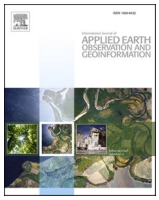
International Journal of Applied Earth Observations and Geoinformation 112 (2022) 102865

Contents lists available at ScienceDirect

International Journal of Applied Earth Observations and Geoinformation

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



Assessing the effects of convolutional neural network architectural factors on model performance for remote sensing image classification: An in-depth investigation

Feihao Chen [a], Jin Yeu Tsou [a,b,*]

[a] School of Architecture, The Chinese University of Hong Kong, Hong Kong Special Administrative Region
[b] Department of Architecture and Civil Engineering, City University of Hong Kong, Hong Kong Special Administrative Region

ARTICLE INFO

Keywords:
CNN architecture
Model performance
Remote sensing
Scene classification

ABSTRACT

Although the application of deep learning in remote sensing (RS) has achieved fruitful results, systematic research on exploring the model performance and guiding the design of new convolutional neural network (CNN) architectures is still lacking. This subject is of great concern to researchers or practitioners in this field because existing CNN structures may not be adequate to deal with complex RS scenarios. In this study, an empirical formula of CNN model performance is delivered based on a literature review. Extensive experiments are conducted on six public RS data sets to investigate the influences of three architectural factors, namely, network depth, width, and cardinality. Two types of CNN architectures, i.e., VGG and ResNet, are adopted as baselines. We monitor and visualize the data distributions and gradients of the utilized CNNs to prevent the gradient vanishing or exploding problem. Grad-CAM is adopted to open the black box of CNNs and to illustrate the effects of adjusting architectural factors. Our experiments indicate that (1) increasing the network depth is beneficial to the semantic feature learning capacity of a CNN model, but excessive depth also leads to a decline of overall accuracy; (2) a partly widening strategy is effective because it can improve the model performance while maintaining the network complexity; and (3) network cardinality has huge potential in achieving a balance between model efficiency and accuracy. Suggestions for improving the CNN model performance and developing new structures are summarized in this paper.

1. Introduction

As the fastest-growing trend in big data analysis, deep learning (DL) has recently gained popularity in various fields, such as natural language processing, computer vision, and speech enhancement. Inspired by models of the biological brain, DL uses simple but massively interconnected units—the neurons—to replicate the perception and cerebral mechanisms (Castelluccio et al., 2015; Chen and Tsou, 2021b). Unlike traditional learning algorithms, DL can learn intrinsic features from raw input data without using hand-crafted features, thus avoiding reliance on domain knowledge (Li et al., 2018). In general, four mainstream DL methods exist (Ball et al., 2017; Yuan et al., 2020), namely, deep belief networks, stacked autoencoder, convolutional neural networks (CNNs), and recurrent neural networks. More recently, generative adversarial networks, a promising unsupervised learning method, have become another emerging trend in this domain (Cheng et al., 2020; Ma et al., 2019; Osco et al., 2021). Of these methods, CNN is the most well-known and most published DL architecture to date (Ball et al., 2017).

The tremendous success of CNN has drawn attention from the remote sensing (RS) community. An increasing number of studies in this domain have utilized CNN models due to their outstanding performance. For example, Penatti et al. (2015) first adopted CNNs for RS-related tasks. Their experiments indicated that pretrained CNNs can be transferred to the RS image classification task and achieved inspiring results (Penatti et al., 2015). Castelluccio et al. (2015) applied two CNN architectures, i.e., CaffeNet and GoogLeNet, on two RS data sets (Castelluccio et al., 2015). Three training modalities are utilized, namely, full training, fine-tuning, and using CNNs as feature extractors. Their research illustrated that GoogLeNet with careful fine-tuning achieved the highest accuracy on UCMerced, whereas GoogLeNet trained from scratch (full training) exhibited the most satisfactory performance on Brazilian Coffee Scenes. Nogueira et al. (2017) conducted further experiments following this

* Corresponding author at: Department of Architecture and Civil Engineering, City University of Hong Kong, Hong Kong Special Administrative Region.
E-mail addresses: feihaochen@link.cuhk.edu.hk (F. Chen), jytsou@cityu.edu.hk (J.Y. Tsou).

https://doi.org/10.1016/j.jag.2022.102865
Received 15 April 2022; Received in revised form 18 May 2022; Accepted 9 June 2022
Available online 24 June 2022

direction. They explored a variety of existing CNN architectures on three RS data sets and concluded that fine-tuning is the best training strategy in different situations (Nogueira et al., 2017).

In addition to scene classification, CNNs could be applied to almost every step of RS image processing (Ma et al., 2019). Major applications of CNNs in the RS domain include data fusion (Shao and Cai, 2018; Wei et al., 2017; Yang et al., 2018), image registration (He et al., 2018; Wang et al., 2018b), information construction and prediction (Das and Ghosh, 2017; Wu et al., 2019; Zhang et al., 2018), land use and cover mapping (Jia et al., 2019; Mohammadimanesh et al., 2019; Yang et al., 2019; Yoo et al., 2019), change detection (Cao et al., 2019; Wang et al., 2018a), and environmental parameter retrieval (Jiang et al., 2019; Jin et al., 2020). Land use and cover classification with CNNs is the most published application, as indicated by (Ma et al., 2019). Scene-wise classification and semantic segmentation are the two commonly used mapping approaches, with state-of-the-art results being achieved (Chen and Tsou, 2021a; Liu and Shi, 2020; Mohammadimanesh et al., 2019; Qiu et al., 2020).

Although great achievements have been made, challenges arise concerning the capacity of existing CNN structures for RS tasks. Researchers from the RS domain prefer directly applying ready-made CNN architectures; however, these networks may not be adequate to deal with complex RS scenarios (Ball et al., 2017). Classic CNNs are mainly designed to intake RGB images, which are different from RS images regarding sourced sensors, spectral bands, and resolutions. For example, (Penatti et al., 2015) showed that pre-trained CNN models were outperformed by some low-level descriptors on some RS data sets. Chen and Tsou (2021a, b) compared the land surface mapping accuracies of two classic CNN structures (i.e., CaffeNet and GoogLeNet) and the random forest (RF) algorithm with a moving window. They found that the performance of CNN schemes was surpassed by the improved RF models (Chen and Tsou, 2021b).

Very recently, attempts on proposing novel CNN structures for RS tasks have emerged (Liu and Shi, 2020; Qiu et al., 2020). However, developing novel CNN architectures is still not easy for researchers with little or no computer science background. Detailed research from the RS domain focusing on structure factors that affect model performance is lacking. Review papers have pointed out that this kind of research is urgent, because no theoretical basis for network structure selection exists (Ball et al., 2017; Yao et al., 2017). Qiu et al. (2020) evaluated the influences of network depth and width on a data set (Qiu et al., 2020), but the robustness of their conclusions should be tested further due to several reasons. First, only two factors (i.e., network depth and width) are selected as variables in their experiments, and other important factors such as network cardinality are neglected. Second, instead of some widely used CNN structures, they utilized a model that they proposed for evaluations. Thus, findings may not be broadly representative. Third, their experiments are conducted on only one data set, which also impedes the generalization of their conclusions. Therefore, further explorations should be encouraged. Unsolved questions include, but are not limited to: 1) What are the key factors of a CNN architecture that are highly related to model performance? 2) How do these factors affect model performance? 3) How can these factors be balanced when improving model performance or developing a new CNN structure?

To address these gaps, we select two widely used CNN architectures, namely, VGG (Simonyan and Zisserman, 2014) and ResNet (He et al., 2016), to investigate factors that affect model performance for RS classification tasks. Meta-analyses (Kattenborn et al., 2021; Neupane et al., 2021) indicate that VGG and ResNet are the most commonly applied CNN backbones in terms of DL applications in RS. Therefore, experimental results based on these two backbone networks are reliable and broadly representative. Extensive experiments are conducted on six public RS data sets, namely, UCMerced Land Use (Yang and Newsam, 2010), Brazilian Coffee Scenes (Penatti et al., 2015), SIRI-WHU (Zhao et al., 2016), EuroSAT (Helber et al., 2018), Hurricane Damage (Cao and Choe, 2020), and CLRS data sets (Li et al., 2020). These public data sets are different from each other in terms of spatial resolution, image size, release year, total number of images, and number of categories, representing a wide range of RS scenarios. Thus, the findings and conclusions of this paper are universally robust. Grad-CAM proposed by (Selvaraju et al., 2017) is applied to illustrate the interpretation process of CNN models and to understand how architectural factors affect model performance. Metrics such as overall accuracy (OA), the total number of parameters (Params), parameter size (Param_size), and floating point operations (Flops) are utilized to fully consider these factors when developing a new CNN architecture.

The main contributions of this paper are threefold.

1) We deliver an empirical formula of the CNN model performance and specifically focus on factors that directly come from network architectures. Extensive experiments are conducted on six public RS data sets to investigate their potential effects. To the best of our knowledge, this paper is the first in the RS domain to focus on this topic and conduct comprehensive experiments.
2) We illustrate the interpretation process of CNN models by opening the black box using state-of-the-art visualizing techniques. The interpretability of CNNs has been a hot spot in recent years. Few published papers are available concerning this issue in the RS field; thus, this paper could serve as an important supplement.
3) We summarize some useful tricks of developing CNN structures and provide insightful conclusions. The findings of this study could be pivotal to researchers and practitioners who are interested in improving model performance of CNNs or proposing novel CNN structures for RS classification tasks.

The remainder of this paper is structured as follows: Section 2 proposes an empirical formula of CNN model performance and briefly reviews the elements in the formula, with particular focus on the architectural factors. Two widely adopted CNN structure are also introduced. Section 3 presents the utilized data sets, evaluation metrics, and implementation details. Section 4 exhibits the experimental results in detail, and Section 5 illustrates the interpretation process of CNN models by using some state-of-the-art visualizing techniques. Rules of thumb are summarized for researchers to improve the model performance or develop new CNN architectures. Section 6 provides the conclusions of this paper.

2. Related research

2.1. Factors affecting model performance

The last decade has witnessed significant advances in CNNs. A breakthrough was made by Krizhevsky et al. in 2012, who proposed a novel structure called AlexNet, winning the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)-2012 competition by a large margin (Krizhevsky et al., 2012). The success of AlexNet has accelerated the development of CNNs. Since then, milestone CNNs with good performance have been emerging, pushing the frontiers continuously.

Let $\mathscr{A}$, $\mathscr{I}$, $\mathscr{T}$, and $\mathscr{N}$ be the CNN architecture, input data, training settings, and measurement noise, respectively. Based on the work by (Wightman et al., 2021), we propose that the model performance of a CNN can be expressed as follows:

$$Accuracy(model) = f(\mathscr{A}, \mathscr{I}, \mathscr{T}, \mathscr{N}) \quad (1)$$

Input data are foundational to model performance and are consistently undervalued (Sambasivan et al., 2021). High quality and large amount of training data (i.e., ImageNet data set) have contributed to the prosperity of DL in recent decades. Data augmentation techniques are widely adopted to avoid overfitting when limited input data are available (Shorten and Khoshgoftaar, 2019). Research indicates that multiscale and high-resolution input data have positive effects on model performance (Hu et al., 2015; Tong et al., 2018). In addition to input

data, appropriate training settings are crucial to model accuracy. Trends in training a CNN model have changed over time, including the development of different optimizers (Loshchilov and Hutter, 2018; Sutskever et al., 2013; You et al., 2019), the replacement of waterfall scheduler with more progressive schedulers (Dollár et al., 2021; Dosovitskiy et al., 2020), and the utilization of other loss functions except for cross-entropy loss (Beyer et al., 2020). In fact, no optimal training setting applies to all CNNs. Learning rate, batch size, weight decay, and the number of training epochs are some important hyperparameters that affect model accuracies. As for the measurement noise, good practices to mitigate $\mathscr{N}$ include measuring the standard deviation with different seeds, using a separate evaluation data set, and evaluating models on transfer tasks (Wightman et al., 2021).

This paper specifically focuses on factors that directly come from the CNN architecture (i.e., the algorithm itself). A recent study conducted by MIT scientists indicated the importance of algorithm improvement. Their findings showed that the improvement to performance from algorithms outpaced those from improved hardware (Sherry and Thompson, 2021). In real-world scenarios, input data are often limited even after augmentation, and improvements by adjusting training settings might not be significant. Therefore, a functional CNN architecture is crucial under this circumstance. To investigate the effects of these architectural factors solely, we disentangled $\mathscr{A}$ from other elements in formula 1 by keeping $\mathscr{I}$, $\mathscr{T}$, and $\mathscr{N}$ unchanged. Formula 1 is thus denoted as.

$$Accuracy_{\mathscr{A}}(model) = \max_{\mathscr{A}} f(\mathscr{A}|\mathscr{I},\mathscr{T},\mathscr{N}). \tag{2}$$

Three fundamental factors of CNN architecture are selected to investigate their influences on model performance, namely, network depth, width, and cardinality. Major CNN performance improvements over the past few decades came from these three dimensions.

2.1.1. Network depth

The depth of a CNN refers to the number of its convolutional and fully connected layers. A widely accepted assumption is that with the increase in depth, a CNN model can better approximate the target function with a number of nonlinear mappings and more enriched feature hierarchies (Bengio, 2013; Khan et al., 2020). Empirically, a deeper model means stronger nonlinear interpretation capacity; thus, it can represent certain classes of function more efficiently than its shallow counterparts (Khan et al., 2020; Montufar et al., 2014). In addition, as the network goes deeper, the function to be approximated by each layer becomes simpler. Network structures such as VGG further strengthen the idea that depth is essential for model performance.

However, deep CNNs may suffer from several severe problems. For example, overfitting becomes common when a network consists of dozens or hundreds of layers without regularizations. The gradient exploding or vanishing problem is another drawback associated with an excessively deep network. Network degradation is another major concern when the depth of a CNN architecture is increased (K. He et al., 2016). In summary, despite the fact that generally, network depth improves model generalization capacity, it also poses negative influences on model performance when a network is excessively deep.

2.1.2. Network width

Network width is related to the number of channels of each CNN layer. During 2012–2015, research from the computer vision domain has mainly focused on exploiting the power of network depth. However, Szegedy et al. (2015) showed for the first time that width is another essential factor to improve model performance. Their proposed wide CNN structure, GoogLeNet, achieved the best classification accuracy in the ILSVRC-2014 competition (Szegedy et al., 2015), suggesting the importance of network width along with depth. Generally, a wider network allows each layer to learn richer features such as textures in different directions and frequencies. Studies have indicated that CNNs with ReLU activation function have to be wide enough to hold universal approximation property, and if the maximum network width is not larger than the input dimension, then they cannot well approximate a class of continuous functions on a compact set (Hanin and Sellke, 2017; Khan et al., 2020; Lu et al., 2017).

Notably, excessively wide networks will lead to high computational cost and memory requirement. As a result, they may not be deployed in resource-constrained environments. Some recent studies also revealed that an excessively wide CNN generates poor classification accuracy (Qiu et al., 2020).

2.1.3. Network cardinality

For a long time, the common strategy to improve model accuracy is to deepen or widen the network. However, as the depth or width increases, the network complexity and computational cost will also increase significantly. Xie et al. identified a new dimension called cardinality, i.e., the size of the set of transformations, as an essential factor in addition to depth and width (Xie et al., 2017). Their idea was inspired by the split-transform-merge strategy (Szegedy et al., 2015), that is, the input is first split into multiple embeddings, then transformed by a set of kernels, and finally merged by element-wise addition (Qiu et al., 2020; Xie et al., 2017) (Fig. 1). Through experiments, they concluded that increasing cardinality is a more effective way of gaining accuracy than going deeper or wider, especially when depth and width start to give diminishing returns for models (Xie et al., 2017). One of the biggest advantages of increasing cardinality is that model improvement could be obtained while maintaining or even reducing the network complexity.

