remote sensing

MDPI

Article

One-Shot Dense Network with Polarized Attention for Hyperspectral Image Classification

Haizhu Pan [1,*], Moqi Liu [1], Haimiao Ge [1] and Liguo Wang [2]

[1] College of Computer and Control Engineering, Qiqihar University, Qiqihar 161000, China; 2020935653@qqhru.edu.cn (M.L.); 01557@qqhru.edu.cn (H.G.)
[2] College of Information and Communication Engineering, Dalian Nationalities University, Dalian 116000, China; wangliguo@hrbeu.edu.cn
* Correspondence: panhaizhu@qqhru.edu.cn

Abstract: In recent years, hyperspectral image (HSI) classification has become a hot research direction in remote sensing image processing. Benefiting from the development of deep learning, convolutional neural networks (CNNs) have shown extraordinary achievements in HSI classification. Numerous methods combining CNNs and attention mechanisms (AMs) have been proposed for HSI classification. However, to fully mine the features of HSI, some of the previous methods apply dense connections to enhance the feature transfer between each convolution layer. Although dense connections allow these methods to fully extract features in a few training samples, it decreases the model efficiency and increases the computational cost. Furthermore, to balance model performance against complexity, the AMs in these methods compress a large number of channels or spatial resolutions during the training process, which results in a large amount of useful information being discarded. To tackle these issues, in this article, a novel one-shot dense network with polarized attention, namely, OSDN, was proposed for HSI classification. More precisely, since HSI contains rich spectral and spatial information, the OSDN has two independent branches to extract spectral and spatial features, respectively. Similarly, the polarized AMs contain two components: channel-only AMs and spatial-only AMs. Both polarized AMs can use a specially designed filtering method to reduce the complexity of the model while maintaining high internal resolution in both the channel and spatial dimensions. To verify the effectiveness and lightness of OSDN, extensive experiments were carried out on five benchmark HSI datasets, namely, Pavia University (PU), Kennedy Space Center (KSC), Botswana (BS), Houston 2013 (HS), and Salinas Valley (SV). Experimental results consistently showed that the OSDN can greatly reduce computational cost and parameters while maintaining high accuracy in a few training samples.

Keywords: hyperspectral image classification; convolutional neural network; one-shot dense networks; polarized attention mechanisms

Citation: Pan, H.; Liu, M.; Ge, H.; Wang, L. One-Shot Dense Network with Polarized Attention for Hyperspectral Image Classification. *Remote Sens.* **2022**, *14*, 2265. https://doi.org/10.3390/rs14092265

Academic Editor: Edoardo Pasolli

Received: 28 March 2022
Accepted: 6 May 2022
Published: 8 May 2022

1. Introduction

Benefiting from the increased spectral resolution of remote sensing sensors, the hyperspectral imaging technique shows great potential for obtaining high-quality land-cover information. Hyperspectral image (HSI) contains much spectral and spatial information, and each pixel contains hundreds of continuous and narrow spectral bands ranging from visible to near-infrared. Therefore, it has been widely used in many fields, such as urban planning [1], precision agriculture [2], and mineral exploration [3]. Among these applications, HSI classification is an important technical tool that aims to assign a unique class to each pixel [4]. However, due to the insufficient labeled samples and much redundant information, HSI classification remains a challenging task [5].

In the last decade, various methods have been proposed for HSI classification. These classification methods can be divided into two main categories: traditional machine-learning-based (ML-based) and modern deep-learning-based (DL-based) methods [6].

Generally, in ML-based methods, researchers first perform feature extraction on the raw HSI and then use classifiers to classify the extracted features. According to the types of features, they can be further divided into the spectral-based method and the spatial–spectral-based method. Commonly, the spectral-based method directly classifies the spectral vector of each pixel, such as random forest [7], k-nearest neighbors [8], and support vector machine (SVM) [9]. Moreover, many methods focus on reducing redundant spectral dimensions, which aim to map the high-dimensional spectral vector into a low-dimensional space, such as principal component analysis (PCA) [10], linear discriminant analysis [11], and independent component analysis [12]. However, it is difficult to identify the land-cover types using spectral features alone. The classification results are often filled with much salt-and-pepper noise. Alternatively, many researchers have discovered that spatial features can provide additional useful information for classification tasks. On the basis of this consideration, researchers have proposed a series of spatial–spectral-based methods for HSI classification, such as Gabor wavelet transform [13], local binary patterns [14], and morphological profiles [15]. Although the above methods can improve the classification accuracy, the feature extraction process relies on a priori knowledge and appropriate parameter settings. These limitations may affect the robustness and discrimination of the extracted features, making it difficult to achieve satisfactory results in complex scenarios [16].

In recent years, with the continuous improvement of computing power, the development of deep learning techniques has been greatly promoted. Deep neural network models can automatically extract highly robust and discriminative features from the raw data. They have made significant breakthroughs in many computer vision tasks, including image classification [17], semantic segmentation [18], and remote sensing image processing [19]. Naturally, in the field of HSI classification, research methods are gradually converging to state-of-art deep learning techniques. Currently, many effective classification models based on deep learning methods have been proposed. Chen et al. [20] proposed a stacked autoencoder deep neural network for spatial–spectral classification. It is the first application of DL-based methods to HSI classification. After that, many DL-based classification methods were proposed, and especially convolutional neural networks have attracted much attention.

Convolutional neural network (CNN) with multiple hidden layers has a powerful feature learning capability. It can provide more discriminative features with fine quality for HSI classification. Hu et al. [21] first used a one-dimensional (1-D) CNN to extract deep spectral features from each pixel for HSI classification. In addition, Yu et al. [22] proposed an improved 1-D CNN framework, which embeds pre-extracted hashing features in the network. To fully utilize the spatial context information, some two-dimensional (2-D) CNN has been applied to HSI classification and achieved desirable performance. Chen et al. [23] extracted the first principal component from the HSI data by PCA along the spectral dimension and then fed it into a 2-D CNN model to extract the spatial depth features. Yu et al. [24] applied a multiple 2-D CNN layer with a 1×1 convolutional kernel to extract deep spatial features for HSI classification. However, the high spectral dimension in HSI may increase the number of learnable parameters of the 2-D CNN model, and the correlation of local spectra may be neglected. Compared with the 2-D CNN model, the three-dimensional (3-D) CNN model can simultaneously extract joint spatial–spectral features. Mei et al. [25] proposed an unsupervised 3-D convolutional autoencoder to extract the joint spatial-spectral feature. Roy et al. [26] proposed a hybrid 3-D and 2-D CNN model for HSI classification (HYNN). This model first uses 3-D CNN to extract shallow joint spatial-spectral features and then uses 2-D CNN to extract more abstract spatial texture features. Moreover, to reduce the computational cost of 3-D CNN, Zhang et al. [27] proposed a 3-D depth-wise separable CNN for HSI classification. Recently, inspired by the residual network [28], Zhong et al. [29] proposed a spectral–spatial residual network (SSRN), which uses spectral and spatial 3-D residual blocks to learn deep-level features of HSI. Subsequently, inspired by SSRN and DenseNet [30], Wang et al. [31] proposed an end-to-end fast densely connected spectral-spatial classification framework (FDSS),

which can more effectively reuse features in a few training samples. Although these CNN-based classification models can extract rich spatial and spectral features of HSI, since the convolution kernel is localized, it needs to expand the field of perception by stacking convolution layers, which may lead to a large number of useless features propagated to the deeper convolutional layers. Those useless features will affect the learning efficiency of the model and eventually lead to a decrease in classification accuracy. Thus, finding and focusing on the discriminative features of HSI is an important problem.

Inspired by the human visual system, many researchers have introduced the attention mechanism to computer vision tasks, such as object detection [32], image caption [33], and image enhancement [34]. Since the attention mechanism can pay attention to valuable features or regions in the feature map, some researchers have successfully introduced it to HSI classification. Fang et al. [35] proposed a densely connected spectral-wise attention mechanism network, in which the squeeze-and-excitation (SE) attention module [36] is applied to recalibrate each spectral contribution. Later, many similar spectral attention modules were introduced for HSI classification to highlight valuable spectral and suppress unless ones. For example, Li et al. [37] proposed a spectral band attention module through the adversarial learning method, in which the attention module can explore the contribution of each band and avoid the spectral distortion. Roy et al. [38] proposed a fused SE attention module, in which two different squeezing operations, global pooling and max pooling, are used to generate the excitation weight. To make the network simultaneously boost and suppress features in both spectral and spatial dimensions, many networks based on spectral–spatial attention modules have been proposed for HSI classification. Inspired by SSRN and convolutional block attention module (CBAM) [39], Ma et al. [40] proposed a double-branch multi-attention network (DBMA), in which the spectral and spatial branches are equipped with spectral-wise attention and spatial-wise attention, respectively. Subsequently, Li et al. [41] constructed a double-branch dual attention (DBDA) network for HSI classification, in which the dual attention network (DANet) [42] is inserted separately into two branches. Compared with CBAM, DANet can adaptively integrate local features and global dependencies. In addition, to obtain the long-distance spatial and spectral features, Shi et al. [43] proposed a 3-D coordination attention mechanism network, and the 3-D attention module could be better adapted to the 3-D structure of the HSI. Li et al. [44] proposed a spectral–spatial global context attention [45] network (SSGC) with less time cost to capture more discriminative features. Moreover, in [46], Shi et al. proposed a pyramidal convolution and iterative attention network (PCIA), in which each branch can extract hierarchical features. Although the above three attention-based methods can achieve good classification results, they compress a large spatial or spectral resolution in obtaining the attention feature map. Meanwhile, the feature extraction process requires a high computational cost due to their simple application dense connection modules.

