Information Sciences 608 (2022) 900–916

Contents lists available at ScienceDirect

Information Sciences

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

Multi-task convolutional neural network with coarse-to-fine knowledge transfer for long-tailed classification

Zhengyu Li [a,b], Hong Zhao [a,b,*], Yaojin Lin [a,b]

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

ARTICLE INFO

Article history:
Received 11 September 2021
Received in revised form 3 May 2022
Accepted 2 July 2022
Available online 6 July 2022

Keywords:
Long-tailed classification
Coarse-to-fine knowledge transfer
Multi-task convolutional neural network
Multi-granularity

ABSTRACT

Long-tailed classifications make it very challenging to deal with class-imbalanced problems using deep convolutional neural networks (CNNs). Existing solutions based on re-balancing methods perform well and use single-task CNNs to train each fine-grained class independently. However, classification tasks are multiplex and involve a coarse-to-fine hierarchical relation. In this paper, we propose a coarse-to-fine knowledge transfer based multi-task CNN, which utilizes the coarse-to-fine structure to promote long-tailed learning. First, we construct a tail hierarchical structure in a coarse-to-fine way to pay greater attention to tail classes than head classes. Second, a multi-task CNN is adopted to simultaneously train coarse- and fine-grained tasks to extract a more generalized knowledge representation than the single-task CNN. Third, we design a coarse-to-fine knowledge transfer strategy to adaptively adjust the task weights to improve fine-grained performance. Extensive experiments on benchmark datasets show that our model achieves better gains than the re-balancing methods. In particular, the proposed model is 3.25% more accurate than the second-best method on the long-tailed tieredImageNet dataset.

1. Introduction

In classification tasks, success is dependent on the ability of deep Convolutional Neural Networks (CNNs) to process class-balanced datasets [1]. The CNNs are powerful tools for analyzing, understanding, and representing data in many fields such as disease prevention [2], image classification [3], and object detection [4]. The advantage of CNNs is that they can extract high-quality characteristics of classes, which helps to obtain better representations and improve classification accuracy [5]. Early, neural networks were employed to classify handwritten digits [6]. As computing power increased, GoogleNet was designed to explore deeper convolutions and be trained on larger-scale data [7]. Subsequently, ResNet adopted a residual learning framework to ease the training of networks [8]. Nowadays, CNNs have been widely utilized to deal with various tasks and applications.

Real-world datasets often have long-tailed distributions for fine-grained class labels [9,10], where head classes have a large number of samples and tail classes have few. This poses great challenges to CNNs that have excellent performance on class-balanced datasets [11]. Moreover, classification task complexity increases with the scale of the datasets, *i.e.*, the

* Corresponding author.
E-mail address: hongzhaocn@163.com (H. Zhao).

https://doi.org/10.1016/j.ins.2022.07.015
0020-0255/

number of samples and labels and the target class similarity [12]. In such scenarios, CNNs suffer from generalizing to tail classes due to the data-hungry restrictions of CNN and the leading role of head classes [13].

Re-balancing methods are effective in promoting the CNN generalization and consist of re-sampling and re-weighting. Re-sampling strategies attempt to build uniform distributions from original imbalanced datasets. Re-sampling consists of over-sampling and under-sampling: (1) Over-sampling is a natural approach using repeating tail data, which has proven to be effective [14,15]. Simply over-sampling tail samples easily leads to over-fitting. Thus, Chawla et al. [16] proposed a synthetic minority over-sampling technique (SMOTE) to augment data in the feature space. Further, several variants of SMOTE based on hierarchical clustering have been suggested accordingly [17,18]. (2) Under-sampling abandons data to avoid a bias toward head classes [19,20]. Considering information loss from the head classes, Lin et al. [21] developed a clustering-based under-sampling strategy to reduce the risk of removing useful data from the head classes.

Re-weighting [22,23] is another effective strategy, which allocates weights according to inverse class frequency. Its use has become popular with complex and large-scale datasets. For cases with very high sample numbers, Cui et al. [24] assigned weights considering the inverse effective number of classes to avoide data overlap. Instead of managing all classes, Lin et al. [25] lowered the weights of well-classified head samples and reshaped the standard cross-entropy loss to Focal loss. In contrast, Cao et al. [26] encouraged tail classes to have higher margins and raised label-distribution-aware margin loss. These methods promote the CNN training, which is in expectation closer to the test distribution. Therefore, they help to alleviate the long-tailed fine-grained challenge.

However, a hidden hypothesis behind most of the aforementioned schemes is that tasks for distinguishing all target classes are independent of each other. This may not be inconsistent with close inter-dependent relations and strong inter-class visual similarities among classes. Concretely, a traditional single-task CNN is designed as an N-way flat classifier (N is the total number of target classes) to conduct a single classification task. The classifier is expected to tell one target from the other of N - 1 classes directly when starting from a random initialization point.

An illustration of a single-task CNN is shown in Fig. 1 to explain its fine-grained visual complexity: (1) Intuitively, it is tough to distinguish *Bike* from *Motor* on account of their similar shapes, textures, and colors. (2) Despite their distinct appearances, it is difficult for a flat CNN to distinguish *Motor* from *Dolphin* due to the latter being rarely represented in the long-tailed distribution. (3) It is very hard to judge between *Dolphin* and *Whale* when both of the above conditions

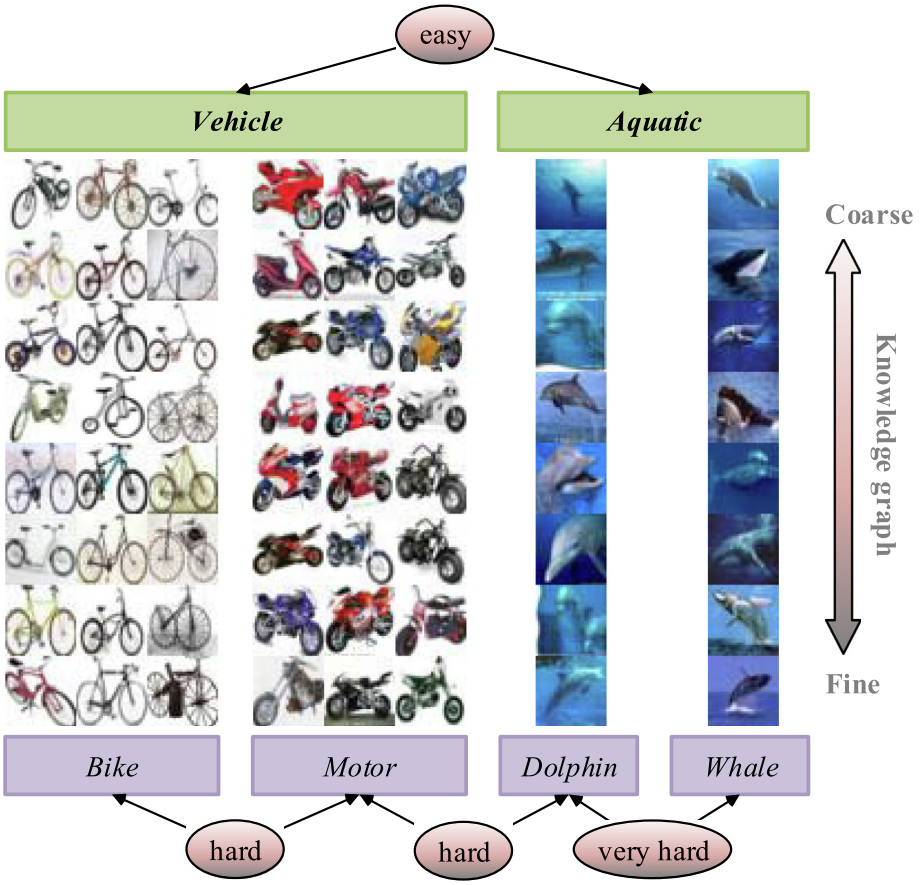

Fig. 1. Comparison of visual complexity between fine granularity and coarse granularity on the long-tailed CIFAR dataset.

are met. Similarly, Kuang et al. [27] suggested that some classes with strong inter-class relations are not acceptable for use with an *N*-way flat CNN. Moreover, random initialization of the CNN undoubtedly worsens the above limitations. As Erhan et al. [28] showed, a CNN easily becomes stuck in local minima when starting from the random initialization. Therefore, a single-task flat CNN has difficulty in generalization with long-tailed classification tasks.

Fortunately, a knowledge graph usually exists in vast classes [29,30], which contributes to organizing classes effectively and discovering inter-class knowledge relations rapidly. One of the most popular knowledge graphs is the semantic hierarchical tree [31,32]. This groups fine-grained words into coarse-grained sets by WordNet [33,34]. In Fig. 1, both classes *Bike* and *Motor* belong to the same coarse-grained class, *Vehicle*, and the remaining classes belong to another coarse-grained class, *Aquatic*, defined within the long-tailed CIFAR [35] dataset. Thus, similar fine-grained classes are grouped into discriminative coarse-grained classes. This is the reason why visual complexity is much easier to process when coarse-grained than fine-grained. Many studies [36,37] have also proven that the classification ability is improved by exploiting the knowledge graph in a coarse-to-fine way. In general, the knowledge graph has the following advantages: (1) The CNN easily classifies samples into coarse-grained classes based on their distinctive shapes and relatively uniform distribution. (2) The CNN can extract a coarse-grained knowledge representation that helpfully distinguishes fine-grained target classes.

