Article

RSCNet: An Efficient Remote Sensing Scene Classification Model Based on Lightweight Convolution Neural Networks

Zhichao Chen $^{1,2}$, Jie Yang $^{1,3,*}$, Zhicheng Feng $^{1,2}$ and Lifang Chen $^{4}$

1 Department of Electrical Engineering and Automation, Jiangxi University of Science and Technology, Ganzhou 341000, China
2 Jiangxi Provincial Key Laboratory of Maglev Technology, Ganzhou 341000, China
3 Ganjiang Innovation Academy, Chinese Academy of Sciences, Ganzhou 341000, China
4 Department of Science, Jiangxi University of Science and Technology, Ganzhou 341000, China
* Correspondence: yangjie@jxust.edu.cn

Abstract: This study aims at improving the efficiency of remote sensing scene classification (RSSC) through lightweight neural networks and to provide a possibility for large-scale, intelligent and real-time computation in performing RSSC for common devices. In this study, a lightweight RSSC model is proposed, which is named RSCNet. First, we use the lightweight ShuffleNet v2 network to extract the abstract features from the images, which can guarantee the efficiency of the model. Then, the weights of the backbone are initialized using transfer learning, allowing the model to learn by drawing on the knowledge of ImageNet. Second, to further improve the classification accuracy of the model, we propose to combine ShuffleNet v2 with an efficient channel attention mechanism that allows the features of the input classifier to be weighted. Third, we use a regularization technique during the training process, which utilizes label smoothing regularization to replace the original loss function. The experimental results show that the classification accuracy of RSCNet is 96.75% and 99.05% on the AID and UCMerced_LandUse datasets, respectively. The floating-point operations (FLOPs) of the proposed model are only 153.71 M, and the time spent for a single inference on the CPU is about 2.75 ms. Compared with existing RSSC methods, RSCNet achieves relatively high accuracy at a very small computational cost.

Keywords: remote sensing; classification; lightweight neural networks; attention mechanism; artificial intelligence; transfer learning

1. Introduction

In recent years, as the remote sensing imaging technology develops, the resolution of remote sensing image (RSI) keeps getting higher and higher, and the utilization of RSI has been widely concerned [1–3]. RSI contains rich texture features and scene semantic information, which is of great application value in agricultural production, disaster warning, national defense security, etc., [4]. RSSC technology is of great importance to the interpretation and understanding of RSI and is a crucial research branch in RSI processing. However, the RSSC algorithm still faces the following two challenges. In terms of algorithm accuracy, the characteristics of RSI with many scene categories and high similarity between categories, makes the RSSC is challenging [5]. In terms of algorithm efficiency, when faced with massive remote sensing data, computing equipment needs to perform large-scale, intelligent, and real-time calculations [6,7]. The high complexity of the models relies on high performance computing devices, resulting in significant computational costs. Therefore, it is particularly important to consider the accuracy and computational efficiency of the classification algorithm in the study of RSSC.

Currently, deep learning has made significant achievements in computer vision tasks [8–10], which is the critical technology to realizing artificial intelligence. In computer vision tasks, deep learning mainly uses convolution neural network (CNN) to build models
and automatically updates the model parameters based on the gap with the target during the model training phase. There exist many typical convolution neural networks, such as the residual structure-based ResNet [11], the dense connection-based DenseNet [12], and the neural structure search technology-based MnasNet [13]. Additionally, there is SO-UNet [14,15] networks constructed based on the encoder-decoder and self-supervised learning. CNN models have refreshed the list of classification tasks repeatedly. However, the scale of these models is also increasing, making increasingly high demands on the computational resources of the devices. With the purpose of ensuring computational efficiency, researchers turned to design the lightweight models. In 2016, Iandola et al. [16] proposed the first lightweight model, which was SqueezeNet. The classification accuracy of this model was close to AlexNet, and the number of parameters was only 1/510 of AlexNet. So far, there are a series of lightweight networks such as SqueezeNet, Xception [17], MobileNet [18,19], ShuffleNet [20], and EfficientNet [21]. The satisfactory performance of the lightweight model in terms of accuracy, stability, and speed provides theoretical and technical support for the implementation of efficient RSSC.

Large-scale deep neural networks with excellent performance are typically computationally intensive and often need to be run in computationally powerful GPU devices. This means that using these models requires a higher threshold for practical applications and it increases the cost. The purpose of this paper is to apply lightweight neural networks to RSSC. On the one hand, a lightweight ShuffleNet v2 [20] model suitable for CPU-level computing is chosen. On the other hand, it is optimized using transfer learning, attention mechanism and label smoothing regularization (LSR) techniques. The comprehensive performance of the proposed method for RSSC tasks exceeds that of many classical models, such as the typical large networks VGG16 [22], ResNet-50 [11], DenseNet-121 [12], and the lightweight networks SqueezeNet [16] and MobileNet v2 [19]. The following are the study’s primary contributions:

1. For the feature extraction network, we propose to combine a ShuffleNet v2 feature extractor with a channel attention mechanism and train it for the task of scene classification in remote sensing images.

2. In terms of training strategies, the proposed model makes two improvements, including transfer learning and LSR loss function. Transfer learning uses the knowledge of the model on big data to better initialize the model weights, which is conducive to improving the accuracy. The LSR takes into account to some extent the loss calculation in all dimensions and changes the optimization direction of the model.

3. This paper puts emphasis on model efficiency, which is recently gaining a lot of attention, not only in the growing field of “green AI”. The ablation and comparison experiments confirm the feasibility of the proposed model in RSSC.

The overall organization of this paper is as follows. Section 2 introduces the current research status of RSSC methods and the technical background of ShuffleNet v2. Section 3 describes the overall framework of the proposed RSCNet. Section 4 describes the dataset and the experimental environment. Section 5 presents the experimental results and analyzes the results. Finally, Section 6 concludes our paper and gives an outlook.

2. Related Work

Most of legacy RSSC methods are based on manual features, which are low-level visual attributes (color, texture, spectrum, etc.) extracted from images using various feature operators, such as scale invariant feature transform (SIFT) [23], histogram of oriented gradient (HOG) [24], local binary pattern (LBP) [25]. Ren et al. [26,27] extracted SIFT features of RSI and achieved a classification accuracy of 77.71% and 77.38% on the UCMerced_LandUse dataset, respectively. Ren et al. [28] achieved a classification accuracy of 88.20% on UCMerced_LandUse by optimizing high-dimensional LBP features. Xia et al. [29] conducted experiments on RSSC based on manual features, in which the classification accuracy of SIFT-based and LBP-based classification methods on the AID dataset was only 16.76% and 29.99%. It can be seen that manual features require large amounts of prior knowledge,
and hardly to describe feature objects with complex spatial distribution, so the classification accuracy is low.