To solve the above problems, inspired by the latest technology and predecessors, we propose a one-shot dense network with polarized attention for HSI classification. Instead of following the 3-D dense connection method used by predecessors to extract features from HSI, we propose a one-shot dense connection block that maintains good classification accuracy and consumes less computational cost. Meanwhile, we add residual connections in this block, enhancing feature transfer and mitigating the gradient disappearance problem. In addition, the latest proposed polarized attention mechanism (PAM) [47] is introduced in the network to mine finer and higher quality features. Compared with other attention mechanisms [36,39,42,45], it can maintain a relatively high resolution in spectral and spatial dimensions and thus reduce the loss of features. Furthermore, the proposed network is composed of two branches that can perform feature extraction in spectral and spatial realms, respectively. The channel-only and spatial-only attention mechanisms are inserted into each branch to recalibrate feature maps. After extracting the enhanced features from the two branches, we fuse them with a concatenation operation to obtain the spectral–spatial features. Finally, the fused features are fed into the fully connected layer to obtain the classification results. The main contributions of this paper are summarized as follows:

(1) We propose a novel spectral–spatial network based on one-shot dense block and polarized attention for HSI classification. The proposed network has two independent feature extraction branches: the spectral branch with channel-only polarized attention applied to obtain spectral features, and the spatial branch with spatial-only polarized attention used to capture spatial features.
(2) By one-shot dense block, the number of parameters and computational complexity of the network are greatly reduced. Meanwhile, the residual connection is added to the block, which can alleviate the performance saturation and gradient disappearance problems.
(3) We apply both channel-only and spatial-only polarized attention in the proposed network. The channel-only polarized attention emphasizes valuable channel features and suppresses useless ones. The spatial-only attention is more focused on areas with more discriminative features. In addition, the attention mechanism can preserve more resolution in both channel and spatial dimensions and consume less computational costs.
(4) Some advanced technologies, including cosine annealing learning rate, Mish activation function [48], Dropout, and early stopping, are employed in the proposed network. For reproducibility, the code of the proposed network is available at https://github.com/HaiZhu-Pan/OSDN (accessed on 5 May 2022).

To show the effectiveness of the proposed network, a large number of experiments were carried out on five real-world HSI datasets, namely, PU, KSC, BS, HS, and SV. The experimental results consistently demonstrate that the proposed network can achieve better accuracy than several widely used ML- and DL-based methods in a few training samples and computational resources.

The remainder of this article is structured as follows: Some close backgrounds are reviewed in Section 2. In Section 3, our proposed network is presented with three parts in detail. In Sections 4 and 5, comparative experiments and ablation analyses are performed to demonstrate the effectiveness of the proposed network. Finally, Section 6 provides some concluding remarks and suggestions for future work.

2. Background

In this section, we briefly introduce some important background techniques involved in the proposed HSI classification model, including 3D convolutional operation, ResNet and DenseNet, and attention mechanism.

2.1. 3-D Convolution Operation

Generally, convolutional operations are the core of CNNs. At present, there are three types of convolution operations in the CNN-based HSI classification model, which are 1-D CNN, 2-D CNN, and 3-D CNN. There are some drawbacks of using 1-D CNN or 2-D CNN, such as lack of spatial relationship features or very complex networks [26]. The main reason is that HSI is a 3-D data cube enriched with a large amount of spatial and spectral information. The 1-D CNN alone cannot extract good discriminative features from the spatial dimension. Similarly, a deep 2-D CNN is more computationally complex and may miss some spectral information between adjacent bands. This is our motivation for using the 3-D convolution operation, which can make up for the shortcomings of the first two convolution operations. The process of 3D convolution operation is shown in Figure 1.

As shown in Figure 1, the input data for the 3-D convolution operation is a 4-D tensor $h^x \in h^n \times h^n \times s^n \times k^n$, where $h^n \times h^n \times s^n$ is the size of the input data, and k^n is the number of channels (feature maps). The 3-D convolution operation contains k^{n+1} convolutional kernels of size $\alpha^{n+1} \times \alpha^{n+1} \times d^{n+1}$, and the stride of subsampling is (s, s, s_1). The output size of the 3-D convolution operation is also a 4-D tensor $h^{x+1} \in h^{n+1} \times h^{n+1} \times s^{n+1} \times k^{n+1}$. More specifically, the spatial size of the output data

is $h^{n+1} = \lfloor 1 + (h^n - \alpha^{n+1})/s \rfloor$, and the depth $s^{n+1} = \lfloor 1 + (s^n - d^{n+1})/s_1 \rfloor$. The 3-D convolution operation is be defined as follows:

$$h_{l,i}^{x,y,z} = M\left(\sum_m \sum_{h=0}^{H_l-1} \sum_{w=0}^{W_l-1} \sum_{d=0}^{D_l-1} k_{l,i,m}^{h,w,d} \times h_{(l-1),m}^{(x+h),(y+w),(z+d)} + b_{l,i}\right) \quad (1)$$

where M is the Mish activation function. In addition, the height, the width, and the depth of the convolution kernel are denoted by H_l, W_l, and D_l, respectively. Furthermore, $k_{l,i,m}^{h,w,d}$ denotes the weight of the ith convolution kernel at position (h, w, d) on the mth feature map in the lth convolution layer. Moreover, $h_{(l-1),m}^{(x+h),(y+w),(z+d)}$ denotes the neuron value at position $(x + h, y + w, z + d)$ on the mth feature map in the $(l - 1)$th layer.

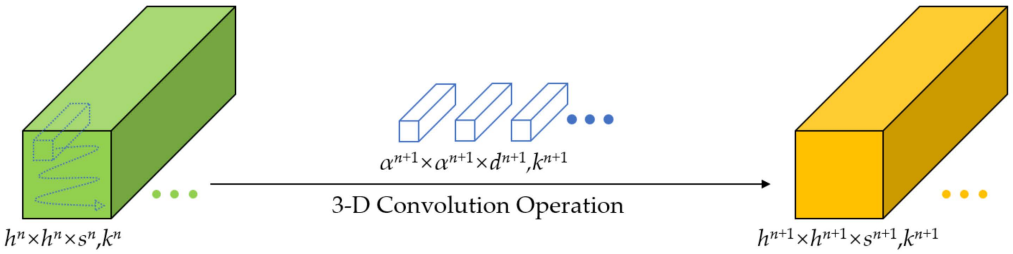


Figure 1. Illustration of the 3-D convolution operation.

2.2. ResNet and DenseNet

Commonly, a trained deep neural network can extract features layer by layer to complete the classification task. However, as the number of convolutional layers increases, two main problems arise: gradient dispersion/explosion and network degradation. Numerous studies have shown that ResNet [28] and DenseNet [30] can alleviate the above problems and achieve feature reuse.

As illustrated in Figure 2, a shortcut connection is added to the base CNN structure in the residual block. The shortcut connections, also known as identity mapping, enable input features to be passed from a lower level to a higher level in a summative way. The output features of the lth residual block are defined as follows:

$$x_l = f_l(x_{l-1}) + x_{l-1} \quad (2)$$

where $f_l(\cdot)$ denotes hidden layers, including convolution, batch normalization (BN), and Mish activation layers.

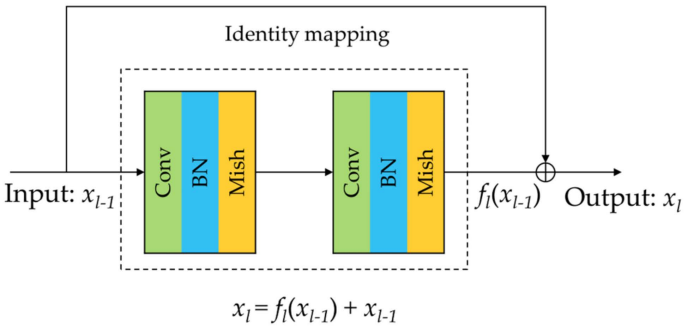


Figure 2. Illustration of the residual block in the ResNet.

