Networks and Heterogeneous Media

NHM, 18(4): 1600–1629.
DOI: 10.3934/nhm.2023070
Received: 18 June 2023
Revised: 13 August 2023
Accepted: 22 August 2023
Published: 5 September 2023

http://www.aimspress.com/journal/nhm

Research article

Simple is best: A single-CNN method for classifying remote sensing images

Huaxiang Song* and Yong Zhou

School of Geography Science and Tourism, Hunan University of Arts and Science, Changde, China

* **Correspondence:** Email: cn11028719@huas.edu.cn.

Abstract: Recently, researchers have proposed a lot of methods to boost the performance of convolutional neural networks (CNNs) for classifying remote sensing images (RSI). However, the methods' performance improvements were insignificant, while time and hardware costs increased dramatically due to re-modeling. To tackle this problem, this study sought a simple, lightweight, yet more accurate solution for RSI semantic classification (RSI-SC). At first, we proposed a set of mathematical derivations to analyze and identify the best way among different technical roadmaps. Afterward, we selected a simple route that can significantly boost a single CNN's performance while maintaining simplicity and reducing costs in time and hardware. The proposed method, called RE-EfficientNet, only consists of a lightweight EfficientNet-B3 and a concise training algorithm named RE-CNN. The novelty of RE-EfficientNet and RE-CNN includes the following: First, EfficientNet-B3 employs transfer learning from ImageNet-1K and excludes any complicated re-modeling. It can adequately utilize the easily accessible pre-trained weights for time savings and avoid the pre-training effect being weakened due to re-modeling. Second, RE-CNN includes an effective combination of data augmentation (DA) transformations and two modified training tricks (TTs). It can alleviate the data distribution shift from DA-processed training sets and make the TTs more effective through modification according to the inherent nature of RSI. Extensive experimental results on two RSI sets prove that RE-EfficientNet can surpass all 30 cutting-edge methods published before 2023. It gives a remarkable improvement of 0.50% to 0.75% in overall accuracy (OA) and a 75% or more reduction in parameters. The ablation experiment also reveals that RE-CNN can improve CNN OA by 0.55% to 1.10%. All the results indicate that RE-EfficientNet is a simple, lightweight and more accurate solution for RSI-SC. In addition, we argue that the ideas proposed in this work about how to choose an appropriate model and training algorithm can help us find more efficient approaches in the future.

Keywords: RE-EfficientNet; RE-CNN; remote sensing image classification; deep learning

Abbreviations: CAM: class activation mapping; CNN: convolutional neural network; DA: data augmentation; DARS: DA and regularization strategy, DL: deep learning; LR: learning rate; m-CutMix: modified CutMix; m-OLS: modified Online Label Smoothing; M: million; ML: machine learning, NPSC: number of parameter-setting combinations; OA: overall accuracy; OLS: online label smoothing; RE: regularization-enhanced; RSI: remote sensing images; SC: semantic classification; SE: squeeze-and-excitation; SOTA: state-of-the-art; TR: training ratio; TT: training trick; ViT: vision transformer.

1. Introduction

Remote sensing is a crucial technique for Earth observation. With the expansion of onboard, ground-based [1] and underwater sensors [2], monitoring data has become too large to be interpreted all by hand. Consequently, machine learning (ML) algorithms have played a prominent role in satisfying automatic and intelligent recognition needs. As the deep learning (DL) era comes, CNNs totally dominate the recognition tasks of RSI, including but not limited to semantic classification (SC), segmentation [3] and object detection or recognition [4]. As the fundamental algorithm, researchers have proposed a lot of CNN methods for RSI-SC in the past decade. To boost the model's performance, these methods have commonly paid more attention to human re-modeling strategies, including modifying pre-trained models, fusing features from models, creating combined models and so on, resulting in a remarkable increase in hardware and time costs. However, these ingenious methods still achieved insignificant performance advances due to some imperceptible flaws. The reasons are given as follows:

First, a model's training can be viewed as searching in the parameter space for an optimal solution. On this premise, the shallow model of traditional ML corresponds to a small search space, and consequently, its training can employ exhaustive computation. Hence, all the strategies using human modeling can be proven effective or not for shallow models. Given the far larger capacity, however, it is impossible to perform an exhaustive search for deep CNNs. Besides, re-modeling will enlarge the search space due to increased parameters. Therefore, it is more difficult to identify the effectiveness of modeling strategies if the methods encounter an expanded parameter space but still employ finite training steps in searching.

Second, the benchmark RSI sets are commonly much smaller than natural images. For example, the Aerial Image Dataset (AID) and Northwestern Polytechnical University (NWPU) [5,6] RSI sets only have smaller amounts of 10,000 and 31,500, while the ImageNet-1K has one million more. Currently, all the off-the-shelf CNNs for RSI recognition are developed on ImageNet-1K. Hence, a CNN is easy to overfit on RSI datasets due to its large capacity. To avoid overfitting, data augmentation (DA) was widely employed in previous studies. However, a DA-processed training set will have a data distribution shift compared to the original one. With this shift problem unhandled, a CNN will easily achieve suboptimal performance [7]. To the authors' best knowledge, the problem has been commonly ignored in previous studies.

Third, some emerging TTs can boost a CNN model's performance, but they are also developed based on the attributes of natural images. Nonetheless, several previous methods for RSI-SC have employed TTs in training and achieved better performance. However, these studies only copied the TTs' usage from ImageNet-1K without considering the inherent difference in RSI. Therefore, this simplistic operation may also be suboptimal due to the domain gap. In addition, previous methods commonly employed transfer learning as the technical route. However, most of them break several basic rules of transfer learning, e.g., using a larger learning rate (LR) of 1E-02. Some evidence reveals that a larger LR can easily make a CNN overfit on RSI sets [8].

To solve these problems, we proposed a simple experiment by training a lightweight CNN [9] (i.e., EfficientNet-B0) on RSI sets. In detail, an EfficientNet-B0 using pre-trained weights on ImageNet-1K is retrained on AID and NWPU using a smaller LR of 1E-04. The whole training only includes 60 epochs without any secondary modeling. Given the same training ratios (TRs), the OA results are compared to some previous state-of-the-art (SOTA) CNN methods published before 2023 (see Table 1, where "None" means no data presented in related studies). As shown in the table, we can see that the single EfficientNet-B0 method can clearly surpass all other methods by using beginner-friendly transfer learning with much fewer parameters. Therefore, it reveals that there may be a more efficient but accurate solution for RSI-SC by using a single CNN.

Table 1. OA (%) comparison of CNN methods on AID and NWPU.

Methods	Roadmap	Param Size (M)	AID	NWPU
			TR–20%	TR–10%
[10] at 2018	Fine-tuning loss	23.4	90.82 ± 0.16	89.22 ± 0.50
[11] at 2019	Feature fusion	276	95.02 ± 0.28	91.30 ± 0.18
[12] at 2019	CNNs ensemble	167	None	92.44 ± 0.37
[13] at 2020	Attention aided	14	95.73 ± 0.22	92.70 ± 0.32
[14] at 2020	Feature fusion	24	None	92.17 ± 0.08
[15] at 2021	Attention aided	24	94.45 ± 0.76	None
[16] at 2022	Models combination	54	96.19 ± 0.48	93.67 ± 0.21
[17] at 2022	Attention aided	8	95.43 ± 0.32	93.39 ± 0.39
[18] at 2022	CNNs ensemble	47	96.20 ± 0.15	93.24 ± 0.15
[this work]	Single EfficientNet-B0	5.3	96.08 ± 0.07	93.47 ± 0.05

To tackle the problem, we proposed a simple, lightweight but accurate method for RSI-SC named regularization-enhanced EfficientNet (RE-EfficientNet). It consists of a single EfficientNet-B3 model and a concise training algorithm called RE-CNN. The EfficientNet-B3 has better performance on ImageNet-1K and much fewer parameters if compared to other classical CNNs such as Resnet and so on. Some experiments also prove that EfficientNet-B3 can achieve better performance on RSI sets compared to other CNNs [19]. The RE-CNN algorithm includes some special ideas to handle the aforementioned problems coming from DA and a simple copy of TTs' usage. It presents an effective combination of regularizations that can significantly boost the CNN's performance. The experimental results on AID and NWPU show that RE-EfficientNet can prevail over 30 SOTA CNN and ViT methods published before 2023. The study's contributions are summarized as follows:

First, the work presents a simple, lightweight but more accurate single-CNN method for classifying RSI. The RE-EfficientNet only consists of an off-the-shelf EfficientNet-B3 with 12 M parameters and employs a replicable open-source training algorithm through transfer learning. Extensive experimental results on AID and NWPU prove that RE-EfficientNet can surpass other SOTA methods with not only remarkable OA improvements but also much fewer parameters.

Second, this work demonstrates that modified TTs based on the inherent characteristics of RSI can significantly improve a CNN's performance for RSI-SC. Ablation experimental results prove that the RE-CNN algorithm can boost a CNN's performance by 1% OA if the TTs are effectively combined.

Third, the work proves that classification tasks for RSI can be better performed by using a simpler roadmap at lower hardware and time costs. The easily accessible EfficientNet-B3 and transfer learning pipeline mean that RE-EfficientNet is a time-saving and beginner-friendly solution. In addition, based on the authors' viewpoints, the model training ideas proposed in this paper can also help us develop more efficient approaches in the future.

The paper's following sections are as follows: Section 2 is a brief review of related works. Sections 3 and 4 present the proposed theory and method. Section 5 presents experimental results, and Sections 6 and 7 present the discussion and conclusions.

2. Related works

In the beginning, CNNs were used as a fixed feature extractor for RSI-SC due to a lack of sufficient knowledge accumulation. For example, with the pre-trained CNNs from ImageNet-1K, Chaib et al. [20] directly took the model's outputs as the representation of RSI but skipped retraining. The method greatly improved its OA if compared to previous approaches using shallow models. However, the roadmap also lacks effective retraining on RSI sets, which undoubtedly results in poor performance. Afterward, some researchers proposed another roadmap by retraining the CNN on RSI sets using fine-tuned objective functions. For example, Cheng et al. [9], Liu et al. [21,22] and Bazi et al. [23] proposed new auxiliary losses for training CNNs. These ideas consist of complex human-modeling procedures and indicators but do surpass the extractor roadmap.

Following that, more creative methods were tried to find better solutions. These ideas mainly include architecture fine-tuning, feature fusion and model combination. For example, Zhang et al. [24] proposed a method by combining two CNNs with capsule nets to verify the effectiveness of multi-mode combinations. Xie et al. [25] proposed another approach by fine-tuning CNN's last pooling layer to make the model capable of processing arbitrary resolutions of input images. With the layer's activated information, Zhu et al. [10], Guo et al. [11] and Li et al. [14] proposed similar methods by choosing relatively important features that can boost the model's performance. Sun et al. [26] proposed a method by fusing outputs from a CNN's different layers to generate the so-called semantic and appearance features that can boost the model's performance. These methods presented meaningful novelty, but the improvements were not very remarkable.

With the success of vision transformers (ViTs), attention mechanisms were widely introduced into a CNN's architecture for gaining advances. For example, Tong et al. [13], Guo et al. [27] and Tang et al. [28] proposed similar methods by adding attention modules to a pre-trained CNN and verified that attention-enhanced CNNs can be better for RSI-SC than the original models without attention modules. Moreover, by using pre-trained models with built-in attention modules, Alhichri et al. [15], Li et al. [16] and Chen et al. [17] proposed other methods by fusing the CNN's outputs with human-engineered indicators or feeding the CNN's outputs into a sequential model to boost the method's performance. All these methods are creative and achieve better performance to some extent, but the final performance still lacks exciting advancement.

In addition, Minetto et al. [12] and Zhao et al. [18] proposed two different ensemble methods using multiple CNNs. The first method employs twelve CNNs coarsely trained on RSI sets and uses the weighted outputs of individual models as votes for prediction. The second method employs a single CNN as the main classifier but embeds multiple different CNNs as functional branch modules to boost the method's performance. These two methods are competitive for RSI-SC to some extent. However, the two methods also have the disadvantages of high hardware costs and tedious training procedures. As the ViT arose, Bazi et al. [29], Zhang et al. [30] and Wang et al. [31] proposed different methods by using a single ViT and proved ViTs can be competitive for RSI-SC as CNNs. However, all these methods' performances are still not exciting if compared to the results shown in Table 1.