Given the remarkable performance of CNNs for image classification tasks, there are a lot of researchers have achieved CNN-based RSSC [30–32]. Li et al. [33] fused the multilayer features of VGG16 to obtain the VGG-VD16 model. The model had an accuracy of 98.81% on the UCMerced_LandUse. In addition, combined with the feature layer of AlexNet, the model was able to improve the classification accuracy by 0.24%. Shawky et al. [34] used transfer learning and augmented fully connected layers to improve the Inception model, and the accuracy of the proposed method on UCMerced_LandUse was 99.86%. Tang et al. [35] proposed a dual-branch network ACNet with a classification accuracy of 95.38% on the AID dataset. Ma et al. [36] used an evolutionary algorithm to search for a large SceneNet_UCM model, which achieved 99.1% classification accuracy on the UCMerced_LandUse dataset. From the results of the RSSC study, the CNN-based classification method significantly outperforms the traditional manual features regarding the overall recognition accuracy and training difficulty. However, most approaches to designing CNN models are by adding network branches or stacking network modules. The pursuit of accuracy has led to oversized models, and there is less research and analysis on model complexity and actual deployment.

Ma et al. [20] proposed four guidelines for designing efficient lightweight models and proposed ShuffleNet v2. With the similar model complexity levels, the classification accuracy of ShuffleNet v2 can exceed MobileNet v2, DenseNet, Xception, and other models on the ImageNet dataset. Due to the efficiency of ShuffleNet v2, several studies [37–39] are based on this model to satisfy the accuracy and fast inference for recognition. Chen et al. [40] proposed an improved ShuffleNet v2 for garbage classification that achieved an accuracy of 97.9%. The accuracy of the model exceeded that of ResNet-101, and the computational cost was only about 1/30 of that of ResNet-101. Tang et al. [41] proposed a classification model of ShuffleNet v2 combined with squeeze and excitation (SE) attention mechanism, which achieves efficient grape disease classification for mobile devices. The proposed model achieved a classification accuracy of 95.28% and the hard disk storage consumption of the model was 5.3 MB. Therefore, ShuffleNet v2 could provide a new approach for RSSC.

Synthesizing the above problems and methods, on the basis of previous studies, this study further explores the design of a lightweight and efficient RSSC model based on lightweight ShuffleNet v2. In addition, the complexity and practical deployment of various models in RSSC tasks are conducted to examine and analyze.

3. Methodology
3.1. The Basics of ShuffleNet v2

Ma et al. [20] showed four design guidelines for efficient networks and improved ShuffleNet v1 to propose ShuffleNet v2. The corresponding four guidelines are as follows: (1) keeping the input and output channels equal during the convolution operation as much as possible; (2) excessive using group convolution also increases memory access cost (MAC); (3) branching and fragmented network structures lead to a decrease in the parallelism of the model, which slows down the inference; (4) element operations on feature maps cannot be ignored, and although the FLOPs for some element-level operations are small, the MAC is large.

The basic unit of ShuffleNet v2 is shown in Figure 1. Depthwise convolution (DW-Conv) [42], a special case of grouped convolution, which is usually followed by accessing a standard convolution of size 1 × 1 to form a depthwise separable convolution. The depthwise convolution is calculated as follows:

\[ G_{i,j,m} = \sum_{w,h} W_{w,h,m} \cdot X_{i+w,j+h,m} \]
where $G$, $K$ represent the output feature matrix and the convolution kernel weight matrix, respectively. $i$, $j$, $w$, $h$ are used to represent the coordinates of the associated matrix. Depthwise separable convolution is able to replace the standard convolution with less computation and is almost standard for lightweight models [17,20,42]. Its computation compared to the standard convolution is as follows:

\[
Q_1 = D_f^2 D_k^2 M + D_f^2 M N \tag{2}
\]

\[
Q_2 = D_f^2 D_k^2 M N \tag{3}
\]

\[
\frac{Q_1}{Q_2} = \frac{D_f^2 D_k^2 M + D_f^2 M N}{D_f^2 D_k^2 M N} = \frac{1}{N} + \frac{1}{D_k^2} \tag{4}
\]

where $Q_1$ and $Q_2$ are the calculation amount of depthwise convolution and standard convolution, respectively. $D_f$ and $D_k$ represent the side sizes of the feature matrix and the convolution kernel. $N$ and $M$ represent the number of input and output feature map channels. In the feature extraction process, the convolution operation of size 3 × 3 is used extensively. Therefore, the calculation amount of depthwise separable convolution is only about 1/9 of that of regular convolution.

![Diagram](image)

**Figure 1.** Two types of bottleneck structures for ShuffleNet v2. (a): The basic unit of ShuffleNet v2 in the case of stride of 1. (b): The basic unit of ShuffleNet v2 in the case of stride of 2.

The channel shuffle technique [43] is an important innovation point proposed by ShuffleNet, which can realize the information exchange of channels in the feature extraction process with a small computational cost. The basic principle of channel shuffling is shown in Figure 2. The reorganized feature set contains the channel features of each grouping, thus the features after group convolution are related in the input and output dimensions.
3.2. Improved ShuffleNet v2

This study explores the design of efficient RSSC models from the perspective of lightweight networks. The overall architecture of the proposed model is shown in Figure 3. The proposed model uses a lightweight ShuffleNet v2 as the backbone and utilizes a transfer learning strategy to initialize the backbone parameters. Then, the efficient channel attention (ECA) [44] is embedded behind the backbone to suppress useless features and achieve weighted processing of features for subsequent classification processing. While maintaining the lightness, the LSR loss function is implemented to consider the multidimensional loss calculation and further improve the noise immunity of the model. To summarize, RSCNet is based on lightweight network that ensures the speed of inference while introducing optimization strategies to improve its accuracy.

3.2.1. Backbone via Transfer Learning

The process of transfer learning works by having the model trained on a larger domain dataset to gain a priori knowledge. This prior knowledge is then used as a basis to continue training in the target domain dataset to improve the training starting point of the model [45]. In the RSSC task, the flow of the transfer learning implementation is shown in Figure 4, i.e., given the source domain $D_s$ and the source task $D_t$, the corresponding target domain $D_t'$ and the target task $T_t$. In the process, the source and target domains are expected with similarity, and the source domain is generally larger than the target domain. ImageNet [46] is a large definitive image classification dataset with 1000 classes of classification. There are a lot of studies [47–49] based on this dataset, which effectively improve the accuracy of the model in the target domain. Therefore, for the RSSC task, the ShuffleNet v2 model with training weights on the ImageNet dataset is used for transfer learning.
For the RSSC task, before the model training, the model is matched with the pre-trained weights by network layer names to complete the initialization of the weights. At first, the pre-trained weights are loaded into the variables of the dictionary. Then, key-value pairs of pre-trained weights are iterated to find the key names and value sizes that can match the RSSC model and save them to the new dictionary. Finally, the new dictionary is loaded in RSSC model.