2.2. Vgg

Proposed by (Simonyan and Zisserman, 2014), VGG is composed of 3×3 convolutional layers, 2×2 max-pooling layers with a stride of 2, and three fully connected layers at the end. VGG takes inspiration from the work by (Zeiler and Fergus, 2014), and it uses only 3×3 convolutional filters, thus being famous for its simplicity and homogeneous topology (Khan et al., 2020; Qiu et al., 2020). Fig. 2 illustrates the network structure of VGG16. It contains five blocks that consist of 3×3 convolutional layers. The number of convolutional layers in each block is [2, 2, 3, 3, 3]. A max-pooling layer is found between each block to halve the size of feature maps. The output feature size of the fifth block is 1/32 of the input image size.

2.3. ResNet

ResNet revolutionized the CNN architecture race by introducing the concept of residual learning, that is, shortcut connection (K. He et al., 2016). Shortcut connection connects the output of at least one convolutional layer to the original input (Kumar et al., 2020). Research indicates that shortcut connections are effective in mitigating the degradation problem and reducing the computational complexity (K. He et al., 2016). Residual learning can be expressed using the following equation, where X_{i-1} and X_i are the input and output of the ith convolutional layer F_i, respectively:

$$X_i = F_i(X_{i-1}) + X_{i-1} \tag{3}$$

The authors designed two types of different blocks for ResNet. One is the basic building module, as illustrated in Fig. 3. The identity mapping skips two 3×3 convolutional layers and connects the previous outputs with input layer. The other is the bottleneck building module, where three convolutional layers (i.e., 1×1, 3×3, and 1×1) are stacked for each residual function instead of two (K. He et al., 2016). Their research revealed that performance differences exist between ResNet_basic (ResNet with basic building modules) and ResNet_bottleneck (ResNet with bottleneck building modules). Thus, these two types of ResNet are adopted in this paper to make our research conclusions more

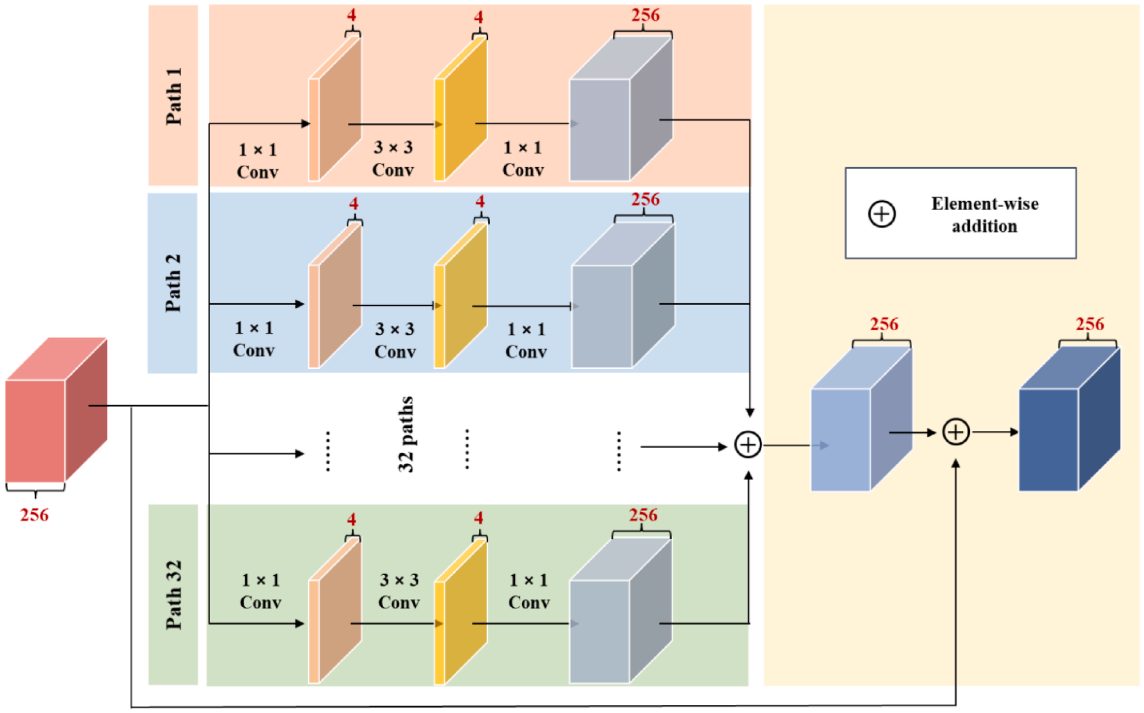


Fig. 1. Illustration of the split-transform-merge strategy. The input channels are first split into multiple paths and then transformed by convolutional kernels (e.g., 1 × 1 and 3 × 3 kernels). After that, they are merged by shortcut connections. In this case, the number of paths is 32. Therefore, the cardinality of this module is 32.

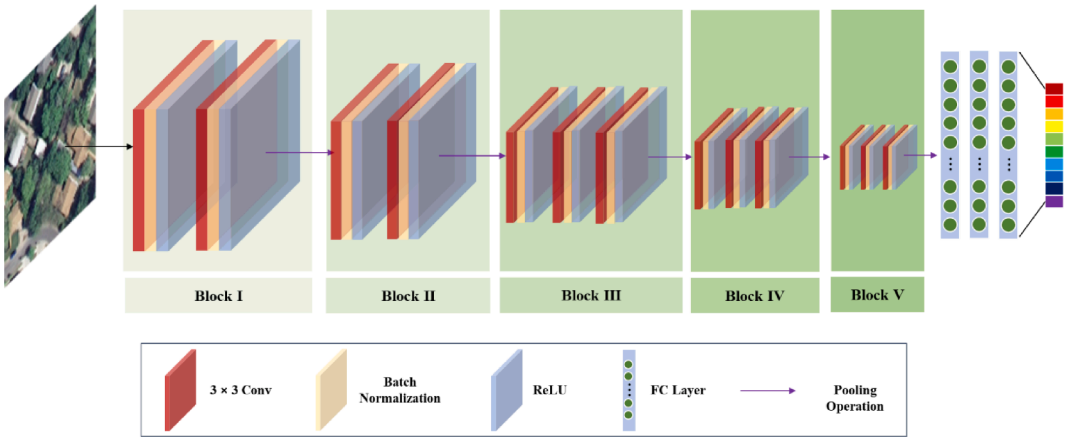


Fig. 2. Illustration of network architecture of VGG16. VGG16 contains 13 convolutional layers and 3 fully connected layers in total. Therefore, the network depth of VGG16 is 16.

representative and reliable.

Fig. 3 illustrates the architecture of ResNet34 or ResNet50. These two networks share an identical structure, and each of their blocks contains the same number of modules, i.e., [3, 4, 6, 3]. After each block, the size of feature maps will be halved. The only difference between these two networks is that ResNet34 applies basic building modules in all the blocks, whereas ResNet50 adopts bottleneck building modules.

3. Proposed method

Although studies on CNN model scaling exist, most of them are from the computer vision domain and focus on some broad aspects such as depth, width, and image resolution (Dollár et al., 2021; Kawaguchi et al., 2019; Tan and Le, 2019). In this paper, we specifically focus on architectural factors and disentangle them from other types of factors in formula 1. Extensive experiments are conducted on six public RS data sets to evaluate the influences of three essential architectural factors on model performance, namely, network depth, width, and cardinality. VGG and two types of ResNet are utilized as baseline networks in this research.

3.1. Data sets

Six public RS data sets are utilized in this paper, namely, UCMerced Land Use, Brazilian Coffee Scenes, SIRI-WHU, EuroSAT, Hurricane Damage, and CLRS. Notably, most of these data sets consist of satellite images, such as Brazilian Coffee Scenes, EuroSAT, and Hurricane Damage. The RS images of SIRI-WHU and CLRS were acquired from Google Earth, where the data sources are a combination of satellite images and aerial images. As for the UCMerced Land Use data set, it is composed of large-size aerial images. Table 1 presents the detailed information of these data sets. As shown in the table, they are different

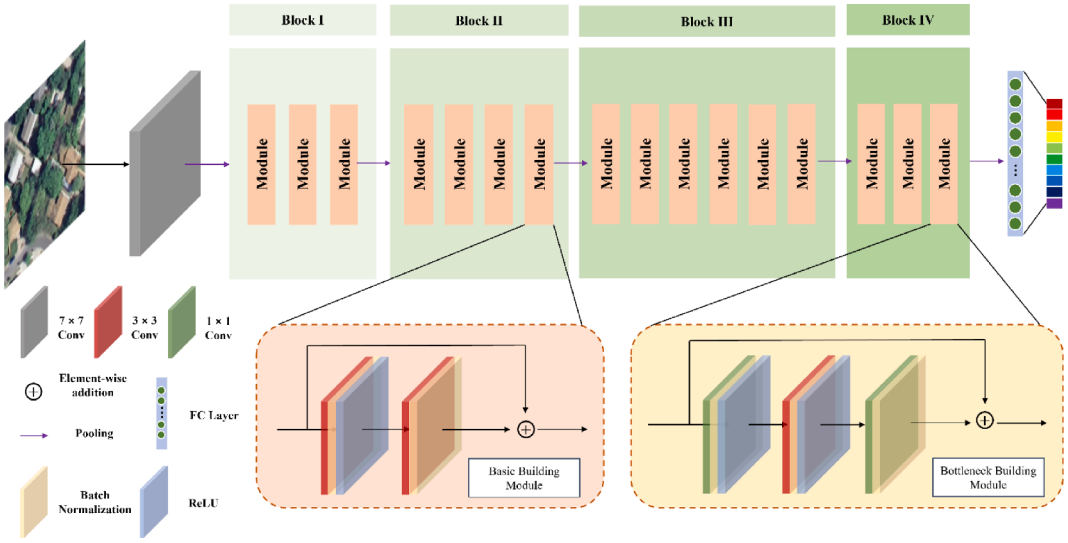


Fig. 3. Illustration of network architecture of ResNet34 and ResNet50. They share an identical architecture. The only difference is that ResNet34 applies basic building modules, whereas ResNet50 utilizes bottleneck building modules.

Table 1
Details of the data sets used in this study. Baseline CNNs are trained on these data sets from scratch.

Data set	Year	Classes	Number of samples			Spatial resolution	Image size
			Training	Validating	Testing		
UCMerced Land Use	2010	21	1,680	210	210	1 ft/pixel	256 × 256
Brazilian Coffee Scenes	2015	2	2,300	288	288	10 m/pixel	64 × 64
SIRI-WHU	2016	12	1,920	240	240	2 m/pixel	200 × 200
EuroSAT	2018	10	21,600	2,700	2,700	10 m/pixel	64 × 64
Hurricane Damage	2018	2	10,000	2,000	2,000	–	128 × 128
CLRS	2020	25	12,000	1,500	1,500	0.26–8.85 m/pixel	256 × 256

from each other concerning the release year, image size, total number of images, number of categories, and spatial resolution. For instance, UCMerced Land Use, SIRI-WHU, and CLRS are high-resolution RS image data sets, while Brazilian Coffee Scenes and EuroSAT feature small-patch and low-resolution RS images. EuroSAT is a large data set with 27,000 labeled images, whereas UCMerced Land Use, Brazilian Coffee Scenes, and SIRI-WHU only contain over 2,000 images. Unlike other data sets, images in Brazilian Coffee Scenes are not optical (green–red–infrared instead of red–green–blue). Due to their significant differences, these data sets represent a wide range of different RS scenarios. Thus, the experiment results are robust and generalizable.

3.2. Evaluation metrics

OA, Params, Param_size, and Flops are used to evaluate the overall effects of different architectural factors on model performance. OA is the percentage of correctly classified samples and all samples in the entire testing data set, and it can be expressed as follows:

$$OA = \frac{\sum_a P_{aa}}{\sum_a t_a}, \tag{4}$$

where $t_a = \sum_b P_{ab}$ computes the total number of patches that belong to class a, P_{ab} is the number of samples of class a that are predicted to belong to class b, and P_{aa} is the number of samples of class a that are correctly predicted to belong to class a.

Params and Param_size are often used to measure the consumption of computer memory by a CNN model. A larger Params/Param_size indicates more consumption of computer memory resources. For a convolutional layer with bias and batch normalization (BN) operation, if the input channel is C_i, the output channel is C_o, and the kernel size is $K \times K$, then the Params of this convolutional layer can be denoted as.

$$Params\_conv = K^2{*}C_i{*}C_o + 3{*}C_o. \tag{5}$$

For a fully connected layer with bias, if the number of input neurons is I and the number of output neurons is O, then its Params can be expressed as.

$$Params\_fc = I{*}O + O. \tag{6}$$

The conversion between Params and Param_size is as follows. Assuming that the Params of a CNN model is X, and each parameter is a floating-point number. Because each parameter occupies 4 bytes of memory, the total amount of occupied memory of this model is $4X$, which is $4X/(1024^2)$ megabytes (MB). In this study, the Param_size of each adopted network is automatically calculated by PyTorch toolkits.

Flops refers to the number of multiplication and addition operations in forward reasoning. It is widely used to evaluate the computational complexity of a model or an algorithm. The Flops of a convolutional layer with bias and BN can be denoted by formula 7, where H and W are the height and width of the output feature maps, respectively.

$$Flops\_conv = K^2{*}C_i{*}C_o{*}H{*}W \tag{7}$$

Correspondingly, the Flops of a fully connected layer with bias can be expressed using the following formula:

$$Flops\_fc = 2{*}I{*}O \tag{8}$$

In this paper, we record the OAs of schemes with different architectural factor settings and investigate how these factors affect model performance of CNNs. We also focus on the computational cost and memory requirement brought about by these factors. Suggestions that are useful for developing new CNN architectures are summarized based

on our experiment results.

3.3. Implementation details

The impacts of architectural factors on model performance can be evaluated by adjusting the network depth, width, and cardinality of the baseline CNNs. When one architectural factor of a baseline network is adjusted, the other factors remain unchanged to minimize their distracted effects. The training configurations (i.e., initial learning rate, batch size, total epochs, and dropout rate) of the three baseline architectures are set to the same.

3.3.1. Experimental setup for network depth

As illustrated in Figs. 2 and 3b, both VGG and ResNet contain several blocks, after which the size of feature maps will be reduced. To evaluate the influence of network depth on model performance, we strictly add or reduce convolutional layers/modules in the final block of these baseline networks. The potential effects of the feature map size are therefore eliminated.

For VGG, we train and test VGG14, VGG15, VGG16, VGG17, and VGG18 on six data sets. The numbers of convolutional layers in each block of these five networks are [2, 2, 3, 3, 1], [2, 2, 3, 3, 2], [2, 2, 3, 3, 3], [2, 2, 3, 3, 4], and [2, 2, 3, 3, 5], respectively. For ResNet_basic, we utilize ResNet34, 36, 38, 40, and 42 for comparison. The numbers of basic building modules in each block of these networks are [3, 4, 6, 3], [3, 4, 6, 4], [3, 4, 6, 5], [3, 4, 6, 6], and [3, 4, 6, 7], respectively. As for ResNet_bottleneck, ResNet50, 53, 56, 59, and 62 are adopted. ResNet_bottleneck and ResNet_basic have an identical structure (Fig. 3); therefore, the numbers of bottleneck building modules of ResNet50, 53, 56, 59, and 62 are the same as those of ResNet34, 36, 38, 40, and 42. Table 2 and 3 display the depth, number of convolutional layers or modules in each block, Params, Param_size, and Flops of the utilized VGG and ResNet models, respectively.

3.3.2. Experimental setup for network width

To investigate the influence of network width, VGG16, ResNet38, and ResNet56 are selected as representatives of VGG, ResNet_basic, and ResNet_bottleneck, respectively. For each selected model, we adjust the width by setting different channels for convolutional layers/modules in their blocks. For example, the channels of each block in a standard VGG16 are [64, 128, 256, 512, 512]. Three variant models of VGG16 are proposed, namely, VGG16_w1, VGG16_w2, and VGG16_w3, which contain channels of [16, 32, 64, 128, 128], [32, 64, 128, 256, 256], and [128, 256, 512, 1024, 1024], respectively. Together with VGG16, these four networks are tested on six data sets (Table 4). Similarly, we conducted experiments and tested ResNet38, ResNet38_w1, ResNet38_w2, and ResNet38_w3 (Table 5) as well as ResNet56, ResNet56_w1, ResNet56_w2, and ResNet56_w3 (Table 6).