To further promote the flow of features in the network, Huang et al. [30] proposed a densely connected network, in which the shortcut connections are used to concatenate the

input features and output features at each layer. This structure is shown in Figure 3. The output features of the lth dense block are computed as follows:

$$x_l = D_l[x_0, x_1, x_2, \ldots, x_{l-1}, x_l] \tag{3}$$

where $D_l(\cdot)$ includes BN, Mish activation function, and convolution operation. $[\cdot]$ is the connected operation. In particular, DenseNet with layer l has $l(l+1)/2$ connections, while CNN with the same layer has only l connections.

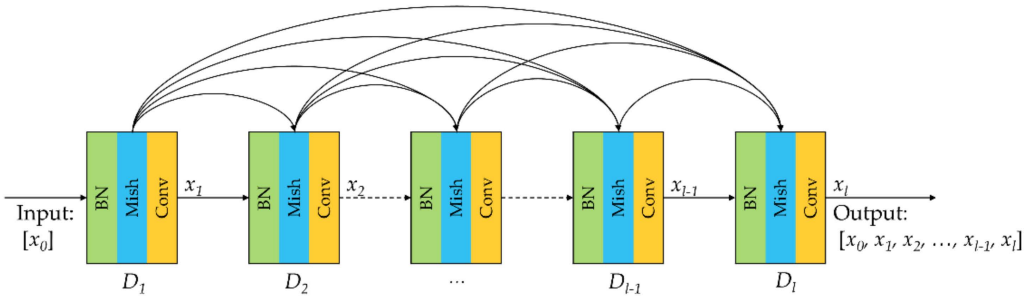


Figure 3. Illustration of the dense block in the DenseNet.

2.3. Attention Mechanism

The attention mechanism is a common data processing method in deep learning. It helps the model assign different weights to each part of the feature maps to extract more critical and discriminative features, thereby enabling the model to make more accurate judgments without imposing more overhead on the computation and storage of the model. The existing attention mechanisms can be roughly divided into two types, i.e., soft attention and hard attention. The former is more attention to the channel or spatial information of the image, while the latter is more attention to the information of a certain position in the image. Most importantly, the soft attention mechanism is differentiable, in which the weight parameters can be updated by backpropagation during the training process. Therefore, soft attention is widely used in the field of computer vision. For example, the SE attention module [36] can recalibrate each channel's contribution to the network. GCNet [45] not only extracts global contextual information but is also lightweight like SENet. In addition, CBAM [39] and DANet [42] can extract attention maps in both the channel and spatial dimensions. However, these attention models have a low internal attention resolution, which loses a great quantity of channels or spatial information. Moreover, these attention models are computationally intensive when paying attention to the channel and spatial dimensions. To alleviate these problems, the PAM [47] employs a distinctive filtering method to reduce the complexity of the model while maintaining high internal attention resolution in both the channel and spatial dimensions. The detailed implementation of the channel-only PAM and spatial-only PAM is described in Sections 3.1 and 3.2

3. Methodology

3.1. Channel-Only Polarized Attention Mechanism

As shown in Figure 4, the channel-only PAM is constructed using the channel relations of the feature map. We assume that the input feature maps $A_c \in \mathbb{R}^{h \times w \times c}$ are independent, where h, w, and c denote height, width, and channel, respectively. First, A_c is fed into a 2-D convolution layer with the kernel size of 1×1. Next, a new feature map $B_c \in \mathbb{R}^{h \times w \times c/2}$ is generated. After that, B_c is reshaped to $D_c \in \mathbb{R}^{n \times c/2}$, where $n = h \times w$. Simultaneously, A_c is also fed into a 2-D convolution layer with the kernel size of 1×1. A new feature map $C_c \in \mathbb{R}^{h \times w \times 1}$ is generated. Then, C_c is reshaped to $E_c \in \mathbb{R}^{1 \times 1 \times n}$, and the SoftMax function is applied to enhance attention scope. Subsequently, the matrix multiplication operation is performed on matrices D_c and E_c, and the generated feature map $F_c \in \mathbb{R}^{1 \times 1 \times c/2}$. After that, F_c is fed into a bottleneck feature

transform layer, which consists of two 1×1 convolution layers, a layer normalization operation, and a ReLU activation function to obtain the dependency of each channel and raise the channel dimension from $c/2$ to c. Next, the Sigmoid function is used to keep the channel weights $G_c \in \mathbb{R}^{1 \times 1 \times c}$ between 0 and 1. Finally, a channel-wise multiplication operation is performed between H_c and A_c to generate the final channel-only polarized attention map $H_c \in \mathbb{R}^{h \times w \times c}$. The overall channel-only PAM implementation process can be defined as follows:

$$G_c = F_{SG}[W_3((\zeta_1(W_1(A_c)) \times F_{SM}(\zeta_2(W_2(A_c)))))] \quad (4)$$

where W_1, W_2, and W_3 are 1×1 convolution layers; ζ_1 and ζ_2 are two tensor transformation operations; $F_{SG}(\cdot)$ is a Sigmoid activation function; and $F_{SM}(\cdot)$ is a SoftMax activation function. The internal channel resolution between $W_1|W_2$ and W_3, is $c/2$. The final output of the channel-only PAM is formulated as

$$H_c = G_c \odot_c A_c \in \mathbb{R}^{h \times w \times c} \quad (5)$$

where $\odot_c$ is dot multiplication operation.

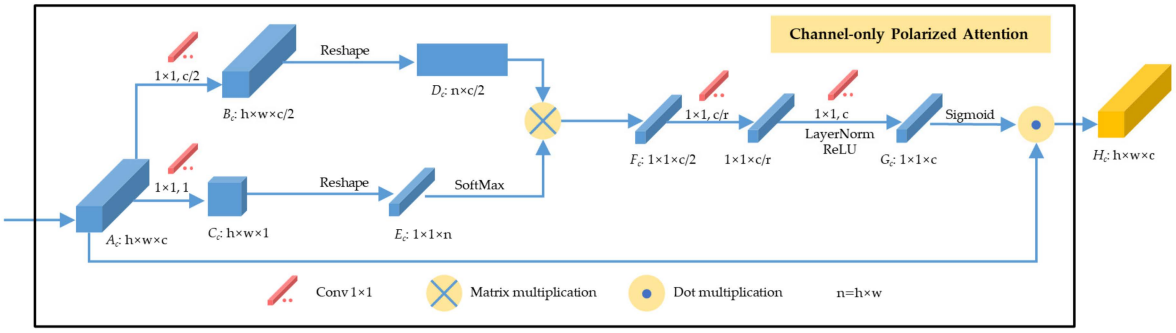


Figure 4. Details of the channel-only polarized attention mechanism in our network.

3.2. Spatial-Only Polarized Attention Mechanism

As is shown in Figure 5, the spatial-only PAM is constructed by the spatial contextual position relationship of the feature map, given an input tensor $A_s \in \mathbb{R}^{h \times w \times c}$. First, A_s is fed into two 1×1 convolution layers to generate feature maps $B_s \in \mathbb{R}^{h \times w \times c/2}$ and $C_s \in \mathbb{R}^{h \times w \times c/2}$, respectively. Next, B_s is reshaped to $D_s \in \mathbb{R}^{c/2 \times n}$, where $n = h \times w$. Second, the global average pooling operation is used in C_s to compress the global spatial features into a feature vector $E_s \in \mathbb{R}^{1 \times 1 \times c/2}$; meanwhile, since the spatial features of C_s are compressed, we use the SoftMax function to perform feature enhancement on E_s. After that, a spatial-wise multiplication operation is conducted on attention maps D_s and E_s. The generated feature map $F_c \in \mathbb{R}^{1 \times 1 \times n}$. Through reshape and Sigmoid operations, the spatial attention weight $G_s \in \mathbb{R}^{h \times w \times 1}$ is generated. The overall spatial-only PAM implementation process can be defined as follows:

$$G_s = F_{SG}[\zeta_3(F_{SM}(\zeta_1(F_{GP}(W_2(A_s))))) \times \zeta_2(W_1(A_s)))] \quad (6)$$

where W_1 and W_2 are two standard 1×1 convolution layers, ζ_1 and ζ_2 are two tensor transformation operations, $F_{GP}(\cdot)$ is a global average pooling operation, $F_{SM}(\cdot)$ is a SoftMax active operation, and $F_{SG}(\cdot)$ is a Sigmoid active operation. The final output of the spatial-only PAM is formulated as

$$H_s = G_c \odot_s A_s \in \mathbb{R}^{h \times w \times c} \quad (7)$$

where $\odot_c$ is dot multiplication operation.