The main contribution of this paper is the application of a knowledge graph that improves the CNN generalization of tail classes. We propose a multi-task CNN based on coarse-to-fine knowledge transfer (MCKT). First, we establish a tail hierarchical structure for employing the coarse-to-fine relations among tail classes. The construction not only splits a task into a group of small sub-classification tasks but also makes the CNN focus on the tail classes. Second, we employ a multi-task CNN to take care of both coarse- and fine-grained learning simultaneously. The architecture extracts a shared knowledge representation and learns discriminative information, which are both important for the final fine-grained task. Third, a coarse-to-fine knowledge transfer strategy is developed for adaptively learning each task. Thus, the multi-task CNN can effectively transfer the useful coarse-grained knowledge to the fine-grained task.

We evaluate our MCKT technique on six long-tailed datasets including the real-world dataset iNaturalist. We conduct ablation studies to demonstrate the effectiveness of our MCKT in terms of different evaluation measures. In addition, our MCKT is proven to be effective in comparison with re-balancing methods. Moreover, our model is comparable with some existing state-of-the-art methods, such as Focal. In particular, our model improves the accuracy by 3.25% compared with Focal on the long-tailed tieredImageNet dataset.

The remainder of this paper is organized as follows. In Section 2, we describe details of our approach. In Section 3, we present experimental datasets and implementation details. In Section 4, we analyze experimental results and compare them with state-of-the-art methods. In Section 5, we conclude this paper.

2. Coarse-to-fine knowledge transfer via multi-task CNN

In this section, we firstly present a general framework of the proposed MCKT and then elaborate on its components.

2.1. Framework of the MCKT model

A general framework of the MCKT model is shown in Fig. 2:

(1) Tail hierarchical structure establishment: We obtain coarse-grained head and tail class labels in different ways, and integrate them into the tail hierarchical structure.

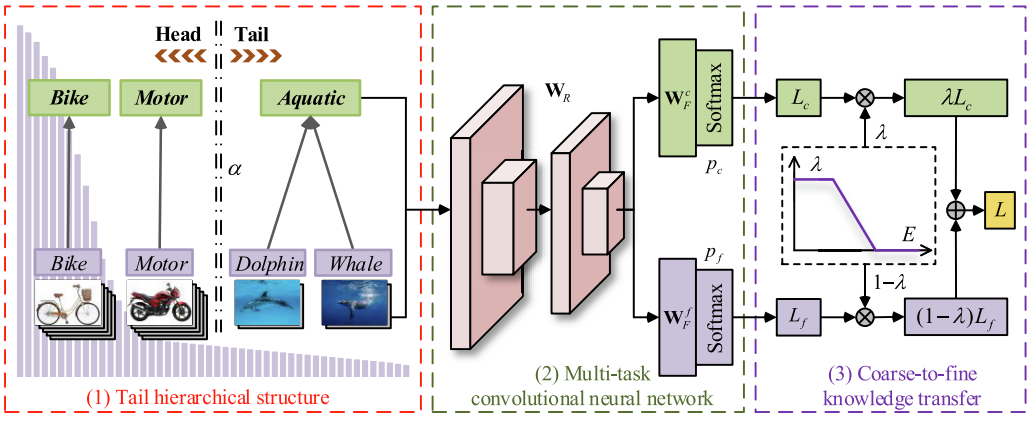

Fig. 2. Framework of the MCKT model.

(2) Multi-task CNN construction: Input data with fine- and coarse-grained labels are fed into the multi-task CNN for training.
(3) Coarse-to-fine knowledge transfer: The training process of the multi-task CNN is divided into three stages: coarse-grained pre-training, multi-grained knowledge transfer stage, and fine-grained target-domain, which are performed adaptively by assigning different learning weights to each stage.

2.2. Tail hierarchical structure establishment

Coarse-to-fine knowledge structures usually exist in classification tasks. The structure often appears in the form of a hierarchical tree containing fruitful knowledge relations and granularity information among classes.

Consider a hierarchical tree knowledge graph $\mathbf{G} = \{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_N\}$ that contains granularity relations for N classes. We denote the granularity relation for the i^{th} class as $\mathbf{g}_i = \left\{y_{c_1}, y_{c_2}, \ldots, y_{c_{h_i}}\right\}$, where $i \in \{1, 2, \ldots, N\}$, h_i denotes the height and $y_{c_{h_i}}$ is the granularity label corresponding to the height h_i. Thus, a greater h_i value represents finer granularity.

We use Example 1 to provide an intuitive interpretation of the hierarchical tree **G**.

Example 1. Fig. 3 depicts an example of a hierarchical tree. This is a subtree of the PASCAL Visual Object Classes (VOC) dataset, where the maximum height is 3. We focus on two different perspectives for the hierarchical tree. (1) From the vertical perspective, class descriptions vary at different heights. The class description closer to the leaf node is more specific, while that farther away the leaf node is more abstract. For example, $\mathbf{g}_{Bottle} = \{ROOT, Household, Bottle\}$, *Household* is a coarse-grained class with height = 2 and its child node is *Bottle* where the height is 3 (the finest level of granularity). (2) From the horizontal perspective, some classes belonging to the same parent node have similar properties with a high probability, while those that do not satisfying the sibling relationships have often discriminative appearances. For example, *Car* and *Bus* have the similar shape since both are part of the *Vehicles* class, while *Bottle* has a distinctive shape and belongs to the Household class. In general, the coarser granularity class represents the more abstract. In addition, parent–child relations in a hierarchical tree are also described as coarse-to-fine relations.

Let $S = \sum_{i=1}^{N} S_i$ denote the total target sample size of the classification task, where S_i is the number of the i^{th} target class, $i \in \{1, 2, \ldots, N\}$. Without loss of generality, we assume that these classes are sorted by cardinality in decreasing order, *i.e.*, $S_1 \geqslant S_2 \geqslant \ldots \geqslant S_N$. Let $t = \lfloor \alpha N \rfloor$ denote a boundary for counts, where $\lfloor \cdot \rfloor$ is an operation for taking an integer and α is a hyper-parameter acting on the number of classes N. We divide the classes into two groups according to the boundary t. They are defined as head classes (*i.e.*, $S_1 \geqslant S_2 \geqslant \ldots \geqslant S_{t-1}$) and tail classes (*i.e.*, $S_t \geqslant \ldots \geqslant S_{N-1} \geqslant S_N$), where S_t is the number of the t^{th} class. For these tail classes, we employ the hierarchical tree knowledge graph to obtain their corresponding coarse-grained classes. For these head classes, we generate the coarse-grained classes, and keep their granularity descriptions consistent with the fine-grained target classes. We use parameter α to control the tail hierarchical structure using different initial settings. Parameter α alters the descriptions for coarse-grained classes.

Continuing with Example 1, we use Example 2 to explain how to obtain the tail hierarchical structure.

Example 2. As shown in Fig. 4, we assume that the classes *Bottle* and *Bicycle* belong to the head class while other classes belong to the tail class. Thus, we translate their coarse-grained descriptions *Household* and *Vehicles* to *Bottle* and *Bicycle*.

The proposed tail hierarchical structure has the following advantages: (1) From the granularity perspective, distinguishing a target directly from a large number of classes is a complex and difficult classification task. A classification task is divided into a group of small sub-classification tasks inspired by the human multi-grained thinking mechanism. Each neuron on a CNN handles a small sub-classification task with a few classes. All these neurons unite to conduct the difficult task in a coarse-to-fine manner. Therefore, the task decomposition under the hierarchical structure can aid CNN training. (2) From the perspective of long-tailed distributions, we pay attention to tail data and aggregate these few classes into a group of abstract sets. Thus, coarse-grained classes tend to be approximately equally represented. This avoids both the data-hungry limitation of CNNs and the extreme class-imbalanced dilemma of long-tailed distributions.

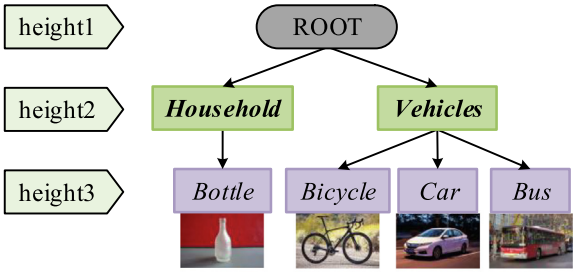

Fig. 3. Hierarchical tree knowledge graph of the VOC dataset.

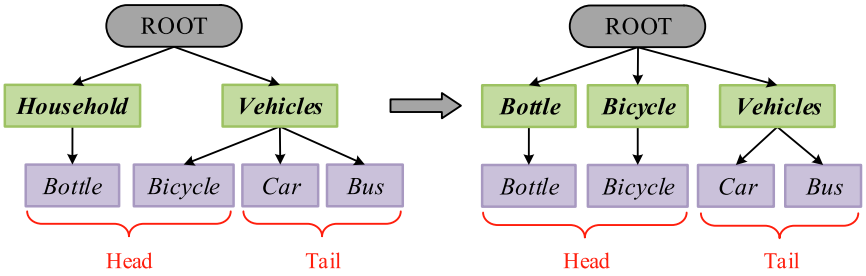

Fig. 4. Tail hierarchical structure establishment.

2.3. Multi-task CNN construction

Input data contains multi-grained labels based on the tail hierarchical structure **G**. Then, we adopt a multi-task CNN architecture to train the data.

We consider a classification task with N fine-grained target classes and M coarse-grained classes from a dataset $\mathscr{D} = \left\{\left(x_i, y_i^f, y_i^c\right)\right\}_{i=1}^{S}$, where x denotes an input, y^f denotes the fine-grained target label and y^c denotes the corresponding coarse-grained label. A multi-task CNN is designed to perform both coarse- and fine-grained learning tasks simultaneously. Subsequently, the input data is fed into shared representation learning layers and task-specific learning layers in the multi-task CNN.