As can be seen from Tables 4, 5, and 6, if we double the width of every block in a network, then its Params, Param_size, and Flops will increase drastically. This approach is computationally inefficient even if a higher accuracy could be achieved. Therefore, we conduct further experiments to investigate the performance of a baseline model with partly increased network widths. Take ResNet38 as an example (Table 5). The channels of each block in ResNet38 are [64, 128, 256, 512]. We propose four variants of ResNet38, i.e., ResNet38_1w ([128, 128, 256, 512]), ResNet38_2w ([64, 256, 256, 512]), ResNet38_3w ([64, 128, 512, 512]), and ResNet38_4w ([64, 128, 256, 1024]). On the basis of this strategy, we also propose variant networks for VGG16 and ResNet56, as presented in Table 4 and 6.

3.3.3. Experimental setup for network cardinality

In this study, we also select VGG16, ResNet38, and ResNet56 as baselines and discuss how network cardinality affects model performance. For each of the three baselines, the cardinality is 1. We increase the network cardinality from 1 to 64 (i.e., cardinality = 1, 2, 4, 8, 16, 32, 64) by setting the number of grouped convolutions (Xie et al., 2017) in each block and record their performance. For example, the cardinality of Resnet38_c4 and ResNet56_c32 is 4 and 32, respectively. That is, for all blocks of ResNet38_c4 and ResNet56_c32, their number of grouped convolutions is 4 and 32, respectively. An exception is the VGG models. Since the number of input channels must be divisible by the cardinality (i.e., the number of grouped convolutions), for VGG16 and its variants, the number of input channels in the first block is 3 (i.e., RGB three channels), which cannot be divisible by the pre-set cardinality. Thus, the cardinality of the first block of VGG16 and its variants remains unchanged. Take VGG16_c4 as an example. Except that the number of grouped convolutions of Block I is 1, the grouped convolutions of Block II, III, IV, and V are all 4. Tables 7 and 8 present detailed information of the baselines and their variants with different cardinalities.

A schematic process workflow of this paper is shown in Fig. 4 for the reader's convenience.

4. Experimental results

4.1. Influence of network depth

Fig. 5 displays the OA of VGGs with various network depths on six data sets. Overall, with the increase of model depth, the performance of VGG exhibits a trend of first increasing and then decreasing. For example, on UCMerced, VGG16 outperformed VGG14, VGG15, VGG17, and VGG18, achieving the highest OA (69.05%). A similar trend of model performance can be observed on both SIRI and CLRS (Fig. 5). Although the fluctuation ranges of OA curves on EuroSAT and Hurricane are much smaller than those on the other data sets, the model performance on these two data sets is in line with this trend. A slightly different case is Brazilian Coffee Scenes, on which VGGs showed an OA curve of rising, descending, and then rising again. In summary, increasing the network depth of VGG usually improves model performance for RS classification tasks. However, excessive depth could also lead to a decline in the overall accuracy.

Fig. 6 exhibits the performance of ResNet_basic networks (i.e., ResNet34–42). On the UCMerced and Brazilian Coffee Scenes data sets, when the network depth is increased, the OA curves increase first and then decrease. This result is consistent with our observations from the performance of VGGs. However, the performance improvement brought about by increasing the network depth is not prominent on CLRS and SIRI. In addition, a clear descending trend in terms of OA is observed when the number of network depth is larger than 38 on these two data sets. Model performance on EuroSAT and Hurricane is similar to that of

Table 2
VGGs with various network depths utilized in this study. We recorded the Params, Param_size, and Flops of the network on the UCMerced Land Use data set.

	Depth	Number of Convolutional Layers in Each Block					Params (Million)	Param_size (MB)	Flops
		Block I	Block II	Block III	Block IV	Block V			
VGG14	14	2	2	3	3	1	35.3	134.5	1.9×10^{10}
VGG15	15	2	2	3	3	2	37.6	143.5	2.0×10^{10}
VGG16	16	2	2	3	3	3	40.0	152.5	2.0×10^{10}
VGG17	17	2	2	3	3	4	42.3	161.5	2.1×10^{10}
VGG18	18	2	2	3	3	5	44.7	170.5	2.1×10^{10}

Table 3
Two types of ResNets with various network depths utilized in this study. We recorded the Params, Param_size, and Flops of the network on the UCMerced Land Use data set.

	Depth	Module Type	Numbers of Modules in Each Block				Params (Million)	Param_size (MB)	Flops
			Block I	Block II	Block III	Block IV			
ResNet34	34	basic	3	4	6	3	21.3	81.2	4.8×10^9
ResNet36	36	basic	3	4	6	4	26.0	99.2	5.1×10^9
ResNet38	38	basic	3	4	6	5	30.7	117.3	5.4×10^9
ResNet40	40	basic	3	4	6	6	35.5	135.3	5.7×10^9
ResNet42	42	basic	3	4	6	7	40.2	153.3	6.0×10^9
ResNet50	50	bottleneck	3	4	6	3	23.6	89.8	5.4×10^9
ResNet53	53	bottleneck	3	4	6	4	28.0	106.9	5.7×10^9
ResNet56	56	bottleneck	3	4	6	5	32.5	123.9	6.0×10^9
ResNet59	59	bottleneck	3	4	6	6	36.9	140.9	6.2×10^9
ResNet62	62	bottleneck	3	4	6	7	41.4	157.9	6.5×10^9

Table 4
VGG16 and its variants with different network widths utilized in this study. We recorded the Params, Param_size, and Flops of the network on the UCMerced Land Use data set.

	Number of Channels (Width) of Convolutional Layers in Each Block					Params (Million)	Param_size (MB)	Flops
	Block I	Block II	Block III	Block IV	Block V			
VGG16_w1	16	32	64	128	128	19.9	75.9	1.3×10^9
VGG16_w2	32	64	128	256	256	24.8	94.4	5.1×10^9
VGG16	64	128	256	512	512	40.0	152.5	2.0×10^{10}
VGG16_w3	128	256	512	1024	1024	92.5	352.9	8.0×10^{10}
VGG16_1w	128	128	256	512	512	40.2	153.2	2.9×10^{10}
VGG16_2w	64	256	256	512	512	40.8	155.6	3.0×10^{10}
VGG16_3w	64	128	512	512	512	45.0	171.7	3.7×10^{10}
VGG16_4w	64	128	256	1024	512	57.7	220.0	3.6×10^{10}
VGG16_5w	64	128	256	512	1024	64.9	247.5	2.4×10^{10}

Table 5
ResNe38 and its variants with different network widths utilized in this study. We recorded the Params, Param_size, and Flops of the network on the UCMerced Land Use data set.

	Module Type	Number of Channels (Width) of Modules in Each Block				Params (Million)	Param_size (MB)	Flops
		Block I	Block II	Block III	Block IV			
ResNet38_w1	basic	16	32	64	128	1.9	7.4	3.7×10^8
ResNet38_w2	basic	32	64	128	256	7.7	29.4	1.4×10^9
ResNet38	basic	64	128	256	512	30.7	117.3	5.4×10^9
ResNet38_w3	basic	128	256	512	1024	122.9	468.7	2.1×10^{10}
ResNet38_1w	basic	128	128	256	512	31.4	119.9	7.9×10^9
ResNet38_2w	basic	64	256	256	512	34.2	130.6	8.7×10^9
ResNet38_3w	basic	64	128	512	512	51.8	197.8	1.1×10^{10}
ResNet38_4w	basic	64	128	256	1024	95.8	365.3	9.6×10^9

Table 6
ResNet56 and its variants with different network widths utilized in this study. We recorded the Params, Param_size, and Flops of the network on the UCMerced Land Use data set.

	Module Type	Number of Channels (Width) of Modules in Each Block				Params (Million)	Param_size (MB)	Flops
		Block I	Block II	Block III	Block IV			
ResNet56_w1	bottleneck	16	32	64	128	2.1	7.8	4.1×10^8
ResNet56_w2	bottleneck	32	64	128	256	8.1	31.1	1.5×10^9
ResNet56	bottleneck	64	128	256	512	32.5	123.9	6.0×10^9
ResNet56_w3	bottleneck	128	256	512	1024	129.7	494.7	2.3×10^{10}
ResNet56_1w	bottleneck	128	128	256	512	33.2	126.8	8.7×10^9
ResNet56_2w	bottleneck	64	256	256	512	36.4	139.0	9.7×10^9
ResNet56_3w	bottleneck	64	128	512	512	55.0	210.0	1.1×10^{10}
ResNet56_4w	bottleneck	64	128	256	1024	98.9	377.2	1.0×10^{10}

VGGs on Brazilian Coffee Scenes, i.e., a trend of first increasing, then decreasing and then increasing again can be observed (Fig. 6).

Compared with VGG and ResNet_basic networks, the benefit of increasing depth is not significant for ResNet_bottleneck models (i.e., ResNet50–62) on the UCMerced, Brazilian Coffee Scenes, Hurricane, and CLRS data sets. As displayed in Fig. 7, ResNet50 achieved the highest OA on these four data sets, i.e., 65.95% on UCMerced, 89.58% on Brazilian Coffee Scenes, 97.10% on Hurricane, and 83.13% on CLRS. Although ResNet59 achieved the same OAs as ResNet50 on Brazilian Coffee Scenes and CLRS, the Params and Flops of Resnet59 are much larger than those of ResNet50. Therefore, excessive network depth is a burden rather than a plus for ResNet_bottleneck models on these data

Table 7
VGG16 and its variants with different network cardinalities. We recorded the Params, Param_size, and Flops of the network on the UCMerced Land Use data set.

	Cardinality (Number of grouped convolutions)	Params [Million]	Param_size (MB)	Flops
VGG16	1	40.0	152.5	2.0×10^{10}
VGG16_c2	2	32.6	124.5	1.1×10^{10}
VGG16_c4	4	29.0	110.6	7.0×10^{9}
VGG16_c8	8	27.1	103.6	4.8×10^{9}
VGG16_c16	16	26.2	100.1	3.7×10^{9}
VGG16_c32	32	25.8	98.3	3.2×10^{9}
VGG16_c64	64	25.5	97.4	2.9×10^{9}

Table 8
Two types of ResNets and their variants with different network cardinalities. We recorded the Params, Param_size, and Flops of the network on the UCMerced Land Use data set.

	Cardinality (Number of grouped convolutions)	Params [Million]	Param_size (MB)	Flops
ResNet38	1	30.7	117.3	5.4×10^{9}
ResNet38_c2	2	15.5	59.0	2.8×10^{9}
ResNet38_c4	4	7.8	29.9	1.5×10^{9}
ResNet38_c8	8	4.0	15.4	8.4×10^{8}
ResNet38_c16	16	2.1	8.1	5.2×10^{8}
ResNet38_c32	32	1.2	4.5	3.6×10^{8}
ResNet38_c64	64	0.7	2.6	2.7×10^{8}
ResNet56	1	32.5	123.9	6.0×10^{9}
ResNet56_c2	2	17.7	67.5	3.3×10^{9}
ResNet56_c4	4	10.3	39.3	2.0×10^{9}
ResNet56_c8	8	6.6	25.2	1.3×10^{9}
ResNet56_c16	16	4.8	18.2	1.0×10^{9}
ResNet56_c32	32	3.8	14.6	8.4×10^{8}
ResNet56_c64	64	3.4	12.9	7.6×10^{8}

sets.

4.2. Influence of network width

We conducted four groups of experiments for each baseline model to evaluate the influences of increasing the network width. Take VGG as an example. When the network width is increased from VGG16_w1 to VGG16, a slow rising trend of OA can be observed on EuroSAT and CLRS (Fig. 8). For example, VGG16_w1 achieved an OA of 96.43% on EuroSAT, whereas VGG16_w2 and VGG16 achieved an OA of 97.06% and 97.50%, respectively. This trend is more rapid on SIRI. The OA of VGG16_w1, VGG16_w2, and VGG16 on SIRI is 81.67%, 83.33%, and 85.42%, respectively. That is, when the network width is doubled, the OA is improved by approximately 2%. However, when we further doubled the width of VGG16 to VGG16_w3, model performance stagnated or decreased slightly on Brazilian Coffee Scenes (+0.35%), Hurricane (+0.30%), EuroSAT (+0.13%), SIRI (−0.42%), and CLRS (−1.34%).

Compared with VGG16, the model performance of ResNet38 and ResNet56 is more sensitive to network width. For ResNet38, when the network width is increased from [16, 32, 64, 128] to [64, 128, 256, 512], an ascending trend of OA is observed on UCMerced, Hurricane, SIRI, EuorSAT, and CLRS. The improvement of OA is approximately 3%–5% when the width is doubled (Fig. 9). This number is approximately 2%–6% for ResNet56 (Fig. 10). Similar to VGGs, when the network width is doubled from [64, 128, 256, 512] to [128, 256, 512, 1024], the performance of ResNet38 stagnates on UCMerced (+0.24%), EuroSAT (+1.00%), and CLRS (+0.47%) or degrades on Brazilian Coffee Scenes (−1.04%), Hurricane (−0.10%), SIRI (−1.25%). For ResNet56, the changes of OA are relatively large when its width is doubled, i.e., −7.62% on UCMerced, −0.44% on EuroSAT, −0.42% on SIRI, +0.05% on Hurricane, +2.2% on CLRS, and + 2.43% on Brazilian Coffee Scenes.

Therefore, when a network is relatively thin, doubling its width is an effective way to improve OA, which contributes to an improvement of approximately 2%–6%. However, this improvement becomes insignificant if we double the width of the standard baseline models (i.e., VGG16, ResNet38, and ResNet56) because standard baselines are the optimal architectures that have been adjusted and tested on many large data sets. Increasing the width hardly improves their performance, but leads to a large amount of extra redundancy in terms of Params and Flops. Thus, we conducted additional experiments to investigate whether a partly widened network performs well.

Figs. 11, 12, and 13 display the test accuracies of VGG16, ResNet38, ResNet56, and their partly widened variants on all the data sets. As shown in these figures, a partly widened baseline network can improve model performance and achieve a higher OA than the original model and its fully widened counterparts.

Fig. 11, together with Fig. 8, illustrates that VGG16_2w and VGG16_3w outperformed VGG16 and VGG16_w3 on Brazilian Coffee Scenes, Hurricane, SIRI, and EuroSAT. Notably, the Params of VGG16_2w and VGG16_3w are 40.8 million and 45.0 million, respectively, which are about 44.1% and 48.6% of that of VGG16_w3, respectively. In addition, the Flops of VGG16_2w and VGG16_3w account for only 37.5% and 46.3% of that of VGG16_w3. Figs. 9 and 12 illustrate that ResNet38_1w achieved the highest OA on UCMerced, i.e., 84.46%, outperforming ResNet38 (80.71%) and ResNet38_w3 (80.95%) by a large margin. It also exhibited a strong classification capacity on SIRI, EuroSAT, and CLRS. This partly widening strategy also works well with ResNet56 (Fig. 13). Several variant networks, such as ResNet56_1w, ResNet56_2w, and ResNet56_3w, achieved the best performance on four out of six data sets, namely, UCMerced, Hurricane, SIRI, EuroSAT, and CLRS. On Brazilian Coffee Scenes, the OA of ResNet56_2w is 89.24%. Although it was slightly outperformed by ResNet56_w3 (89.58%), the OA of ResNet56_2w is still much higher than that of ResNet56 (87.15%). This result is inspiring considering that the Params and Flops of ResNet56_2w are similar to those of ResNet56 and significantly smaller than those of ResNet56_w3 (Table 6).

4.3. Influence of network cardinality

Our experiments revealed that network cardinality is crucial for both model performance and efficiency. As shown in Tables 7 and 8, increasing cardinality considerably reduces the Params and Flops of a CNN model, leading to a lightweight CNN architecture that is suitable for deployment on memory-limited devices.