### 3.2.2. Channel Attention Mechanism

The visual attention mechanism draws on human visual characteristics to focus on important information in images, which is beneficial for improving model performance. Visual attention mechanisms bring accuracy improvements to CNN by weighting the output features, but mostly at the cost of increasing the complexity, such as convolutional block attention module (CBAM) [50], SE [51]. The reference [44] improves ResNet-101 with ECA, SE, CBAM, and AANet [52] modules, respectively. The above methods improve the classification accuracy on ImageNet dataset by 1.82%, 0.79%, 1.66%, and 1.87%, respectively. Meanwhile, the increase in the number of parameters of ECA is less than 0.01 M, while the number of parameters of SE, CBAM and AANet increases by 4.52 M, 4.52 M and 2.91 M, respectively. ECA [44] is a lightweight attention mechanism module that borrows ideas from SE to create a channel attention mechanism, which can be embedded in CNN to participate in end-to-end training. ECA uses one-dimensional convolution for feature extraction, which avoids feature downscaling and effectively captures cross-channel information interactions.

The working principle of the ECA module is demonstrated in the Figure 5. Suppose the input feature matrix is \( F \in \mathbb{R}^{C \times H \times W} \), and \( C \), \( H \), and \( W \) represent the number of channels, height, and width of the input features, respectively. The input matrix is first processed by a global average pooling layer which results in a channel feature description matrix \( F \in \mathbb{R}^{C \times 1 \times 1} \). Then, feature extraction is performed using 1D convolution and the output is processed using a nonlinear activation function The computational expressions are as follows:

\[
M_c(F) = \sigma(f_{1d}(F_{avg}))
\]  

where \( \sigma \) for the sigmoid function and \( f_{1d} \) for the 1D convolution operation function. Finally, the input features are multiplied with the attention weights in the channel dimension. This is calculated as follows:

\[
F' = F \otimes M_c
\]
where $\otimes$ denotes elemental multiplication with a broadcast mechanism, i.e., during the operation, $M_c$ is copied along the spatial dimension to obtain a $C \times H \times W$ feature matrix, which is then point multiplied with another matrix.

![Diagram of ECA channel attention mechanism](image)

**Figure 5.** ECA channel attention mechanism.

The ECA belongs to the channel attention [51,53], which allows to assign weights to the channels of the feature map, making the network focus on the more important channels. In this study, the model will generate a 1024-dimensional feature vector after the Conv5 layer and use it for the classifier. In order to focus the input of the classifier on important dimensional and to ensure that the backbone network performs complete transfer learning. Therefore, the ECA module is embedded after this layer.

### 3.2.3. Label Smoothing Regularization Loss

In classification tasks, one-hot encoding is usually used to label the true labels and the loss calculation uses the cross-entropy function. One-hot encoding with true labels and cross-entropy loss function is presented below:

\[
y_i = \begin{cases} 
1, & i = c \\
0, & i \neq c 
\end{cases}
\]  

(7)

\[
L_{ce} = -y_c \log(p_c)
\]  

(8)

where $L_{ce}$ is loss value, $y_i$ is the true label of the $i$-th category, $p_i$ represents the prediction confidence of the $i$-th category, $k$ is the total number of categories, and $c$ represents the true category.

According to the formula of the cross-entropy loss function, only the $y_i$ with a true label of 1 is involved in the calculation of the loss, while all other dimensions are ignored. In fact, labels may have an inter-class similarity or certain labeling errors. One-hot encoding is just a simplification of the real classification situation. It will lead to poor generalization performance of the model when facing confusing classification tasks. This research focuses on multi-classification of RSI, where similarities easily exist between different scenes. To suppress the overfitting of the model and enhance the anti-noise ability. The label smoothing regularization [54] strategy is used to optimize the one-hot encoding, which adds noise to the real labels and gives the labels a certain error tolerance rate. The encoding method is as follows:

\[
y_i = \begin{cases} 
1 - \varepsilon, & i = c \\
\frac{1}{k-1}, & i \neq c 
\end{cases}
\]  

(9)

where $\varepsilon$ is a smoothing factor, which is a preset hyperparameter. This study is based on the [54], taking 0.1. The loss function after regularization is as follows:

\[
L_i = \begin{cases} 
(1 - \varepsilon)L_{ce}, & i = c \\
\frac{1}{k-1} \sum_{i=1}^{k-1} \log(p_i), & i \neq c 
\end{cases}
\]  

(10)

It can be seen that the loss function after LSR optimization increases the hyperparameter $\varepsilon$. When the value is 0, the encoding method is one-hot encoding. When is not equal to 0, all dimensions participate in the loss calculation. During the training process, through
the LSR strategy, other mispredicted positions participate in the calculation of the loss to a certain extent. Therefore, during training, the utilization of the LSR loss function allows the model to be optimized toward improving the prediction accuracy and reducing the prediction error rate.

4. Experiments

4.1. Dataset Producing

The AID dataset [29], published by Wuhan University in 2016, was a large dataset for RSSC. The AID was mainly collected through Google Earth, with an image resolution of $600 \times 600$ pixels, containing a total of 30 categories of scenes and a total of 10,000 images, with a varying number of 220–420 images in each category. A partial sample of the AID dataset was shown in Figure 6.

![Sample of the AID dataset.](image-url)
UCMerced_LandUse [26] was a RSSC dataset released by the University of California in 2010. The dataset contained 21 categories of scenarios. The image resolution was 256 × 256 with 100 images per category.

4.2. Experimental Environment and Parameter Setup

The hardware for the experiment was as follows: The manufacturer of our computer was Lenovo, and the GPU was made by Nvidia. CPU was Intel Core i5-8500 (3 GHZ), configured with 1 NVIDIA TITAN RTX graphics card with 24 G graphics memory. The operating system was Centos 7, with CUDA version 11.0, and the deep learning framework was Pytorch 1.7.0, and the inference engine was ONNX Runtime 1.10.0. To reflect the lightweight nature of the model and its speed advantage on the CPU, the speed test of the model was conducted on the Intel i5-8500.

Model training used NVIDIA TITAN RTX for accelerated training, and an environment was a single machine and a single card. The model optimizer chosen SGD [55], and the starting learning rate for each model was 0.02 and decays to 1/3 of the original every 20 training epochs. The dataset samples were divided into training and test sets in the ratio of 8:2. The image enhancement of the training set was done by random cropping and random flipping. The final epoch was set to 200, and batch size was 16.