Shared Representation Learning Layers. The input data for the representation learning comes from the fine-grained task, where each sample is assigned a single fine-grained label. The fine-grained task retains attributes of the original distribution and reflects the specific characteristics of each sample. Meanwhile, the tail hierarchical structure aims to mitigate imbalanced distributions by using relatively uniform distribution, whose input data comes from a coarse-grained task. For the coarse-grained task, each sample is attached to an abstract description and converges to the same coarse-grained classes. Therefore, the coarse-grained task alleviates the data-hungry limitation of representation learning and extracts abstract knowledge. Generally, representation learning layers contain convolutional layers and pooling layers for a CNN. We execute multiple tasks in parallel and employ the shared representation learning layers to train the data from different granularity tasks. The shared representation learning is formulated as

$$\mathbf{Z}_R = f(x, \mathbf{W}_R), \tag{1}$$

where $\mathbf{Z}_R$ denotes the output of multi-task CNN and $f(x, \mathbf{W}_R)$ is implemented by the multi-task CNN with parameter $\mathbf{W}_R$. The shared process takes advantage of related tasks in different granularities, which helps the multi-task CNN extract a more generalized representation.

Task-specific Learning Layers. A fully connected (FC) layer of the fine-grained task is concatenated to weight the output $\mathbf{Z}_R$ of the representation learning process. Output logits of the fine-grained FC layer are represented as

$$\mathbf{Z}_F = \mathbf{W}_F \mathbf{Z}_R + \mathbf{b}, \tag{2}$$

where $\mathbf{W}_F$ is the weight vector of FC layer under the fine-grained task, $\mathbf{b}$ is a bias and $\mathbf{Z}_F$ is a predicted output, *i.e.*, $[z_1, z_2, \ldots, z_N]^{\mathrm{T}}$. For each fine-grained class $i \in \{1, 2, \ldots, N\}$, a softmax function calculates the i^{th} class probability by

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}. \tag{3}$$

The output probability distribution of the fine-grained task is denoted as $\mathbf{p} = [p_1, p_2, \ldots, p_N]^{\mathrm{T}}$, where $p_i \in [0, 1] \forall i$. We perform two specific tasks (the coarse- and fine-grained tasks) in a multi-task CNN. Each task is attached to a specific FC layer to learn different discriminative knowledge, which is denoted as $\mathbf{W}_F^c$ and $\mathbf{W}_F^f$ for coarse- and fine-grained tasks, respectively. The predicted class probabilities are calculated as $\mathbf{p}_c$ and $\mathbf{p}_f$ for coarse- and fine-grained classes, respectively. Then, the mixture of coarse- and fine-grained knowledge is implemented on the subsequent loss layer of the multi-task CNN.

2.4. Coarse-to-fine knowledge transfer strategy

We propose a loss function to combine coarse- and fine-grained tasks. Furthermore, we also design a knowledge transfer strategy for shifting learning attention between the two tasks.

The loss function $\mathscr{L}$ of our MCKT approach combines coarse- and fine-grained tasks, which is written as

$$\mathscr{L} = \lambda \mathscr{L}_c + (1-\lambda)\mathscr{L}_f, \tag{4}$$

where λ is an adaptive weighted parameter, $\mathscr{L}_c$ denotes a loss for coarse-grained learning and $\mathscr{L}_f$ denotes a loss for fine-grained learning. We propose the knowledge transfer strategy for λ to shift training attention between the two losses $\mathscr{L}_c$ and $\mathscr{L}_f$. The two loss functions are designed to learn the coarse- and fine-grained tasks. Parameter λ is automatically generated and altered according to the number of training epochs. The weighted losses $\lambda\mathscr{L}_c$ and $(1-\lambda)\mathscr{L}_f$ are integrated into the mixture loss $\mathscr{L}$ by controlling the weights for $\mathscr{L}_c$ and $\mathscr{L}_f$. Loss $\mathscr{L}$ is our optimization goal and the whole multi-task CNN is end-to-end trainable.

In detail, the knowledge transfer strategy divides the multi-task CNN training process into three stages: (1) Coarse-grained pre-trained stage. The input data and coarse-grained labels are fed into the multi-task CNN to generate a generalized tail knowledge representation. (2) Multi-grained knowledge transfer stage. The coarse- and fine-grained tasks are simultaneously learned in a multi-task way, where the model not only makes the two tasks benefit each other but also gradually transfers the obtained tail knowledge to the fine-grained task. (3) Fine-grained target-domain stage. Starting from pre-trained network parameters rather than from random initialization, the fine-grained task is individually trained and aims to achieve a more accurate understanding of the target domain (*i.e.*, fine-grained classes).

Parameter λ is automatically obtained by

$$\lambda = \begin{cases} 1 & 0 < E \leqslant E_c, \\ 1 - \frac{E-E_c}{E_{max}-E_c-E_f} & E_c < E \leqslant E_{max}-E_f, \\ 0 & E_f < E \leqslant E_{max}, \end{cases} \tag{5}$$

where E_{max} denotes the number of total training epochs, E is the current epoch, E_c is the total epochs of coarse-grained stage and E_f is the total epochs of fine-grained stage. In particular, λ is set to be constant at 1 or 0 to focus on the coarse-grained pre-trained stage or the fine-grained target-domain stage. Otherwise, the multi-grained knowledge transfer stage is adaptively performed and gradually urges the coarse-grained stage to induce learning of the fine-grained stage, where λ gradually decreases with increases in the number of training epochs E.

Intuitively, we conjecture the effectiveness of the knowledge transfer results from two aspects: (1) We perform a coarse-grained pre-trained stage based on the idea that good coarse-grained representation is a foundation for fine-grained target classes. The generalized tail knowledge representation is utilized to prevent the CNN from training from random initialized starting points. We conduct a study to verify this claim in Section 4.2. (2) Our model has to find a representation that captures each task and transfers coarse-grained knowledge to the fine-grained task on the multi-grained stage, which reduces the risk of overfitting on the target task. In this respect, λ descends linearly with the transfer of knowledge, where the role of the coarse- and fine-grained tasks are symmetric. Although the coarse- and fine-grained learning deserve equal focus, the emphasis of our MCKT approach is to gradually turn from the coarse-grained task to the targets. The learning attention of MCKT changes from the coarse-grained pre-trained stage to the fine-grained target-domain stage as λ decreases, which greatly improves long-tailed classification accuracy. Unlike transfer strategies [38] [39], our parameter λ ensures that our model treats the two tasks without bias in the multi-grained stage, while the learning attention is individually paid to other stages to focus on corresponding tasks. The strategy avoids effects on one task when the model performs training for another task. We discuss a more detailed analysis in Section 4.3.

In the experiment, $\mathscr{L}$ is a simple usage to the cross-entropy loss that we consider as an experimental baseline for our MCKT in Section 4.4. The loss of MCKT is

$$\mathscr{L} = \lambda \mathscr{L}_{CE}\left(x, \mathbf{W}_R, \mathbf{W}_F^c\right) + (1-\lambda)\mathscr{L}_{CE}\left(x, \mathbf{W}_R, \mathbf{W}_F^f\right), \tag{6}$$

where $\mathscr{L}_{CE}(\cdot)$ denotes the standard cross-entropy loss function.

Algorithm 1: Long-tailed classification learning via MCKT

Input: A dataset $\mathscr{D} = \left\{\left(x_i, y_i^f, y_i^c\right)\right\}_{i=1}^{S}$ with N fine-grained target classes and M coarse-grained classes, where x denotes an input, y^f denotes the fine-grained target label, y^c denotes the corresponding coarse-grained label and S denotes the target sample size of the dataset. The number of total training epochs E_{max}, the total epochs of coarse-grained stage E_c and the total epochs of fine-grained stage E_f.

Output: A parameterized multi-task CNN $\mathbf{W} = \left[\mathbf{W}_R; \mathbf{W}_F^c; \mathbf{W}_F^f\right]$.

1: Initialize a multi-task CNN parameters $\mathbf{W} = \left[\mathbf{W}_R; \mathbf{W}_F^c; \mathbf{W}_F^f\right]$ randomly;
2: **for** $i = 1 : E_{max}$ **do**
3: Update learning rate l;
4: **if** $i \leqslant E_c$ **then**
5: $\mathscr{L}(\mathbf{W}) \leftarrow \frac{1}{S}\sum_{(x,y^c)\in\mathscr{D}} \mathscr{L}_{CE}\left(x, \mathbf{W}_R, \mathbf{W}_F^c\right)$;

(*continued on next page*)

6: **else if** $E_c < i \leqslant E_{max} - E_f$ **then**
7: Update λ by Eq. (5);
8: $\mathscr{L}(\mathbf{W}) \leftarrow \frac{1}{S}\sum_{(x,y^f,y^c)\in\mathscr{D}}\left(\lambda\mathscr{L}_{CE}(x,\mathbf{W}_R,\mathbf{W}_F^c)+(1-\lambda)\mathscr{L}_{CE}\left(x,\mathbf{W}_R,\mathbf{W}_F^f\right)\right)$;
9: **else**
10: $\mathscr{L}(\mathbf{W}) \leftarrow \frac{1}{S}\sum_{(x,y^f)\in\mathscr{D}}\mathscr{L}_{CE}\left(x,\mathbf{W}_R,\mathbf{W}_F^f\right)$;
11: **end if**
12: Update $\mathbf{W} \leftarrow$ A stochastic gradient descent optimizer $\mathbf{W} - l\nabla_{\mathbf{W}}\mathscr{L}(\mathbf{W})$;
13: **end for**

Algorithm 1 is the training process for multi-task via coarse-to-fine knowledge transfer. Line 1 means that a multi-task CNN is trained starting from random initialization. Line 3 suggests that the learning rate is altered with the increase of training epochs. Lines 4 to 11 imply that a loss is calculated via the coarse-to-fine knowledge transfer. Line 12 describes the loss of backpropagation under an optimizer.