Figs. 14, 15, and 16 present the OA of the three types of baseline networks and their variants with different cardinalities. As can be seen from these figures, VGG16 and ResNet56 worked very well on the selected data sets when their cardinality is increased. Testing accuracies of VGG16 on UCMerced and SIRI are significantly improved, i.e., an improvement of 3.5%–10% on UCMerced and 1%–4% on SIRI,

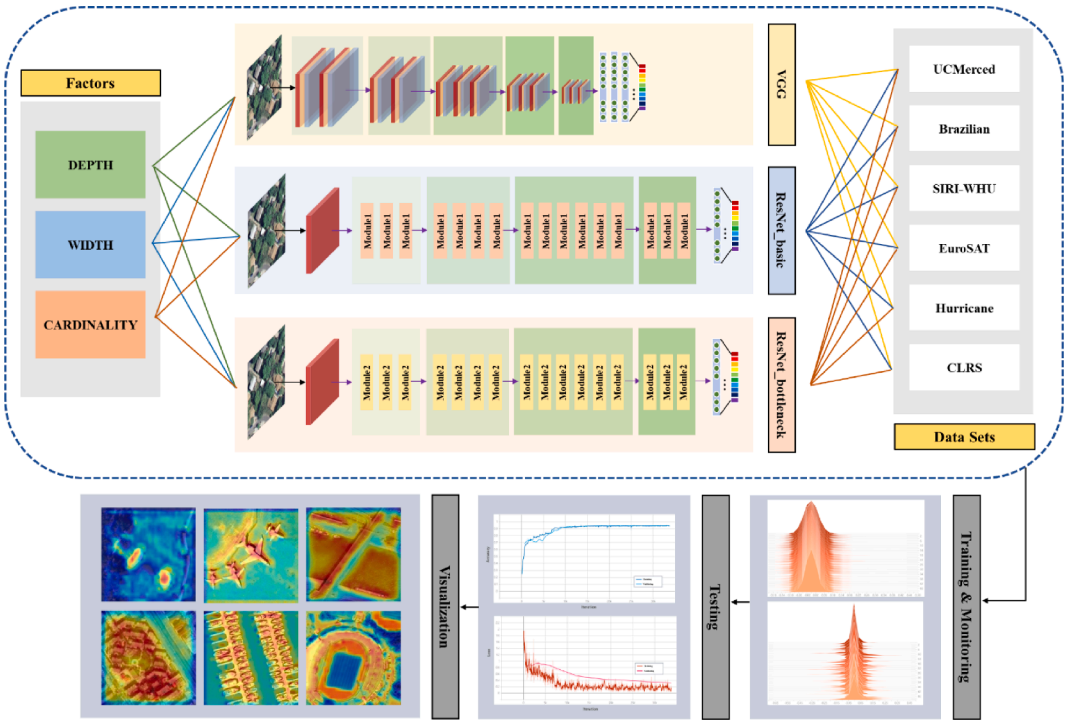
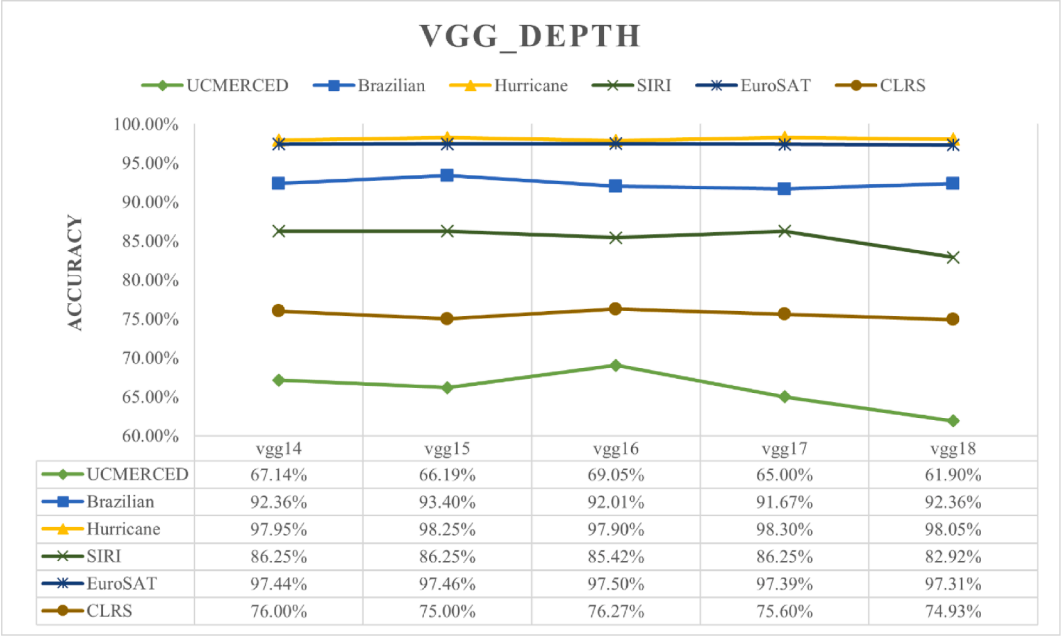


Fig. 4. Schematic process flow showing the implementation steps of this paper.



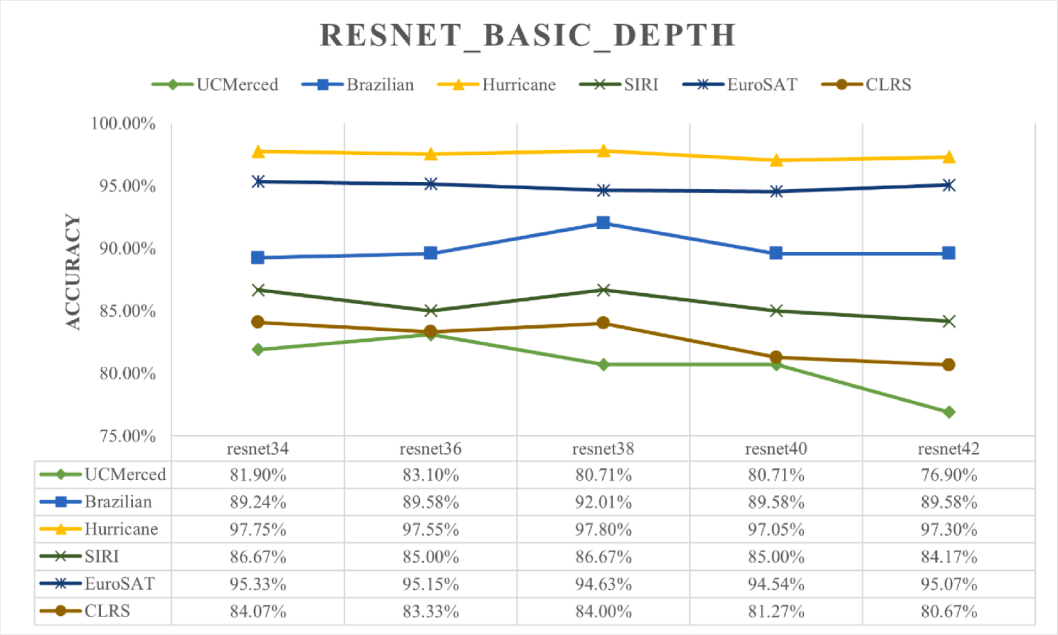
	vgg14	vgg15	vgg16	vgg17	vgg18
UCMERCED	67.14%	66.19%	69.05%	65.00%	61.90%
Brazilian	92.36%	93.40%	92.01%	91.67%	92.36%
Hurricane	97.95%	98.25%	97.90%	98.30%	98.05%
SIRI	86.25%	86.25%	85.42%	86.25%	82.92%
EuroSAT	97.44%	97.46%	97.50%	97.39%	97.31%
CLRS	76.00%	75.00%	76.27%	75.60%	74.93%

Fig. 5. Testing accuracy of VGG networks with different depth on six data sets.

respectively (Fig. 14). On Brazilian Coffee Scenes and Hurricane, a slight improvement of OA can be observed when the cardinality of VGG16 is increased. Although increasing network cardinality hardly improves the performance of VGG16 on EuroSAT and CLRS, it is still encouraging considering the large portion of reduced Params and Flops.

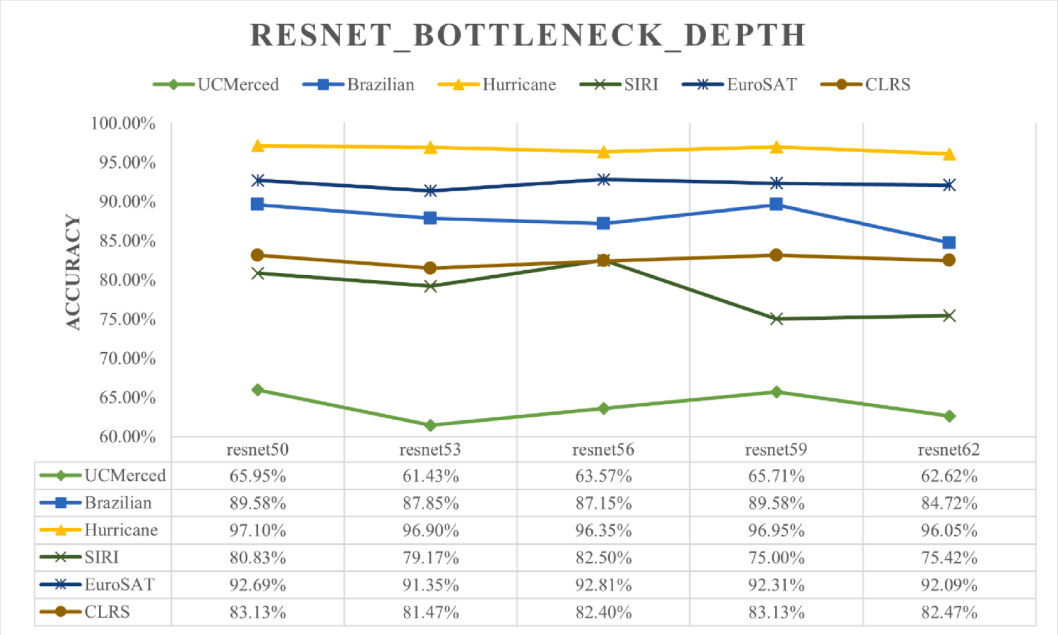
ResNet56 also benefits from this strategy (Fig. 15). On UCMerced, the improvement of OA brought about by increasing the cardinality is 3%–10%. On SIRI, the OA of ResNet56_c16 is 85.83%, which is 3% higher than that of ResNet56. On Brazilian, ResNet56_c4, ResNet56_c8, and ResNet56_c16 achieved the same testing accuracies, i.e., 88.89%, which outperformed that of ResNet56 by about 1.7%. On Hurricane, all the variant models except Resnet56_c8 achieved a higher OA than ResNet56. However, the advantage of increasing network cardinality is not significant on CLRS, i.e., only ResNet56_c2 (83.47%) outperformed RresNet56 (82.40%) by approximately 1%. On EuroSAT, a negative influence on model performance can be observed. ResNet56 achieved the highest OA (92.81%), marginally outperforming all its variants by about 0.5%–1%.

Increasing the network cardinality improved the classification results of ResNet38 to some extent, although they are not as impressive as



	resnet34	resnet36	resnet38	resnet40	resnet42
UCMerced	81.90%	83.10%	80.71%	80.71%	76.90%
Brazilian	89.24%	89.58%	92.01%	89.58%	89.58%
Hurricane	97.75%	97.55%	97.80%	97.05%	97.30%
SIRI	86.67%	85.00%	86.67%	85.00%	84.17%
EuroSAT	95.33%	95.15%	94.63%	94.54%	95.07%
CLRS	84.07%	83.33%	84.00%	81.27%	80.67%

Fig. 6. Testing accuracy of ResNet_basic networks with different depth on six data sets.



	resnet50	resnet53	resnet56	resnet59	resnet62
UCMerced	65.95%	61.43%	63.57%	65.71%	62.62%
Brazilian	89.58%	87.85%	87.15%	89.58%	84.72%
Hurricane	97.10%	96.90%	96.35%	96.95%	96.05%
SIRI	80.83%	79.17%	82.50%	75.00%	75.42%
EuroSAT	92.69%	91.35%	92.81%	92.31%	92.09%
CLRS	83.13%	81.47%	82.40%	83.13%	82.47%

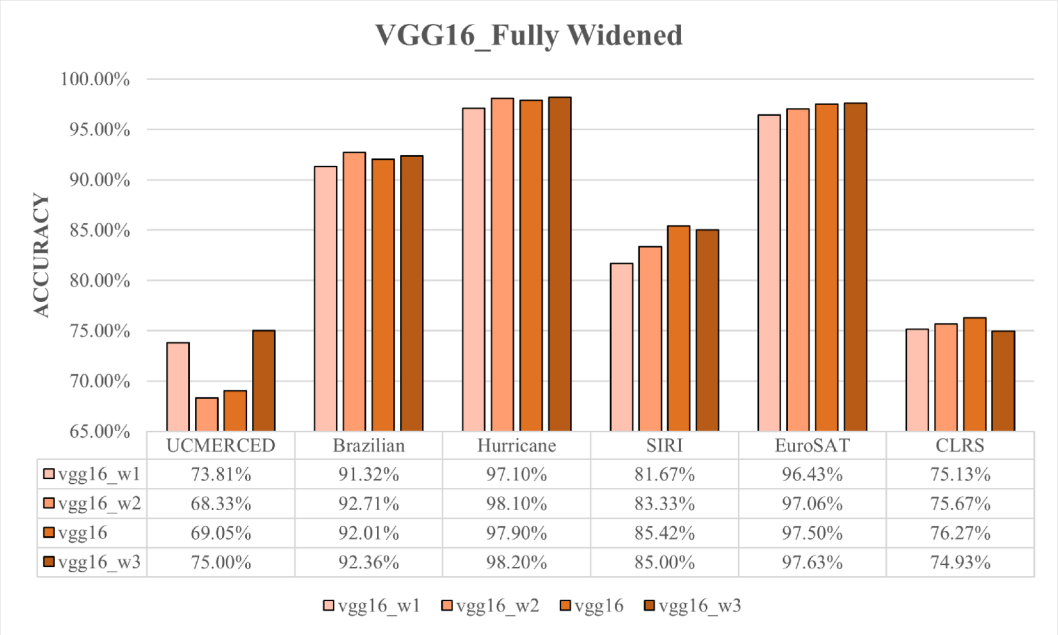
Fig. 7. Testing accuracy of ResNet_bottleneck models with different depth on six data sets.

VGG16 and ResNet56 (Fig. 16). The OA of ResNet38 on UCMerced is 80.71%, which is outperformed by all the variant models except ResNet38_c64. On EuroSAT, ResNet38_c2 and ResNet38_c4 exhibited satisfactory classification results, i.e., 95.41% and 95.07%, respectively, which is slightly higher than ResNet38 (94.63%). The cardinality-increased variants achieved comparable accuracies to ResNet38 on Hurricane, ranging from 97.00% to 97.70%. On Brazilian Coffee Scenes, Hurricane, SIRI, and CLRS, ResNet38 outperformed its variant networks (Fig. 16). This result is a trade-off between model performance and efficiency given that increasing the network cardinality can reduce the Params and Flops of the baseline model by at least 50%. Overall, increasing cardinality maintains or improves the model performance while reducing the network redundancy significantly.