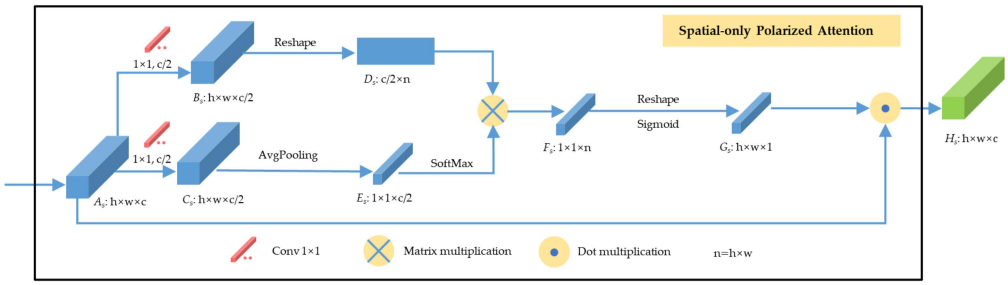


Figure 5. Details of the spatial-only polarized attention mechanism in our network.

3.3. One-Shot Dense Network with Polarized Attention

In this subsection, we describe in detail the proposed network, which consists of spectral feature extraction, spatial feature extraction, and spectral–spatial feature fusion. The structure of the proposed network is shown in Figure 6. In the following, we use the PU dataset as an example to illustrate the three components of the proposed network in detail.

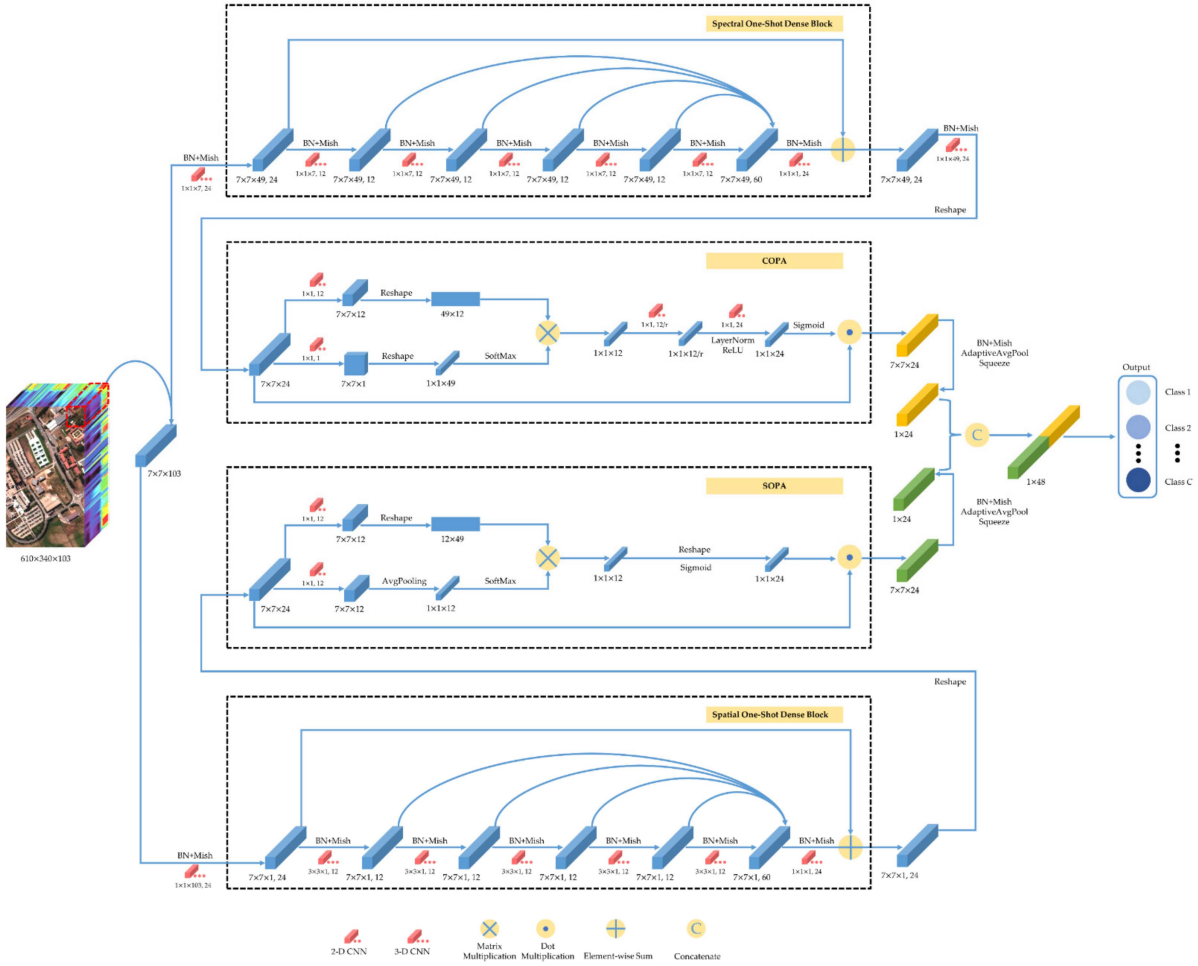


Figure 6. The structure of the proposed network.

3.3.1. Spectral and Spatial Feature Extraction of One-Shot Dense Block

As shown in Figure 6, this part contains two independent feature extraction processes, including the spectral feature extraction process and the spatial feature extraction process. In the process of feature extraction, inspired by ResNet and DenseNet, we propose a one-shot dense block. Unlike the dense block, the feature maps produced by each convolution (*Conv*) layer are concatenated only once, and each *Conv* layer has an equal number of

input and output feature maps. Furthermore, we also insert the skip connection in the one-shot dense block, which enables this block to extract deeper features of HSI. Instead of an individual pixel vector, we first randomly select a 3-D patch cube $7 \times 7 \times 103$ from the PU dataset as the network's input. In this way, the network can consider both spatial background information and spectral information around the central pixel of the 3-D patch cube during the classification process. Before passing through the spectral one-shot dense block, we first use a 3-D *Conv* layer with BN and Mish to reduce the spectral dimension of the input data. The kernel size is $(1 \times 1 \times 7)$; the filters is 24; the stride is (1, 1, 2); no padding operation. After that, the output size of the generated feature maps is $(7 \times 7 \times 49, 24)$. Next, they are fed into the spectral one-shot dense block, which consists of a one-shot connected part and a residual connected part. The kernel size, filters, stride, and padding of all 3-D *Conv* layers in the one-shot connected part is $(1 \times 1 \times 7)$, 12, (1, 1, 1), and (0, 0, 3), respectively. Then, we connect the generated feature maps through the channel dimension, and thus the feature maps with the size of $(7 \times 7 \times 49, 60)$ are generated. Meanwhile, to implement the residual connected part, we use a $1 \times 1 \times 1$ 3-D *Conv* layer to reduce the channel dimension from 60 to 24 and then add it to the last feature maps of the one-shot connected part. Finally, after the last 3-D *Conv* layer with a kernel size of $(1 \times 1 \times 49)$, a $(7 \times 7 \times 1, 24)$ feature map is generated.

Similar to the spectral feature extraction process, we only focus on the spatial features of the input data in the spatial feature extraction process. The input data size of the spatial one-shot dense block is $(7 \times 7 \times 1, 24)$. All hyperparameters are the same as the spectral one-shot dense block except that the kernel size of the spatial one-shot dense block is $(3 \times 3 \times 1)$. The detailed spectral and spatial feature extraction processes are listed in Tables 1 and 2.

Table 1. Detailed steps of the spectral one-shot dense block.