Moreover, researchers also proposed other creative approaches to find better solutions for RSI-SC. Shi et al. [32] proposed a single-CNN method by using a so-called self-compensating module to obtain a lightweight model that can achieve acceptable performance compared to ready-made CNNs. Chen et al. [33] proposed a single-CNN approach by inserting two different attention modules as functional branches in a pre-trained CNN to find performance advances. Deng et al. [34] proposed a dual-model combining method by fusing the outputs of a pre-trained CNN and ViT and showed competitive performance. Miao et al. [35] proposed a multi-granularity decoupling CNN method by using class-imbalanced pseudo-label selection. It aims to tackle CNN's dependence on a great number of training samples and shows acceptable performance. Song et al. [36] proposed a single CNN method by applying clustering to the extracted features and demonstrated improved performance when training samples were limited. Wang et al. [37] proposed a multi-CNN method by employing two CNNs as cooperative classifiers and an adaptive image size transformer. The model can achieve competitive performance when processing images with different resolutions. Xu et al. [38] proposed a knowledge distillation method by distilling out the long-range features contained in a ViT teacher and transferring them to a CNN student. However, all these methods still share a poor tradeoff between hardware costs and accuracy.

Currently, regularization and DA are commonly used in CNN training, although these techniques also need professional implementation [39]. For example, with finite training epochs, too strong regularization results in poor fitting, and likewise, too weak regularization leads to over fitting. Similarly, a DA-processed training set will have a significant shift in data distribution, making the model suboptimal if the shift is well handled [7]. Moreover, most of the TTs are developed on ImageNet-1K, which has a domain gap compared to RSI. Nevertheless, TTs are very meaningful due to their obviously low hardware dependency and common open-source nature. Therefore, we should conduct careful research on the usability of TTs before using them. However, most of the previous methods simply copied the TT's procedure from natural images without any modifications.

To tackle the above problems and enhance the findings of previous studies, we first proposed some novel ideas for the model choice and training algorithm based on a deeply theoretical analysis. Then, we selected a simple pipeline that not only can achieve better performance for RSI-SC but also has much lower hardware and time costs. First, the proposed RE-EfficientNet method only employs an EfficientNet-B3 as the base model and excludes any re-modeling procedures to keep the search in parameter space unexpanded. Second, to adequately utilize the pre-trained weights on ImageNet-1K, this method still employs transfer learning to reduce time costs in training. Third, the RE-CNN training algorithm employs two different combinations of DA transformations in the whole training process to ensure the shift problem of data distribution coming from DA can be alleviated. Fourth, we modified two TTs named CutMix [40] and Online Label Smoothing (OLS) [41] according to the inherent properties of RSI. Then, the RE-CNN algorithm employs the combination of modified CutMix (m-CutMix) and OLS (m-OLS) as regularization in training to boost the model's performance. In brief, the RE-EfficientNet method is a simple, lightweight but more accurate approach for RSI-SC. The proposed method's pipeline is thoroughly different from the previous studies.

3. Methodologies

3.1. Theoretical basis

The CNN learns the image features through the operation of convolution. In the architecture, the implementation of convolution is called the convolutional layer. Let $X \in \mathbb{R}^{C,H,W}$ be the input signals for a convolutional layer, and C, H and W denote its channel number, height and width in pixels. Let $Y \in \mathbb{R}^{C_O,H_O,W_O}$ be the layer's output, with C_0, H_0 and W_0 denoting the same definitions as C, H and W. Then, the convolution processing can be described as follows:

$$Y = \sum_{n=1}^{\bar{}} Wt_{C_O} \times X + Bias_{C_O} \tag{3.1}$$

where Wt denotes the adjustable weights of every convolution kernel, and $Bias$ denotes a random variable.

In training, based on the common gradient descent technique, we employ the back-propagation algorithm to fit a set. Let Z be the output of the following layer defined in Eq (3.1), with the same Y, Wt and X as in Eq (3.1). Then, the gradient of Z at Wt can be described as

$$\frac{\partial Z}{\partial Wt} = \frac{\partial Z}{\partial Y}\frac{\partial Y}{\partial Wt} \tag{3.2}$$

The convolution processing, as shown in Eq (3.1), is an element-wise multiplication and accumulation operation in mathematics. Then, according to the chain rule in differential calculus, the gradient of Z at w can also be described as

$$\frac{\partial Z}{\partial Wt_i} = \frac{\partial Z}{\partial Y_1}\frac{\partial Y_1}{\partial Wt_i} + \frac{\partial Z}{\partial Y_2}\frac{\partial Y_2}{\partial Wt_i} + \cdots + \frac{\partial Z}{\partial Y_n}\frac{\partial Y_n}{\partial Wt_i}. \tag{3.3}$$

According to Eqs (3.1) and (3.3), we can derive that every parameter of a convolutional kernel shares the same gradient value in a back-propagation process due to the addition operation. Hence, in the simplest case, we can treat every kernel of a layer as a single meta-function. Let F be the meta-function, with the same definitions for X, Y, H_O, W_O and C_O in Eq (3.1). Then, the sliding window operation of a convolutional layer can be described as

$$Y_{C'} = \begin{bmatrix} F_{C_O,1,1}(X) & \cdots & F_{C_O,W_O,1}(X) \\ \vdots & \ddots & \vdots \\ F_{C_O,1,H_O}(X) & \cdots & F_{C_O,W_O,H_O}(X) \end{bmatrix}. \tag{3.4}$$

Currently, the architecture of deep CNNs commonly consists of cascaded convolutional layers with a fully connected classifier at the tail. Let $Pred$ be the prediction of a CNN model with i neurons and k layers. Let L denote the operation of every convolutional layer of the model, with the same definition of X as in Eq (3.1). Then, in the simplest case, the predicting process of the CNN can be described as follows:

$$Pred = \sum_{n=1}^{i} L_k(L_{k-1} \dots (L_1(X)). \tag{3.5}$$

In the back-propagation process, we can set all the parameters of a CNN to a certain value at each training step. The state of all these parameters, according to the definitions of permutation and combination in probability theory, is a combination problem. Let us hypothesize that the setting value of every parameter, in the simplest case, can only be 0 or 1. Let C_{NPSC} be the number of parameter-setting combinations (NPSC). Then, the NPSC for a certain CNN layer with a number of N parameters can be described as follows:

$$C_{NPSC} = 2^N. \tag{3.6}$$

Let $C_{NPSC-CNN}$ be the NPSC for a whole CNN, with a number of M layers and the same number of N parameters in each layer. Then, according to Eq (3.6), its NPSC can be described as

$$C_{NPSC-CNN} = 2^{N \times M}. \tag{3.7}$$

Note that the interval for a CNN's single parameter is 0 to 1 in the real training state. As a result, the real NPSC is much larger than $C_{NPSC-CNN}$ in Eq (3.7). The CNN predicting function F in Eq (3.5) is a complex composite function consisting of multidimensional variables. To date, we commonly employ the average gradient of a mini-batch to train a CNN with finite training epochs because an exhaustive search is unrealistic. Hence, a CNN method is essentially a non-convex problem that has a number of different locally optimal solutions achieved in training. Therefore, a method with a larger NPSC apparently corresponds to a lower frequency to gain locally optimal solutions unless the searching process employs some efficient modes, such as regional sampling or greedy searching. Unfortunately, to the authors' best knowledge, such an efficient and common enough training algorithm has not emerged yet. As a result, we can simply conclude that a CNN with a smaller NPSC corresponds to a higher probability of achieving its best potential performance in finite training epochs, or, in other words, a shallow model easily achieves convergence.

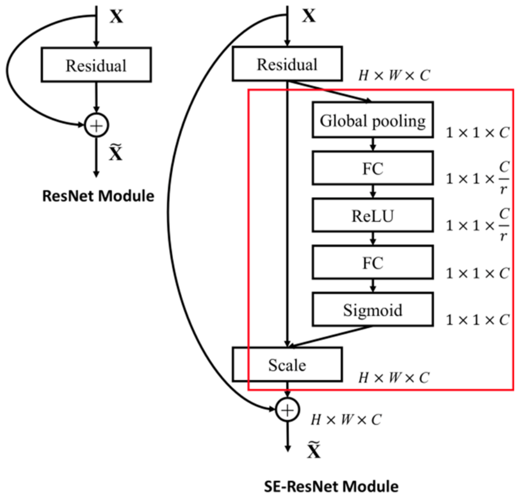

Figure 1. Squeeze-and-excitation block.

The attention module in the CNN's architecture commonly employs two techniques, including channel-wise or spatial-wise attention, to re-weight each output stream of the current layer. For example, Hu et al. [42] proposed the squeeze-and-excitation (SE) block as channel-wise attention to boost a CNN's performance. The SE structure, as shown in the red rectangle of Figure 1, performs SE and another scale operation on the inputs and outputs of the current layer, respectively. Let X and X_{Scale} be the original and weighted signals as defined in Eq (3.1), with the same W, H and C. Then, the scale can be described as follows:

$$X_{Scale_C}{}' = X_c \odot \begin{pmatrix} Wt_{SE_C} & \cdots & Wt_{SE_C} \\ \vdots & \ddots & \vdots \\ Wt_{SE_C} & \cdots & Wt_{SE_C} \end{pmatrix}, \quad (3.8)$$

where $Wt_{SE} \in [0, 1]$ denotes the SE output weight matrix. Note that the Wt_{SE} still has C channels after channel-wise compression, and in the same channel, all weights are equal.

After pre-training, the weights of Wt_{SE} in Eq (3.8) are set to larger or smaller values correlated to the channel-wise feature's importance. In Eq (3.8), X multiplied by Wt_{SE} yields X_{Scale}. As a result, X_{Scale} multiplied by a larger weight becomes more important. In the back-propagation process, according to Eq (3.3), the gradient value of prediction is more sensitive to the parts with great information entropy. Therefore, with the SE attention activated, each iteration step in training becomes sensitive to partial channel-wise regions of the CNN's architecture, resulting in the other regions being slightly updated. Then, according to Eq (3.7), with the attention module inside, the NPSC of a pre-trained CNN is essentially smaller than the one without attention because the former's searching process always ignores its partial channels. Therefore, given finite training steps, a CNN with built-in attention modules commonly performs better. In contrast, we more often employ the class activation mapping technique [43] to intuitively prove this selective importance.

In the feature fusion method, however, the NPSC is always larger than the one without fusion if the feature extractors are hooked in the same back propagating chain. Let $C_{NPSC-FF}$ be the NPSC of a fusion method, with k types of deep features extracted from different layers in a single CNN. Let us hypothesize that the method concatenates all the fused features as the input for the last classifier, or more accurately, hooks all the features in the chain. Then, its NPSC, according to Eq (3.7), can be described as follows:

$$C_{NPSC-FF} = \sum_{j=1}^{k} C_{NPSC-ET}, \tag{3.9}$$

where $C_{NPSC-ET}$ is the NPSC of a certain extractor. Comparing Eqs (3.7) and (3.9), we can easily find that, with the larger NPSC, these fusion methods concatenating all the fused features will have a lower frequency to gain more locally optimal solutions than the ones without fusion if equal training steps are given. Therefore, with finite training epochs, the common fusion strategy may face a worse probability of achieving better performance unless it employs some efficient searching modes or unhooks the features.

Similarly, the method of combining CNNs will have a much larger NPSC if the combined CNNs are in cascaded mode. Theoretically, the method's NPSC is the product of the NPSCs of each CNN. Similarly, with multiple CNNs inside, the NPSC of the combined method in parallel mode is the same as the cascaded CNNs if the combined CNNs are hooked. Therefore, a combining-CNN method also requires some efficient searching algorithms to ensure its true performance can be achieved in finite training steps.

However, the ensemble of CNNs does not increase the NPSC of a single CNN due to its independent training process for its individual classifiers. Similarly, this rule is also right if a method has multiple CNNs but employs independent training processes for each model, or more accurately, unhooks its multiple CNNs. Nonetheless, these two strategies are more complicated and costly than a single CNN method.

The prevalence of human re-modeling mainly derives from two ideas: First, human modeling does succeed in ML fields. Second, based on the different imaging conditions compared to ImageNet-1K, the domain gap may weaken the pre-trained CNN's performance on RSI. These viewpoints, however, are not exactly as correct as we think. The reason is twofold. First, as concluded before, some complex human re-modeling strategies reduce the probability of finding better solutions. Meanwhile, a deep CNN is easy to overfit on the smaller RSI sets. Second, the invariant feature in ImageNet-1K does have similarities to RSI. Given an all-layer-frozen CNN, we can still achieve an OA of approximately 75% on AID and NWPU if the last classifier can be trained (see Table 2).

Based on the above mathematical derivations and analyses, this work employed a simpler transfer learning strategy by using a pre-trained CNN that has built-in attention modules. In particular, the proposed method excludes any architecture modification or feature fusion to maintain the unexpanded NPSC. Given this premise, this work also cautiously evaluated the real value of the invariant feature in the smaller RSI sets. More importantly, it employed a smaller LR and an effective combination of TTs as regularization to avoid overfitting.