Our algorithm adopts a common convolutional network architecture, and adjusts the training of the architecture by a specific loss function. This architecture is applied in most of the existing methods, such as Focal [25] and LDAM [26] losses. For example, Focal loss decreases the losses of head classes, which optimizes the training process of the architecture. Therefore, our model is algorithmically similar to these existing methods. Moreover, we design a shared learning structure that reduces the unnecessary network training time. In addition, we perform the same experimental settings and training epochs. In terms of time complexity, our model is at the same level as these existing methods.

3. Experimental settings

In this section, we describe datasets and implementation details used in our experiments. Then, we introduce some state-of-the-art methods and evaluation measures.

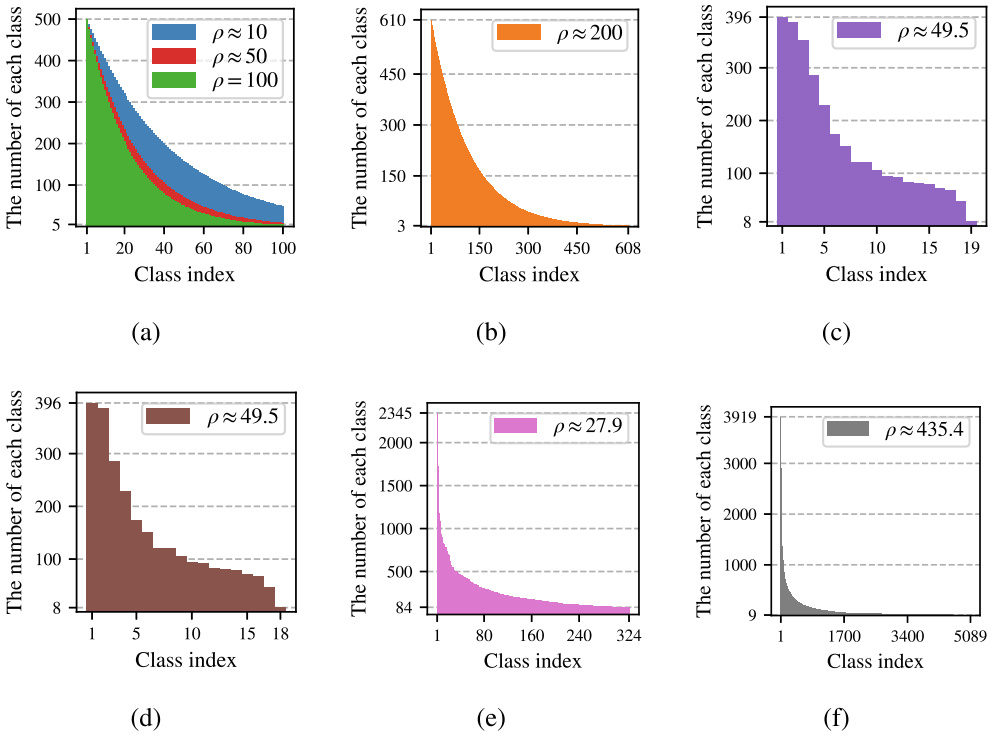

Fig. 5. Histograms on number of training samples per class on different datasets: (a) LT-CIFAR; (b) LT-tIN; (c) VOC-19; (d) VOC-18; (e) SUN-324; and (f) iNaturalist.

Table 1
Descriptions of the experimental datasets.

Dataset	Original height	Coarse-grained class	Fine-grained target class	Training size	Imbalance
CIFAR-IR10	3	20	100	19,572	Artificial
CIFAR-IR50	3	20	100	12,607	Artificial
CIFAR-IR100	3	20	100	10,847	Artificial
tIN-IR200	3	34	608	69,545	Artificial
VOC-19	5	3	19	2,937	Real-world
VOC-18	5	3	18	2,585	Real-world
SUN-324	4	15	324	85,147	Real-world
iNaturalist	3	13	5089	579,184	Real-world

3.1. Datasets

We evaluate our model on six long-tailed datasets. For all these experimental datasets, (1) samples are single-labeled for both fine- and coarse-grained classes, and (2) classes are built in a coarse-to-fine manner by a knowledge graph. Unless otherwise stated, the default class is the fine-grained class. Fig. 5 illustrates distributions of classes for all datasets considered in this paper. Table 1 summarizes descriptions of these datasets.

Long-tailed CIFAR (LT-CIFAR). The 100 classes version of CIFAR [35] consists of 60,000 color images of size 32×32 in 100 classes. Each class has 600 images that are divided into 500 training and 100 test images. We artificially create imbalanced training versions as the same as those used in [26,40]: (1) We reduce the training sample number of each class except the first class. (2) We denote the imbalance ratio as $\rho = S_1/S_N$, so that S_i between S_1 and S_N follows an exponential decay across i. We keep the test size unaltered, *i.e.*, this is still class-balanced. The imbalance ratios ρ used in theexperiments are 10, 50, and 100, *e.g.*, CIFAR-IR50. Moreover, the 100 fine-grained classes in CIFAR are grouped into 20 coarse-grained classes by WordNet.

Long-tailed tieredImageNet (LT-tIN). tieredImageNet [41] is a subset of ImageNet2012 [42]. This groups 608 classes into 34 coarse-grained classes by the ImageNet hierarchy [43]. There are at least 732 images of size 84×84 in each class. We randomly choose 610 images as training images and 122 images as test images for each class, and artificially create tIN-IR200.

VOC-19 and VOC-18. The 2012 version of VOC [44] contains 20 classes that are built by a five-level knowledge graph. We remove multi-label images since this kind of classification task is not discussed in this paper. Further, we obtain VOC-19 with 19 classes (excluding class *Person*) and VOC-18 with 18 classes (further excluding class *Bird*). The official split of training and test images are utilized.

SUN-324. The extensive Scene UNderstanding (SUN) database [45,46] contains 397 classes that are built by a four-level knowledge graph. We retain 324 classes by removing multi-label classes on the SUN dataset. The 324 fine-grained classes are grouped into 15 coarse-grained classes at the third level and three coarse-grained classes at the second level.

iNaturalist. The 2017 version of iNaturalist [47] contains 675,170 images with 5,089 classes that are built by a three-level knowledge graph. We use the official split containing 579,184 training and 95,986 test images.

3.2. Implementation details

For all datasets, we employ a data augmentation technique [48]: (1) We firstly convert an original image to a size of 36×36 by padding or resizing. (2) Then we randomly crop an image of size of 32×32 from it or its horizontal flip. We ensure that the input images are all size of 32×32 and are normalized.

For all experiments, we choose a residual network [8] with 32 layers (*i.e.*, ResNet-32) as the backbone network. We use a stochastic gradient descent optimizer (*i.e.*, SGD) with a momentum of 0.9, weight decay of 2×10^{-4} and batch size of 128. The hyper-parameter α is set to 0.8. The learning rate is set to 0.1 across the coarse-grained pre-trained stage with 55 epochs and the multi-grained knowledge transfer stage with 45 epochs. For the fine-grained target-domain stage with 200 epochs, the initial learning rate is set to 0.1 where a linear warm-up learning rate schedule [24] is adopted in the first 5 epochs. Then, the learning rate is decayed by 0.01 at the last 40 epochs and again at the last 20 epochs [38].

We train all models on a single NVIDIA GeForce RTX 2080Ti GPU. Our code is developed using the PyTorch framework and is available at http://github.com/fhqxa/lzy_MCKT.

3.3. Comparison methods

We consider a wide range of re-balancing methods that consist of losses and strategies.
There are three models based on **Losses:**

(1) cross-entropy loss (CE): A standard training and each training sample has an equal probability of being selected;

(2) focal loss (Focal) [25]: Training samples with Focal that automatically lowers the contribution of easy head samples and rapidly focuses the model on hard tail samples, instead of CE;
(3) label-distribution-aware margin loss (LDAM) [26]: Training samples with LDAM that regularizes tail classes more strongly than the head classes, instead of CE.

There are six models based on **Strategies:**

(1)re-sampling (RS) [49]: Each class has an equal probability of being selected, seen as the combination of under-sampling on head samples and over-sampling on tail samples in data space;
(2) re-weighting (RW) [22]: Each class is attached to weight by inverse class frequency in a loss function;
(3) class-balanced re-weighting (CB) [24]: A variant of RW depending on the inverse effective number in each class, defined as $(1-\beta^{S_N})/(1-\beta)$, instead of inverse class frequency. Here, we use β = 0.9999;
(4)deferred re-sampling (DRS), (5) deferred re-weighting (DRW) and (6) deferred class-balanced re-weighting (DCB) [26]: RS, RW, and CB are deferred until the later stage of training of a model, respectively.

When one loss and one strategy are combined, we concatenate them with a dash as a new technique [26,38,40]. For example, LDAM-DRW, a current state-of-the-art method, combines the LDAM loss and the DRW strategy.

3.4. Evaluation measures

We mainly use accuracy (ACC) to evaluate our model. Moreover, we also report hierarchical evaluation measures [50].