4.3. Evaluation Metrics

The model evaluation metrics used in this study are shown in Table 1. Where $T_P$ represents the number of predicted results that are consistent with the positive sample. $F_P$ represents the number of samples for which the predicted outcome is positive but inconsistent with the true outcome. $T_N$ represents the number of positive samples predicted as negative samples. $F_N$ represents the number of predicted results that are consistent with the negative samples. $p_o$ is calculated the same as $A$. $p_e$ is the sum of the true number of each category multiplied by the predicted number of that category divided by the square of the total number of all categories.

| Metric          | Symbol | Calculation Formula | Meaning                                                                 |
|-----------------|--------|---------------------|------------------------------------------------------------------------|
| Accuracy        | $A$    | $\frac{T_P + T_N}{T_P + F_P + T_N + F_N}$ | The proportion of correctly predicted samples to the total samples.   |
| Precision       | $P_P$  | $\frac{T_P}{T_P + F_P}$ | Precision is the proportion of samples predicted to be positive that are actually positive. |
| Recall          | $R_e$  | $\frac{T_P}{T_P + F_N}$ | Recall is the proportion of actual positive samples that are predicted to be positive. |
| Specificity     | $S_P$  | $\frac{T_N}{T_N + F_P}$ | Specificity describes the ability to predict negative cases.          |
| MCC             | $M_{CC}$ | $\frac{T_N T_P - F_P F_N}{\sqrt{(T_P + F_P)(T_P + F_N)(T_N + F_P)(T_N + F_N)}}$ | MCC describes the similarity of predicted and actual results.         |
| Cohen’s Kappa   | $K_P$  | $\frac{p_o - p_e}{1 - p_e}$ | Kappa describes the consistency of predicted and actual results.      |

5. Results and Discussion

5.1. Transfer Learning Necessity Validation and Backbone Selection

In order to select the backbone network and verify the necessity of using transfer learning in RSSC. The experiment selected the lightweight models ShuffleNet v2 [20], SqueezeNet [16], MobileNet v2 [19], and the large models DenseNet-121 [12], ResNet-50 [11], VGG-16 [22] for comparison. On the AID dataset, using the above model, training without and with transfer learning, i.e., weights are initialized randomly and initialized using pre-trained weights on the ImageNet dataset.
The training result curves of each model are shown in Figure 7. It can be seen, after transfer learning, the models are able to converge faster and obtain higher classification accuracy. The test results for various models on two public datasets are recorded in Table 2. It can be seen that ShuffleNet v2 has the best overall performance among the compared models. The accuracy of ShuffleNet v2 on the AID dataset is 95.33% and the FLOPs are only 0.15 G. The calculation amount of the model is only about 1/2, 2/11 of the lightweight models MobileNet v2, SqueezeNet, and 1/19, 1/27, 1/103 of the large models DenseNet-121, ResNet-50, VGG-16, respectively. In addition, ShuffleNet v2 achieves similar results on the UCMerced_LandUse dataset, which has a better overall performance in terms of classification accuracy and FLOPs.

**Figure 7.** Comparison of training curves for each model with and without transfer learning.

**Table 2.** Performance comparison of various backbone networks after using transfer learning. The training rate is 0.8. FLOPs is floating-point operations, and $P$ is the number of parameters of the model. $A_n$ represents accuracy without using transfer learning and $A_t$ represents accuracy with transfer learning.

| Model            | AID $A_n$ (%) | AID $A_t$ (%) | UCM $A_n$ (%) | UCM $A_t$ (%) | $P/M$ | FLOPs/G |
|------------------|---------------|---------------|---------------|---------------|-------|---------|
| ShuffleNet v2 [20] | 92.15          | 95.33         | 93.57         | 97.52         | 1.3   | 0.15    |
| SqueezeNet [16]  | 83.05          | 93.99         | 84.76         | 95.95         | 1.3   | 0.82    |
| MobileNet v2 [19] | 92.21          | 95.45         | 93.65         | 96.83         | 2.3   | 0.31    |
| DenseNet-121 [12] | 92.97          | 96.25         | 93.51         | 97.69         | 7     | 2.86    |
| ResNet-50 [11]   | 92.19          | 95.31         | 93.09         | 97.62         | 25.6  | 4.11    |
| VGG-16 [22]      | 90.75          | 95.38         | 93.05         | 97.15         | 102   | 15.46   |
To further explore the contribution of transfer learning with small training samples of remote sensing images, we train each model with different training ratios, and the results are shown in Table 3. It can be found that at a training ratio of 0.05, the classification accuracy of ShuffleNet v2, SqueezeNet, MobileNet v2, DenseNet-121, ResNet-50, and VGG16 improved by 24.81%, 26.25%, 22.15%, 23.14%, 25.18%, and 24.3%, respectively. As the proportion of training data increases, for example, with a training ratio of 0.2, the accuracy improvement of ShuffleNet v2 using transfer learning is about 9%. This indicates that the use of transfer learning is necessary for the lack of remote sensing image data. Moreover, ShuffleNet v2 has the highest accuracy for each training ratio used. Thus, ShuffleNet v2 is chosen as the baseline model for this study.

Table 3. Comparison results of various models with different training ratios on AID dataset. Tr is the training data ratio.

| Model           | Tr = 0.05 | Tr = 0.1 | Tr = 0.15 | Tr = 0.2 |
|-----------------|-----------|----------|-----------|----------|
|                 | A_n (%)   | A_t (%)  | A_n (%)   | A_t (%)  |
| ShuffleNet v2   | 59.76     | 84.57    | 73.69     | 89.22    | 79.8     | 91.62    | 83.79    | 92.83    |
| SqueezeNet      | 45.67     | 71.92    | 51.12     | 80.32    | 55.07    | 83.04    | 60.83    | 86.04    |
| MobileNet v2    | 58.33     | 80.48    | 73.19     | 86.3     | 78.75    | 86.38    | 83.5     | 91.35    |
| DenseNet-121    | 55.75     | 78.89    | 69.32     | 88.25    | 73.75    | 89.17    | 80.51    | 89.65    |
| ResNet-50       | 54.18     | 79.36    | 64.89     | 85.43    | 69.96    | 85.07    | 77.19    | 91.19    |
| VGG-16          | 52.05     | 76.35    | 67.80     | 86.5     | 76.16    | 90.18    | 78.59    | 91.53    |

5.2. Ablation Experiment

5.2.1. Training Results of the Proposed Model

The selected benchmark model is the ShuffleNet v2. First of all, each of the proposed improvement points is combined with the benchmark model, respectively. The training model is conducted on the AID and its performance is recorded to show the contribution of the improvement method to the model. Then, for a full evaluation of the final model, all improvement points were incorporated into the benchmark model.