Table 2. OA (%) comparison of frozen tests on AID and NWPU.

Methods	Roadmap	AID		NWPU	
		TR-20%	TR-50%	TR-10%	TR-20%
Frozen CNN layers with a trainable classifier	An EfficientNet-B0	78.66	83.42	72.74	76.87

3.2. Training framework

In this paper, we propose a concise and beginner-friendly training framework, as shown in Figure 2. In summary, the training process consists of 300 successive training epochs in total but can be viewed as two steps because the DA and regularization strategy (DARS) in training have some differences. In detail, the DARS of the first 60 epochs, named Step 1, only consists of a cascading combination of routine geometric transformations, coupled with m-OLS as regularization. The DARS of the second 240 epochs, called Step 2, also includes a combination of cascading routine geometric transformations. It additionally uses m-OLS and m-CutMix as regularizations.

The training procedures, as shown in Figure 2, start with the red arrows with the DARS of Step 1 activated. Then, at epoch 61, the model undergoes another 240 training epochs using the DARS of Step 2, just as the blue arrows indicate.

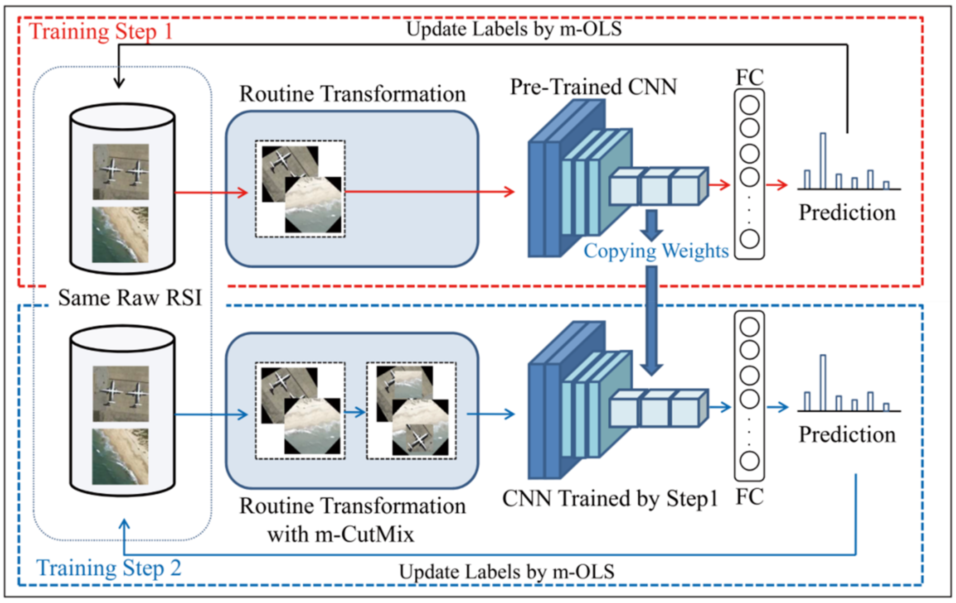

Figure 2. The proposed training framework.

3.3. Training algorithm

The procedures of RE-CNN are presented in Algorithm 1. In training, the training and testing images' resolutions are 256^2. In detail, the combination of transformations in Steps 1 and 2 is the same, consisting of a resize, a color jitter, a horizontal and vertical flip and a rotation. The mini-batch size in training is 30. The initial LRs in Steps 1 and 2 are both 1E-04. The LR optimizer is a cosine decay algorithm with maximum iterations equal to the training epochs. The object function is cross-entropy. The error's back-propagating algorithm is Adam-W [44], with a weight decay of 1E-06.

Algorithm 1: The procedures of RE-CNN.

```
    Definition.
    X: a mini-batch of image samples; Y: the corresponding labels of X; TNI: the total
    number of images in the training dataset; CT: a combination of transformations;
    CM: the m-CutMix transformation; CM-Loss: the corresponding loss function of
    CM; OLS-Loss: the corresponding loss function of m-OLS; F: the forward
    function of the CNN model; Update: the parameter updating function of CNN
    through back propagating; Acc: the model's prediction accuracy; Results: the
    dictionary of Acc and its weights file.
1   CT = { resize, color Jitter, horizontal and vertical flip, rotation }
2   Initialize (model, pre-trained weighs file from ImageNet-1k )
3   for Epoch = 1, …, 60 do
4           for batch = 1, …, TNI / 30 do
5                   Z = F ( CT (X) )
6           Grad = OLS-Loss (Z, Y)
7           Update (model, Grad)
8           end for
9       Acc = F (Testing Dataset)
10      If Acc is the best then
11          Save the weights file of the best Acc in Results
12  end for
13  Initialize (model, the weights file of the best Acc in Results)
14  for Epoch = 61, …, 300 do
15          for batch = 1, …, TNI / 30 do
16                  Z = F ( CM ( CT (X) ) )
17          Grad = CM-Loss ( OLS-Loss (Z, Y) )
18          Update (model, Grad)
19          end for
20      Acc = F (Testing Dataset)
21      If Acc is the best then
22          Save the weights file of the best Acc in Results
23  end for
```

3.4. Model's architecture

RE-EfficientNet employs a single EfficientNet-B3 model as the base model, while EfficientNet-B0 is also used to support some viewpoints. EfficientNets are well-known deep CNN models with built-in SE blocks, and their architecture charts can be found in [8]. The attention mechanism, as concluded before, helps EfficientNets gain better performance on ImageNet-1K. In this work, all the CNN models, as well as the pre-trained weights on ImageNet-1K, follow the original settings in PyTorch without any modification.

3.5. OLS modification

Szegedy et al. [45] proposed label smoothing as regularization to boost performance and avoid overfitting. This technique converts the one-hot hard label of 1 or 0 to a soft version. For example, a value of 0.9 denotes the probability of a certain sample belonging to class A, while a residual value of 0.1 denotes the sum of likelihoods belonging to the other classes. In this setting of 0.1, the likelihoods share an equal average value without considering the difference in similarity between classes. To tackle this problem using an online-updated training style, Zhang et al. [41] proposed the OLS as a solution to learn category similarity from training sets. Let L_{OLS} be the final label of a subclass, L_H denote the hard label and L_S denote the value of OLS. Then, the computation of a final label can be described as follows:

$$L_{OLS} = \alpha \times L_H + (1 - \alpha) \times L_S \ , \tag{3.10}$$

where α (∈ [0, 1]) is a hyperparameter to balance the hard and soft labels. In training, the OLS stores and accesses the labels in a matrix variable of the categories' similarity. Let $Matrix_S$ be this soft label matrix and n be the number of classes in a dataset. Then, the data shape of $Matrix_S$ can be described as follows:

$$Matrix_S = \begin{bmatrix} label_{1,1} & \cdots & label_{1,n} \\ \vdots & \ddots & \vdots \\ label_{1,n} & \cdots & label_{n,n} \end{bmatrix} , \tag{3.11}$$

where $label_{1,n}$ denotes the updated probability that the first-class sample belongs to class n. The updating algorithm for $Matrix_S$ can be found in [41].

Experiments on ImageNet-1K in [41] set the α in Eq (3.10) at 0.5 and proved that its effectiveness surpasses label smoothing. In this paper, based on the obvious larger difference in similarity between subclasses, the α is given an empirical value of 0.9 for the RSI.

3.6. CutMix modification

Mixup, proposed by Zhang et al. [46], can help a CNN avoid overconfidence on the training dataset. In implementation, it overlaps an A-class sample onto another B-class sample with a fuzzy overlap rather than a complete replace. Consequently, in training, the model will learn an unnatural image. As an improved version, Yun et al. [40] proposed another, CutMix, giving its operation algorithm in the paper. Different from Mixup, CutMix directly replaces an equal proportion of a class-B sample with an A-class sample's cut patch. Following that, CutMix also modifies the label of the cut-and-mixed image according to the proportion of patches. Let L_{CM} be the label of a cut-and-mixed image, L_O be the label of the original image and λ be the proportion of patches. Then, the L_{CM} can be described as follows:

$$L_{CM} = \lambda \times L_{O_A} + (1 - \lambda) \times L_{O_B} . \tag{3.12}$$

Experiments on ImageNet-1K proved CutMix combined with OLS can further improve CNN performance. However, with the same experiments on RSI, it may not work that well. The reason lies in Figure 3.

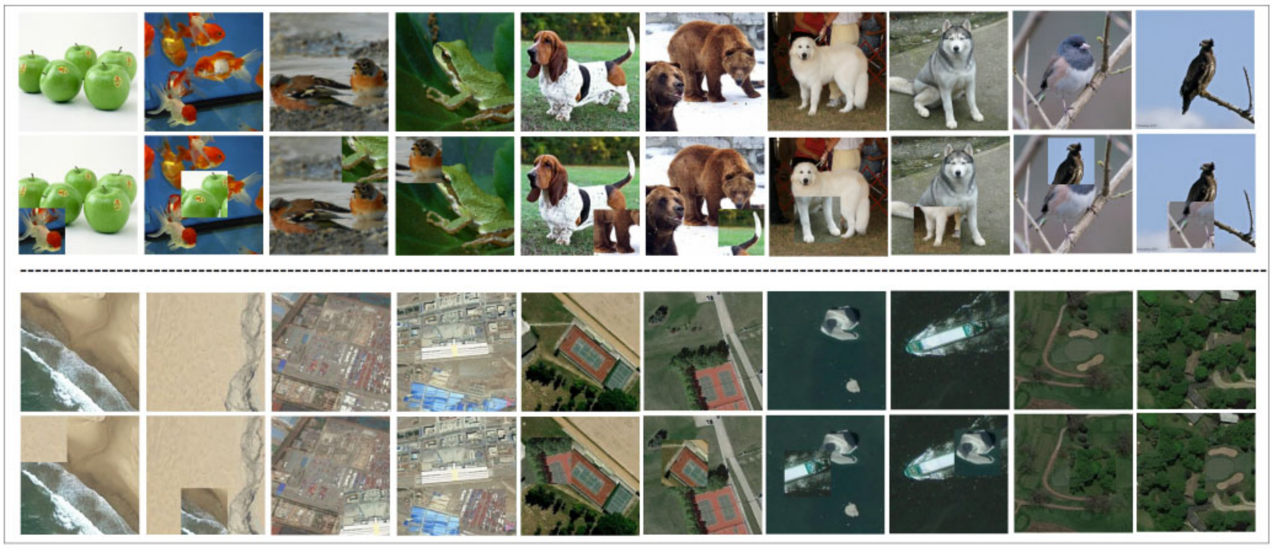

Figure 3. The cut-and-mixed images.

In Figure 3, rows 1 and 2 are images from ImageNet-1K, with originals in the first and cut-and-mixed ones in the second. At the bottom of Figure 3, rows 3 and 4 are originals and cut-and-mixed images from the RSI dataset. Without zooming in, taking the natural images for instance, we can easily tell where the image patch is and classify the patch to a certain class, as well as the background. However, with a quick look at the RSI without zooming out, it will become more difficult to locate the patch and tell the cut-and-mixed image's category as the semantic scenes change from left to right.

This contradiction in Figure 3 clearly explains the large difference in similarity between natural images and RSI. With the definition in Eq (3.12), it can work well for natural images; however, it is not absolutely right for RSI. More specifically, taking the RSI into account, the patch of A-class images does have some similarity with the cut-and-mixed background of B-class. In training, if we assign a B-class label to an A-class patch, the CNN model will correlate some features in the patch to the B-category and eliminate the correlation with the A-category. As a result, with the impaired features of the A-class samples, the model will show poor performance when classifying those samples similar to the A-category. Hence, we argue that we should modify TTs based on the inherent difference in RSI. Therefore, we rebuilt several parts of CutMix, including its algorithm and usage, as follows:

First, based on the greater similarity of RSI, this paper sets a new parameter to control the probability of a cut-and-mixed operation in training. In detail, this parameter has an empirical value of 0.1 based on extensive experimental results. Second, based on the similarity of RSI, this paper further utilizes the hierarchical similarity among categories contained in the $Matrix_S$ as defined in Eq (3.11). More specifically, in each training epoch, the image patch of m-CutMix is only cut from those images belonging to the 15 classes in the lowest order of similarity. The m-CutMix algorithm is shown in Algorithm 2.