5. Discussion

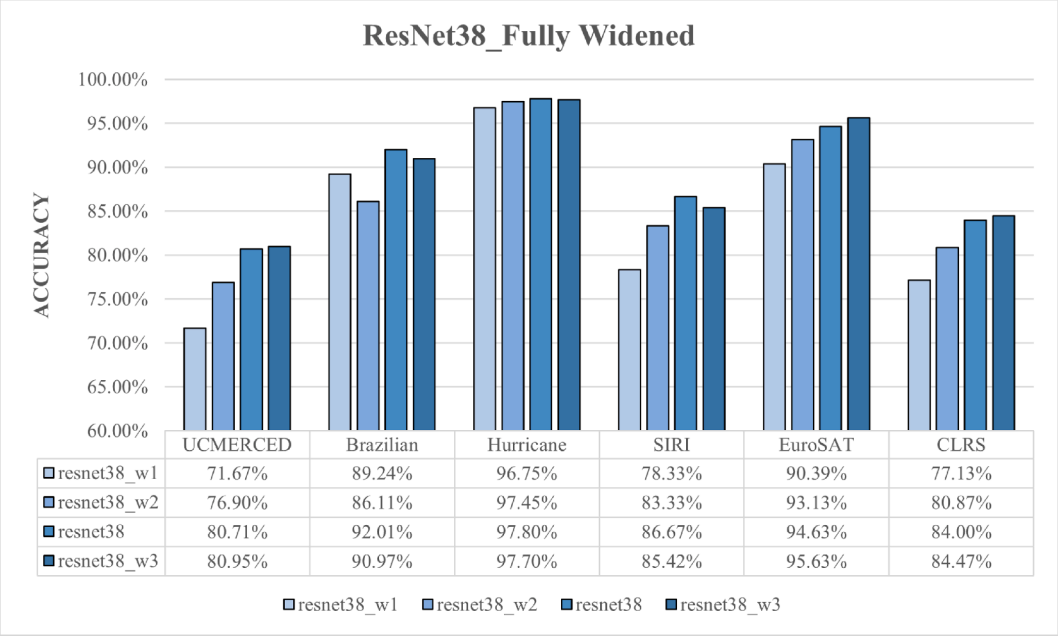
5.1. Rethinking the roles of network depth, width, and cardinality

Network depth and width are the two most widely concerned aspects of CNN architectures. For a long time, the development direction of CNNs has been deepening and widening. However, our experiments indicated that excessive depth or width could be more of a burden than a plus for CNN models on RS classification tasks. When the network depth is increased, the model performance of three baseline models exhibited a trend of first rising then decreasing. This result is in line with existing research from the computer science domain. Experimental results also suggested that shallow baseline networks benefit more than their deep



	UCMERCED	Brazilian	Hurricane	SIRI	EuroSAT	CLRS
vgg16_w1	73.81%	91.32%	97.10%	81.67%	96.43%	75.13%
vgg16_w2	68.33%	92.71%	98.10%	83.33%	97.06%	75.67%
vgg16	69.05%	92.01%	97.90%	85.42%	97.50%	76.27%
vgg16_w3	75.00%	92.36%	98.20%	85.00%	97.63%	74.93%

Fig. 8. Testing accuracy of VGG16 and its variants with different width on six data sets.

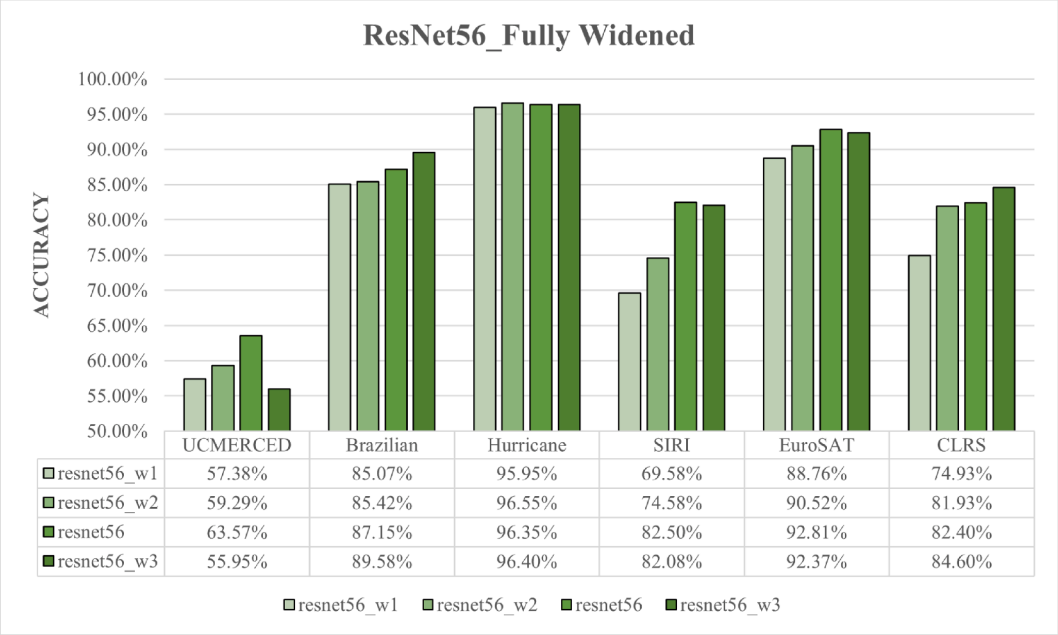


	UCMERCED	Brazilian	Hurricane	SIRI	EuroSAT	CLRS
resnet38_w1	71.67%	89.24%	96.75%	78.33%	90.39%	77.13%
resnet38_w2	76.90%	86.11%	97.45%	83.33%	93.13%	80.87%
resnet38	80.71%	92.01%	97.80%	86.67%	94.63%	84.00%
resnet38_w3	80.95%	90.97%	97.70%	85.42%	95.63%	84.47%

Fig. 9. Testing accuracy of ResNet38 and its variants with different width on six data sets.

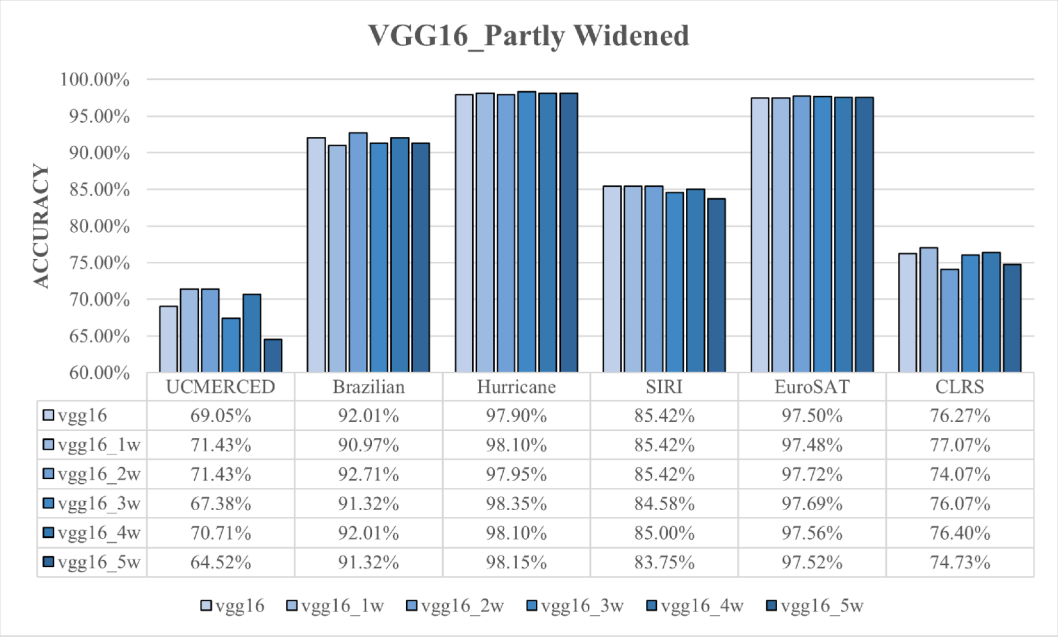
counterparts when the model depth is increased. For example, ResNet34 and ResNet50 have an identical structure; however, when the number of modules in their last block is increased from 3 to 5, the model performance of ResNet34 is improved on four out of six data sets, whereas the model performance of ResNet50 is improved on only two data sets.

In terms of network width, our experiments revealed that model performance of a thin baseline network is sensitive to width. For a thin model such as ResNet38_w1, doubling its width improves the OA by about 1%–5%. However, this improvement is negligible if we double the width of a standard network (e.g., ResNet38) and, in some cases, it may result in a decline in the model performance. In addition, doubling the network width of a standard baseline model will lead to a significant increase of Params and Flops, making it computationally inefficient. Given this situation, we argue that a partly widened baseline network is an optimal choice because it can achieve a satisfactory classification result while maintaining relatively low Params and Flops. Our comparison experiments also revealed that widening the first or second block of a baseline model contributes to the best model performance, e.g., VGG16_1w or VGG16_2w achieved the highest or second highest OA on almost all the utilized data sets. A plausible explanation is that widening the first few blocks of a baseline model helps the network learn richer shallow features. The importance of shallow features has long been underestimated. Standard baseline models follow the protocol of gradually widening the network from shallow to deep layers, e.g., the width of modules in each block of ResNet56 is [64, 128, 256, 512]. That is, only 64 and 128 channels are utilized in the first and second blocks,



	UCMERCED	Brazilian	Hurricane	SIRI	EuroSAT	CLRS
resnet56_w1	57.38%	85.07%	95.95%	69.58%	88.76%	74.93%
resnet56_w2	59.29%	85.42%	96.55%	74.58%	90.52%	81.93%
resnet56	63.57%	87.15%	96.35%	82.50%	92.81%	82.40%
resnet56_w3	55.95%	89.58%	96.40%	82.08%	92.37%	84.60%

Fig. 10. Testing accuracy of ResNet56 and its variants with different width on six data sets.



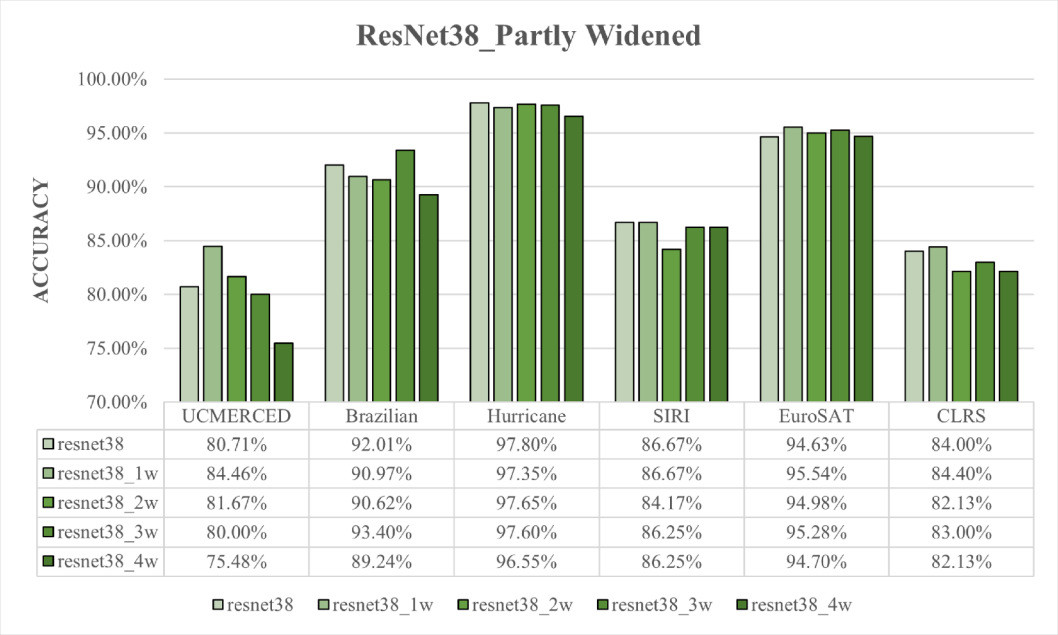
	UCMERCED	Brazilian	Hurricane	SIRI	EuroSAT	CLRS
vgg16	69.05%	92.01%	97.90%	85.42%	97.50%	76.27%
vgg16_1w	71.43%	90.97%	98.10%	85.42%	97.48%	77.07%
vgg16_2w	71.43%	92.71%	97.95%	85.42%	97.72%	74.07%
vgg16_3w	67.38%	91.32%	98.35%	84.58%	97.69%	76.07%
vgg16_4w	70.71%	92.01%	98.10%	85.00%	97.56%	76.40%
vgg16_5w	64.52%	91.32%	98.15%	83.75%	97.52%	74.73%

Fig. 11. Testing accuracy of VGG16 and its partly widened variants on six data sets.

which are inadequate to capture the rich shallow features such as texture and color. Our findings suggested that more channels should be utilized in the first few layers or blocks when developing a new CNN structure for RS classification tasks.

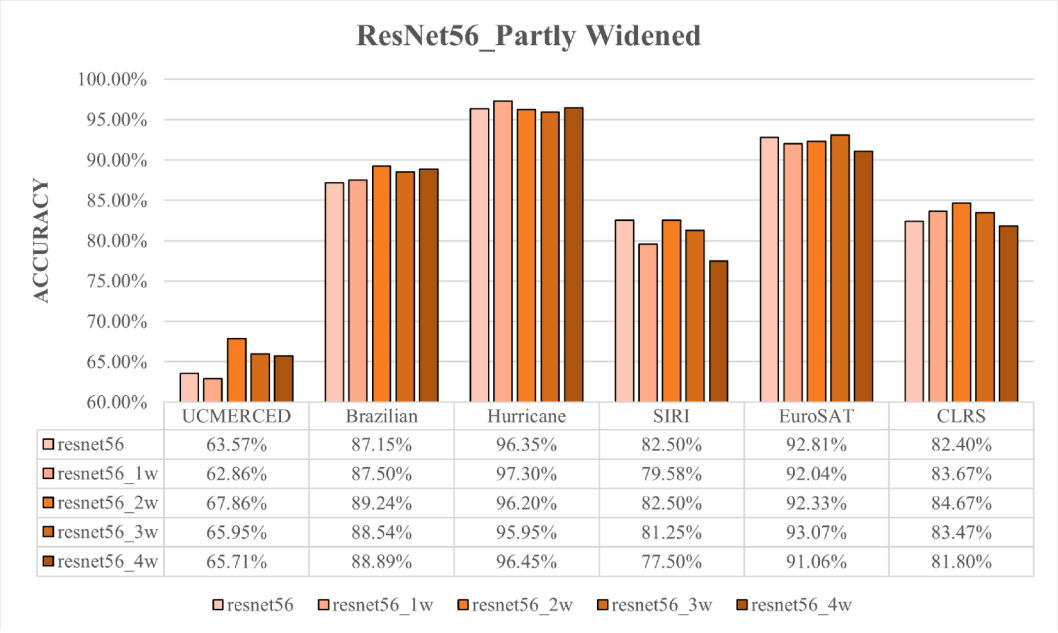
Network cardinality is a crucial factor to balance model performance and efficiency. Experimental results indicated that increasing the network cardinality of VGG16 and ResNet56 improves their performance on the utilized data sets. This improvement is not significant for ResNet38; however, it is still attractive considering the large amount of reduced Params and Flops. When we double the cardinality of a standard baseline network (e.g., VGG16 or ResNet38), at least 45% of Flops can be saved, which greatly reduces the training and testing time. In real-world applications, lightweight models that are suitable for resource-constrained environments are even more important than outstanding model performance. Therefore, we suggest that network cardinality and other architectural factors be considered jointly when adjusting CNN structures.

As increasing the width of a baseline network often leads to a huge increase in Params and Flops, we conducted additional experiments, i.e., doubling its width while increasing its cardinality, to investigate whether this modified network performances well. ResNet38 was adopted as the baseline model, and experiments were conducted on a randomly selected data set, i.e., EuroSAT. Table 9 presents the OA, Params, Param_size, and Flops of ResNet38 and its variant models. As displayed in the table, when we double the width of ResNet38 to ResNet38_w3, the OA is improved by 1%. However, this also leads to a



	UCMERCED	Brazilian	Hurricane	SIRI	EuroSAT	CLRS
resnet38	80.71%	92.01%	97.80%	86.67%	94.63%	84.00%
resnet38_1w	84.46%	90.97%	97.35%	86.67%	95.54%	84.40%
resnet38_2w	81.67%	90.62%	97.65%	84.17%	94.98%	82.13%
resnet38_3w	80.00%	93.40%	97.60%	86.25%	95.28%	83.00%
resnet38_4w	75.48%	89.24%	96.55%	86.25%	94.70%	82.13%

Fig. 12. Testing accuracy of ResNet38 and its partly widened variants on six data sets.