The training curves are shown in Figure 8. Note that the formula for calculating the loss has changed after using the label smoothing loss function. As can be noticed in Equation (10), the LSR loss is not only calculated in the dimension where the true label is located, but also in other dimensions using the hyperparameter $\epsilon$. So the final loss value will be higher than using the normal cross-entropy loss function. As can be observed from the curves, each model completes the training in the convergence state. All three improvement strategies improve the classification accuracy of the models, which proves that each improvement point is effective for the model accuracy improvement.

To further illustrate the classification performance of the model, the ROC curves of the proposed model on the two datasets are plotted. As shown in Figure 9, it can be seen that the upper left point on the ROC curve of RSCNet is closer to the (0,1) point compared with the original model on both the AID and UC_MerceLandUse datasets. It indicates that the proposed model is able to obtain good recall and specificity at higher classification confidence threshold in the face of remote sensing image classification. Moreover, the area formed by the ROC curve of RSCNet with the x-axis is higher than that of the original model, which proves that the proposed model has better classification performance.
5.2.2. Ablation Experimental Test Results of the Improvement Points

To further test the effectiveness of the proposed combination of improvement points, we conducted more detailed ablation experiments on both datasets. The test results of the models on the AID dataset are shown in Table 4. When the training data ratio is 0.8, for the baseline model, it can be seen that by introducing the ECA module, the classification accuracy of the model improves by 0.6%, while the FLOPs increase only a little (0.06 M). When the weights are initialized using the transfer learning approach, the classification accuracy is improved by 3.18%. After replacing the original loss function with label smoothing loss, the classification accuracy is improved by 1.35%. It can be found that the classification accuracy of the final model is improved by 1.42% compared to that of the transfer learning model. To verify the feasibility of optimizing ECA and LSR with small data, the ablation experiments were conducted on the AID dataset using a training ratio of 0.1, and the results are shown in Table 5. It can be found that there is a 3.01% improvement in the final model compared with the model after transfer learning. This indicates that the contribution of ECA and LSR to the transfer learning optimized model would be more significant in the case of smaller data.
Similarly, our improvement points are validated on the UCMerced_LandUse dataset, as shown in Table 6. Finally, the classification accuracy of the proposed RSCNet on the AID and UCMerced_LandUse datasets is 96.75% and 99.05%, respectively, which is 4.6% and 6.07% higher than the baseline model, respectively. All metrics for measuring the precision of the classification model are better, and the FLOPs of the model are only 153.72M. The proposed model achieves an improvement in prediction accuracy while maintaining its lightweight.

Table 4. Test results of the proposed method on the AID dataset. The training data ratio is 0.8.

| Model   | ECA | Transfer | LSR | A (%) | Pr (%) | Re (%) | Sp (%) | M_CC (%) | KP (%) | FLOPs/M |
|---------|-----|----------|-----|-------|--------|--------|--------|----------|--------|---------|
| ×       | ×   | ×        | 92.15 | 92.49 | 91.89  | 99.73  | 91.75  | 91.91    | 153.66 |
| ✓       | ×   | ×        | 92.75 | 92.84 | 92.3   | 99.75  | 92.25  | 92.48    | 153.72 |
| ×       | ✓   | ×        | 95.33 | 95.46 | 95.00  | 99.84  | 95.04  | 95.27    | 153.66 |
| ShuffleNet v2 | ×   | ✓        | ✓    | 93.50 | 93.88 | 93.23 | 99.78 | 93.17    | 153.66 |
| ✓       | ✓   | ×        | 95.69 | 95.83 | 95.41 | 99.85 | 95.41 | 95.53    | 153.72 |
| ✓       | ×   | ✓        | 94.37 | 94.47 | 94.04 | 99.81 | 94.01 | 94.16    | 153.72 |
| ×       | ✓   | ✓        | 96.04 | 96.03 | 95.8  | 99.86 | 95.75 | 95.90    | 153.66 |
| RSCNet  | ✓   | ✓        | ✓    | 96.75 | 96.83 | 96.48 | 99.89 | 96.51    | 153.72 |

Table 5. Test results of the proposed method on the AID dataset. The training data ratio is 0.1.

| Model   | ECA | Transfer | LSR | A (%) | Pr (%) | Re (%) | Sp (%) | M_CC (%) | KP (%) | FLOPs/M |
|---------|-----|----------|-----|-------|--------|--------|--------|----------|--------|---------|
| ×       | ×   | ×        | 73.69 | 74.63 | 73.54  | 99.09  | 73.04  | 72.75    |        |
| ✓       | ×   | ×        | 75.94 | 77.49 | 75.99  | 99.17  | 75.51  | 75.08    |        |
| ×       | ✓   | ×        | 89.22 | 89.24 | 88.77  | 99.63  | 88.57  | 88.84    |        |
| ShuffleNet v2 | ×   | ✓        | ✓    | 76.90 | 78.57 | 76.71 | 99.20 | 76.19    | 70.09 |
| ✓       | ✓   | ×        | 90.01 | 90.14 | 89.62 | 99.66 | 89.43 | 89.66    | 75.69 |
| ✓       | ×   | ✓        | 79.44 | 80.54 | 79.43 | 99.29 | 79.43 | 78.71    |        |
| ×       | ✓   | ✓        | 90.99 | 90.94 | 90.57 | 99.69 | 90.39 | 90.67    |        |
| RSCNet  | ✓   | ✓        | ✓    | 92.23 | 92.82 | 92.23 | 99.73 | 91.99    | 91.84 |

Table 6. Test results of the proposed method on the UCMerced_LandUse dataset. The training data ratio is 0.8.

| Model   | ECA | Transfer | LSR | A (%) | Pr (%) | Re (%) | Sp (%) | M_CC (%) | KP (%) | FLOPs/M |
|---------|-----|----------|-----|-------|--------|--------|--------|----------|--------|---------|
| ×       | ×   | ×        | 92.98 | 93.04 | 92.98  | 99.65  | 92.58  | 92.63    |        |
| ✓       | ×   | ×        | 93.48 | 93.74 | 93.48  | 99.67  | 93.16  | 93.16    |        |
| ×       | ×   | ✓        | 98.25 | 98.32 | 98.25  | 99.91  | 98.17  | 98.16    |        |
| ShuffleNet v2 | ×   | ✓        | ✓    | 93.74 | 93.94 | 93.98 | 99.70 | 93.78    | 93.68 |
| ✓       | ✓   | ×        | 98.50 | 98.60 | 98.50  | 99.92  | 98.45  | 98.42    |        |
| ✓       | ×   | ✓        | 93.98 | 94.38 | 93.98  | 99.70 | 93.78  | 93.68    |        |
| ×       | ✓   | ✓        | 98.75 | 98.83 | 98.75  | 99.94 | 98.71  | 96.68    |        |
| RSCNet  | ✓   | ✓        | ✓    | 99.05 | 99.07 | 99.00 | 99.95 | 98.97    | 98.95 |

5.3. Model Comparison and Analysis

5.3.1. Performance on the AID Dataset

There exist many image classification models based on CNN. In order to demonstrate the model performance advantages of RSCNet, a variety of popular CNN models are selected for comparison experiments. Lightweight networks ShuffleNet v2 [20], MobileNet v2 [19], SqueezeNet [18], and MnasNet [13], and large networks DenseNet-121 [12], ResNet-50 [11], and VGG-16 [22] were selected for the experiments. In addition, models from the references [56–59] were compared. For a fair comparison, the weights of the above models were all initialized using transfer learning. The above models were trained sepa-
rately on the AID dataset, and the test accuracy of each epoch was recorded. The training
curves of each model are shown in Figure 10.