Algorithm 2: The cut-and-mix procedures of m-CutMix.

```
   Definition.
   X: a mini-batch of samples; Z: a mini-batch of cut-and-mixed samples; CMO: the
   original CutMix function; P: the probability of a cut-and-mix operation in training;
   $Matrix_S$: the matrix contains the hierarchical similarity among categories updated
   by OLS; sort: the sorting function; random: the random pick-one-out function.
1  for x in X do
2      if P > 0.1 then
3          index = sort ($Matrix_S$ (x), type = ascent)
4          target = random ( index [0 : 15] )
5  .....z = CMO ( x, X [target] )
6          Z.append(z)
7      else
8      Z.append(x)
```

In Step 2, the m-CutMix is active in conjunction with the m-OLS. Hence, the label of each training sample should be recalculated according to Eqs (3.10) and (3.12). Let L_{Step2} be the sample's label in Step 2, and then its calculation can be described as follows:

$$L_{Step2} = L_{CM}\left(L_{O_A}, \quad L_{O_B}\right) = \lambda \times L_{O_A} + (1-\lambda) \times L_{O_B}, \tag{3.13}$$

where L_O and L_{CM} are the same as in Eqs (3.10) and (3.12).

3.7. Datasets and evaluation criteria

This study employs AID and NWPU as benchmarks to verify the method's effectiveness. The introduction to the two datasets can be found in [5,6]. As a brief summary, these two sets are both cropped from Google Earth. AID has 30 categories of 10,000 images with a fixed resolution of 600^2, while NWPU contains 45 categories of 31,500 images with a fixed resolution of 256^2. With categories numbered, the representative images for each AID category are shown in Figure 4, while those for NWPU are shown in Figure 5.

In this study, to get a fair comparison, the TRs of two datasets are used with the same settings as previous studies, i.e., AID is 20% and 50%, and NWPU is 10% and 20%. The training subsets are also selected at random, using the rest of the samples as testing subsets. This paper also employs the OA and confusion matrix as criteria for different methods' effectiveness evaluations. In detail, the OA can be described as follows:

$$OA = \frac{N_C}{N_T}, \tag{3.14}$$

Where N_C is the total number of accurately classified samples, and N_T is the total number of tested samples.

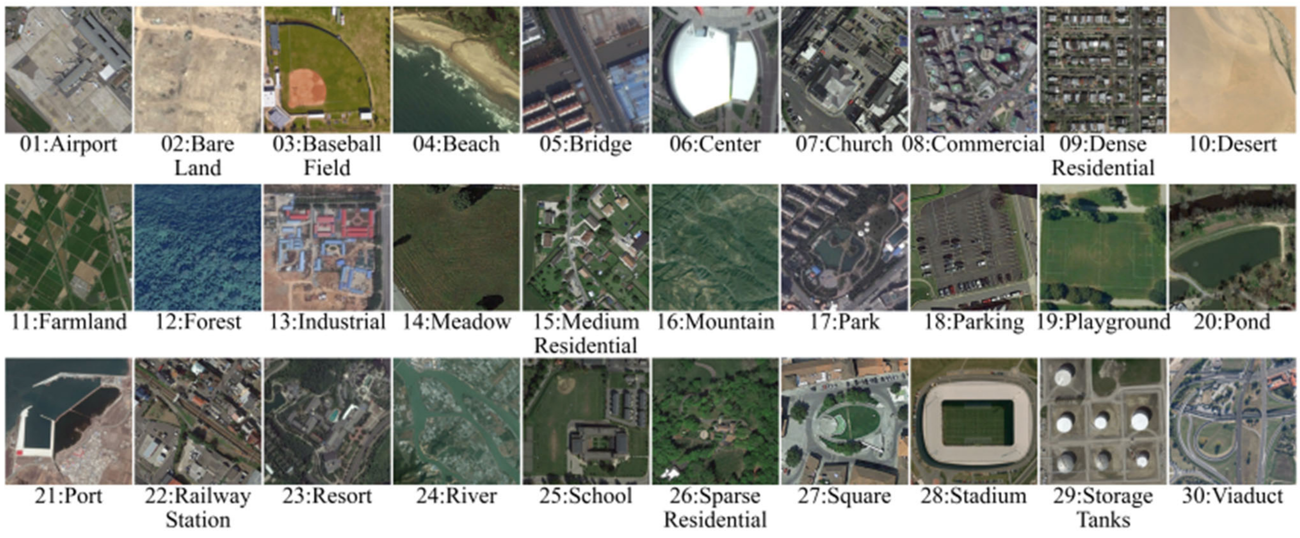

Figure 4. Sample images for each AID category.

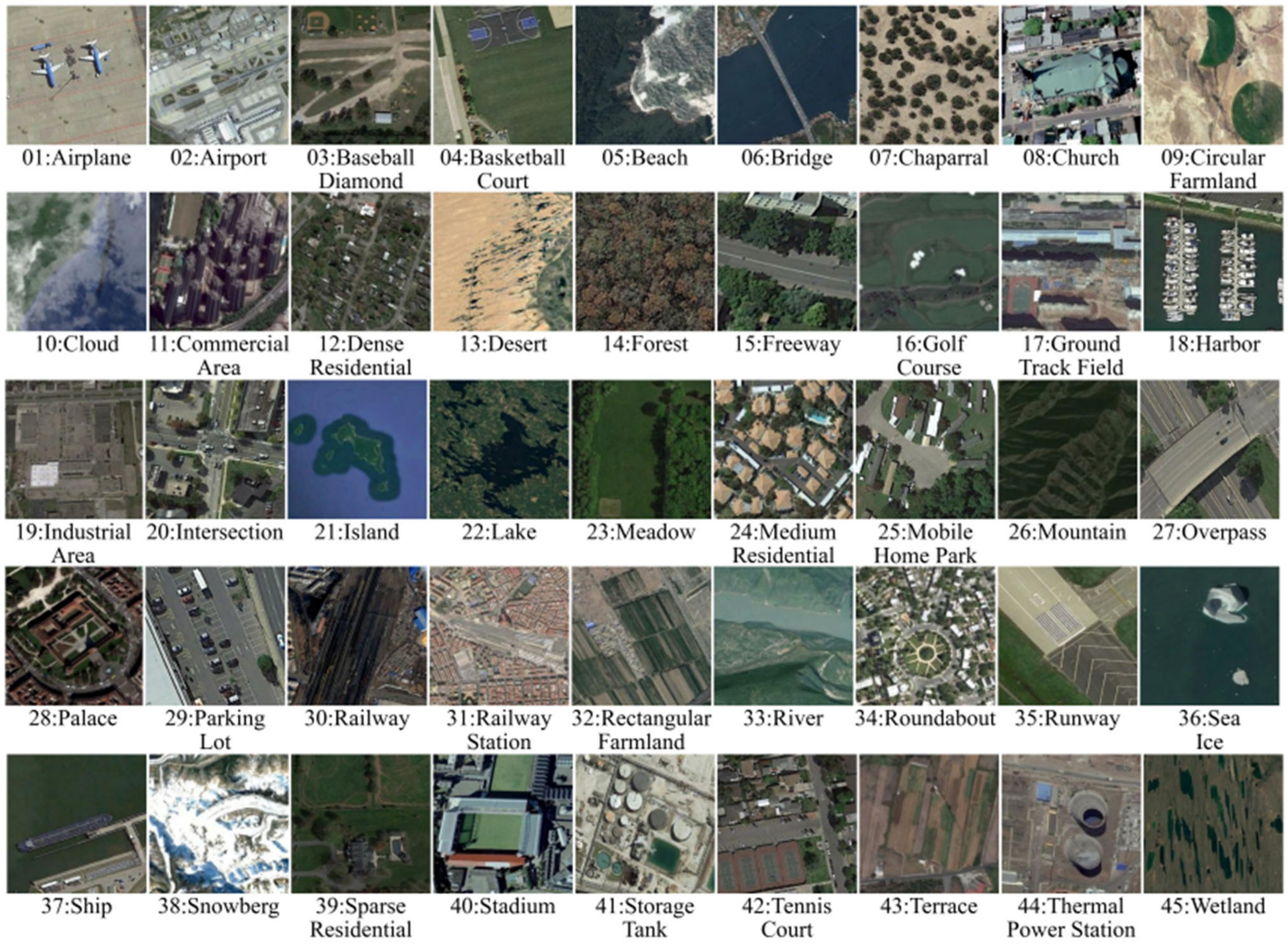

Figure 5. Sample images for each NWPU category.

3.8. Hardware and software environments

The experiments were performed on four computers, each equipped with a single RTX 2060. PyTorch 1.11.0 was running with the Compute Unified Device Architecture 11.5 on Windows 10. All the experimental results were averaged over five runs.

4. Experimental results

4.1. Training curves

The training curves at different TRs are shown in Figure 6, with the top two charts corresponding to loss curves and the bottom two charts being testing accuracy curves. As shown in Figures 6a and 6b, the training losses of AID and NWPU both show a fast decrease in Step 1, rebound with obvious amplitude at the beginning of Step 2 and then present a slow decrease in the following epochs. As mentioned before, the loss calculations in Steps 1 and 2 are very different according to Eqs (3.10) and (3.13).

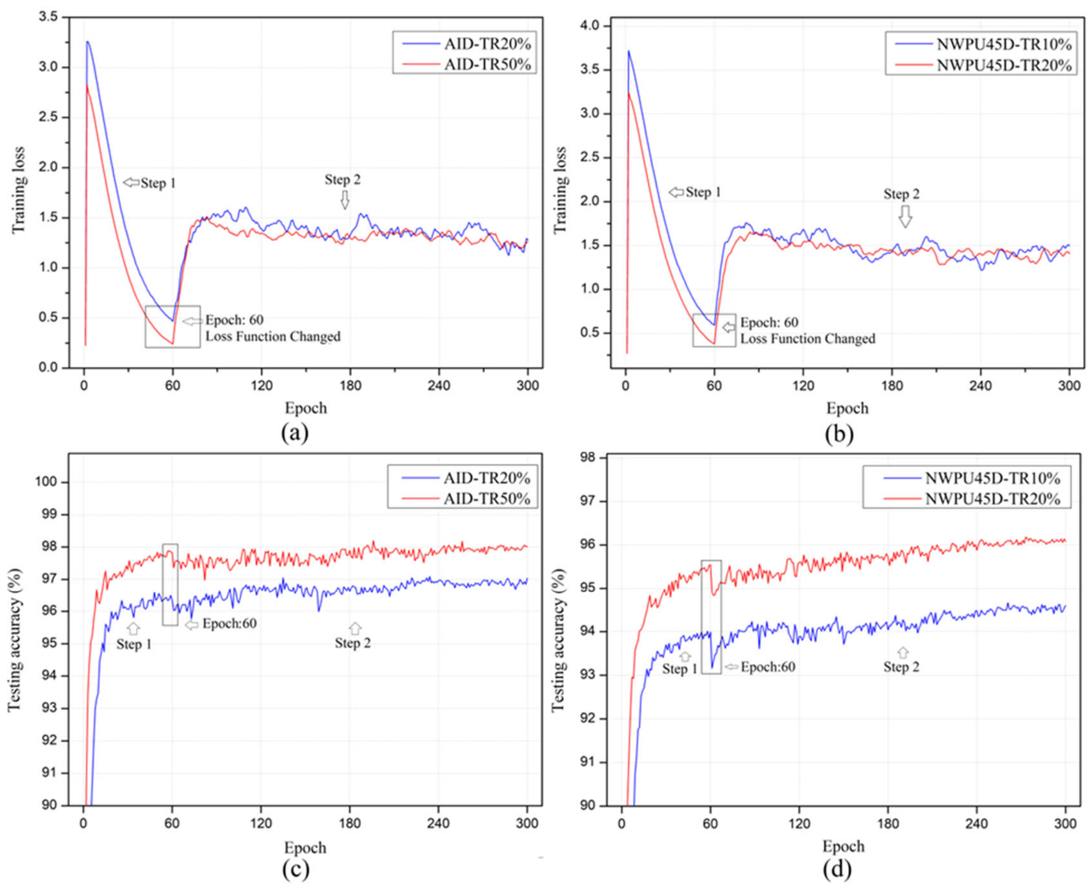

Figure 6. The fitting curves of AID and NWPU.

Hence, as shown in Figure 6c,d, the testing accuracy (i.e., the OA) in Step 2 achieves visible improvement on AID as well as NWPU. Similarly, as the m-CutMix was activated, the OA curves showed a clear but slight decline at the beginning of Step 2. In other words, the cut-and-mix operation does have a shocking effect on the model's performance, but with a cautious reconstruction, it can still help the CNN model gain better accuracy. This phenomenon indicates that a higher frequency of cut-and-mix operations may generate greater impacts on the loss and gradient changes. As some evidence in the experimental results [8] suggests, it will result in larger updates to the parameters through backpropagation and make the CNN suboptimal.

