strategies, in: Artificial Intelligence, 2000, pp. 111–117.

[34] M. Ren, W. Zeng, B. Yang, R. Urtasun, Learning to reweight examples for robust deep learning, in: International Conference on Machine Learning, 2018, pp. 4334–4343.

[35] Y. Cui, Y. Song, C. Sun, A. Howard, S. Belongie, Large scale fine-grained categorization and domain-specific transfer learning, in: IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4109–4118.

[36] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, D. Meng, Meta-weight-net: Learning an explicit mapping for sample weighting, Adv. Neural Inf. Process. Syst. 32 (2019) 1919–1930.

[37] M.A. Jamal, M. Brown, M. Yang, L. Wang, B. Gong, Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 7610–7619.

[38] Y. Zhang, X. Wei, B. Zhou, J. Wu, Bag of tricks for long-tailed visual recognition with deep convolutional neural networks, in: AAAI Conference on Artificial Intelligence, 2021, pp. 3447–3455.

[39] S. Li, K. Gong, C.H. Liu, Y. Wang, F. Qiao, X. Cheng, MetaSAug: Meta semantic augmentation for long-tailed visual recognition, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 5212–5221.

[40] R.R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, D. Batra, Grad-CAM: Visual explanations from deep networks via gradient-based localization, Int. J. Comput. Vis. 128 (2016) 336–359.

Wei Zhao is currently pursuing the M.S. degree from the School of Computer Science, Minnan Normal University, Zhangzhou, China. His current research interests include granular computing, machine learning, and long-tailed learning for hierarchical classification.

Hong Zhao is currently a Professor of the School of Computer Science and the Key Laboratory of Data Science and Intelligence Application, Minnan Normal University, Zhangzhou, China. She received the Ph.D degree from Tianjin University, Tianjin, China. Her current research interests include few-shot learning, granular computing, and data mining for hierarchical classification.