Figure 10. Performance comparison of various models on AID.

The performance of the models are compared in Table 7. As can be observed from the re-
sults, although ShuffleNet v2 is a lightweight model, it still achieves excellent comprehen-
sive performance in the remote sensing scene classification task. It achieves satisfactory
classification accuracy with very low computation (153.66 M FLOPs). RSCNet is improved
based on the ShuffleNet v2 network, and the classification accuracy is higher than that
of MobileNet v2, SqueezeNet, MnasNet, and DenseNet-121 by 1.3%, 2.76%, 0.26%, and
0.5%. In addition, RSCNet maintains the lightness of ShuffleNet v2, and the calculation
amount of the model is only 153.72M FLOPs. In summary, RSCNet is an efficient model
for remote sensing scene classification.

Table 7. Results of various models on AID dataset.

| Training Rate | Model                  | A(%) | P/M | FLOPs/M |
|---------------|------------------------|------|-----|---------|
| 0.8           | RSCNet                 | 96.75| 1.3 | 153.72  |
|               | ShuffleNet v2 [20]     | 95.33| 1.3 | 153.66  |
|               | MobileNet v2 [19]      | 95.45| 2.3 | 312.95  |
|               | SqueezeNet [16]        | 93.99| 0.8 | 734.99  |
|               | MnasNet [13]           | 96.49| 3.1 | 324.08  |
|               | DenseNet-121 [12]      | 96.25| 7   | 2860    |
|               | ResNet-50 [11]         | 95.31| 25.6| 4110    |
|               | VGG-16 [22]            | 95.38| 102 | 15466   |
| 0.5           | RSCNet                 | 96.24| 1.3 | 153.72  |
|               | SF-CNN with AlexNet [56]| 94.32| >57.1| >710    |
|               | SF-CNN with VGG16 [56] | 96.66| >102| >15466  |
|               | Inception v3 with CapsNet [57]| 96.63| >24.4| >6356   |
|               | DCNN with VGG16 [58]   | 96.89| >102| >15466  |
|               | BiMobileNet [59]       | 96.87| 2.52| 340     |
5.3.2. Performance on the UCMerced_LandUse Dataset

To further analyze the performance of RSCNet in other remote sensing scenes, this study adds the UCMerced_LandUse dataset for model comparison experiments. The classification accuracy of each model on the UCMerced_LandUse dataset is shown in Table 8. It can be seen that among the compared models, RSCNet’s performance on the UCMerced_LandUse dataset has a classification accuracy of 99.05%. RSCNet has the lowest calculation costs, with only 153.71 M. Therefore, RSCNet has the best overall performance among the compared models and can obtain high classification accuracy with very low calculation costs.

Table 8. Results of various models on UCMerced_LandUse dataset. The training data rate is 0.8.

| Model                           | A(%) | P/M | FLOPs/M |
|---------------------------------|------|-----|---------|
| RSCNet                          | 99.05| 1.3 | 153.71  |
| ResNet-50                       | 97.62| 25.6| 4110    |
| VGG-16                          | 97.15| 102 | 15466   |
| SqueezeNet                      | 95.95| 0.7 | 734.2   |
| SF-CNN with AlexNet [56]        | 96.98| >57.1| >710    |
| SF-CNN with VGG16 [56]          | 99.05| >102| >15466  |
| Inception v3 with CapsNet [57]  | 99.05| >24.4| >6356   |
| DCNN with VGG16 [58]            | 98.93| >102| >15466  |
| BiMobileNet [59]                | 99.03| 2.52| 340     |

Figures 11–13 show the confusion matrix for the four models, where the diagonal data indicate the number of correct classifications for each category, and the data beyond the diagonal line indicate the number of samples where confusion occurs for each category. It can be seen that VGG-16 has an error rate of 15% for category 7. ResNet-50 has a prediction error rate of 10% on categories 5, 9, and 11. RSCNet’s predictions fall mostly on the diagonal, with 5% error rate for category 1 and category 7, 10% error rate for category 5, and the rest of the categories being correct, with a low overall confusion rate. According to the experimental comparison results, RSCNet is a lightweight and efficient model for RSSC.
Figure 11. Confusion matrix of RSCNet on UC_MerceLandUse test set.
Figure 12. Confusion matrix of ResNet-50 on UC_MerceLandUse test set.
Figure 13. Confusion matrix of VGG-16 on UC_MercedLandUse test set.
5.4. Testing of Model Speed

The inference speed of the models is directly related to the reliability of the practical application of remote sensing scene classification. We deploy RSCNet, MobileNet v2, SqueezeNet, DenseNet-121, ResNet-50, and VGG16 models on the CPU (Intel Core i5-8500). In order to perform efficient inference and address the framework limitations of deep learning, ONNX Runtime is used as the inference engine.

For the above model, the model inference is called 100 times consecutively using the program, where the input batch size is 1. Then, the inference time spent for each time is recorded, and the obtained recognition speed is shown in Figure 14. In the figure, \( t \) represents the single inference time spent and \( N \) is the number of inferences. It can be seen that the average inference time of RSCNet is only 2.75ms, while the lightweight models MobileNet v2 and SqueezeNet are 3.82 ms and 7.43 ms, respectively. And the average inference time of large models VGG-16, ResNet-50, and DenseNet-121 are 76.28 ms, 21.74 ms, and 19.67 ms, respectively. RSCNet’s inference speed is only about 4/11 of SqueezeNet and 1/27 of VGG-16, and the test results show that RSCNet has the shortest average inference time among the compared models. In addition, because RSCNet is a lightweight network and has fewer network branches, there is less fluctuation in the inference process in CPU devices. Some models have very large FLOPs or a lot of branching structures in the network, which can make continuous computation unstable on CPU with tight computational resources.

![Figure 14. Comparison of inference time for various models.](image)

6. Conclusions

Classification methods based on deep CNNs form a crucial technical basis in RSI processing. However, large CNN models face the challenge of large model sizes while achieving high recognition accuracy. Therefore, this study focuses on the efficiency of the RSSC model. Optimization of lightweight ShuffleNet v2 is taken as the core, and it is improved by using transfer learning, attention mechanism and label smoothing regularization.