4.2. Accuracy results

The OA comparison of different methods published from 2017 to 2023 is shown in Table 3. In summary, the OA results include 30 previous studies, which consist of 23 CNN and 7 ViT methods. The "parameter size" column is the exact value self-reported or a minimum estimate according to the open information in the related papers. In addition, "not mentioned" means lacking exact information, while "none" corresponds to no relative information.

As shown in Table 3, RE-EfficientNet shows an obvious advance on AID and NWPU, while only the CNN ensemble in [18] and the ViT in [30] show a partial lead in a TR of 50% on AID. Even with the advantage of more parameters, the methods [18,30] still show a notable OA gap when the testing samples are getting larger in the fourth, sixth and last columns of Table 3. Hence, based on the authors' viewpoint, the partial leading in [18,30] may come from a reduced total number of testing samples. If taking RE-EfficientNet's OA as the baseline, the improvement on the 20% TR of AID is 0.50% over the leading method [31], while the improvements on the 10% and 20% TRs of NWPU are 0.70% and 0.75% over the other leading method [34], respectively. We can see that RE-EfficientNet's lead is more notable when the testing sets are larger. Taking the parameter size into account, we can find that RE-EfficientNet only has a quarter of the parameters in [31] and far fewer than the parameters in [34], which has several CNNs and a ViT inside. Hence, as a fair comparison, the results demonstrate that RE-EfficientNet is more effective and lightweight than the previous ones.

In Table 3, given the OA results between different technical routes, we can find out some valuable facts as follows:

First, the strategy of feature fusion in [10,11,14,20,24,25] commonly falls behind, while the one using attention modules in [13,15,17,27,33] is more likely to perform better if the feature fusion is abandoned. Attention modules, however, do not guarantee better performance when the feature fusion is coupled with attention modules like [23] does.

Second, putting the complex training procedure and huge parameter size aside, we can find that the strategy of model combinations in [16,26,28] has not shown significant improvements over the single CNN with built-in attention modules. However, in [28], the combination strategy using individually trained models can get a slight lead over its competitor with built-in attention modules if its hardware-extensive budget is acceptable.

Third, despite the complex training procedure, the ensemble of CNNs in [18] does show competitive performance. In [18], each CNN of the ensemble has newly added attention modules, and for this reason, we can see that the ensemble in [12] falls behind due to its inferior individual classifiers without attention modules.

Fourth, regardless of the larger parameter size, the ViT methods commonly perform better on the larger NWPU than the other single-CNN methods except RE-EfficientNet. In other words, the ViT has not shown persuasive advantages when training sets are smaller. Anyhow, considering the "data-hungry" effect of the ViT architecture widely shown in natural images [47,48], we can easily accept these results. In addition, it is commonly considered that a CNN can have inductive biases inside, such as translation invariance, that can quicken the model's fitting [49]. Contrary to CNNs, however, the ViT's advantage is the long-range dependence of different features and sequence information [50]. Hence, based on the authors' viewpoint, the transfer learning tasks of RSI are easier and cheaper by employing CNNs, unless the hardware-expensive ViT has other acceptable advantages in special tasks.

Fifth, the two TTs, i.e., label smoothing and Mixup, are used in several previous methods [13,17,27] without any modification. More specifically, the study in [37] even proposed an adaptive label updating pipeline. However, in Table 3, these four methods do not show exciting performance.

Table 3. OA (%) comparison of different methods on AID and NWPU.

Methods	Roadmap	Parameter size (M)	AID		NWPU	
			TR20%	TR50%	TR10%	TR 20%
[10] in 2019	Feature fusion	23.4	93.68 ± 0.29	94.75 ± 0.24	90.58 ± 0.19	91.91± 0.23
[11] in 2019		276	95.02 ± 0.28	96.66 ± 0.19	91.30± 0.18	93.45 ± 0.17
[20] in 2017		138	None	89.71 ±0.33	None	None
[24] in 2019		145	91.63 ± 0.19	94.74 ± 0.17	89.18 ± 0.14	85.08 ± 0.13
[25] in 2019		138	93.60 ± 0.12	96.66 ± 0.11	89.89 ± 0.16	92.55 ± 0.14
[14] in 2020		24	93.33 ± 0.29	95.38 ± 0.29	92.17 ± 0.08	92.46 ± 0.09
[23] in 2019	A single EfficientNet-B3 with feature fusion	> 12	94.19 ± 0.15	None	91.08 ± 0.14	None
[9] in 2018	Fine-tuning Loss	138	90.82 ± 0.16	91.89 ± 0.22	89.22 ± 0.50	91.89 ± 0.22
[21] in 2019	Models combination with	633	None	96.37 ±0.30^{*1}	None	93.27 ± 0.17^{*1}
[22] in 2019	fusion and fine-tuning loss	237	None	97.24 ± 0.32^{*1}	None	None
[13] in 2020	Attention aided	14	95.73 ± 0.22	97.16 ± 0.26	92.70± 0.32	94.58 ± 0.26
[17] in 2022		8	95.43 ± 0.32	97.39 ± 0.24	93.39 ± 0.39	94.95 ± 0.36
[15] in 2021		24	94.45 ± 0.76	96.56 ± 0.12	None	None
[27] in 2020		89	None	None	93.15 ± 0.09	94.86 ± 0.15
[33] in 2022		26	95.60 ± 0.17	97.14 ± 0.03	92.32 ± 0.15	94.66 ± 0.11
[28] in 2021	Models combination	276	93.33 ± 0.29	95.38 ± 0.29	91.09 ± 0.13	92.42 ± 0.16
[26] in 2020		138	92.20 ± 0.23	95.48 ± 0.12	None	None
[16] in 2022		54	96.19 ± 0.48	97.48 ± 0.39	93.67 ± 0.21	95.32 ± 0.36
[12] in 2019	CNNs ensemble	167	None	None	92.44 ± 0.37	None
[18] in 2022		47	96. 0 ± 0.15	98.54 ± 0.17	93.24 ± 0.15	95.58 ± 0.08
[29] in 2021	ViT	307	95.86 ± 0.28	None	93.83 ± 0.46	None
[30] in 2021		46	95.54 ± 0.18	98.48 ± 0.06	93.06 ± 0.11	95.56 ± 0.20
[31] in 2022		28	96.61 ± 0.07	98.08 ± 0.03	93.90 ± 0.07	95.29 ± 0.12
[47] in 2022		40	95.56 ± 0.17	96.98±0.16	92.72±0.04	94.66±0.10
[32] in 2022	A self-designed CNN	0.5	93.15 ± 0.25	97.31 ± 0.10	92.02 ± 0.50	94.39 ± 0.16
[34] in 2022	CNNs combined with ViT	Not mentioned	96.25 ± 0.10	97.70 ± 0.11	93.90 ± 0.14	95.40 ± 0.15
[35] in 2023	Decoupling CNNs	Not mentioned	86.52 ± 0.18	None	84.81 ± 0.36	91.41 ± 0.69
[36] in 2022	CNN using clustering	Not mentioned	91.32^{*2}	None	91.96^{*2}	None
[37] in 2022	CNNs using adaptive transform	16	94.55 ± 0.27	96.72 ± 0.23	90.25 ± 0.14	93.05 ± 0.12
[38] in 2022	CNN combined with ViT	Not mentioned	95.58	96.88	92.72	94.50
RE-EfficientNet [this work]	Single EfficientNet-B3	12	97.11 ± 0.06	98.15 ± 0.10	94.60 ± 0.05	96.15 ± 0.03

*1: The methods used a testing ratio of 30% on AID and 70% on NWPU.

*2: The method only used a testing ratio of 20% on AID and NWPU, respectively.

Therefore, all the results validate the effectiveness and superiority of RE-EfficientNet. Meanwhile, based on the comparable results and analysis, it is revealed that the systematic reasons related to the leading performance of RE-EfficientNet come from several aspects. First, the attention modules in EfficientNet-B3 provide a great chance for RE-EfficientNet to achieve better performance if compared to the other classical CNNs without attention. Second, the unmodified EfficientNet-B3 in RE-EfficientNet avoids an expanded search space and can effectively take advantage of better starting points in the search space provided by pre-training on ImageNet-1K. Third, RE-EfficientNet has well handled the data distribution shift from DA by using different effective combinations of transformations. Fourth, the modified TTs according to the nature of RSI are more effective than the simple usage in previous studies. Fifth, compared to ViTs, the inductive biases of CNNs have enhanced RE-EfficientNet's performance for classifying the smaller RSI sets like AID and NWPU.

4.3. Confusion matrixes

The confusion matrixes for AID and NWPU are presented in Figures 7 and 8, while the most confusing categories are marked with a red arrow. As concluded before, RE-EfficientNet shows more obvious leads when the training dataset is smaller. Hence, to get a good understanding of how RE-EfficientNet works, this paper selectively shows the matrixes of AID and NWPU with the same TR of 20%.

As shown in Figure 7, the confusion mainly occurs in the categories including center, church, industry, park, resort, school and square. Meanwhile, except for medium residential, which has an OA of 96.60%, all the OAs of other classes are higher than the mean value of 97.11%. Comparing the confusion result to the feature fusion methods [11,24,25], we can see remarkable improvements in all categories. However, making the same comparison to another fusion method [10], we can find that RE-EfficientNet still shows obvious leads in most categories, excluding resort, school and square. More specifically, in this paper, the OA result of the three classes is less than the results in [10]. This fact proves that feature fusion may have a positive impact on CNN's performance in a certain class. With a close look at the CNN methods with attention modules [13,17,33], we can see that RE-EfficientNet also shows clear leads in all categories. However, when looking at the ViTs [30,38], we will find another contradiction of confusing categories. In detail, in [30], the most confusing classes are bare land and desert, with OAs of 87% and 51%, respectively. Yet, in [38], the most confusing classes are the same as RE-EfficientNet, with a slight OA gap. Because the other ViT methods [29,31] lack related information, we cannot further analyze this deviation. In addition, among the other methods providing necessary information, the CNN ensemble [18] and model combination [28] also have suboptimal performance compared to RE-EfficientNet. In particular, in [18], the ensemble shows the same partial improvements on three categories as [10] does and also has worse performance on the other categories.

As shown in Figure 8, at the 20% TR of NWPU, the most confusing categories include the church, dense residential, industry, medium residential, palace, rectangular farmland and wetland, with OA less than 93%. Meanwhile, another ten categories have an OA greater than 93% but below the mean value of 96.15%, including commercial area, desert, freeway, island, lake, meadow, mountain, railway, railway station and runaway. Comparing the RE-EfficientNet's results to the other methods [10,11,13,24,25,30,33,38], we can see an overall improvement for all the most confusing categories, and the temporary lead on AID in [10] does not exist on NWPU. However, in [18], the CNN ensemble's most confusing categories are quite different from all others, including sparse residential, forest and overpass, with OA less than 90%. Yet, the OAs of these three categories in RE-EfficientNet are all above 96.30%. Hence, this contradiction still proves the local-optimal-solution problem in deep learning.

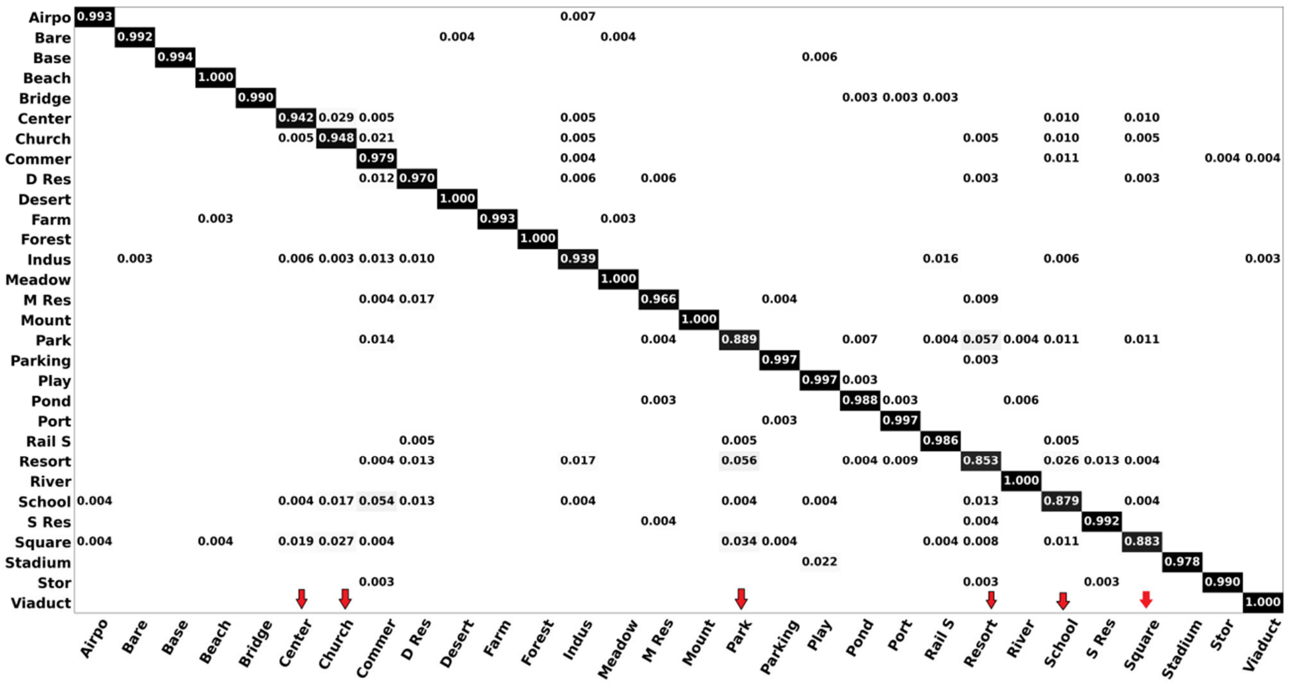

Figure 7. The confusion matrix of AID at a 20% TR.