Input Size	Layer Operations	Kernel Size	Filters	Output Size
(7 × 7 × 103, 1)	BN-Mish-Conv3D	(1 × 1 × 7)	24	(7 × 7 × 49, 24)
(7 × 7 × 49, 24)	BN-Mish-Conv3D	(1 × 1 × 7)	12	(7 × 7 × 49, 12)
(7 × 7 × 49, 12)	BN-Mish-Conv3D	(1 × 1 × 7)	12	(7 × 7 × 49, 12)
(7 × 7 × 49, 12)	BN-Mish-Conv3D	(1 × 1 × 7)	12	(7 × 7 × 49, 12)
(7 × 7 × 49, 12)	BN-Mish-Conv3D	(1 × 1 × 7)	12	(7 × 7 × 49, 12)
(7 × 7 × 49, 12)	BN-Mish-Conv3D	(1 × 1 × 7)	12	(7 × 7 × 49, 12)
(7 × 7 × 49, 12)/(7 × 7 × 49, 12)/(7 × 7 × 49, 12) /(7 × 7 × 49, 12)/(7 × 7 × 49, 12)	Concatenate	/	/	(7 × 7 × 49, 60)
(7 × 7 × 49, 60)	BN-Mish-Conv3D	(1 × 1 × 1)	24	(7 × 7 × 49, 24)
(7 × 7 × 49, 24)/(7 × 7 × 49, 24)	Element-wise Sum	/	/	(7 × 7 × 49, 24)
(7 × 7 × 49, 24)	BN-Mish-Conv3D	(1 × 1 × 49)	24	(7 × 7 × 1, 24)

Table 2. Detailed steps of the spatial one-shot dense block.

Input Size	Layer Operations	Kernel Size	Filters	Output Size
(7 × 7 × 103, 1)	BN-Mish-Conv3D	(1 × 1 × 103)	24	(7 × 7 × 1, 24)
(7 × 7 × 1, 24)	BN-Mish-Conv3D	(3 × 3 × 1)	12	(7 × 7 × 1, 12)
(7 × 7 × 1, 12)	BN-Mish-Conv3D	(3 × 3 × 1)	12	(7 × 7 × 1, 12)
(7 × 7 × 1, 12)	BN-Mish-Conv3D	(3 × 3 × 1)	12	(3 × 3 × 1, 12)
(7 × 7 × 1, 12)	BN-Mish-Conv3D	(3 × 3 × 1)	12	(3 × 3 × 1, 12)
(7 × 7 × 1, 12)	BN-Mish-Conv3D	(3 × 3 × 1)	12	(3 × 3 × 1, 12)
(7 × 7 × 1, 12)/(7 × 7 × 1, 12)/(7 × 7 × 1, 12) /(7 × 7 × 1, 12)/(7 × 7 × 1, 12)	Concatenate	/	/	(7 × 7 × 1, 60)
(7 × 7 × 1, 60)	BN-Mish-Conv3D	(1 × 1 × 1)	24	(7 × 7 × 1, 24)
(7 × 7 × 1, 24)/(7 × 7 × 1, 24)	Element-wise Sum	/	/	(7 × 7 × 1, 24)

3.3.2. Spectral and Spatial Feature Enhancement of Polarized Attention Mechanism

After the spectral and spatial feature extraction process, the feature maps are enriched with a large amount of spectral and spatial information. However, different channels and positions in these feature maps may make different contributions to the classification results. Therefore, as shown in Figure 6, to enhance valuable features and suppress non-valuable features, the feature maps are fed into the channel-only polarized attention (COPA) block and spatial-only polarized attention (SOPA) block. The input size of both the COPA block and the SOPA block is ($7 \times 7 \times 24$). A detailed description of these two attention mechanisms is given in Sections 3.1 and 3.2. In addition, the detailed implementation of the feature enhancement process is listed in Tables 3 and 4.

Table 3. Detailed steps of the channel-only polarized attention block.

Input Size	Layer Operations	Kernel Size	Filters	Output Size
$(7 \times 7 \times 24)$	Conv2D	(1×1)	12	$(7 \times 7 \times 12)$
$(7 \times 7 \times 12)$	Reshape	/	/	(49×12)
$(7 \times 7 \times 24)$	Conv2D	(1×1)	1	$(7 \times 7 \times 1)$
$(7 \times 7 \times 1)$	Reshape	/	/	$(1 \times 1 \times 49)$
$(1 \times 1 \times 49)$	SoftMax	/	/	$(1 \times 1 \times 49)$
$(1 \times 1 \times 49)/(49 \times 12)$	Matrix Multiplication	/	/	$(1 \times 1 \times 12)$
$(1 \times 1 \times 12)$	Conv2D	(1×1)	12/r	$(1 \times 1 \times 12/r)$
$(1 \times 1 \times 12/r)$	LayerNorm and ReLu	/	/	$(1 \times 1 \times 12/r)$
$(1 \times 1 \times 12/r)$	Conv2D	(1×1)	24	$(1 \times 1 \times 24)$
$(1 \times 1 \times 24)$	Sigmoid	/	/	$(1 \times 1 \times 24)$
$(1 \times 1 \times 24)/(7 \times 7 \times 24)$	Dot Multiplication	/	/	$(7 \times 7 \times 24)$

Table 4. Detailed steps of the spatial-only polarized attention block.

Input Size	Layer Operations	Kernel Size	Filters	Output Size
$(7 \times 7 \times 24)$	Conv2D	(1×1)	12	$(7 \times 7 \times 12)$
$(7 \times 7 \times 12)$	Reshape	/	/	(12×49)
$(7 \times 7 \times 24)$	Conv2D	(1×1)	12	$(7 \times 7 \times 12)$
$(7 \times 7 \times 12)$	AvgPooling	/	/	$(1 \times 1 \times 12)$
$(1 \times 1 \times 12)$	SoftMax	/	/	$(1 \times 1 \times 12)$
$(1 \times 1 \times 12)/(12 \times 49)$	Matrix Multiplication	/	/	$(1 \times 1 \times 49)$
$(1 \times 1 \times 49)$	Reshape	/	/	$(7 \times 7 \times 1)$
$(7 \times 7 \times 1)$	Sigmoid	/	/	$(7 \times 7 \times 1)$
$(7 \times 7 \times 1)/(7 \times 7 \times 24)$	Dot Multiplication	/	/	$(7 \times 7 \times 24)$

3.3.3. Spectral and Spatial Feature Fusion and Classification

After the spectral and spatial feature enhancement process, the resulting feature maps are separately fed into an adaptive average pooling (AdaptiveAvgPool) layer with BN and Mish. Compared to the fully connected layer, the AdaptiveAvgPool layer can reduce the computation cost. The output size of this layer is (1×24). Finally, we fuse the two feature maps along the channel dimension and then feed the fused feature maps into the linear layer to obtain the classification results. Since we use the cross-entropy loss in PyTorch as the loss function of the network, which automatically contains the probability distribution of the labels, we no longer use the SoftMax layer to obtain the final classification results. The detailed implementation of the feature fusion and classification process is listed in Table 5.

Table 5. Detailed steps of the feature fusion and classification process.

Input Size	Layer Operations	Output Size
$(7 \times 7 \times 24)$	AdaptiveAvgPool-BN-Mish and Squeeze	(1×24)
$(7 \times 7 \times 24)$	AdaptiveAvgPool-BN-Mish and Squeeze	(1×24)

Table 5. *Cont.*

Input Size	Layer Operations	Output Size
$(1 \times 24)/(1 \times 24)$	Concatenate	(1×48)
(1×48)	Dropout-Linear	(1×9)

4. Experiment

4.1. Hyperspectral Dataset Description

In this paper, we employed five well-known HSI datasets, namely, PU, KSC, BS, HS, and SV, to validate the generality and effectiveness of our proposed method. A detailed description of the above five datasets is presented as follows:

PU: The PU dataset was photographed by the Reflective Optics System Imaging Spectrometer (ROSIS) sensor over the University of Pavia. Its spatial dimensions and geometric resolutions are 610×340 and 1.3 m, respectively. Every pixel includes 115 spectral bands ranging from 430 nm to 860 nm. After dropping 12 noise-contaminated spectral bands, the number of spectral bands used for the experiment was 103. The ground truth consists of nine urban land-cover types with 42,776 labeled samples.

KSC: The KSC dataset was taken by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) sensor over the Kennedy Space Center, Florida, on 23 March 1996. The spatial dimensions and resolutions are 512×614 and 18 m, respectively. Each pixel includes 176 spectral bands ranging from 400 to 2500 nm. In addition, this dataset includes 13 land-cover types with 5211 labeled pixels.

BS: The BS dataset was acquired by the NASA EO-1 satellite over the Okavango Delta, Botswana, on 31 May 2001. The spatial size of this dataset is 1476×256, and the spatial resolution is 30 m. Furthermore, the dataset contains 145 spectral bands ranging from 400 to 2500 nm. The dataset contains 3248 labeled pixels, which are divided into 14 classes.

HS: The HS dataset was captured over the University of Houston campus and the neighboring urban area on 23 June 2012, through the NSF-funded Center for Airborne Laser Mapping (NCALM). Its height and width are 349 and 1905, respectively, and its spatial resolution is up to 2.5 m. This dataset consists of 144 spectral bands in the 380 to 1050 nm region. This dataset has 664,845 pixels with 15,029 labeled samples, divided into 15 land-cover types.