Experimental results show that transfer learning is effective and necessary in remote sensing scene classification, and that a lightweight network using transfer learning can achieve satisfactory classification results. The embedded attention mechanism could weight the output features with a minor calculation cost, which helps to improve the model accuracy. Adding noise to the labels by a label smooth regularization strategy improves the generalization ability of the model. The proposed model has higher classification accuracy and faster processing speed in two public remote sensing datasets, which provides
the basic theory and key technical support for conducting fast classification of enormous amount of remote sensing images.

The future research directions will be considered for improvement in the following directions:

(1) This study is mainly focused on improving the computational efficiency of the model from the perspective of lightweight network structure. In the future, we can study how to further improve computing efficiency by pruning and quantization techniques.

(2) Explore the application of lightweight backbone in remote sensing image segmentation or detection models. An efficient backbone will be beneficial to remote sensing detection and segmentation models.

**Author Contributions:** Conceptualization, J.Y.; Data curation, Z.C. and L.C.; Formal analysis, J.Y. and Z.F.; Investigation, Z.F.; Methodology, Z.C.; Project administration, J.Y.; Software, Z.F. and L.C.; Validation, Z.C.; Writing original draft, Z.C.; Writing—review and editing, Z.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by a grant from the Research Projects of Ganjiang Innovation Academy, Chinese Academy of Sciences (No. E255J001), the National Natural Science Foundation of China (No. 62063009), and the Jiangxi Postgraduate Innovation Special Fund Project (YC2022-S648).

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. He, N.; Fang, L.; Li, S.; Plaza, J.; Plaza, A. Skip-Connected Covariance Network for Remote Sensing Scene Classification. *IEEE Trans. Neural Netw. Learn. Syst.* 2020, 31, 1461–1474. [CrossRef]

2. Yuan, X.; Shi, J.; Gu, L. A review of deep learning methods for semantic segmentation of remote sensing imagery. *Expert Syst. Appl.* 2021, 169, 114417. [CrossRef]

3. Ma, W.; Karakuş, O.; Rosin, P.L. AMM-FuseNet: Attention-Based Multi-Modal Image Fusion Network for Land Cover Mapping. *Remote Sens.* 2022, 14, 1158. [CrossRef]

4. Zhang, L.; Cai, Y.; Huang, H.; Li, A.; Yang, L.; Zhou, C. A CNN-LSTM Model for Soil Organic Carbon Content Prediction with Long Time Series of MODIS-Based Phenological Variables. *Remote Sens.* 2022, 14, 4441. [CrossRef]

5. Broni-Bediako, C.; Murata, Y.; Mormille, L.H.B.; Atsumi, M. Searching for CNN Architectures for Remote Sensing Scene Classification. *IEEE Trans. Geosci. Remote Sens.* 2021, 60, 1–13. [CrossRef]

6. Al-Khasawneh, M.A.; Uddin, I.; Shah, S.A.A.; Khasawneh, A.M.; Abualigah, L.; Mahmoud, M. An improved chaotic image encryption algorithm using Hadoop-based MapReduce framework for massive remote sensed images in parallel IoT applications. *Clust. Comput.* 2022, 25, 999–1013. [CrossRef]

7. Karadal, C.H.; Kaya, M.C.; Tunçer, T.; Dogan, S.; Acharya, U.R. Automated classification of remote sensing images using multileveled MobileNetV2 and DWT techniques. *Expert Syst. Appl.* 2021, 185, 115659. [CrossRef]

8. Leonardi, R.; Giudice, A.L.; Isola, G.; Spampinato, C. Deep Learning and Computer Vision: two promising pillars, powering the future in Orthodontics. *Semin. Orthod.* 2021, 27, 62–68. [CrossRef]

9. Chai, J.; Zeng, H.; Li, A.; Ngai, E.W. Deep learning in computer vision: A critical review of emerging techniques and application scenarios. *Mach. Learn. Appl.* 2021, 6, 100134. [CrossRef]

10. Liu, H.; You, K. Research on image multi-feature extraction of ore belt and real-time monitoring of the tabling by semantic segmentation of DeepLab V3. In Proceedings of the Advances in Artificial Intelligence and Security, Quinghai, China, 15–20 July 2022; Springer International Publishing: Cham, Switzerland, 2022; pp. 35–49. [CrossRef]

11. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]

12. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2261–2269. [CrossRef]

13. Tan, M.; Chen, B.; Pang, R.; Vasudevan, V.; Sandler, M.; Howard, A.; Le, Q.V. MnasNet: Platform-aware neural architecture search for mobile. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 2815–2823. [CrossRef]

14. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. CoRR 2015. Available online: http://xxx.lanl.gov/abs/1505.04597 (accessed on 1 October 2022).