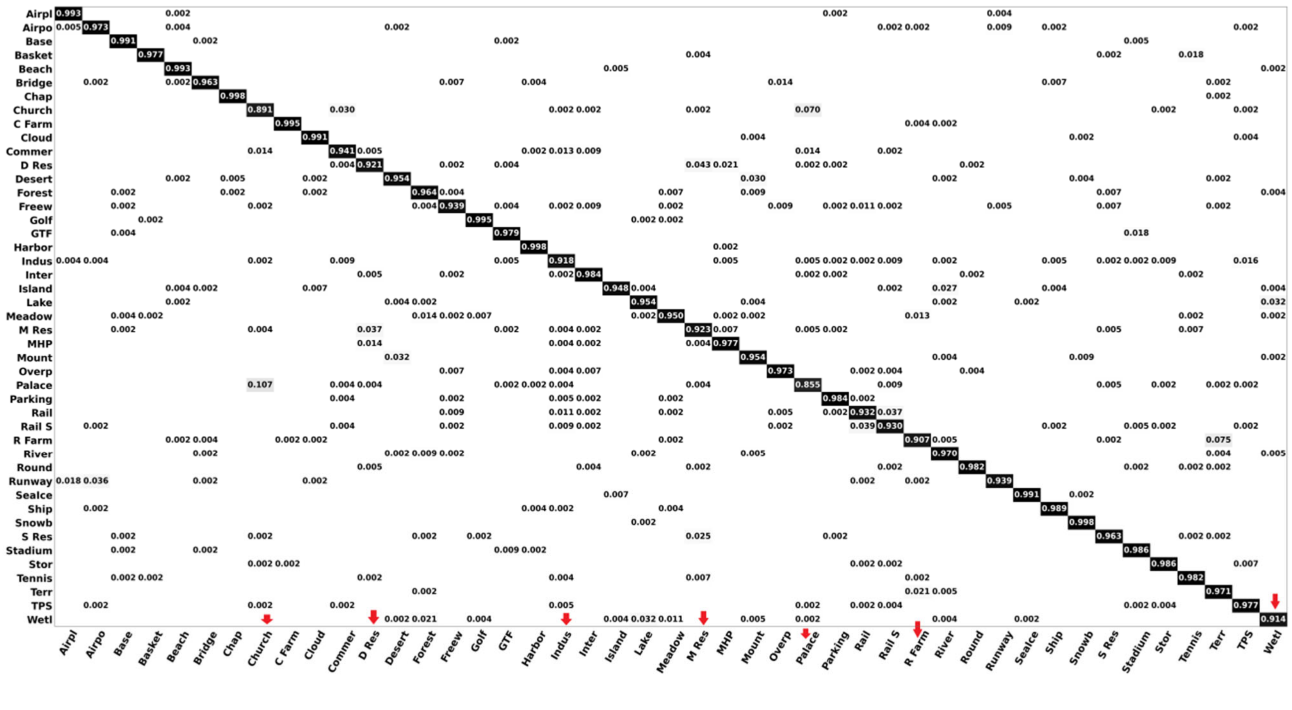

Figure 8. The confusion matrix of NWPU at a 50% TR.

In summary, based on all the comparisons, we can draw two conclusions. First, comparing the previous studies, RE-EfficientNet does perform better on AID and NWPU with fewer training samples, and the improvements are clear for all categories. Second, some previous methods have found partial advances for several confusing categories when the RSI set is small. However, the improved performance will also disappear if the RSI set becomes larger.

4.4. Changes in m-OLS labels

To understand the difference in regularization between RSI and ImageNet-1K, this work also studied the changes in soft labels modified by the m-OLS, and the results are shown in Figure 9. In the figure, the category number has the same definition as in Figures 4 and 5, while "Step" is also the same as in Figure 2.

As shown in Figure 9a, we can see three facts. First, with a 20% TR, the smallest category labels in Step 1 are school and resort. In contrast, the smallest labels in Step 2 are the railway station and storage tank, and all 30 category labels share a small gap of less than 0.1. Second, with a 50% TR, the school and resort in Step 1 still have the smallest label values, but the values are obviously higher than the ones of the 20% TR. However, the categories with the smallest label values in Step 2 change to the church and school, corresponding to the confusing categories in Figure 7. Third, we can see that all the labels of Step 2 are obviously less than the ones of Step 1, while the values of Step 1 show greater volatility with a 20% TR. This phenomenon is mainly because the m-OLS algorithm dynamically updates label values according to the model's prediction results for training sets (see [41]).

Similarly, as shown in Figure 9b, we can see two facts. First, all 45 label values of Step 1 show greater volatility and are smaller than the values of Step 2. Second, the categories with the smallest label, including the church, medium residential and palace, are the same in Steps 1 and 2 but still correspond to the confusing categories shown in Figure 8. Taking Figure 10 as a whole, we can find that the m-OLS value shows smoother fluctuations as the sum of training samples increases. This phenomenon is also because the model's increasing prediction accuracy on training sets has caused the m-OLS algorithm to make slight changes to the category's similarity.

Based on all the facts, we can understand that the m-OLS is an adaptive soft label generator based on the similarity of training sets. It can prevent a CNN from being overconfident based on hard labels. Meanwhile, comparing these m-OLS labels to the ones for nature images shown in [41], we can see a larger gap of approximately 0.2. Hence, we should modify the usage of OLS formed in ImageNet-1K to exclude larger fluctuations first and then design an algorithm with proper training epochs to get the soft labels closer to the similarity distribution nature of RSI.

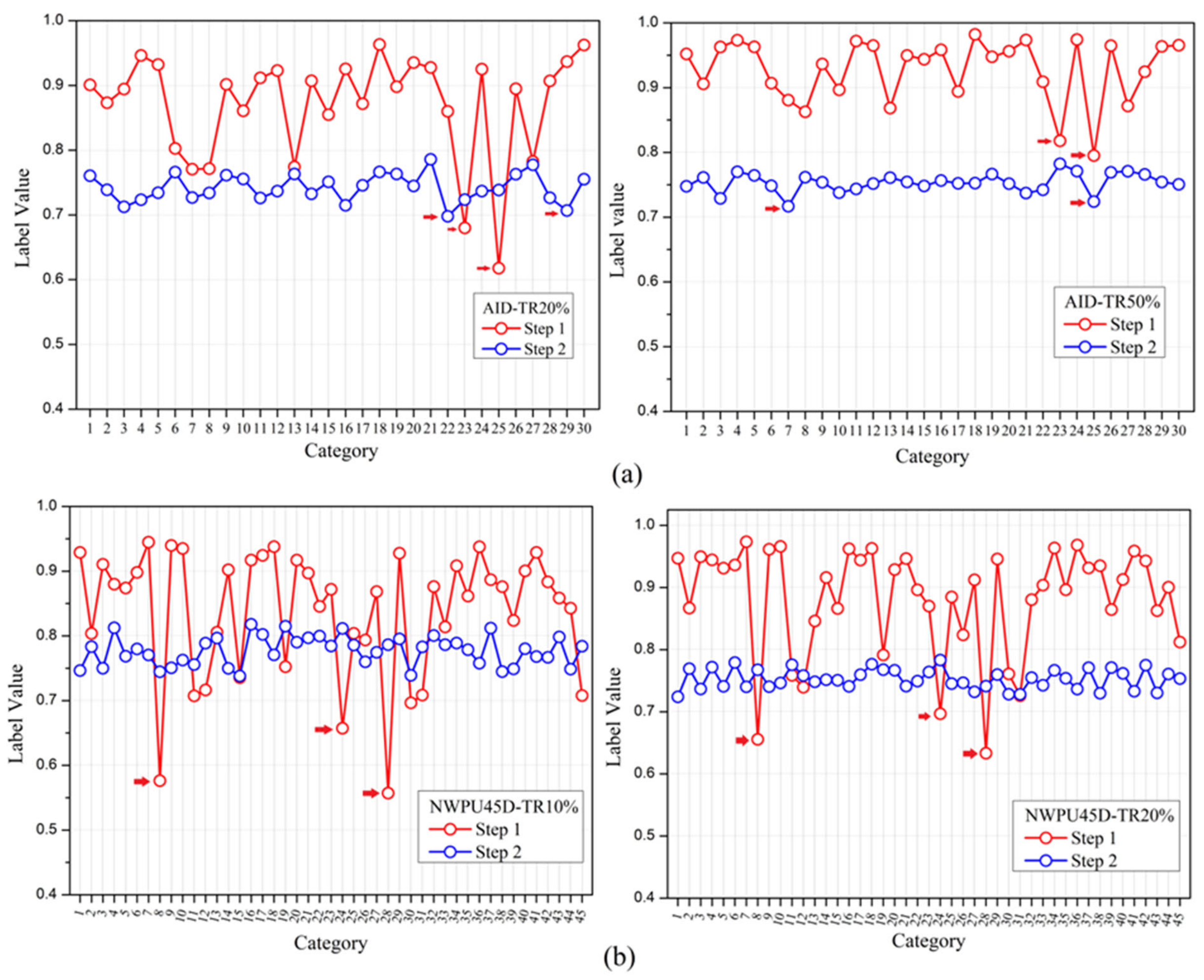

Figure 9. Changes in the category labels modified by m-OLS.

4.5. Ablation experiments

To validate the effectiveness of modified TTs in training, as shown in Table 4, this work performed ablation experiments for RE-EfficientNet. In the table, a single selected baseline denotes the same training procedures described in Algorithm 1 but excludes m-OLS and m-CutMix. Similarly, with the baseline and m-OLS chosen at the same time, it shows that the training procedures have the m-OLS activated. In other words, the last row of Table 4 represents the entire training procedure described in Algorithm 1.

Table 4. OAs (%) of ablation experiments with different combinations of DARS.

DA and regularization			AID		NWPU	
Baseline	m-OLS	m-CutMix	TR20%	TR50%	TR10%	TR20%
✓			96.09 ± 0.16	97.61 ± 0.13	93.52 ± 0.10	95.37 ± 0.06
✓	✓		96.41 ± 0.01	97.74 ± 0.02	93.89 ± 0.06	95.66 ± 0.11
✓		✓	96.84 ± 0.08	98.08 ± 0.02	94.38 ± 0.07	96.03 ± 0.04
✓	✓	✓	97.11 ± 0.06	98.15 ± 0.10	94.60 ± 0.05	96.15 ± 0.03

As shown in the first row of Table 4, we can see that the baseline training strategy can also let a single EfficientNet-B3 model achieve competitive performance compared to the other previous methods shown in Table 1. Comparing the baseline result to that of the more lightweight EfficientNet-B0 in Table 1, however, we can find no obvious improvements, though the EfficientNet-B3 has a larger capacity. This reveals that, without proper regularization, a CNN model is easily overfitted on the same RSI dataset. As shown in the second row of Table 4, with m-OLS activated, a single EfficientNet-B3 can achieve clearly improved OAs by 0.10% to 0.30%. The advances are more visible on those smaller TRs. As shown in the third row of Table 4, when m-CutMix is active, a single EfficientNet-B3 model can gain more OA by 0.3% to 0.7%, while similarly, the improvements are clearer in the scenario with smaller TRs. At the end, we can see that RE-EfficientNet can achieve remarkable improved OAs by 0.55% to 1.1% compared to the baseline model trained without TTs.