SV: The SV dataset was also gathered by the AVIRIS sensor, but it was collected in the Salinas Valley region of California. Its spatial dimensions and resolutions are 512×217 and 3.7 m, respectively. The raw SV dataset has 224 spectral bands ranging from 400 to 2500 nm. Twenty water absorption bands are abandoned. Therefore, this article uses 204 bands for the experimental dataset. This dataset contains 16 land-cover types with 54,129 labeled samples.

4.2. Experimental Evaluation Indicators

In this work, three evaluation indicators, namely, overall accuracy (*OA*), average accuracy (*AA*), and *Kappa* coefficient (*Kappa*), are used to assess the classification performance of the proposed method [49]. *OA* refers to the percentage of correctly classified labeled samples to the total labeled samples. *AA* is the average accuracy for each class, which assigns the same importance to each category. *Kappa* is the consistency between classification results and ground truth. It is calculated from -1 to 1, but usually, it falls between 0 and 1. All in all, the closer the above three indicators are to 1, the better the classification model will be.

To explain the above three evaluation indicators more intuitively, we first define the confusion matrix. In the confusion matrix, each column represents the predicted label, and

each row represents the actual label. The composition of the confusion matrix ($A_{n \times n}$) is defined as follows:

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \tag{8}$$

where element a_{ij} indicates the number of samples in class i classified as class j, and $\sum_{i}^{n} a_{ij}$ and $\sum_{j}^{n} a_{ij}$ indicate the sum of samples in each row and column, respectively. Then, the values of OA, AA, and $Kappa$ can be defined as follows:

$$OA = \frac{\sum_{i=1}^{n} a_{ii}}{\sum_{i}^{n} \sum_{j}^{n} a_{ij}} \tag{9}$$

$$AA = \frac{1}{n} \times \sum_{i=1}^{n} \frac{a_{ii}}{a_{ij}} \tag{10}$$

$$Kappa = \frac{OA - \sum_{i=1}^{n} \left(\sum_{i}^{n} a_{ij} \times \sum_{j}^{n} a_{ij} \right)}{1 - \sum_{i=1}^{n} \left(\sum_{i}^{n} a_{ij} \times \sum_{j}^{n} a_{ij} \right)} \tag{11}$$

4.3. Experimental Setting

The experiments were implemented on a deep learning workstation with a 2× Intel Xeon E5-2680 v4 processor, 35 M of L3 cache, a clock speed of 2.4 GHz, and 14 physical cores/28 way multitask processing. Furthermore, it is equipped with 128 GB of DDR4 RAM and 8× NVIDIA GeForce RTX 2080Ti super graphical processing unit (GPU) with 11 GB of memory. The software environment is CUDA v11.2, PyTorch 1.1.0, and Python 3.7.

To validate the effectiveness of our proposed method, we selected seven representative methods for comparison: one representative ML-based method and seven state-of-the-art DL-based methods. All comparison methods are briefly described as follows:

(1) SVM: The SVM with radial basis function (RBF) kernel is employed as a representative of the traditional method for HSI classification. It is implemented by scikit-learn [50]. Each labeled sample in the HSI has a continuous spectral vector. They are directly fed into the SVM without feature extraction and dimensionality reduction. The penalty parameter C and the RBF kernel width σ are selected by Grid SearchCV, both in the range of $(10^{-2}, 10^{2})$.

(2) HYSN [26]: The HYSN model has three 3-D convolution layers, one 2-D convolution layer, and two fully connected layers. The sizes of the convolution kernels of the 3-D convolution layers are $3 \times 3 \times 7$, $3 \times 3 \times 5$, and $3 \times 3 \times 3$, respectively. The size of the convolution kernel of the 2-D convolution layer is 3×3.

(3) SSRN [29]: The SSRN model consists of two residual convolutional blocks with convolution kernel sizes of $1 \times 1 \times 7$ and $3 \times 3 \times 1$, respectively. They are connected sequentially to extract deep-level spectral and spatial features, in which BN and ReLu are added after each convolutional layer.

(4) FDSS [31]: The network structure of FDSS is connected by three convolutional parts, including a densely connected spectral feature extraction part, a reducing dimension part, and a densely connected spatial feature extraction part. The shapes of the three partial convolution kernels are $1 \times 1 \times 7$, $1 \times 1 \times$ b (b represents the spectral depth of the generated feature map), and $3 \times 3 \times 1$, respectively. Moreover, BN and ReLu are added before each convolutional layer.

(5) DBMA [40]: The DBMA model is designed with a two-branch network structure. Each branch has a dense block and an attention block. Its dense block is the same as in FDSS. Moreover, the attention block is inspired by CBAM [39].

(6) DBDA [41]: The DBDA model uses DANet [42] as the attention mechanism, and the rest of the network structures are the same as DBMA. In particular, it adopts the Mish as the activation function.
(7) PCIA [46]: The PCIA model uses an iterative approach to construct an attention mechanism. This network structure also consists of two branches, but each branch uses a pyramid convolution module to perform feature extraction.
(8) SSGC [44]: The GCNet [45] attention mechanism is introduced to the SSGC. The rest of the network architecture is the same as DBMA.

To ensure the impartiality of the comparison experiments, we took the same hyperparameters on these methods. For the training set of the proposed method, we applied the Adam optimizer [51] to update the parameters for 100 training epochs, where the initial learning rate is 0.0005 for all datasets. The learning rate is dynamically adjusted every 25 epochs by the cosine annealing [52]. Furthermore, if the loss on the validation set does not change within 10 epochs, the network will move to the test session. To balance efficiency and effectiveness, the spatial size of the HSI patch cube was set to 7×7, and the batch size was set to 32. Tables 6–10 provide the detailed distribution of the training, validation, and testing samples of PU, KSC, BS, HS, and SA datasets. To seek reproducibility, the proposed network code is available publicly at https://github.com/HaiZhu-Pan/OSDN (accessed on 5 May 2022).

Table 6. The number of total samples, training samples, validation samples, and testing samples for each category of the PU dataset.

Number	Land Cover Type	Total	Train	Val.	Test
C1	Asphalt	6631	66	66	6499
C2	Meadows	18,649	186	186	18,277
C3	Gravel	2099	21	21	2057
C4	Trees	3064	31	31	3002
C5	Painted metal sheets	1345	13	13	1319
C6	Bare soil	5029	50	50	4929
C7	Bitumen	1330	13	13	1304
C8	Self-blocking bricks	3682	37	37	3608
C9	Shadows	947	9	9	929
	Total	42,776	428	428	41,920

Table 7. The number of total samples, training samples, validation samples, and testing samples for each category of the KSC dataset.

Number	Land Cover Type	Total	Train	Val.	Test
C1	Scrub	761	15	15	731
C2	Willow swamp	243	5	5	233
C3	CP hammock	256	5	5	246
C4	Slash pine	252	5	5	242
C5	Oak/broadleaf	161	3	3	155
C6	Hardwood	229	5	5	219
C7	Swamp	105	2	2	101
C8	Graminoid marsh	431	9	9	413
C9	Spartina marsh	520	10	10	500
C10	Cattail marsh	404	8	8	388
C11	Salt marsh	419	8	8	403
C12	Mud flats	503	10	10	483
C13	Water	927	19	19	889
	Total	5211	104	104	5003

Table 8. The number of total samples, training samples, validation samples, and testing samples for each category of the BS dataset.

Number	Land Cover Type	Total	Train	Val.	Test
C1	Water	270	5	5	260
C2	Hippo grass	101	2	2	97
C3	Floodplain grasses1	251	5	5	241
C4	Floodplain grasses2	215	4	4	207
C5	Reeds1	269	5	5	259
C6	Riparian	269	5	5	259
C7	Fierscar2	259	5	5	249
C8	Island interior	203	4	4	195
C9	Acacia woodlands	314	6	6	302
C10	Acacia shrublands	248	5	5	238
C11	Acacia grasslands	305	6	6	293
C12	Short mopane	181	4	4	173
C13	Mixed mopane	268	5	5	258
C14	Exposed soils	95	2	2	91
	Total	3248	65	65	3118

Table 9. The number of total samples, training samples, validation samples, and testing samples for each category of the HS dataset.

Number	Land Cover Type	Total	Train	Val.	Test
C1	Healthy grass	1251	25	25	1201
C2	Stressed grass	1254	25	25	1204
C3	Synthetic grass	697	14	14	669
C4	Trees	1244	25	25	1194
C5	Soil	1242	25	25	1192
C6	Water	325	7	7	311
C7	Residential	1268	25	25	1218
C8	Commercial	1244	25	25	1194
C9	Road	1252	25	25	1202
C10	Highway	1227	25	25	1177
C11	Railway	1235	25	25	1185
C12	Parking lot 1	1233	25	25	1183
C13	Parking lot 2	469	9	9	451
C14	Tennis court	428	9	9	410
C15	Running track	660	13	13	634
	Total	15,029	301	301	14,427

Table 10. The number of total samples, training samples, validation samples, and testing samples for each category of the SV dataset.