	UCMERCED	Brazilian	Hurricane	SIRI	EuroSAT	CLRS
resnet56	63.57%	87.15%	96.35%	82.50%	92.81%	82.40%
resnet56_1w	62.86%	87.50%	97.30%	79.58%	92.04%	83.67%
resnet56_2w	67.86%	89.24%	96.20%	82.50%	92.33%	84.67%
resnet56_3w	65.95%	88.54%	95.95%	81.25%	93.07%	83.47%
resnet56_4w	65.71%	88.89%	96.45%	77.50%	91.06%	81.80%

Fig. 13. Testing accuracy of ResNet56 and its partly widened variants on six data sets.

fourfold increase in Params, Param_size, and Flops, which is computationally inefficient. To balance the model accuracy and complexity, we adjust the network architecture to ResNet38_c4w3. Experimental results indicated that ResNet38_c4w3 outperformed the other networks and achieved the highest OA (95.69%) with approximately the same number of parameters and Flops as ResNet38 (Table 9).

5.2. *Opening the black box: an in-depth investigation*

To ensure the reliability of the aforementioned experimental results, we visualized the data distributions and gradients of the first and last convolutional layers of every utilized CNN model and observed whether they were fully trained and whether gradient vanishing and gradient exploding problems occurred. Fig. 17 exhibits the data distributions and gradients of ResNet38 on EuroSAT. As shown in this figure, due to the utilization of BN layers, data distributions of the first and last convolutional layers in each epoch are normalized to normal distributions (Fig. 17a and 17c). Correspondingly, with the increase of training epochs, the gradient values of the first and last convolutional layers are kept within a reasonable range (Fig. 17b and 17d), and no gradient vanishing or exploding problem occurs. Figs. 18a and 18b show the accuracy and loss of ResNet38, respectively. The training and validating accuracy curves increase rapidly and then reach a plateau after about 20,000 iterations (Fig. 18a). Correspondingly, the descent of training and validating loss slows down after 15,000 and 20,000 iterations, respectively, indicating that the baseline model has been fully trained.

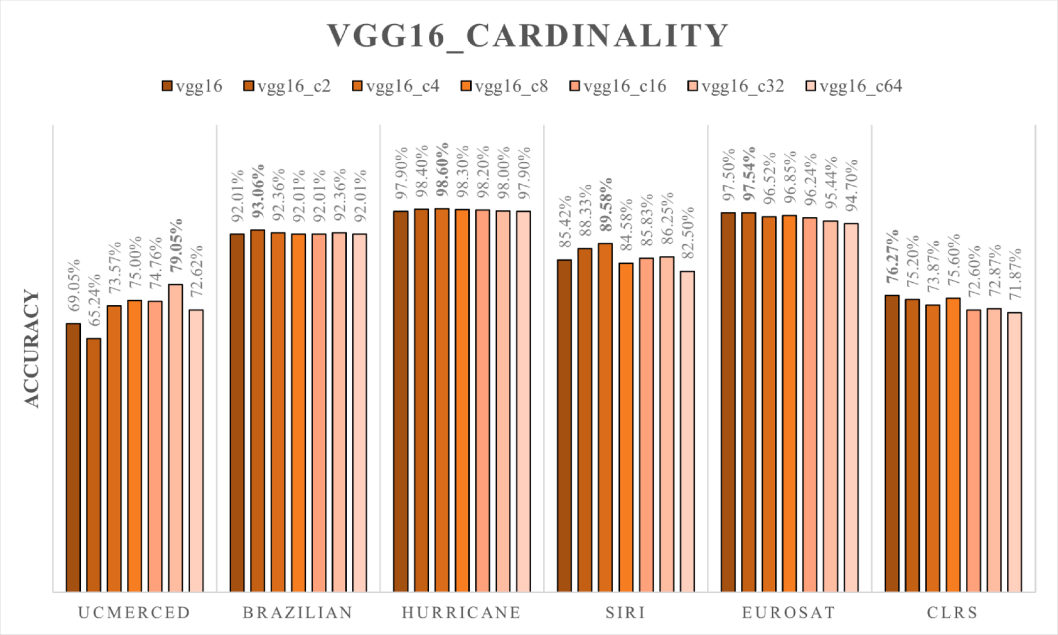


Fig. 14. Testing accuracy of VGG16 and its variants with different cardinality on six data sets.

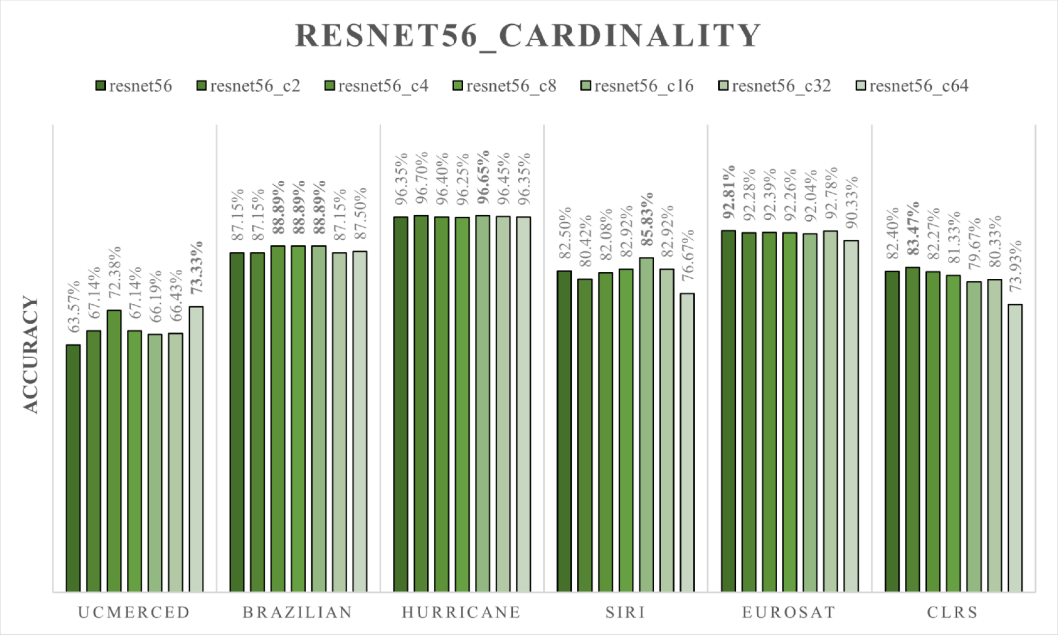


Fig. 15. Testing accuracy of ResNet56 and its variants with different cardinality on six data sets.

Grad-CAM is a class activation mapping approach for intuitive explanations from any CNN-based models, and it can highlight the discriminative regions in an image used to predict the image class (Ba et al., 2019; Wu et al., 2021). As one of the most interpretable and reliable artificial intelligence explanation methods (Kakogeorgiou and Karantzalos, 2021), Grad-CAM is applied in this study to generate the attention maps for baseline networks with different architectural factor settings.

As indicated by (X. Zhang et al., 2021), class activation maps from the middle block are the best representation for Grad-CAM visualizations. Given that five and four blocks of VGG and ResNet models exist, respectively, class activation maps from the third and second blocks of VGG and ResNet are utilized in this research.

Fig. 19 illustrates the attention maps of VGG networks with different depths on UCMerced. Six categories of UCMerced are selected, including airplane, dense residential, golf course, harbor, storage tanks, and tennis court. As presented in these figures, a deeper network helps better exploit semantic information. The attention maps of VGG14 have large and random blue areas that cover almost the entire image, and only a few areas around the main object are brightened with yellow and red (e.g., airplane, harbor, and storage tanks). VGG15 failed to capture meaningful semantic information on almost every class, and only some random points are highlighted in the attention maps. Unlike VGG14, VGG16 and VGG17 focused on the local spatial information. Some edge areas of the main objects are brightened, such as the tail wing of the airplane and the pipeline of the storage tanks. As for VGG18, the dark

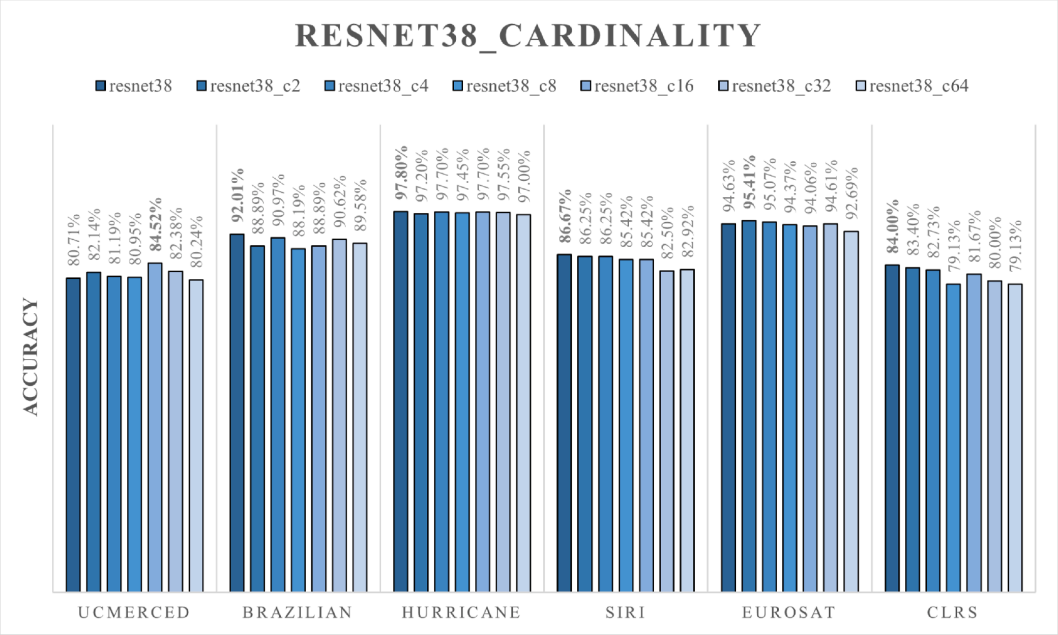


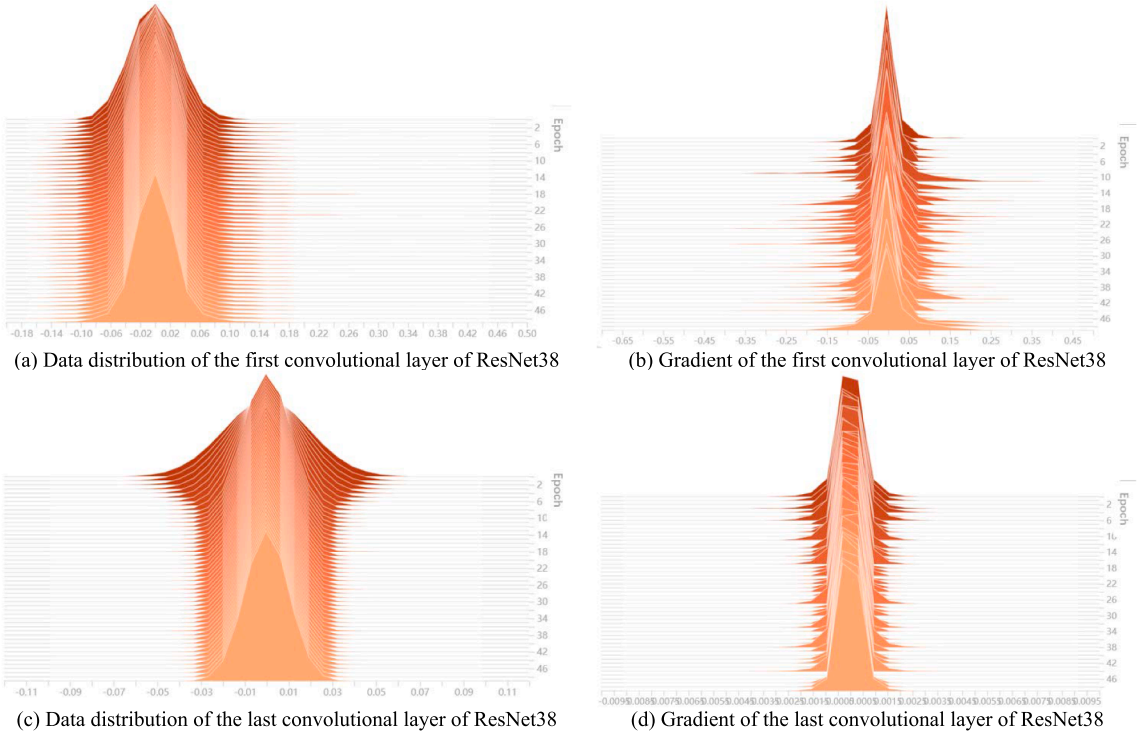
Fig. 16. Testing accuracy of ResNet38 and its variants with different cardinality on six data sets.

Table 9
Comparison of OA, Params, Param_size, and Flops of ResNet38 and its variants on the EuroSAT data set.

	OA	Params (Million)	Param_size (MB)	Flops
ResNet38	94.63%	30.7	117.3	5.4×10^9
ResNet38_w3	95.63%	122.9	468.7	2.1×10^{10}
ResNet38_c4	95.07%	7.8	29.9	1.5×10^9
ResNet38_c4w3	**95.69%**	31.3	119.4	5.7×10^9

blue areas in the attention maps decrease significantly, and the entire body instead of some edges of the main objects is brightened with red and yellow. This finding indicates that VGG18 can successfully capture both the local spatial information and global semantic information.

Fig. 20 are the activation maps of ResNet38 and its variants with different width on SIRI-WHU. The first row of Fig. 20 exhibits the original images of five selected categories, namely, idle land, industrial, overpass, pond, and residential. From the second to fifth rows, we can conclude that network width improves the semantic information mining capacity to a certain extent (e.g., attention maps of Residential).



(a) Data distribution of the first convolutional layer of ResNet38

(b) Gradient of the first convolutional layer of ResNet38

(c) Data distribution of the last convolutional layer of ResNet38

(d) Gradient of the last convolutional layer of ResNet38

Fig. 17. Data distributions and gradients of the first and last convolutional layers of ResNet38 on EuroSAT. The vertical axis represents the epochs, and the horizonal axis denotes the value number.

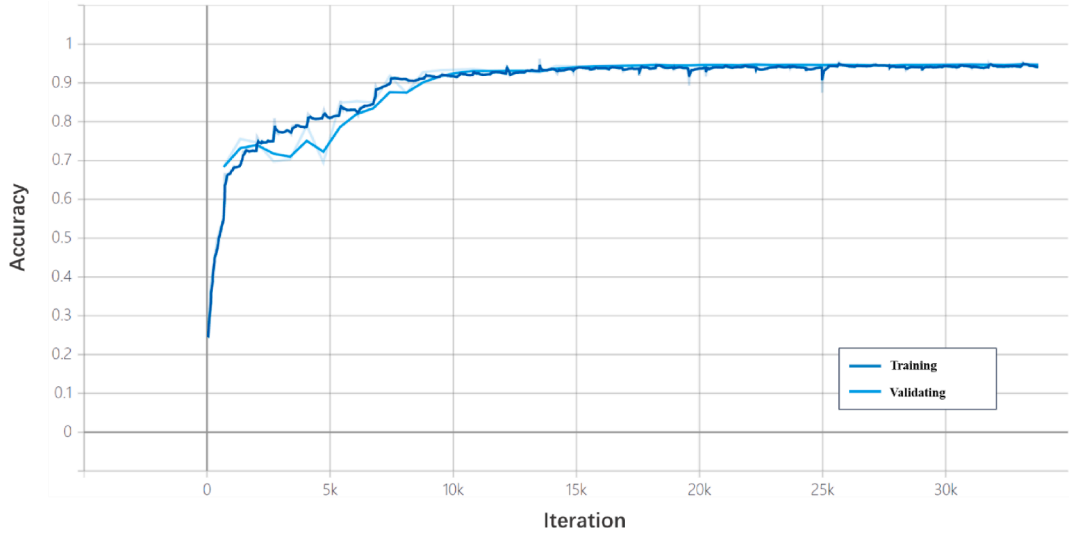


Fig. 18a. Training and testing accuracy of ResNet38 on EuroSAT. The vertical axis is the OA, and the horizontal axis is the number of iterations.