4.6. Visualization and analysis

4.6.1. Class activation mapping

To gain a deeper understanding of RE-EfficientNet, we employed the gradient class activation mapping (CAM) technique [43] to generate visualizations for different cut-and-mixed samples, and the related images are shown in Figure 10. Ten different categories are included in the CAM visualization. In Figure 10, the label of each column at the top corresponds to a certain category, while the pair of two labels surrounded by a red rectangle means the two categories were cut and mixed with each other. Note that all images are separated into five rows: The first and third are the original and cut-and-mixed images; the second is the CAM of the original; and the fourth and fifth are the CAM of the cut-and-mixed, with a prediction corresponding to the first and second label in each label pair.

As shown in the first and second rows of Figure 10, the brighter part of the CNN's CAM, also known as the activation part with more important information, agrees with human semantic cognitive patterns if the RSI scene has a salient ground object: e.g., the waves of a beach, the station of a railway station, the court of a basketball court and so on. The result proves that the model has learned to classify different samples based on invariant features that belong to a certain category. However, without a salient object, we can see that the activation becomes wider and relatively darker. This result still agrees with humans because we also classify this scene through its global pattern, though this recognition pattern is not as discriminative as a salient ground object is. Looking at the cut-and-mixed images, as shown in the fourth and fifth rows of Figure 10, we can find two facts as follows:

First, for the fourth row with a prediction of a first category, its activation is almost the same as the one in the second row, while the activation also gets distracted if the cut-and-mixed image patch closes to its original activating location. Second, for the last row with a prediction of the patch's category, its activation also focuses on the image patch if the patch is part of the activation location shown in the second row.

Hence, based on the CAM results, we can tell that RE-CNN has helped the model focus its attention on the salient object of the RSI scene. Meanwhile, with a restrained cut-and-mix operation, the model's attention has not obviously been distracted. However, if the cut-and-mix operation frequently happens, this learned attention will very likely be weakened, letting the model be suboptimal. Taking this fact into account, the proposed modification for CutMix is quite reasonable.

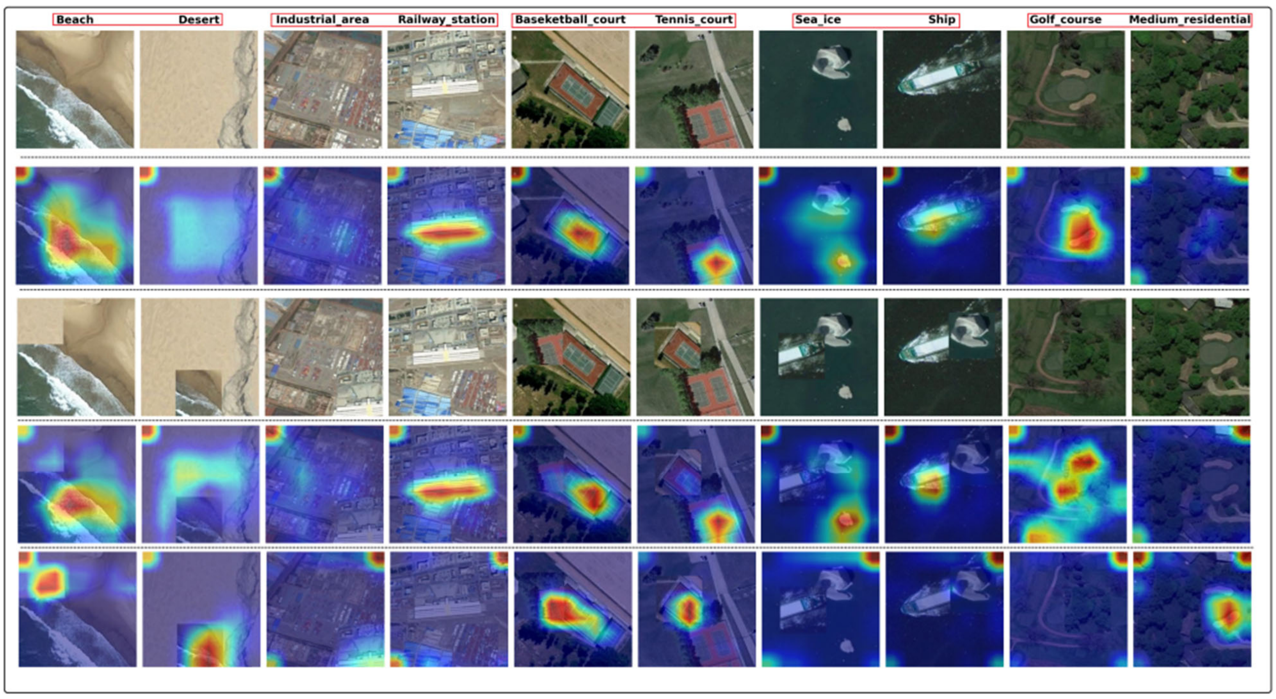

Figure 10. Gradient class activation mapping visualizations.

4.6.2. *Stochastic neighbor embedding*

To validate the effectiveness of extracted features, this paper employs the t-Distributed Stochastic Neighbor Embedding (t-SNE) [51] technique to intuitively visualize the similarity of classified samples, and the contrastive results are shown in Figure 11. Two EfficientNet-B3 models are involved in the comparison, while the left one is trained with a combination of m-OLS and m-CutMix, but the right one is not.

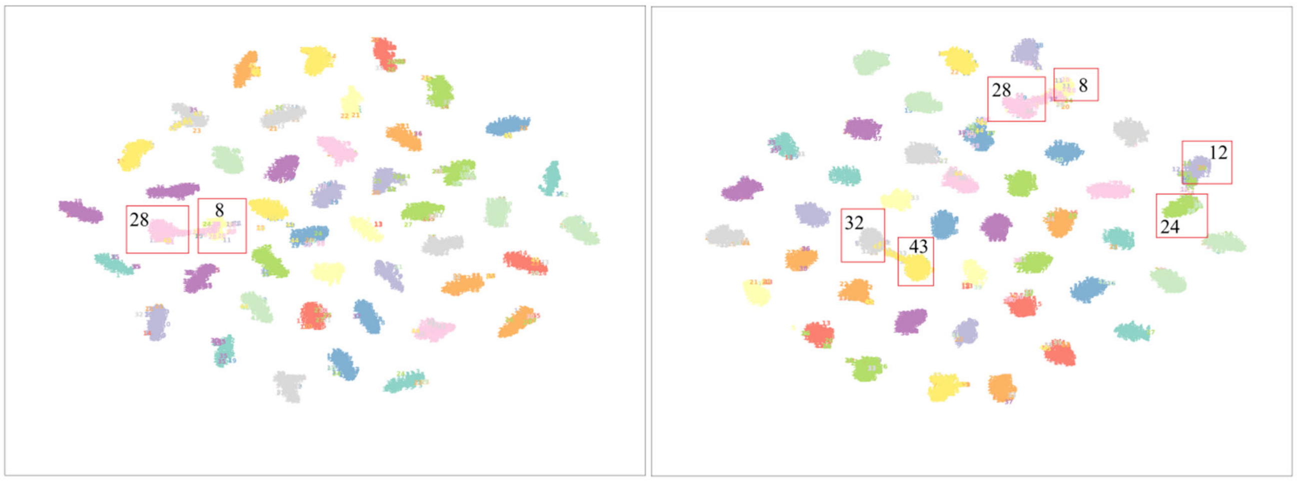

Figure 11. Visualization of the category's similarity through t-SNE.

In Figure 11, each category of NWPU consists of two hundred samples selected at random, while the category name is the same as in Figure 5. As shown in Figure 11, most of the categories are clearly separated from each other. This reveals that the features extracted by RE-EfficientNet are effective. A quick glance at Figure 11 shows that the right model has more overlapping categories (marked with red rectangles). More specifically, the overlapping pairs on the right include the church with palace, the dense residential with medium residential and the rectangular farm with terrace, while the left only consists of the church with palace. Comparing the result to the confusion matrix, we can see consistency. Similarly, this result also clearly visualizes the performance gap between the different training strategies. Comparing the t-SNE visualization to those in previous studies [14,18,33,35,38], we can see that this work shows a better separation of categories. Hence, the t-SNE visualization also proves the effectiveness and superiority of RE-EfficientNet.

5. Discussion

To date, the overwhelming majority of proposed methods for RSI-SC employ deep models pre-trained on natural image datasets. The main reason lies in the fact that the RSI research domain lacks large-scale datasets. However, there are still researchers who claim to be developing and trying to release larger-scale RSI datasets. For example, in [31], Wang et al. have tried to train a ResNet-50 CNN model and a ViT from scratch on an RSI dataset called Million-AID [52], which has about a million images cropped from Google Earth. Using the pre-trained weights on Million-AID, [31] did the classifying tests on AID and NWPU, and the results are shown in Table 5.

Table 5. OA (%) comparison of different models pre-trained on natural images and RSI.

Methods	Roadmap	Parameter size (M)	AID		NWPU	
			20%-TR	50%-TR	10%-TR	20%-TR
Wang et al. [31]	A single Resnet-50 pre-trained on Million-AID	24	96.81 ± 0.03	97.89 ± 0.08	93.93 ± 0.10	95.02 ± 0.06
	A single Vision Transformer pre-trained on Million-AID	19	96.91 ± 0.06	98.22 ± 0.09	94.41 ± 0.11	95.60 ± 0.06
RE-EfficientNet (This work)	A single EfficientNet-B3 pre-trained on ImageNet-1K	12	97.11 ± 0.06	98.15 ± 0.10	94.60 ± 0.05	96.15 ± 0.03

As shown in Table 5, with pre-trained weights on Million-AID, the ViT shows clearly improved OAs compared to the one using pre-trained weights on ImageNet-1K. Compared to the ViT pre-trained on Million-AID, we can see that RE-EfficientNet still performs better, though the OA gaps are slightly reduced. Compared to the ResNet-50 pre-trained on Million-AID, however, we can see that RE-EfficientNet performs much better with improved OAs by 0.26% to 1.13%. As mentioned before, ViTs commonly need more training samples to perform better. Therefore, the result indicates that the time-consuming pre-training in [31] still has not eliminated the performance gap in RSI sets if compared to RE-EfficientNet. Meanwhile, the ResNet-50 in [31] lacks attention modules, and its training algorithm excludes effective combinations of DA transformations and TTs, resulting in poor performance compared to RE-EfficientNet.

Based on the basic rule in deep learning, a CNN model pre-trained on a large-scale RSI set has a big chance of achieving better performance because the domain gaps are larger in ImageNet-1K. Currently, only 10,000 samples of the open Million-AID have category information. Hence, if possible, the authors will test this promising pre-training strategy on large-scale RSI datasets in the future.

6. Conclusions

As deep learning techniques quickly dominate the computer vision task of RSI, a lot of creative CNN and ViT methods have been proposed for SC. Recently, researchers have proposed many effective but costly methods to seek ideal performance. However, there is still a better way to achieve a simple, lightweight and accurate solution.

In the beginning, we presented a set of deep and reasonable studies on theoretical analysis and attempts to explain why it is difficult for those complex and costly methods to achieve ideal solutions as expected. Then, we selected an easy way to gain competitive performance for RSI-SC by using a single CNN and beginner-friendly transfer learning. The proposed RE-Efficient only includes a lightweight EfficientNet-B3 with 12 M parameters. The RE-CNN training algorithm only consists of routine DA transformations and two modified TTs. Experimental results on AID and NWPU prove that RE-Efficient can perform much better. It presents remarkable OA improvements of 0.5% to 0.75% over the leading one among the 30 SOTA methods published before 2023. The ablation experimental results also demonstrate that the proposed combinations of DA transformations and modified TTs are effective. It can boost EfficientNet-B3's performance with improved OAs by 0.55% to 1.1%. Since RE-Efficient only consists of an open-source model and training algorithm, it is easier to reproduce. Given its fewer parameters and transfer-leaning strategy, RE-EfficientNet is more efficient and beginner-friendly. The performance of RE-EfficientNet reveals that a single CNN can perform better for RSI-SC if the whole pipeline and training algorithm is effective. In addition, the authors argue that the proposed mathematical derivation can help us develop more efficient methods for RSI-SC in the future.

There are still some shortcomings in this work. First, RE-EfficientNet may perform better if it has been pre-trained on a large-scale RSI set. Second, the effectiveness of the RE-CNN algorithm for other classical CNNs has not been verified. The authors will make progress on these prospects in the future.

Use of AI tools declaration

The authors declare that no artificial intelligence (AI) tool was used in the creation of this article.

Acknowledgments

Thanks to Hunan University of Arts and Science for doctoral research funding for this study (grant number 16BSQD23).