Number	Land Cover Type	Total	Train	Val.	Test
C1	Brocoli-green-weeds_1	2009	40	40	1929
C2	Brocoli-green-weeds_2	3726	75	75	3576
C3	Fallow	1976	40	40	1896
C4	Fallow-rough-plow	1394	28	28	1338
C5	Fallow-smooth	2678	54	54	2570
C6	Stubble	3959	79	79	3801
C7	Celery	3597	72	72	3435
C8	Grapes-untrained	11,271	225	225	10,821
C9	Soil-vinyard-develop	6203	124	124	5955
C10	Corn-senesced-green-weeds	3278	66	66	3146
C11	Lettuce-romaine-4wk	1068	21	21	1026
C12	Lettuce-romaine-5wk	1927	39	39	1849
C13	Lettuce-romaine-6wk	916	18	18	880

Table 10. *Cont.*

Number	Land Cover Type	Total	Train	Val.	Test
C14	Lettuce-romaine-7wk	1070	21	21	1028
C15	Vinyard-untrained	7268	145	145	6978
C16	Vinyard-vertical-trellis	1807	36	36	1735
	Total	54,129	1083	1083	51,963

4.4. Experimental Results

Tables 11–15 report the classification accuracy of each category, *OA*, *AA*, and *Kappa*, on five datasets. It is clear that the proposed OSDN produces the best *OA*, *AA*, and *Kappa* and provides a significant improvement over the other methods on all datasets. For example, when 1% of the samples are randomly chosen for training on the PU dataset (Table 11), the improvement in *OA* compared to SVM, HYSN, SSRN, FDSS, DBMA, DBDA, PCIA, and SSGC methods are 9.96%, 5.87%, 3.60%, 1.72%, 2.16%, 2.06%, 2.99%, and 1.45%, respectively. Specifically, since SVM only uses spectral information to perform classification, its accuracy on all datasets is much lower than other methods. Conversely, the other eight DL-based methods (i.e., HYSN, SSRN, FDSS, DBMA, DBDA, SSGC, PCIA, and OSDN) all achieved good classification results on five datasets because they could automatically extract deep, high-level, and discriminative spatial–spectral information from the 3-D patch cube. Furthermore, compared to SSRN and HYSN, the *OA* of FDSS was improved approximately by 1–8% on all datasets, which indicates that the densely connected structure can extract features more adequately in a few training samples. In addition, the network structures of DBMA, DBDA, PCIA, and SSGC are very similar. Their classification models are based on two main ideas: dual-branch 3-D dense convolution block and dual-branch attention mechanism. Among these dual-branch attention models, SSGC achieved the best classification results in most datasets due to its ability to focus on global contextual information. In addition, the classification accuracy obtained by OSDN was higher than that of FDSS and SSCG because the PAM module in OSDN not only retained a large amount of spectral and spatial resolution but also dynamically enhanced the feature maps. Finally, compared with the best comparison methods in the five datasets, the *OA* of OSDN was improved by 1.45%, 1.86%, 1.46%, 1.62%, and 0.82%, respectively. At the same time, *AA* and *Kappa* improved to different degrees on the five datasets.

Figures 7–11 show the ground truth, false-color image, and classification maps of all methods on the five datasets. Generally, the outline of each category was smoother and clearer in the proposed OSDN classification map on all datasets. Because the SVM method cannot effectively extract the spatial feature, its classification map had a large amount of salt-and-pepper noise on the five datasets (Figures 7b, 8b, 9b, 10b and 11b). In addition, benefiting from the PAM module, our proposed OSDN was found to be significantly better than other methods in predicting those unlabeled categories. Taking the PU dataset as an example, looking carefully at Figure 7k, we can see that there may have been several trees (C4) in the lower side area of the bare soil (C6). However, no method can predict as many trees in this area as possible. On the contrary, it is clear from Figure 7j that our proposed OSDN can predict eight trees in this area. Similarly, in the left area of these eight trees, the proposed OSDN was able to visualize the area more completely than other methods. All observations validate that our proposed OSDN can accurately predict labeled categories and reasonably predict unlabeled categories on all datasets. Moreover, the above results further verify that the proposed one-shot dense connection can also extract sufficient features in a few training samples, while the PAM module can focus on extracting finer features to perform classification.

Table 11. Classification results of the PU dataset based on 1% training samples.

Number	Color	SVM	HYSN	SSRN	FDSS	DBMA	DBDA	PCIA	SSGC	OSDN
C1		87.65	97.57	97.03	99.61	97.35	95.13	93.55	98.30	98.65
C2		91.78	95.66	98.21	97.85	97.90	98.83	98.56	98.68	99.63
C3		76.13	94.02	71.88	93.58	93.78	90.38	78.62	98.99	98.07
C4		93.81	93.27	98.56	100.0	98.81	97.89	99.64	99.26	99.36
C5		98.14	98.94	99.70	99.92	100.0	99.55	99.92	99.92	99.47
C6		85.81	84.69	96.28	99.49	99.10	95.21	98.12	99.46	99.98
C7		68.39	89.65	99.91	81.94	93.89	100.0	99.79	100.0	100.0
C8		84.88	81.57	82.71	91.33	86.12	91.52	86.73	84.48	92.86
C9		99.89	99.56	99.44	99.04	99.01	99.78	97.37	97.27	100.0
OA (%)		88.87	92.96	95.23	97.11	96.67	96.77	95.84	97.38	98.83
AA (%)		87.39	92.77	93.75	95.86	96.22	96.48	94.70	97.37	98.67
Kappa × 100		85.11	90.69	93.67	96.16	95.57	95.71	94.47	96.52	98.44

Table 12. Classification results of the KSC dataset based on 2% training samples.

Number	Color	SVM	HYSN	SSRN	FDSS	DBMA	DBDA	PCIA	SSGC	OSDN
C1		86.49	99.86	87.58	88.83	90.72	97.22	96.58	87.68	97.96
C2		76.43	86.76	67.41	66.57	88.18	84.55	91.16	93.42	98.60
C3		64.14	86.18	53.19	56.54	80.88	77.32	91.62	80.48	82.09
C4		45.75	55.00	61.54	83.78	61.47	54.86	64.57	71.90	92.35
C5		31.76	26.80	100.0	97.96	69.23	33.33	82.85	70.07	95.60
C6		50.58	96.70	100.0	75.81	72.86	95.81	81.06	94.17	80.78
C7		48.20	67.61	100.0	100.0	84.40	76.80	94.35	83.67	96.12
C8		68.95	83.82	90.89	97.61	86.22	89.74	94.38	99.00	98.06
C9		72.31	82.33	98.24	99.79	86.89	98.81	97.59	100.0	100.0
C10		94.00	98.54	64.09	98.97	100.0	100.0	99.94	100.0	99.47
C11		86.35	87.82	98.53	99.75	100.0	100.0	100.0	99.48	100.0
C12		81.41	81.51	91.08	98.72	98.91	94.97	99.29	99.35	92.26
C13		100.0	97.41	100.0	98.54	100.0	100.0	100.0	100.0	99.78
OA (%)		77.25	82.72	84.99	90.24	90.62	91.85	94.23	93.81	96.09
AA (%)		69.72	80.80	85.58	89.45	86.14	84.88	91.80	90.70	94.85
Kappa × 100		74.68	80.79	83.22	89.10	89.53	90.92	93.57	93.10	95.64