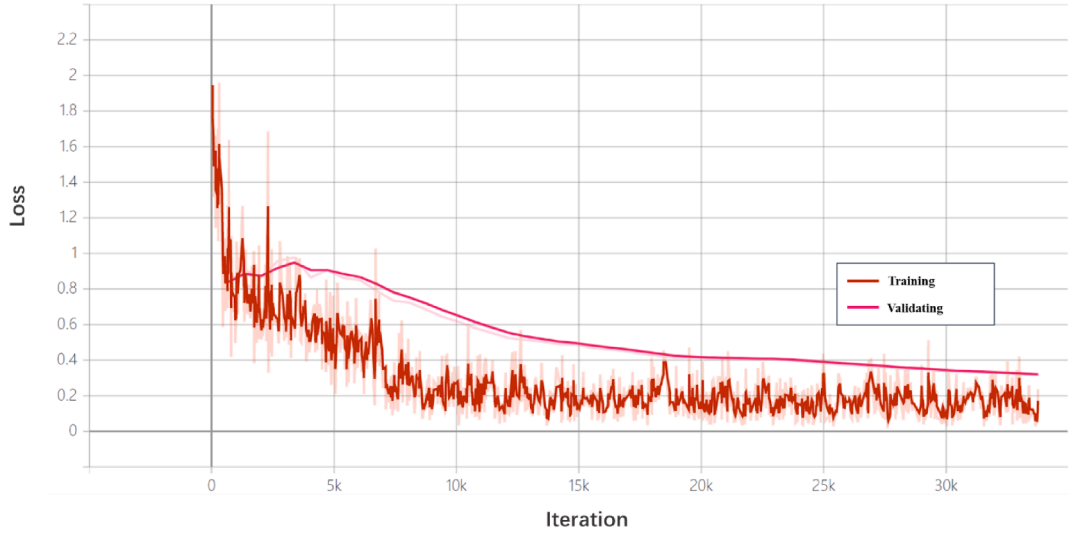


Fig. 18b. Training and testing loss of ResNet38 on EuroSAT. The vertical axis is the loss, and the horizontal axis is the number of iterations.

However, excessive width also impedes the model from focusing on the right areas (e.g., attention maps of Industrial). Compared with ResNet38_w3, some partly widened variants can better "understand" the semantic meanings of the input images. For example, ResNet38_1w is the only network that correctly highlight the water surface of pond (Fig. 20). Unlike ResNet38_w3, ResNet38_3w and ResNet38_4w can exploit the background information and focus on the roof of industrial buildings. Therefore, the attention maps from rows 5 to 9 indicate that a partly widened network is comparable to or even better than a fully widened network in terms of semantic information mining capacity.

Fig. 21 presents the attention maps of ResNet_bottleneck models with different network cardinalities on CLRS. The selected classes of CLRS for visualization include beach, bridge, parking, playground, river, and stadium. In general, ResNet56 has a strong semantic feature learning capacity and focuses on the discriminative regions of the scene. However, some cardinality-increased variants generated more satisfactory attention maps than ResNet56 on some categories. An example is ResNet56_c16; from its activation map of the river, we can see that the river surface is clearly highlighted with red. In addition, for stadium, ResNet56 mainly brightened the roof areas, whereas ResNet56_c16 focused on the roof and grandstand. Therefore, increasing network cardinality will not necessarily hinder the semantic feature learning ability of a CNN model. In some cases, a cardinality-increased network can have a better "understanding" of the global and semantic information of an input image than the original model.

5.3. *Rules of Thumb: Some useful Suggestions*

On the basis of our experimental results, some useful suggestions can be summarized for those who are interested in improving the CNN performance or developing novel CNN structures for RS tasks:

(1) Network depth is important to model performance as it helps the model focus on both the local spatial information and global semantic information. However, excessive depth could be more of a burden than a plus for CNN models. The optimal depths for different models on different data sets are not the same; thus, we suggest determining the suitable model depth by trial and error. When a CNN structure is adjusted, network depth could be the first architectural factor for consideration because adding or removing some layers will not lead to considerable changes in Params and Flops.

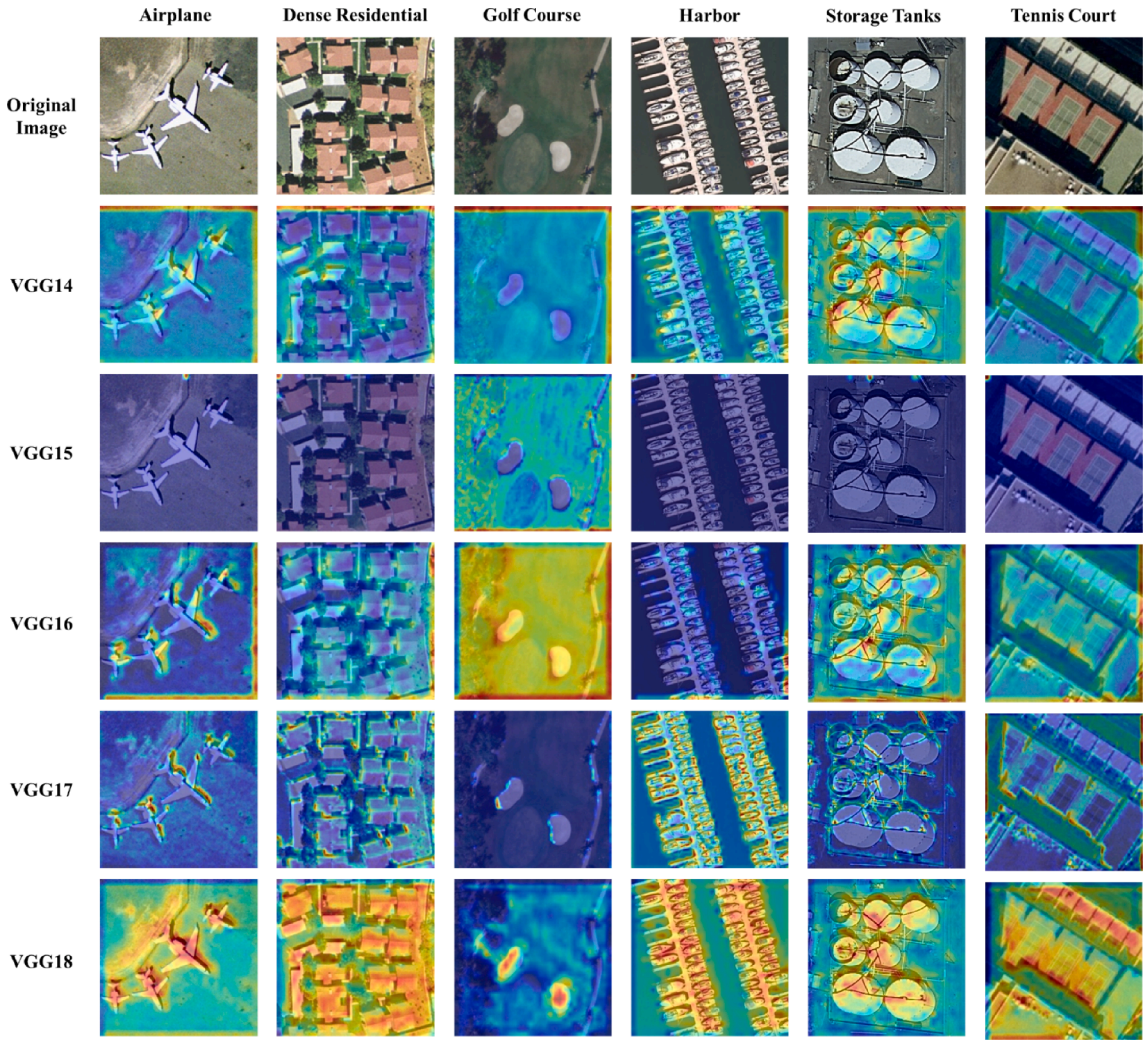


Fig. 19. Attention maps of VGG models with different network depth on UCMerced. These maps are derived from the third block of VGG models.

(2) Mainstream architectures of CNN models follow the protocol of gradually widening the network from shallow to deep layers. If a thin model is adopted as the backbone, then doubling the network width is an effective way to improve model performance. Our experiments on six public RS data sets reveal that thin models are sensitive to network width. For a thin ResNet, doubling its width improves the OA by approximately 2%–6%.

(3) However, when a standard CNN is utilized as the backbone model, doubling its width is not an optimal choice. Experimental results indicate that this approach will lead to a negligible improvement of OA, and, in some cases, it will hinder the model performance. Considering the huge increase of Params and Flops, directly doubling the width of a standard network is computationally inefficient. In that case, a partly widening strategy is suggested. Our research indicates that low- and middle-level features are significant to RS classification results, but they have long been overlooked. More shallow features can be learned by widening the first few blocks of a standard model.

(4) Cardinality is an essential architectural factor for designing new CNN structures. The industry has long been concerned with how to strike a balance between model efficiency and accuracy. Increasing the network cardinality can significantly reduce the model complexity at the expense of a slight decrease in OA. Therefore, we propose to increase the network cardinality while deepening or widening the CNN model. Our additional experiments illustrate that a widened baseline model with increased cardinality improves model performance while maintaining a relatively low number of Params and Flops.

6. Conclusion

In this study, we propose an empirical formula for CNN model performance based on existing literature. Six public RS data sets representing a wide range of RS scenarios are selected, and extensive experiments are conducted on these data sets using two types of widely used CNN architectures, namely, VGG and ResNet. We mainly focus on three architectural factors (i.e., network depth, width, and cardinality) and investigate how they affect model performance. To ensure the reliability of our experiments, we monitor the data distributions and gradients of the baseline models, thus preventing the gradient vanishing or exploding problem. Grad-CAM is also adopted to open the black box of CNN models and illustrate their interpretation process.

On the basis of the experimental results, some rules of thumb are summarized for those who are interested in improving CNN model performance or developing novel CNN architectures for RS tasks. Our experiments highlight the importance of network depth and suggest that it could be the first factor for consideration when adjusting or developing a CNN architecture. The optimal model depth of a network on a specific data set could be determined by trial and error. In addition, we proposed a partly widening strategy and conduct comparative experiments to verify its effectiveness. Results show that a partly widened model can outperform its fully widened counterparts with considerably

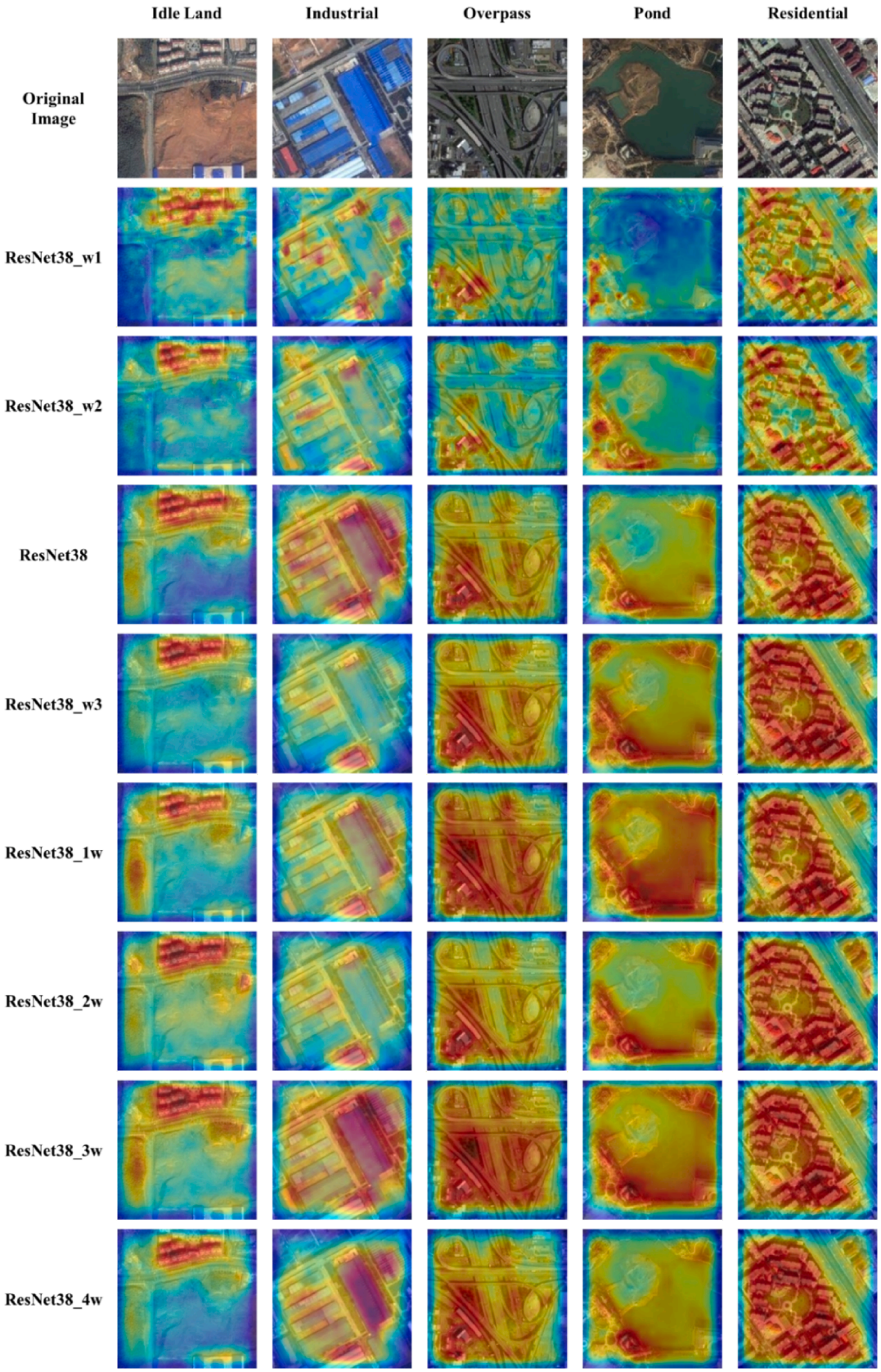


Fig. 20. Attention maps of ResNet38 and its variants with different network depth on SIRI-WHU. These maps are derived from the second block of ResNet models.