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. A. P. Plageras, K. E. Psannis, C. Stergiou, H. Wang, B. B. Gupta, Efficient IoT-based sensor BIG Data collection–processing and analysis in smart buildings. *Future Gener Comput Syst*, **82** (2018), 349–357. https://doi.org/10.1016/j.future.2017.09.082
2. Z. Ahmed, M. Ayaz, M. A. Hijji, M. Z. Abbas, A. Rahim, AUV-Based efficient data collection scheme for underwater linear sensor networks. *Int J Semant Web Inf Syst*, **18** (2022), 1–19. https://doi.org/10.4018/IJSWIS.299858
3. D. Tian, Y. Han, B. Wang, T. Guan, H. Gu, W. Wei, Review of object instance segmentation based on deep learning, *J. Electron. Imag.*, **31** (2021), 041205. https://doi.org/10.1117/1.JEI.31.4.041205
4. K. S. Arikumar, A. D. Kumar, T. R. Gadekallu, S. B. Prathiba, K. Tamilarasi, Real-Time 3D Object detection and classification in autonomous driving environment using 3D LiDAR and camera sensors, *Electronics*, **11** (2022), 4203. https://doi.org/10.3390/electronics11244203
5. H. Song, A more efficient approach for remote sensing image classification, *Comput. Mater. Contin.*, **74** (2023), 5741–5756. https://doi.org/10.32604/cmc.2023.034921
6. H. Song, FST-EfficientNetV2: exceptional image classification for remote sensing, *Comput. Sci. Eng.*, **46** (2023), 3959–3978. https://doi.org/10.32604/csse.2023.038429
7. H. Touvron, A. Vedaldi, M. Douze, H. Jégou, Fixing the train-test resolution discrepancy, arXiv: 1906.06423, [preprint], (2019) [cited 2023 September 05]. Available from: http://arxiv.org/abs/1906.06423
8. H. Song, A Leading but Simple Classification Method for Remote Sensing Images, *AETiC*, **7** (2023), 1–20. https://doi.org/10.33166/AETiC.2023.03.001
9. M. Tan, Q. V. Le, EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, arXiv:1905.11946, [preprint], (2019) [cited 2023 September 05]. Available from: http://arxiv.org/abs/1905.11946
10. G. Cheng, C. Yang, X. Yao, L. Guo, J. Han, When deep learning meets metric learning: remote sensing image scene classification via learning discriminative CNNs, *IEEE Trans Geosci Remote Sens*, **56** (2018), 2811–2821. https://doi.org/10.1109/TGRS.2017.2783902
11. Y. Guo, J. Ji, X. Lu, H. Huo, T. Fang, D. Li, Global-Local attention network for aerial scene classification, *IEEE Access*, **7** (2019), 67200–67212. https://doi.org/10.1109/ACCESS.2019.2918732
12. R. Minetto, M. P. Segundo, S. Sarkar, Hydra: An ensemble of convolutional neural networks for geospatial land classification, *IEEE Trans Geosci Remote Sens*, **57** (2019), 6530–6541. https://doi.org/10.1109/TGRS.2019.2906883
13. W. Tong, W. Chen, W. Han, X. Li, L. Wang, Channel-Attention-Based DenseNet network for remote sensing image scene classification, *IEEE J Sel Top Appl Earth Obs Remote Sens*, **13** (2020), 4121–4132. https://doi.org/10.1109/JSTARS.2020.3009352
14. J. Li, D. Lin, Y. Wang, G. Xu, C. Ding, Deep discriminative representation learning with attention map for scene classification, arXiv:1902.07967, [preprint], (2019) [cited 2023 September 05]. Available from: http://arxiv.org/abs/1902.07967

15. H. Alhichri, A. S. Alswayed, Y. Bazi, N. Ammour, N. A. Alajlan, Classification of remote sensing images using efficientnet-b3 cnn model with attention, *IEEE Access*, **9** (2021), 14078–14094. https://doi.org/10.1109/ACCESS.2021.3051085
16. B. Li, Y. Guo, J. Yang, L. Wang, Y. Wang, W. An, Gated recurrent multiattention network for VHR remote sensing image classification, *IEEE Trans Geosci Remote Sens*, **60** (2022), 1–13. https://doi.org/10.1109/TGRS.2021.3093914
17. W. Chen, S. Ouyang, W. Tong, X. Li, X. Zheng, L. Wang, GCSANet: A global context spatial attention deep learning network for remote sensing scene classification, *IEEE J Sel Top Appl Earth Obs Remote Sens*, **15** (2022), 1150–1162. https://doi.org/10.1109/JSTARS.2022.3141826
18. Q. Zhao, Y. Ma, S. Lyu, L. Chen, Embedded Self-Distillation in compact multibranch ensemble network for remote sensing scene classification, *IEEE Trans Geosci Remote Sens*, **60** (2022), 1–15. https://doi.org/10.1109/TGRS.2021.3126770
19. H. Song, A consistent mistake in remote sensing images' classification literature, *Intell. Autom. Soft Comput.*, **37** (2023), 1381–1398. https://doi.org/10.32604/iasc.2023.039315
20. S. Chaib, H. Liu, Y. Gu, H. Yao, Deep feature fusion for vhr remote sensing scene classification, *IEEE Trans Geosci Remote Sens*, **55** (2017), 4775–4784. https://doi.org/10.1109/TGRS.2017.2700322
21. Y. Liu, C. Y. Suen, Y. Liu, L. Ding, Scene classification using hierarchical wasserstein CNN, *IEEE Trans Geosci Remote Sens*, **57** (2019), 2494–2509. https://doi.org/10.1109/TGRS.2018.2873966
22. Y. Liu, Y. Liu, L. Ding, Scene classification by coupling convolutional neural networks with wasserstein distance, *IEEE Geosci. Remote Sensing Lett.*, **16** (2019), 722–726. https://doi.org/10.1109/LGRS.2018.2883310
23. Y. Bazi, M. M. Al Rahhal, H. Alhichri, N. Alajlan, Simple yet effective fine-tuning of deep CNNs using an auxiliary classification loss for remote sensing scene classification, *Remote Sensing*, **11** (2019), 2908. https://doi.org/10.3390/rs11242908
24. W. Zhang, P. Tang, L. Zhao, Remote sensing image scene classification using CNN-CapsNet, *Remote Sensing*, **11** (2019), 494. https://doi.org/10.3390/rs11050494
25. J. Xie, N. He, L. Fang, A. Plaza, Scale-Free convolutional neural network for remote sensing scene classification, *IEEE Trans Geosci Remote Sens*, **57** (2019), 6916–6928. https://doi.org/10.1109/TGRS.2019.2909695
26. H. Sun, S. Li, X. Zheng, X. Lu, Remote sensing scene classification by gated bidirectional network, *IEEE Trans Geosci Remote Sens*, **58** (2020), 82–96. https://doi.org/10.1109/TGRS.2019.2931801
27. D. Guo, Y. Xia, X. Luo, Scene classification of remote sensing images based on saliency dual attention residual network, *IEEE Access*, **8** (2020), 6344–6357. https://doi.org/10.1109/ACCESS.2019.2963769
28. X. Tang, Q. Ma, X. Zhang, F. Liu, J. Ma, L. Jiao, Attention consistent network for remote sensing scene classification, *IEEE J Sel Top Appl Earth Obs Remote Sens*, **14** (2021), 2030–2045. https://doi.org/10.1109/JSTARS.2021.3051569
29. Y. Bazi, L. Bashmal, M. M. A. Rahhal, R. A. Dayil, N. A. Ajlan, Vision transformers for remote sensing image classification, *Remote Sensing*, **13** (2021), 516. https://doi.org/10.3390/rs13030516
30. J. Zhang, H. Zhao, J. Li, TRS: transformers for remote sensing scene classification, *Remote Sensing*, **13** (2021), 4143. https://doi.org/10.3390/rs13204143

31. D. Wang, J. Zhang, B. Du, G. S. Xia, D. Tao, An empirical study of remote sensing pretraining, arXiv: 2204.02825, [preprint], (2022) [cited 2023 September 06]. Available from: http://arxiv.org/abs/2204.02825
32. C. Shi, X. Zhang, J. Sun, L. Wang, Remote sensing scene image classification based on Self-Compensating convolution neural network, *Remote Sensing*, **14** (2022), 545. https://doi.org/10.3390/rs14030545
33. S. B. Chen, Q. S. Wei, W. Z. Wang, J. Tang, B. Luo, Z. Y. Wang, Remote sensing scene classification via Multi-Branch local attention network, *IEEE Trans. on Image Process.*, **31** (2022), 99–109. https://doi.org/10.1109/TIP.2021.3127851
34. P. Deng, K. Xu, H. Huang, When CNNs meet vision transformer: a joint framework for remote sensing scene classification, *IEEE Geosci. Remote Sensing Lett.*, **19** (2022), 1–5. https://doi.org/10.1109/LGRS.2021.3109061
35. W. Miao, J. Geng, W. Jiang, Multigranularity decoupling network with pseudolabel selection for remote sensing image scene classification, *IEEE Trans Geosci Remote Sens*, **61** (2023), 1–13. https://doi.org/10.1109/TGRS.2023.3244565
36. H. Song, W. Yang, GSCCTL: a general semi-supervised scene classification method for remote sensing images based on clustering and transfer learning, *Int J Remote Sens.*, **43** (2022), 5976–6000. https://doi.org/10.1080/01431161.2021.2019851
37. W. Wang, Y. Chen, P. Ghamisi, Transferring CNN with adaptive learning for remote sensing scene classification, *IEEE Trans Geosci Remote Sens*, **60** (2022), 1–18. https://doi.org/10.1109/TGRS.2022.3190934
38. K. Xu, P. Deng, H. Huang, Vision transformer: an excellent teacher for guiding small networks in remote sensing image scene classification, *IEEE Trans Geosci Remote Sens*, **60** (2022), 1–15. https://doi.org/10.1109/TGRS.2022.3152566
39. T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, M. Li, Bag of tricks for image classification with convolutional neural networks, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, (2019), 558–567.
40. S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, Y. Yoo, CutMix: Regularization strategy to train strong classifiers with localizable features, *Proceedings of the IEEE/CVF international conference on computer vision*, (2019), 6023–6032.
41. C. B. Zhang, P. T. Jiang, Q. Hou, Y. Wei, Q. Han, Z. Li, et al., Delving deep into label smoothing, *IEEE Trans. on Image Process.*, **30** (2021), 5984–5996. https://doi.org/10.1109/TIP.2021.3089942
42. J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, Squeeze-and-Excitation Networks, *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2018), 7132–7141.
43. R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-CAM: visual explanations from deep networks via Gradient-Based localization, *Proceedings of the IEEE international conference on computer vision*, (2017), 618–626.
44. I. Loshchilov, F. Hutter, Decoupled weight decay regularization, arXiv: 1711.05101v3, [preprint], (2017) [cited 2023 September 05]. Available from: http://arxiv.org/abs/1711.05101v3
45. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, 2818–2826.

46. H. Zhang, M. Cisse, Y. N. Dauphin, D. Lopez-Paz, Mixup: beyond empirical risk minimization, arXiv: 1710.09412, [preprint], (2017) [cited 2023 September 05]. Available from: http://arxiv.org/abs/1710.09412
47. Y. H. Liu, E. Sangineto, W. Bi, N. Sebe, B. Lepri, M. Nadai, Efficient training of visual transformers with small datasets, *NIPS*, **34** (2021), 23818–23830.
48. A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, H. Shi, Escaping the big data paradigm with compact transformers, arXiv: 2104.05704, [preprint], (2021) [cited 2023 September 05]. Available from: http://arxiv.org/abs/2104.05704
49. Z. Li, F. Liu, W. Yang, S. Peng, J. Zhou, A survey of convolutional neural networks: analysis, applications, and prospects, *IEEE Trans. Neural Netw. Learning Syst.*, **33** (2022), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827
50. F. Liu, D. Chen, Z. Guan, X. Zhou, J. Zhu, J. Zhou, RemoteCLIP: A vision language foundation model for remote sensing, arXiv: 2306.11029, [preprint], (2023) [cited 2023 September 05]. Available from: http://arxiv.org/abs/2306.11029
51. L van der Maaten, G. Hinton, Visualizing data using t-SNE, *J Mach Learn Res*, **9** (2008), 2579–2605.
52. Y. Long, G. S. Xia, S. Li, W. Yang, M. Y. Yang, X. X. Zhu, et al., On creating benchmark dataset for aerial image interpretation: reviews, guidances, and Million-AID, *IEEE J Sel Top Appl Earth Obs Remote Sens*, **14** (2021), 4205–4230. https://doi.org/10.1109/JSTARS.2021.3070368