ACC. ACC is the most popular evaluation measure for classification tasks. Let a label with the highest probability value denote a predicted label. ACC-top1 is calculated as a ratio of the number of correctly predicted samples to the total number of test samples. ACC-top5 uses the top five labels in probability to pair with a true label. Unless otherwise stated, we default ACC to fine-grained ACC-top1 of all classes, *e.g.*, ACC of tail classes is regarded as fine-grained ACC-top1 of tail classes.

Hierarchical $\mathbf{F_1}$($\mathbf{F_H}$). F_1 used in hierarchical classification tasks is defined as

$$F_H = 2\frac{P_H R_H}{P_H + R_H}, \tag{7}$$

where P_H denotes hierarchical precision and R_H denotes hierarchical recall. They are written as

$$\begin{aligned} P_H &= \frac{|\widehat{D}_{aug} \bigcap D_{aug}|}{|\widehat{D}_{aug}|}, \\ R_H &= \frac{|\widehat{D}_{aug} \bigcap D_{aug}|}{|D_{aug}|}, \end{aligned} \tag{8}$$

where $D_{aug} = D \bigcup An(D), \widehat{D}_{aug} = \widehat{D} \bigcup An\left(\widehat{D}\right), D$ denotes a true label of test sample, $An(D)$ denotes a parent node set of the true class to which sample really belongs, $\widehat{D}$ indicates a predicted label of test sample and $|\cdot|$denotes a count of elements.

Tree-induced error (TIE). Penalties of false labels are not always equal in a hierarchical structure. Let edges in the hierarchical tree denote degree of false labels. TIE reflects the cost of errors and is formulated as

$$TIE(d_{pr}, d_{tr}) = |E_H(d_{pr}, d_{tr})|, \tag{9}$$

where $E_H(d_{pr}, d_{tr})$ denotes a set of edges along a path from a predicted label d_{pr} to true label d_{tr} in the hierarchical tree and $|\cdot|$denotes a count of elements.

4. Experimental results and analysis

In this section, we conduct ablation studies to present a detailed discussion of the proposed model. For a fair comparison, we mainly utilize CIFAR-IR50 as an example in all these experiments. Finally, we demonstrate the effectiveness of our model by comparing it with state-of-the-art methods on different datasets.

4.1. Different tail hierarchical structures

In Section 2.2, we construct a tail hierarchical structure to alleviate the class-imbalanced problem. We explore several α value to generate the different structures on CIFAR-IR50.

We present the fine-grained performance of all classes based on different evaluation measures for models trained by different values of α in Fig. 6. Hyper-parameter α comes from a fixed set of candidates, *i.e.*, $\alpha \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. Specifically, all samples are regarded as head data when α is set to 0.0, which corresponds to a flat CNN.

We have the following observations in terms of ACC:

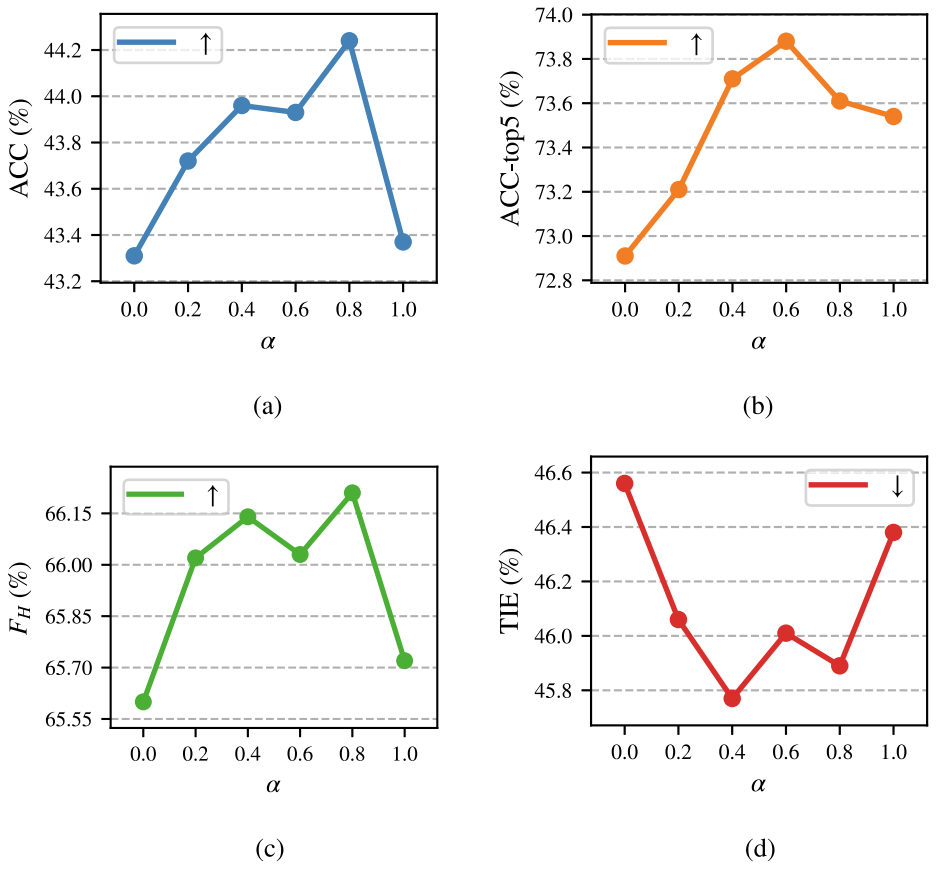

Fig. 6. Four evaluation measures for all classes with different α values on CIFAR-IR50: (a) ACC (↑); (b) ACC-top5 (↑); (c) F_H (↑); and (d) TIE (↓). ("↑" denotes the greater the better and "↓" denotes the smaller the better.).

(1) The ACC is lower with $\alpha = 0.0$ than those with other candidates. Performance is better when using hierarchical structures than the flat structure. The consistent gain indicates that the tail hierarchical structures contribute to the performance of the classification task.
(2) The ACC generally increases with α without regard to $\alpha = 1.0$. The trend suggests that the tail hierarchical structure makes use of the tail class coarse-grained descriptions to prevent the under-fitting of the CNN. This means that the generalized knowledge puts the starting point in a pre-trained parameter space of the CNN that is better than random.
(3) The best α is 0.8 and the ACC decreases significantly when α is raised to 1.0. This implies that the benefits of constructing the tail hierarchical structure on all classes are inferior to those of paying attention to tail classes. This is a desirable property in the context of long-tailed learning.
(4) More importantly, we obtain the observations that are similar to ACC from the perspective of other evaluation measures including ACC-top5, F_H, and TIE. Consistent trends in the four evaluation measures reveal that the tail hierarchical structure promotes the training of a robust CNN.

We also report the ACC of tail classes to further demonstrate the influence of focusing on tail classes in Fig. 7. Tail hierarchical structures with $\alpha = 0.6$ and $\alpha = 0.8$ are better than other candidates, with $\alpha = 0.8$ being the best. This indicates that our structure facilitates tail class learning. More importantly, the model performs worse when focusing on all classes, *i.e.*, $\alpha = 1.0$, than on tail classes. This is consistent with the poor generalization with tail classes due to their infrequent samples and the leading role of head classes. Thus, we pay more attention to the tail than head classes to reduce the dominance of the head over tail classes.

4.2. Effectiveness of coarse-grained task

We impose the knowledge representation extracted on coarse-grained tasks to help generalization on tail classes. We consider an experiment where E_c is set to various values to verify this.

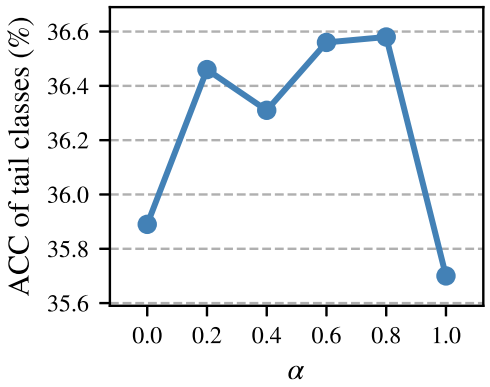

Fig. 7. The ACCs of tail classes with different α values on CIFAR-IR50.

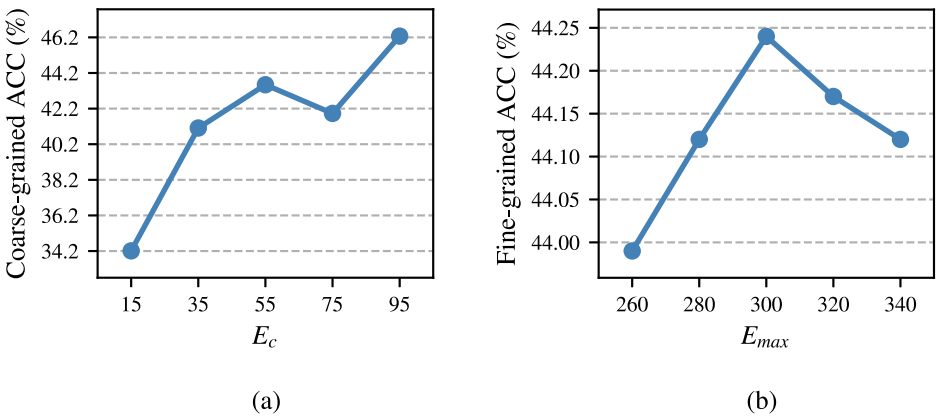

Fig. 8. Influence of the coarse-grained task on fine-grained learning on CIFAR-IR50.