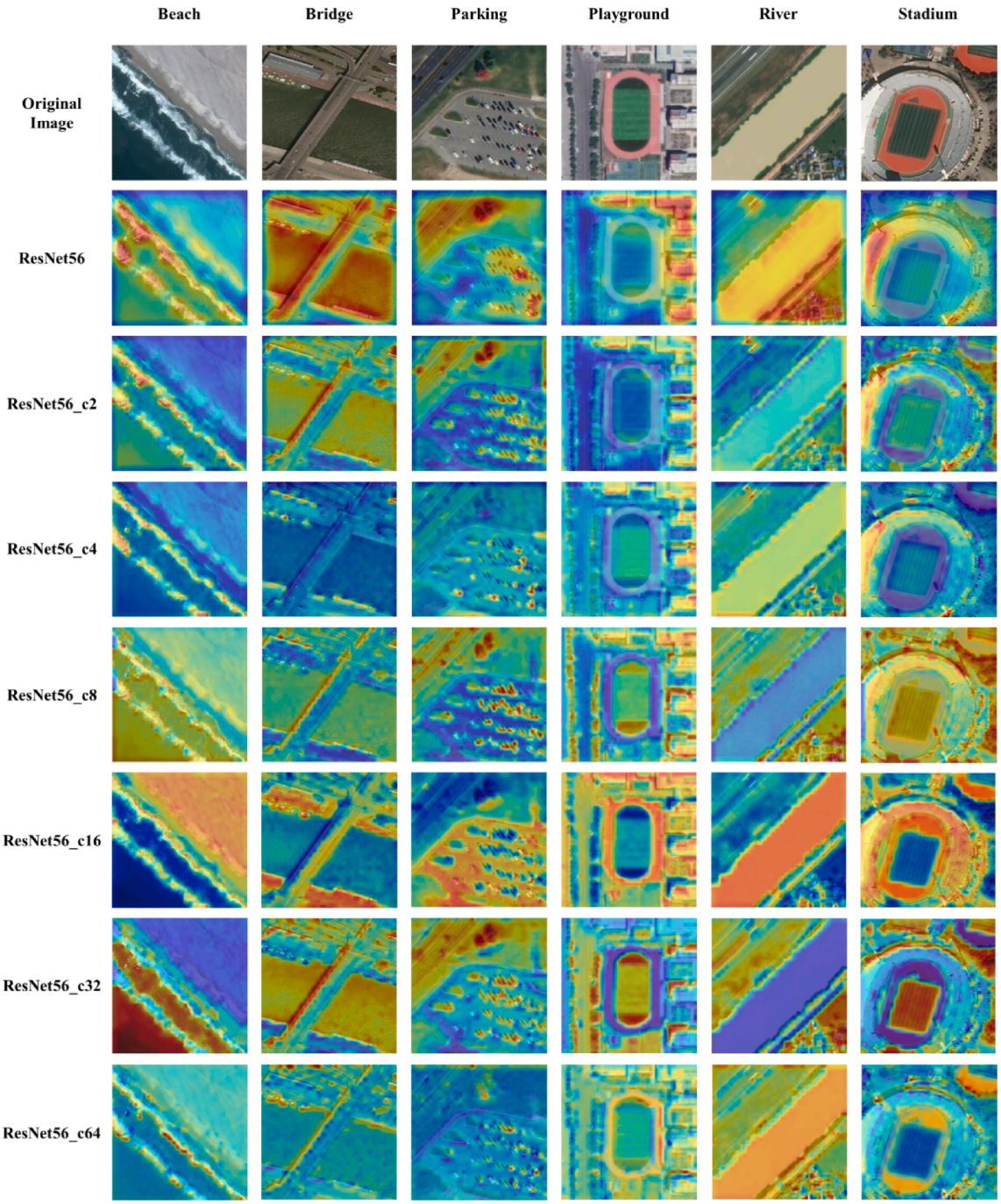


Fig. 21. Attention maps of ResNet56 and its variants with different network cardinality on CLRS. These maps are derived from the second block of ResNet models.

fewer Params and Flops. Moreover, increasing network cardinality is an effective way to reduce the network complexity. Therefore, we suggest that network cardinality and other architectural factors be adjusted jointly. Additional experiments reveal that a widened network with increased cardinality achieved satisfactory classification results. The findings of this study are believed to be robust and generalizable. Thus, they could be pivotal to researchers and practitioners who are interested in improving model performance or proposing novel CNNs for RS classification tasks.

CRediT authorship contribution statement

Feihao Chen: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing – original draft, Visualization. **Jin Yeu Tsou:** Conceptualization, Methodology, Resources, Writing – review & editing, Supervision, Project administration.

Declaration of Competing Interest

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

References

Ba, R., Chen, C., Yuan, J., Song, W., Lo, S., 2019. SmokeNet: Satellite smoke scene detection using convolutional neural network with spatial and channel-wise attention. Remote Sens. 11 (14), 1702.

Ball, J.E., Anderson, D.T., Chan Sr, C.S., 2017. Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community. J. Appl. Remote Sens. 11 (4), 042609.
Bengio, Y., 2013. Deep learning of representations: Looking forward. In: Paper presented at the International conference on statistical language and speech processing.
Beyer, L., Hénaff, O.J., Kolesnikov, A., Zhai, X., Oord, A.V.D., 2020. Are we done with imagenet? arXiv preprint arXiv:.07159.
Cao, C., Dragićević, S., Li, S., 2019. Land-use change detection with convolutional neural network methods. Environments 6 (2), 25.
Cao, Q.D., Choe, Y., 2020. Building damage annotation on post-hurricane satellite imagery based on convolutional neural networks. Nat. Hazards 103 (3), 3357–3376.
Castelluccio, M., Poggi, G., Sansone, C., Verdoliva, L., 2015. Land use classification in remote sensing images by convolutional neural networks. arXiv preprint arXiv: 1508.00092.
Chen, F., Tsou, J.Y., 2021a. DRSNet: Novel architecture for small patch and low-resolution remote sensing image scene classification. Int. J. Appl. Earth Observ. Geoinform. 104, 102577. https://doi.org/10.1016/j.jag.2021.102577.
Chen, F., Tsou, J.Y., 2022. Mapping urban form and land use with deep learning techniques: a case study of Dongguan City. China. Int. J. Oil, Gas Coal Technol. 29 (3), 306–328.
Cheng, G., Xie, X., Han, J., Guo, L., Xia, G.-S., 2020. Remote sensing image scene classification meets deep learning: Challenges, methods, benchmarks, and opportunities. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 13, 3735–3756.
Das, M., Ghosh, S.K., 2017. A deep-learning-based forecasting ensemble to predict missing data for remote sensing analysis. IEEE J. Selected Topics Appl. Earth Observ. Remote Sens. 10 (12), 5228–5236.
Dollár, P., Singh, M., Girshick, R., 2021. Fast and accurate model scaling. Paper presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.
Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Gelly, S., 2020. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:.11929.
Hanin, B., Sellke, M., 2017. Approximating continuous functions by relu nets of minimal width. arXiv preprint arXiv:.11278.
He, H., Chen, M., Chen, T., Li, D., 2018. Matching of remote sensing images with complex background variations via Siamese convolutional neural network. Remote Sens. 10 (2), 355.
He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
Helber, P., Bischke, B., Dengel, A., Borth, D., 2018. Introducing eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. Paper presented at the IGARSS 2018–2018 IEEE international geoscience and remote sensing symposium.
Hu, F., Xia, G.-S., Hu, J., Zhang, L., 2015. Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery. Remote Sens. 7 (11), 14680–14707.
Jia, Y., Ge, Y., Chen, Y., Li, S., Heuvelink, G., Ling, F., 2019. Super-resolution land cover mapping based on the convolutional neural network. Remote Sens. 11 (15), 1815.
Jiang, H., Lu, N., Qin, J., Tang, W., Yao, L., 2019. A deep learning algorithm to estimate hourly global solar radiation from geostationary satellite data. Renew. Sustain. Energy Rev. 114, 109327. https://doi.org/10.1016/j.rser.2019.109327.
Jin, X., Li, Z., Feng, H., Ren, Z., Li, S., 2020. Deep neural network algorithm for estimating maize biomass based on simulated Sentinel 2A vegetation indices and leaf area index. Crop J. 8 (1), 87–97.
Kakogeorgiou, I., Karantzalos, K., 2021. Evaluating explainable artificial intelligence methods for multi-label deep learning classification tasks in remote sensing. Int. J. Appl. Earth Observ. Geoinform. 103, 102520. https://doi.org/10.1016/j.jag.2021.102520.
Kattenborn, T., Leitloff, J., Schiefer, F., Hinz, S., 2021. Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. ISPRS J. Photogramm. Remote Sens. 173, 24–49.
Kawaguchi, K., Huang, J., Kaelbling, L.P., 2019. Effect of depth and width on local minima in deep learning. Neural Comput. 31 (7), 1462–1498.
Khan, A., Sohail, A., Zahoora, U., Qureshi, A.S., 2020. A survey of the recent architectures of deep convolutional neural networks. Artif. Intell. Rev. 53 (8), 5455–5516.
Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Adv. Neural Inform. Process. Syst. 25, 1097–1105.
Kumar, N., Kaur, N., Gupta, D., 2020. Major Convolutional Neural Networks in Image Classification: A Survey. Paper presented at the Proceedings of International Conference on IoT Inclusive Life (ICIIL 2019), NITTTR Chandigarh, India.
Li, H., Jiang, H., Gu, X., Peng, J., Li, W., Hong, L., Tao, C., 2020. CLRS: Continual learning benchmark for remote sensing image scene classification. Sensors 20 (4), 1226.
Li, Y., Zhang, H., Xue, X., Jiang, Y., Shen, Q., 2018. Deep learning for remote sensing image classification: A survey. Wiley Interdisciplinary Rev.: Data Min. Knowledge Discovery 8 (6), e1264.
Liu, S., Shi, Q., 2020. Local climate zone mapping as remote sensing scene classification using deep learning: A case study of metropolitan China. ISPRS J. Photogramm. Remote Sens. 164, 229–242.
Loshchilov, I., Hutter, F., 2018. Fixing weight decay regularization in adam.
Lu, Z., Pu, H., Wang, F., Hu, Z., Wang, L., 2017. The expressive power of neural networks: A view from the width. Adv. Neural Inform. Process. Syst., 30.
Ma, L., Liu, Y.u., Zhang, X., Ye, Y., Yin, G., Johnson, B.A., 2019. Deep learning in remote sensing applications: A meta-analysis and review. ISPRS J. Photogramm. Remote Sens. 152, 166–177.
Mohammadimanesh, F., Salehi, B., Mahdianpari, M., Gill, E., Molinier, M., 2019. A new fully convolutional neural network for semantic segmentation of polarimetric SAR imagery in complex land cover ecosystem. ISPRS J. Photogramm. Remote Sens. 151, 223–236.
Montufar, G.F., Pascanu, R., Cho, K., Bengio, Y., 2014. On the number of linear regions of deep neural networks. Adv. Neural Inform. Process. Syst. 27.
Neupane, B., Horanont, T., Aryal, J., 2021. Deep learning-based semantic segmentation of urban features in satellite images: A review and meta-analysis. Remote Sens. 13 (4), 808.
Nogueira, K., Penatti, O.A.B., dos Santos, J.A., 2017. Towards better exploiting convolutional neural networks for remote sensing scene classification. Pattern Recogn. 61, 539–556.
Osco, L.P., Marcato Junior, J., Marques Ramos, A.P., de Castro Jorge, L.A., Fatholahi, S. N., de Andrade Silva, J., Matsubara, E.T., Pistori, H., Gonçalves, W.N., Li, J., 2021. A review on deep learning in UAV remote sensing. Int. J. Appl. Earth Observ. Geoinform. 102, 102456. https://doi.org/10.1016/j.jag.2021.102456.
Penatti, O.A., Nogueira, K., Dos Santos, J.A., 2015. Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition workshops.
Qiu, C., Tong, X., Schmitt, M., Bechtel, B., Zhu, X.X., 2020. Multilevel Feature Fusion-Based CNN for Local Climate Zone Classification From Sentinel-2 Images: Benchmark Results on the So2Sat LCZ42 Dataset. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 13, 2793–2806.
Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., Aroyo, L.M., 2021. "Everyone wants to do the model work, not the data work": Data Cascades in High-Stakes AI. Paper presented at the proceedings of the 2021 CHI Conference on Human Factors in Computing Systems.
Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D., 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. Paper presented at the Proceedings of the IEEE international conference on computer vision.
Shao, Z., Cai, J., 2018. Remote sensing image fusion with deep convolutional neural network. IEEE J. Selected Topics Appl. Earth Observ. Remote Sens. 11 (5), 1656–1669.
Sherry, Y., Thompson, N.C., 2021. How fast do algorithms improve? Proc. IEEE 109 (11), 1768–1777.
Shorten, C., Khoshgoftaar, T.M., 2019. A survey on image data augmentation for deep learning. J. Big Data 6 (1), 1–48.
Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
Sutskever, I., Martens, J., Dahl, G., Hinton, G., 2013. On the importance of initialization and momentum in deep learning. Paper presented at the International conference on machine learning.
Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Rabinovich, A., 2015. Going deeper with convolutions. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
Tan, M., Le, Q., 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. Paper presented at the International Conference on Machine Learning.
Tong, X.-Y., Xia, G.-S., Lu, Q., Shen, H., Li, S., You, S., Zhang, L., 2018. Learning transferable deep models for land-use classification with high-resolution remote sensing images. *arXiv preprint arXiv:1807.05713*.
Wang, Q., Zhang, X., Chen, G., Dai, F., Gong, Y., Zhu, K., 2018a. Change detection based on Faster R-CNN for high-resolution remote sensing images. Remote Sens. Lett. 9 (10), 923–932.
Wang, S., Quan, D., Liang, X., Ning, M., Guo, Y., Jiao, L., 2018b. A deep learning framework for remote sensing image registration. ISPRS J. Photogramm. Remote Sens. 145, 148–164.
Wei, Y., Yuan, Q., Shen, H., Zhang, L., 2017. Boosting the accuracy of multispectral image pansharpening by learning a deep residual network. IEEE Geosci. Remote Sens. Lett. 14 (10), 1795–1799.
Wightman, R., Touvron, H., Jégou, H., 2021. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:.00476*.
Wu, P., Yin, Z., Yang, H., Wu, Y., Ma, X., 2019. Reconstructing geostationary satellite land surface temperature imagery based on a multiscale feature connected convolutional neural network. Remote Sens. 11 (3), 300.
Wu, X., Huang, T.-Z., Deng, L.-J., Zhang, T.-J., 2021. Dynamic Cross Feature Fusion for Remote Sensing Pansharpening. Paper presented at the Proceedings of the IEEE/CVF International Conference on Computer Vision.
Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K., 2017. Aggregated residual transformations for deep neural networks. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
Yang, C., Rottensteiner, F., Heipke, C., 2019. Towards better classification of land cover and land use based on convolutional neural networks. Int. Arch. Photogramm., Remote Sens. Spatial Inform. Sci.-ISPRS Arch. 42, Nr. 2/W13, 42(2/W13), 139-146.
Yang, J., Zhao, Y.-Q., Chan, J.-C.-W., 2018. Hyperspectral and multispectral image fusion via deep two-branches convolutional neural network. Remote Sens. 10 (5), 800.
Yang, Y., Newsam, S., 2010. Bag-of-visual-words and spatial extensions for land-use classification. Paper presented at the Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems.
Yao, C., Luo, X., Zhao, Y., Zeng, W., Chen, X., 2017. A review on image classification of remote sensing using deep learning. Paper presented at the 2017 3rd IEEE International Conference on Computer and Communications (ICCC).

Yoo, C., Han, D., Im, J., Bechtel, B., 2019. Comparison between convolutional neural networks and random forest for local climate zone classification in mega urban areas using Landsat images. ISPRS J. Photogramm. Remote Sens. 157, 155–170.

You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Hsieh, C.-J., 2019. Large batch optimization for deep learning: Training bert in 76 minutes. arXiv preprint arXiv:.00962.

Yuan, Q., Shen, H., Li, T., Li, Z., Li, S., Jiang, Y., Xu, H., Tan, W., Yang, Q., Wang, J., Gao, J., Zhang, L., 2020. Deep learning in environmental remote sensing: Achievements and challenges. Remote Sens. Environ. 241, 111716. https://doi.org/10.1016/j.rse.2020.111716.

Zeiler, M.D., Fergus, R., 2014. Visualizing and understanding convolutional networks. Paper presented at the European conference on computer vision.

Zhang, Q., Yuan, Q., Zeng, C., Li, X., Wei, Y., 2018. Missing data reconstruction in remote sensing image with a unified spatial–temporal–spectral deep convolutional neural network. IEEE Trans. Geosci. Remote Sens. 56 (8), 4274–4288.

Zhang, X., An, W., Sun, J., Wu, H., Zhang, W., Du, Y., 2021. Best representation branch model for remote sensing image scene classification. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 14, 9768–9780.

Zhao, B., Zhong, Y., Zhang, L., Huang, B., 2016. The Fisher kernel coding framework for high spatial resolution scene classification. Remote Sensing 8 (2), 157.