15. Awad, M.M.; Lauteri, M. Self-Organizing Deep Learning (SO-UNet)—A Novel Framework to Classify Urban and Peri-Urban Forests. *Sustainability* 2021, 13, 5548. [CrossRef]
16. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. arXiv 2016, arXiv:1602.07360.
17. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21-26 July 2017; pp. 1800–1807. [CrossRef]
18. Chen, Z.; Guo, H.; Jie, Y. Fast vehicle detection algorithm in traffic scene based on improved SSD. Measurement 2022, 201, 111655. [CrossRef]
19. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted residuals and linear bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18-23 June 2018; pp. 4510–4520. [CrossRef]
20. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the Proceedings of the European conference on computer vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 116–131. [CrossRef]
21. Tan, M.; Le, Q. EfficientNet: Rethinking model scaling for convolutional neural Networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Volume 97, pp. 6105–6114.
22. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. arXiv 2014, arXiv:1409.1556.
23. Amerini, I.; Ballan, L.; Caldelli, R.; Bimbo, D.A.; Serra, G. A SIFT-Based Forensic Method for Copy–Move Attack Detection and Transformation Recovery. IEEE Trans. Inf. Forensics Secur. 2011, 6, 1099–1110. [CrossRef]
24. Tan, S.; Bhattacharyya, U.; Lu, S.; Su, B.; Tan, C.L. Multilingual scene character recognition with co-occurrence of histogram of oriented gradients. Pattern Recognit. 2016, 51, 125–134. [CrossRef]
25. Wang, X.; Han, T.X.; Yan, S. An HOG-LBP human detector with partial occlusion handling. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009.
26. Yang, Y.; Newsam, S. Bag-of-Visual-Words and Spatial Extensions for Land-Use Classification; Association for Computing Machinery: New York, NY, USA, 2010; Volume 10, pp. 270–279. [CrossRef]
27. Yang, Y.; Newsam, S. Spatial pyramid co-occurrence for image classification. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 3-6 November 2011; pp. 1465–1472. [CrossRef]
28. Ren, J.; Jiang, X.; Yuan, J. Learning LBP structure by maximizing the conditional mutual information. Pattern Recognit. 2015, 48, 3180–3190. [CrossRef]
29. Xia, G.S.; Hu, J.; Hu, F.; Shi, B.; Bai, X.; Zhong, Y.; Zhang, L.; Lu, X. AID: A Benchmark Data Set for Performance Evaluation of Aerial Scene Classification. IEEE Trans. Geosci. Remote Sens. 2017, 55, 3965–3981. [CrossRef]
30. Song, J.; Gao, S.; Zhu, Y.; Ma, C. A survey of remote sensing image classification based on CNNs. Big Earth Data 2019, 3, 232–254. [CrossRef]
31. Dou, P.; Shen, H.; Li, Z.; Guan, X. Time series remote sensing image classification framework using combination of deep learning and multiple classifiers system. Int. J. Appl. Earth Obs. Geoinf. 2021, 103, 102477. [CrossRef]
32. Cheng, X.; He, X.; Qiao, M.; Li, P.; Hu, S.; Chang, P.; Tian, Z. Enhanced contextual representation with deep neural networks for land cover classification based on remote sensing images. Int. J. Appl. Earth Obs. Geoinf. 2022, 107, 102706. [CrossRef]
33. Li, E.; Xia, J.; Du, P.; Lin, C.; Samat. A. Integrating Multilayer Features of Convolutional Neural Networks for Remote Sensing Scene Classification. IEEE Trans. Geosci. Remote Sens. 2017, 55, 5653–5665. [CrossRef]
34. Shawky, O.A.; Hagag, A.; El-Dahshan, E.; Ismail, M.A. Remote Sensing Image Scene Classification Using CNN-MLP with Data Augmentation. Opt. Int. J. Light Electron Opt. 2020, 221, 165356. [CrossRef]
35. Tang, X.; Ma, Q.; Zhang, X.; Liu, F.; Ma, J.; Jiao, L. Attention Consistent Network for Remote Sensing Scene Classification. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 2021, 14, 2030–2045. [CrossRef]
36. Ma, A.; Wan, Y.; Zhong, Y.; Wang, J.; Zhang, L. SceneNet: Remote sensing scene classification deep learning network using multi-objective neural evolution architecture search. ISPRS J. Photogramm. Remote Sens. 2021, 172, 171–188. [CrossRef]
37. Gu, P. A Multi-Source Data Fusion Decision-Making Method for Disease and Pest Detection of Grape Foliage Based on Shufflenet V2. Remote Sens. 2021, 13, 5102. [CrossRef]
38. Li, Y.; Wang, X.; Shi, B.; Zhu, M. Hand Gesture Recognition Using IR-UWB Radar with Shufflenet V2; Association for Computing Machinery: New York, NY, USA, 2021; pp. 126–131. [CrossRef]
39. Ran, H.; Wen, S.; Wang, S.; Cao, Y.; Zhou, P.; Huang, T. Memristor-Based Edge Computing of ShuffleNetV2 for Image Classification. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 2021, 40, 1701–1710. [CrossRef]
40. Chen, Z.; Yang, J.; Chen, L.; Jiao, H. Garbage classification system based on improved ShuffleNet v2. Resour. Conserv. Recycl. 2022, 178, 106900. [CrossRef]
41. Tang, Z.; Yang, J.; Li, Z.; Qi, F. Grape disease image classification based on lightweight convolution neural networks and channelwise attention. Comput. Electron. Agric. 2020, 178, 105735. [CrossRef]
42. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv 2017, arXiv:1704.04861. [CrossRef]
43. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18-23 June 2018; pp. 6848–6856. [CrossRef]
44. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; Hu, Q. ECA-Net: Efficient channel attention for deep convolutional neural networks. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 11531–11539. [CrossRef]

45. Tan, C.; Sun, F.; Kong, T.; Zhang, W.; Yang, C.; Liu, C. A Survey on Deep Transfer Learning. In Proceedings of the International Conference on Artificial Neural Networks, Prague, Czech Republic, 3–6 September 2018. [CrossRef]

46. Krizhevsky, A.; Sutskever, I.; Hinton, G. ImageNet Classification with Deep Convolutional Neural Networks. Adv. Neural Inf. Process. Syst. 2012, 25. [CrossRef]

47. Díaz-Romero, D.; Sterkens, W.; Van den Eynde, S.; Goedemé, T.; Dewulf, W.; Peeters, J. Deep learning computer vision for the separation of Cast- and Wrought-Aluminum scrap. Resour. Conserv. Recycl. 2021, 172, 105685. [CrossRef]

48. Talo, M. Automated classification of histopathology images using transfer learning. Artif. Intell. Med. 2019, 101, 101743. [CrossRef] [PubMed]

49. Relekar, H.; Shanmugam, P. Transfer learning based ship classification in Sentinel-1 images incorporating scale variant features. Adv. Space Res. 2021, 68, 4594–4615. [CrossRef]

50. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cham: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Heraklion, Greece. 27–30 April 2018; pp. 3–19. [CrossRef]

51. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), New York, NY, USA, 15–17 June 2018. [CrossRef]

52. Xu, H.; Zhang, J. AANet: Adaptive aggregation network for efficient stereo matching. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 1956–1965. [CrossRef]

53. Niu, Z.; Zhong, G.; Yu, H. A review on the attention mechanism of deep learning. Neurocomputing 2021, 452, 48–62. [CrossRef]

54. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826. [CrossRef]

55. Loshchilov, I.; Hutter, F. SGDR: Stochastic gradient descent with warm restarts. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 201. Available online: OpenReview.net (accessed on 1 October 2022).

56. Xie, J.; He, N.; Fang, L.; Plaza, A. Scale-Free Convolutional Neural Network for Remote Sensing Scene Classification. IEEE Trans. Geosci. Remote Sens. 2019, 57, 6916–6928. [CrossRef]

57. Zhang, W.; Tang, P.; Zhao, L. Remote Sensing Image Scene Classification Using CNN-CapsNet. Remote Sens. 2019, 11, 494. [CrossRef]

58. Cheng, G.; Yang, C.; Yao, X.; Guo, L.; Han, J. When Deep Learning Meets Metric Learning: Remote Sensing Image Scene Classification via Learning Discriminative CNNS. IEEE Trans. Geosci. Remote Sens. 2018, 56, 2811–2821. [CrossRef]

59. Yu, D.; Xu, Q.; Guo, H.; Zhao, C.; Li, D. An Efficient and Lightweight Convolutional Neural Network for Remote Sensing Image Scene Classification. Sensors 2020, 20, 1999. [CrossRef]