In Fig. 8, the first E_c epochs are conducted in the coarse-grained pre-trained stage and the whole training process ends at the E_{max}^{th} epoch for our multi-task CNN. We fix the total epochs of multi-grained knowledge transfer and the fine-grained target-domain stage. E_c is set to $\{15, 35, 55, 75, 95\}$ and E_{max} is accordingly set to $\{260, 280, 300, 320, 340\}$, which correspond to coarse-grained ACC of the E_c^{th} epoch and fine-grained ACC of the E_{max}^{th} epoch, respectively.

We obtain the following conclusions in Fig. 8:

(1) The coarse-grained ACC constantly improves with increases in E_c when $E_c = \{15, 35, 55\}$. Similarly, the fine-grained ACC is also enhanced with increases in coarse-grained ACC. It is concluded that coarse-grained learning facilitates training and optimization of the model for the fine-grained task. This suggests that the auxiliary information from the related task is beneficial in distinguishing target classes, further proving a fact that hierarchical structures are widely appropriate for coarse-to-fine classifications.

(2) The fine-grained ACC shows a downward trend when E_c is further boosted, (*i.e.*, E_c=75 and 95). This means that we excessively focus on the coarse-grained task over the fine-grained task when fixing the total epochs of the fine-grained and multi-task stages. Although coarse-grained learning extracts generalized knowledge, it is inadvisable to neglect the learning of the fine-grained task. Thus, E_c greatly affects the fine-grained classification ability and a trade-off allocation between coarse- and fine-grained tasks is important for our model.

(3) Coarse-grained classes that are trained for smaller epochs easily achieve a better performance than fine-grained classes. For example, the coarse-grained ACC trained for 95 epochs is better than the fine-grained ACC trained for 200 epochs. This indicates that the CNN indeed obtains better performance on coarse-grained classes. This observation proves our expectation that coarse-grained classes are easier to be distinguished than fine-grained classes. In addition, the available and high-quality generalized knowledge lays a foundation for transfer in a coarse-to-fine fashion.

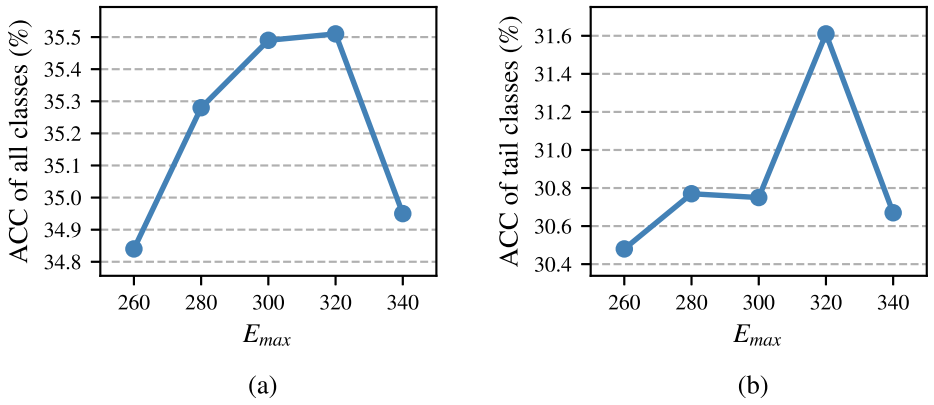

Fig. 9. The ACCs of both all classes and tail classes on SUN-324.

Moreover, we follow the same experiment to examine how the coarse-grained knowledge affects the fine-grained classification using a real-world long-tailed distribution. The experiment is performed on the SUN-324 dataset. In contrast to when using CIFAR-IR50, we report that the ACCs using both all classes and tail classes with different values of E_{max} in Fig. 9.

As expected, ACC of all classes improve with increasing values of $E_{max} = \{260, 280, 300, 320\}$. In terms of fine-grained tail classes, the ACCs of tail classes generally improves with increases in E_{max} until an oversize epoch is reached. This indicates that the generalized tail knowledge extracted by the coarse-grained task helps the classification of target tail classes. Thus, the coarse-grained pre-trained stage is effective in influencing the updating of CNN parameters using long-tailed data. That is the reason why starting the training from a pre-trained stage rather than from random weights yields better CNN performance.

Notably, the best E_c is 55 on CIFAR-IR50. However, the performance when $E_c = 75$ is best on SUN-324. This means that there are different optimal parameters for different datasets. Without loss of generality, we choose $E_c = 55, E_f = 200$ and $E_{max} = 300$ to train our model for all datasets in this paper.

4.3. Different knowledge transfer strategies

In Section 2.4, we develop a knowledge transfer strategy to adaptively transfer beneficial related knowledge to the target task. We employ CIFAR-IR50 as an example to show the ACCs for our model when trained with different strategies.

In Fig. 10, we visualize the coarse-grained loss weights λ and depict curves for different transfer strategies: (1) Single-stage strategies (*e.g.*, cosine decay, linear decay). These strategies progressively adjust λ with the number of training epochs. (2) Multi-stage strategies (*e.g.*, vertical drop, equal weight, parabolic decay, our linear decay). These strategies divide the training into multiple stages, where λ follows the corresponding rules for different stages.

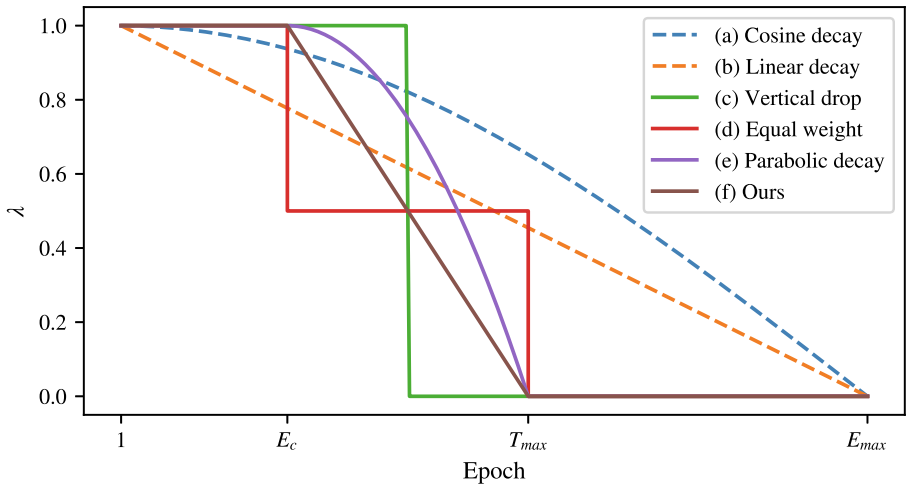

Fig. 10. Visualization of λ based on the different transfer strategies. Single-stage strategies: (a) cosine decay and (b) linear decay and multi-stage strategies: (c) vertical drop; (d) equal weight; (e) parabolic decay; and (f) ours.

Table 2
The ACCs of the different transfer strategies on CIFAR-IR50 and SUN-324 (%).

Type	Transfer strategy	λ	CIFAR-IR50	SUN-324
Single-stage	Cosine decay	$\cos\left(\frac{E}{E_{max}}\cdot\frac{\pi}{2}\right)$	43.38	34.41
	Linear decay	$1-\frac{E}{E_{max}}$	43.59	34.34
Multi-stage	Vertical drop	1/0	44.04	34.68
	Equal weight	0.5	44.05	35.36
	Parabolic decay	$1-\left(\frac{T}{T_{max}}\right)^2$	44.11	35.07
	Linear decay (ours)	$1-\frac{T}{T_{max}}$	**44.24**	**35.49**

Table 2 lists the ACCs of the training model with different transfer curves. The best results are marked in bold. We denote $T = E - E_c$ as the current epoch and $T_{max} = E_{max} - E_c - E_f$ as the ending epoch for on the multi-grained knowledge transfer stage. We observe that:

(1) The multi-stage strategies for generating λ gain better performance than the single-stage strategies. This indicates that the coarse-grained task should be learned intently first, and then the fine-grained task. This proves our assumption that the generalized related knowledge contributes to the target task learning.
(2) The vertical drop strategy yields worse performance than other multi-stage strategies. This suggests that the knowledge should be learned in a multi-task way in the multi-grained stage, where coarse- and fine-grained tasks benefit each other.
(3) Among these strategies, the best way to generate λ is the proposed multi-stage linear decay strategy.
(4) Similar observations are obtained on SUN-324. This means that our strategy is equally useful for the real-world long-tailed scenes.

4.4. Performance comparison of different methods

We conduct extensive experiments on six datasets: (a) The artificial datasets LT-CIFAR and LT-tIN and (b) the real-world datasets VOC-19, VOC-18, SUN-324, and iNaturalist. We mainly report ACC and F_H values for these datasets.
Table 3 provides a performance comparison of the different methods on LT-CIFAR and LT-tIN under different imbalance ratios. The best results are marked in bold. From this table, we can obtain the following observations:

(1) ACC and F_H gradually decrease as the imbalance ratio ρ increases. The number of tail classes is sharply reduced with increases in ρ, while the number of head classes is generally unaltered. This means that it is difficult for the CNN to generalize on tail classes under an extreme class-imbalanced distribution.
(2) Combinations of losses and strategies often provide better performance than individual approaches.
(3) Our MCKT consistently achieves better performance compared with the plain training (without any strategies) with CE. This indicates that our model can improve the classification performance in a long-tailed challenge.
(4) Focal and LDAM are designed to leverage fine-grained classes to adjust imbalanced distributions between head and tail classes. Our MCKT obtains comparable or even better performance than Focal and LDAM. This proves that coarse-grained knowledge can promote fine-grained learning, where the original distribution is retained.
(5) Our MCKT combined with DRW consistently achieves the best performance across all the datasets when compared with the other comparison methods. In particular, the ACC of MCKT-DRW is about 1.31% higher than that of LDAM-DRW on CIFAR-IR50 and 3.25% higher than that of Focal on LT-tIN.

Table 3
Performance comparison among the different methods on LT-CIFAR and LT-tIN datasets under different imbalance ratios (%).

Dataset	LT-CIFAR						LT-tIN	
Imbalance ratio	$\rho \approx 10$		$\rho \approx 50$		$\rho = 100$		$\rho \approx 200$	
Evaluation measure	ACC	F_H	ACC	F_H	ACC	F_H	ACC	F_H
CE	56.52	73.43	43.31	65.60	38.34	62.79	15.11	46.24
CE-RS [49]	55.93	73.15	40.71	64.40	35.36	61.10	12.38	43.39
CE-RW [22]	55.92	73.22	42.01	64.67	32.41	59.18	10.29	41.82
CE-CB [24]	56.26	73.37	40.12	63.91	33.15	59.79	10.29	41.82
Focal [25]	55.68	73.07	45.06	66.75	39.62	63.46	15.33	46.40
Focal-DCB [24]	57.46	74.10	45.87	66.90	38.54	62.93	12.52	43.37
LDAM [26]	57.03	73.75	45.21	66.84	39.98	63.79	14.80	46.08
LDAM-DRW [26]	58.10	74.24	45.42	66.63	41.38	64.33	11.33	43.97
MCKT (ours)	57.15	73.81	44.24	66.17	39.15	63.35	15.27	46.39
MCKT-DRW (ours)	**58.53**	**74.50**	**46.73**	**67.34**	**41.58**	**64.40**	**18.58**	**47.33**

Table 4
Performance comparison among the different methods on real-world long-tailed datasets VOC-19, VOC-18, SUN-324, and iNaturalist (%).

Dataset	VOC-19		VOC-18		SUN-324		iNaturalist	
Imbalance ratio	$\rho \approx 49.5$		$\rho \approx 49.5$		$\rho \approx 27.9$		$\rho \approx 435.4$	
Evaluation measure	ACC	F_H	ACC	F_H	ACC	F_H	ACC	F_H
CE	40.95	68.93	42.64	71.10	35.03	60.08	7.08	46.24
CE-RS [49]	35.46	66.41	38.80	69.34	37.48	60.95	8.04	46.52
CE-RW [22]	37.24	66.93	38.04	68.66	37.83	61.16	6.37	45.49
CE-CB [24]	37.55	67.58	37.58	68.18	38.23	61.50	6.27	45.42
Focal [25]	39.47	68.14	42.45	71.22	35.24	60.48	7.26	46.29
Focal-DCB [24]	39.84	68.43	42.91	71.12	37.91	61.21	6.45	45.47
LDAM [26]	**41.25**	**69.25**	**45.07**	**72.27**	31.46	57.97	7.96	46.72
LDAM-DRW [26]	40.58	69.02	44.43	71.93	35.46	59.70	7.26	46.01
MCKT (ours)	38.89	68.30	42.87	71.18	35.49	60.16	7.11	46.28
MCKT-DRW (ours)	38.46	68.02	42.49	70.95	**39.81**	**62.32**	**8.98**	**46.91**

Table 4 provides a performance comparison among the different methods on VOC-19, VOC-18, SUN-324, and iNaturalist datasets. The best results are marked in bold. We obtain the following conclusions:

(1) On iNaturalist, MCKT-DRW surpasses all comparison methods. This demonstrates the effectiveness of our model under real-world imbalance and with vast labels. In particular, MCKT-DRW improves ACC by 1.90% over the plain training with CE.
(2) On SUN-324, we see results consistent with those on artificial datasets. In addition, our MCKT approach surpasses the plain training with CE, Focal, and LDAM approaches. Moreover, MCKT-DRW consistently obtains better performance than all comparison methods by a large margin (at least 1.29%). In particular, the ACC of MCKT-DRW is 1.29% and 4.78% better than those of CE-DRS and CE, respectively.
(3) Notably, MCKT achieves poor results on the VOC-19 and VOC-18 datasets, which were obtained from the VOC dataset with 20 classes. Our model performs worse than other methods including the plain training with CE. We conjecture that the knowledge graph of VOC is insufficient to reflect coarse-to-fine relations due to its few classes (*i.e.*, 20 classes). Then, the coarse-grained task may learn a poor knowledge representation that is transferred to fine-grained target classes. This leads to a negative knowledge transfer and damages the target representation learning, which is the reason why our MCKT method lowers the classification performance. This observation proves our expectation from another perspective—that a satisfactory coarse-to-fine structure is a foundation for improving long-tailed classification ability.

4.5. Statistical test of different methods

We adopt statistical tests to distinguish the different methods. The approach is used to evaluate various methods on multiple datasets. Given k comparison methods and n experimental datasets, r_i^j is s the rank of the j^{th} method on the i^{th} dataset. We compute the average rank of the j^{th} method on all datasets as

$$R_j = \frac{1}{n}\sum_{i=1}^{n} r_i^j. \tag{10}$$

The results of the average ranks of ACC among the different methods are listed in Table 5.

We take a null hypothesis that the performances of all methods are equivalent. Friedman test is defined as

$$\tau_F = \frac{(n-1)\chi_F^2}{n(k-1)-\chi_F^2}, \tag{11}$$

where τ_F obeys F-distribution with $k-1$ and $(k-1)(n-1)$ degrees of freedom, and

Table 5
Average ranks of the ACCs of the different methods on various datasets.

	CE-RS	CE-RW	CE-CB	Focal-DCB	LDAM-DRW	MCKT-DRW
CIFAR-IR10	55.93(5)	55.92(6)	56.26(4)	57.46(3)	58.10(2)	**58.53(1)**
CIFAR-IR50	40.71(5)	42.01(4)	40.12(6)	45.87(2)	45.42(3)	**46.73(1)**
CIFAR-IR100	35.36(4)	32.41(6)	33.15(5)	38.54(3)	41.38(2)	**41.58(1)**
tIN-IR200	12.38(3)	10.29(5)	10.29(6)	12.52(2)	11.33(4)	**18.58(1)**
SUN-324	37.48(5)	37.83(4)	38.23(2)	37.91(3)	35.46(6)	**39.81(1)**
iNaturalist	8.04(2)	6.37(5)	6.27(6)	6.45(4)	7.26(3)	**8.98(1)**
Avg. rank	4	5	4.83	2.83	3.33	1

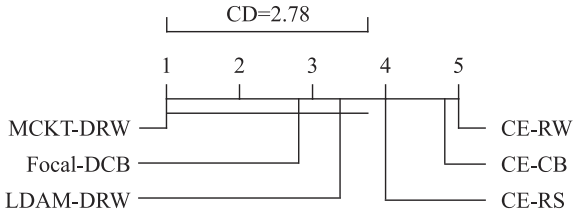

Fig. 11. The statistical tests for the different methods on various datasets.

$$\chi_F^2 = \frac{12n}{k(k+1)}\left(\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right). \tag{12}$$

We obtain $\chi_F^2 = 18.73$ and $\tau_F = 8.31$. Then, $F((k-1),(k-1)(n-1)) = F(5,25)$ for 0.05 α is 2.603 (<8.31), so we reject the null-hypothesis. Thus, these comparison methods have differences in performance.

Further, we use Bonferroni-Dunn post hoc tests to distinguish between these methods. Let critical distance

$$CD_\alpha = q_\alpha\sqrt{\frac{k(k+1)}{6n}} \tag{13}$$

denote the difference between the two methods. Two methods are significantly distinct when the difference between the two average ranks is greater than CD_α. We have $\alpha = 0.05$ and $q_\alpha = 2.576$, so $CD_\alpha = 2.78$. Fig. 11 shows the statistical tests of different methods based on ACC. The performance of MCKT-DRW is statistically different from CE-RW, CE-RS, and CE-CB. In addition, there is no obvious evidence that the performance of MCKT-DRW is significantly superior to Focal-DCB and LDAM-DRW.

5. Conclusions and future work

In this paper, we propose a multi-task CNN to transfer useful coarse-grained knowledge to a fine-grained task (MCKT) for long-tailed classification. We employ a hierarchical tree knowledge graph to organize coarse- and fine-grained classes based on tail data. Then, we utilize a multi-task CNN to learn coarse- and fine-grained tasks jointly and extract generalized representation. Further, we design a coarse-to-fine knowledge transfer strategy to adaptively regulate the multi-task CNN to distinguish between target classes. Extensive experiments show that our MCKT method achieves satisfactory performance on long-tailed data. This suggests that coarse-grained representation can alleviate the under-fitting of fine-grained classes due to insufficient data.

In MCKT, we only use the granularity information to facilitate tail class learning. In the future, we will consider how to transfer head class knowledge to the tail class to make full use of head data and enhances the knowledge capacity. In addition, we will also design a technique to combine the granularity and head information, which will improve the extensibility of the CNN. Moreover, we will explore the impact of network architecture on the classification results and optimize the parameters of the blocks.

CRediT authorship contribution statement

Zhengyu Li: Methodology, Data curation, Writing - original draft. **Hong Zhao:** Methodology, Software, Validation, Writing - review & editing. **Yaojin Lin:** Conceptualization, Data curation, Writing - review & editing.

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.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant Nos. 62141602 and 62076116, and the Natural Science Foundation of Fujian Province under Grant Nos. 2021J011003 and 2021J02049.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, athttps://doi.org/10.1016/j.ins.2022.07.015.

References

[1] J. Tan, C. Wang, B. Li, Q. Li, W. Ouyang, C. Yin, J. Yan, Equalization loss for long-tailed object recognition, in: in: Conference on Computer Vision and Pattern Recognition, 2020, pp. 11662–11671.
[2] S. Varela Santos, P. Melin, A new approach for classifying coronavirus COVID-19 based on its manifestation on chest X-rays using texture features and neural networks, Inf. Sci. 545 (2021) 403–414.
[3] P. Wang, K. Han, X.S. Wei, L. Zhang, L. Wang, Contrastive learning based hybrid networks for long-tailed image classification, in: Conference on Computer Vision and Pattern Recognition, 943–952, 2021a.
[4] Y. Su, K. Zhou, X. Zhang, R. Cheng, C. Zheng, A parallel multi-objective evolutionary algorithm for community detection in large-scale complex networks, Inf. Sci. 576 (2021) 374–392.
[5] Y. Poma, P. Melin, C.I. Gonzlez, G.E. Martnez, Optimization of convolutional neural networks using the fuzzy gravitational search algorithm, Journal of Automation, Mobile Robotics and Intelligent Systems (2020) 109–120.
[6] Y. Le Cun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, L. Jackel, Handwritten digit recognition with a back-propagation network, in: International Conference on Neural Information Processing Systems, 1989, pp. 396–404.
[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Conference on Computer Vision and Pattern Recognition, 1–9, 2015.
[8] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Conference on Computer Vision and Pattern Recognition, 770–778, 2016.
[9] J. Sanz, M. Sesma Sara, H. Bustince, A fuzzy association rule-based classifier for imbalanced classification problems, Inf. Sci. 577 (2021) 265–279.
[10] S. Li, K. Gong, C.H. Liu, Y. Wang, F. Qiao, X. Cheng, MetaSAug: meta semantic augmentation for long-tailed visual recognition, in: Conference on Computer Vision and Pattern Recognition, 5212–5221, 2021a.
[11] G. Andresini, A. Appice, D. Malerba, Autoencoder-based deep metric learning for network intrusion detection, Inf. Sci. 569 (2021) 706–727.
[12] T. Li, G. Kou, Y. Peng, S.Y. Philip, A fast diagonal distance metric learning approach for large-scale datasets, Inf. Sci. 571 (2021) 225–245.
[13] N. Zhang, S. Ying, W. Ding, K. Zhu, D. Zhu, WGNCS: a robust hybrid cross-version defect model via multi-objective optimization and deep enhanced feature representation, Inf. Sci. 570 (2021) 545–576.
[14] G. Dupret, M. Koda, Bootstrap re-sampling for unbalanced data in supervised learning, Eur. J. Oper. Res. 134 (1) (2001) 141–156.
[15] L. Abdi, S. Hashemi, To combat multi-class imbalanced problems by means of over-sampling techniques, Trans. Knowl. Data Eng. 28 (1) (2015) 238–251.
[16] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, Artific. Intell. Res. 16 (2002) 321–357.
[17] X. Wang, Y. Yang, M. Chen, Q. Wang, Q. Qin, H. Jiang, H. Wang, AGNES-SMOTE: an oversampling algorithm based on hierarchical clustering and improved SMOTE, Sci. Programm. 2020 (2020) 8837357.
[18] H. Yi, Q. Jiang, X. Yan, B. Wang, Imbalanced classification based on minority clustering synthetic minority oversampling technique with wind turbine fault detection application, Trans. Industr. Inform. 17 (9) (2021) 5867–5875.
[19] M.A. Tahir, J. Kittler, F. Yan, Inverse random under sampling for class imbalance problem and its application to multi-label classification, Pattern Recogn. 45 (10) (2012) 3738–3750.
[20] Q. Kang, X. Chen, S. Li, M. Zhou, A noise-filtered under-sampling scheme for imbalanced classification, Trans. Cybern. 47 (12) (2017) 4263–4274.
[21] W.C. Lin, C.F. Tsai, Y.H. Hu, J.S. Jhang, Clustering-based undersampling in class-imbalanced data, Inf. Sci. 409 (2017) 17–26.
[22] C. Huang, Y. Li, C.C. Loy, X. Tang, Learning deep representation for imbalanced classification, in: Conference on Computer Vision and Pattern Recognition, 5375–5384, 2016.
[23] Y.X. Wang, D. Ramanan, M. Hebert, Learning to model the tail, in: International Conference on Neural Information Processing Systems, 7032–7042, 2017.
[24] Y. Cui, M. Jia, T.Y. Lin, Y. Song, S. Belongie, Class-balanced loss based on effective number of samples, in: Conference on Computer Vision and Pattern Recognition, 9268–9277, 2019.
[25] T.Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollr, Focal loss for dense object detection, in: International Conference on Computer Vision, 2980–2988, 2017b.
[26] K. Cao, C. Wei, A. Gaidon, N. Arechiga, T. Ma, Learning imbalanced datasets with label-distribution-aware margin loss, International Conference on Neural Information Processing Systems 1567–1578 (2019).
[27] Z. Kuang, J. Yu, Z. Li, B. Zhang, J. Fan, Integrating multi-level deep learning and concept ontology for large-scale visual recognition, Pattern Recogn. 78 (2018) 198–214.
[28] D. Erhan, P.A. Manzagol, Y. Bengio, S. Bengio, P. Vincent, The difficulty of training deep architectures and the effect of unsupervised pre-training, in: Artificial Intelligence and Statistics, 153–160, 2009.
[29] H. Zhao, P. Wang, Q. Hu, P. Zhu, Fuzzy rough set based feature selection for large-scale hierarchical classification, Trans. Fuzzy Syst. 27 (10) (2019) 1891–1903.
[30] Y. Wang, R. Liu, D. Lin, D. Chen, P. Li, Q. Hu, C.L.P. Chen, Coarse-to-fine: progressive knowledge transfer based multi-task convolutional neural network for intelligent large-scale fault diagnosis, Trans. Neural Networks Learn. Syst. (2021) 1–14.
[31] G. Wan, S. Pan, C. Gong, C. Zhou, G. Haffari, Reasoning like human: hierarchical reinforcement learning for knowledge graph reasoning, in: International Joint Conference on Artificial Intelligence, 1926–1932, 2020.
[32] F. Zhang, X. Wang, Z. Li, J. Li, TransRHS: a representation learning method for knowledge graphs with relation hierarchical structure, in: International Joint Conference on Artificial Intelligence, 2987–2993, 2020.
[33] P. Oram, WordNet: an electronic lexical database, Appl. Psycholinguistics 22 (1) (2001) 131–134.
[34] T. Pedersen, S. Patwardhan, J. Michelizzi, et al., WordNet: similarity-measuring the relatedness of concepts, in: International Conference on Artificial Intelligence, 25–29, 2004.
[35] A. Krizhevsky, V. Nair, G. Hinton, Learning multiple layers of features from tiny images, University of Tront (2009) 1–58.
[36] H. Zhao, Q. Hu, P. Zhu, Y. Wang, P. Wang, A recursive regularization based feature selection framework for hierarchical classification, Trans. Knowl. Data Eng. 33 (7) (2021) 2833–2846.
[37] Y. Wang, Q. Hu, P. Zhu, L. Li, B. Lu, J.M. Garibaldi, X. Li, Deep fuzzy tree for large-scale hierarchical visual classification, Trans. Fuzzy Syst. 28 (7) (2019) 1395–1406.
[38] B. Zhou, Q. Cui, X.S. Wei, Z.M. Chen, BBN: bilateral-branch network with cumulative learning for long-tailed visual recognition, in: Conference on Computer Vision and Pattern Recognition, 9719–9728, 2020.
[39] Y. Zhang, Q. Yang, A survey on multi-task learning, Trans. Knowl. Data Eng. (2021) 1–20.
[40] J. Kim, J. Jeong, J. Shin, M2m: imbalanced classification via major-to-minor translation, in: Conference on Computer Vision and Pattern Recognition, 13896–13905, 2020.
[41] 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, 1–15, 2018.
[42] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al, ImageNet large scale visual recognition challenge, Int. J. Comput. Vision 115 (3) (2015) 211–252.
[43] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, L. Fei Fei, ImageNet: a large-scale hierarchical image database, in: Conference on Computer Vision and Pattern Recognition, 248–255, 2009.
[44] M. Everingham, L. Van Gool, C.K. Williams, J. Winn, A. Zisserman, The Pascal Visual Object Classes (VOC) challenge, Int. J. Comput. Vision 88 (2) (2010) 303–338.

[45] J. Xiao, K.A. Ehinger, J. Hays, A. Torralba, A. Oliva, SUN database: exploring a large collection of scene categories, Int. J. Comput. Vision 119 (1) (2016) 3–22.
[46] J. Xiao, J. Hays, K.A. Ehinger, A. Oliva, A. Torralba, SUN database: large-scale scene recognition from abbey to zoo, in: Conference on Computer Vision and Pattern Recognition, 3485–3492, 2010.
[47] Y. Cui, Y. Song, C. Sun, A. Howard, S. Belongie, Large scale fine-grained categorization and domain-specific transfer learning, in: Conference on Computer Vision and Pattern Recognition, 4109–4118, 2018.
[48] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Adv. Neural Inform. Process. Syst. 25 (2012) 1097–1105.
[49] N. Japkowicz, The class imbalance problem: significance and strategies, in: International Conference on Artificial Intelligence, 111–117, 2000.
[50] A. Kosmopoulos, I. Partalas, E. Gaussier, G. Paliouras, I. Androutsopoulos, Evaluation measures for hierarchical classification: a unified view and novel approaches, Data Min. Knowl. Disc. 29 (3) (2015) 820–865.