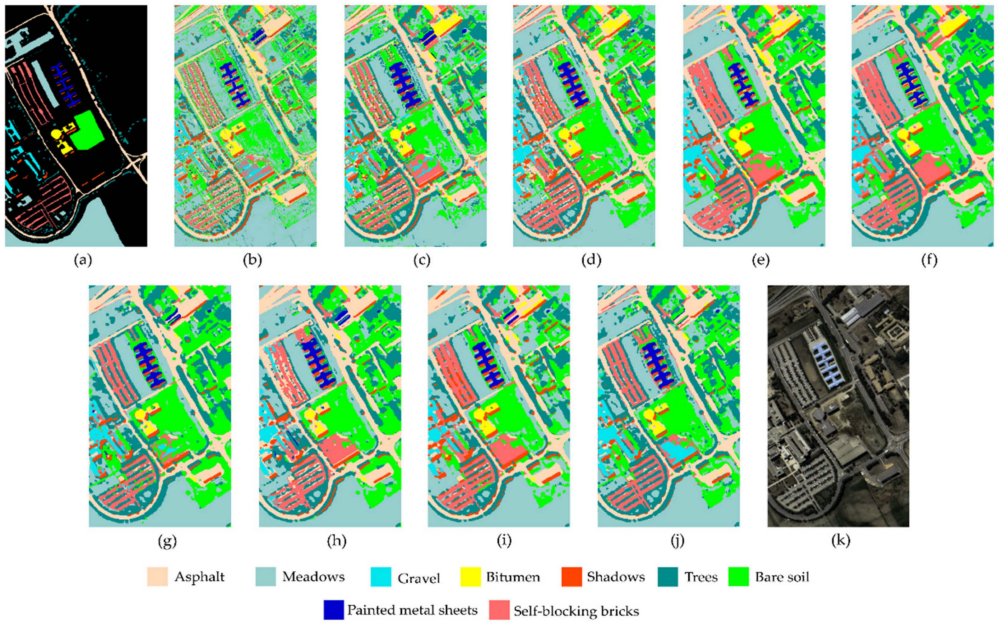


Figure 7. Full-factor classification maps for the PU dataset. (**a**) Ground-truth. (**b**) SVM. (**c**) HYSN. (**d**) SSRN. (**e**) FDSS. (**f**) DBMA. (**g**) DBDA. (**h**) PCIA. (**i**) SSGC. (**j**) OSDN. (**k**) False-color image.

Table 13. Classification results of the BS dataset based on 2% training samples.

Number	Color	SVM	HYSN	SSRN	FDSS	DBMA	DBDA	PCIA	SSGC	OSDN
C1		100.0	78.71	98.86	96.98	97.01	95.57	98.31	98.48	100.0
C2		86.76	97.73	100.0	100.0	100.0	98.00	86.27	100.0	100.0
C3		86.70	88.03	100.0	87.78	100.0	99.58	88.70	100.0	99.17
C4		94.19	87.06	90.95	99.03	95.41	91.96	97.90	91.59	91.86
C5		77.05	90.37	90.28	77.41	87.31	91.96	97.69	92.06	86.75
C6		59.86	57.51	80.08	96.61	83.69	96.07	97.04	91.27	89.71
C7		100.0	88.46	96.48	99.2	100.0	100.0	100.0	100.0	100.0
C8		86.49	91.46	96.26	89.23	98.00	98.43	96.11	97.91	100.0
C9		64.10	69.02	94.89	81.18	96.69	96.74	81.57	90.96	98.67
C10		85.05	92.06	81.60	100.0	99.58	85.14	72.83	91.44	99.57
C11		44.00	89.51	93.31	91.3	100.0	100.0	100.0	94.83	100.0
C12		91.35	90.12	98.05	99.42	100.0	84.91	100.0	100.0	98.08
C13		76.79	98.13	79.50	100.0	83.01	91.05	96.35	92.28	92.13
C14		100.0	95.56	100.0	100.0	100.0	100.0	100.0	100.0	100.0
OA (%)		73.40	84.32	91.45	92.54	94.76	94.63	92.82	94.95	96.41
AA (%)		82.31	86.70	92.88	94.15	95.76	94.96	93.77	95.77	96.85
Kappa × 100		71.07	82.99	90.73	91.91	94.32	94.18	92.22	94.53	96.11

Table 14. Classification results of the HS dataset based on 2% training samples.

Number	Color	SVM	HYSN	SSRN	FDSS	DBMA	DBDA	PCIA	SSGC	OSDN
C1		96.69	91.27	97.81	97.46	91.18	96.47	99.80	96.83	99.91
C2		98.21	86.10	99.92	99.92	94.69	99.00	92.26	98.15	97.86
C3		98.81	94.63	100.0	99.55	100.0	100.0	100.0	100.0	100.0
C4		91.78	89.32	85.14	95.11	99.48	97.21	95.64	97.62	95.34
C5		89.80	92.09	92.39	93.90	93.04	98.66	96.70	94.61	99.66
C6		95.85	91.50	100.0	100.0	100.0	100.0	100.0	96.91	96.53
C7		70.96	81.38	94.68	85.11	95.00	92.80	80.88	95.52	94.64
C8		69.36	87.98	99.52	96.71	97.15	92.33	88.54	92.46	99.63
C9		71.47	80.53	81.45	82.37	95.81	95.48	88.15	96.17	91.96
C10		76.44	86.94	67.22	93.09	82.49	80.42	89.38	93.61	88.59
C11		80.71	86.33	94.26	89.55	92.77	97.58	89.07	92.95	97.92
C12		71.96	83.42	91.84	89.60	92.09	85.77	91.91	88.09	92.84
C13		29.25	94.00	95.51	87.12	71.40	85.81	97.62	77.99	96.16
C14		92.73	90.79	95.77	100.0	100.0	100.0	100.0	100.0	100.0
C15		99.53	90.75	98.15	98.76	94.17	99.68	94.43	99.21	100.0
OA (%)		82.82	87.52	90.30	92.81	92.85	93.99	92.18	94.66	96.28
AA (%)		82.24	88.47	92.91	93.88	93.29	94.75	93.62	94.67	96.74
Kappa × 100		81.41	86.51	89.51	92.23	92.27	93.50	91.55	94.23	95.98

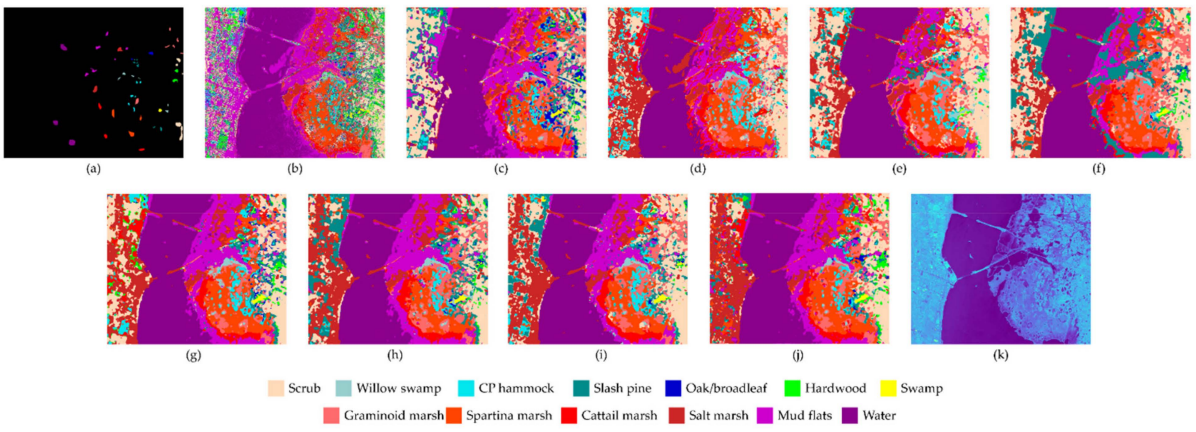


Figure 8. Full-factor classification maps for the KSC dataset. (**a**) Ground-truth. (**b**) SVM. (**c**) HYSN. (**d**) SSRN. (**e**) FDSS. (**f**) DBMA. (**g**) DBDA. (**h**) PCIA. (**i**) SSGC. (**j**) OSDN. (**k**) False-color image.

Table 15. Classification results of the SA dataset based on 2% training samples.

Number	Color	SVM	HYSN	SSRN	FDSS	DBMA	DBDA	PCIA	SSGC	OSDN
C1		99.90	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
C2		98.62	99.94	100.0	100.0	99.97	100.0	100.0	100.0	99.92
C3		91.69	96.38	100.0	97.57	100.0	100.0	99.36	100.0	100.0
C4		97.32	99.15	99.93	99.78	99.33	99.32	100.0	95.61	97.39
C5		97.27	96.55	99.92	99.81	99.65	99.42	91.18	100.0	100.0
C6		99.97	99.95	100.0	100.0	99.92	100.0	100.0	99.97	100.0
C7		98.92	99.47	99.88	100.0	100.0	100.0	100.0	100.0	99.83
C8		76.15	85.39	88.47	96.93	95.44	96.55	95.14	90.45	98.78
C9		98.88	98.35	99.75	99.97	99.45	99.80	100.0	99.82	99.52
C10		92.24	95.68	98.27	99.48	96.41	99.20	99.77	99.46	98.39
C11		96.16	97.93	100.0	95.75	100.0	100.0	95.32	100.0	99.90
C12		95.97	96.39	99.84	99.04	99.89	99.51	99.30	99.68	99.84
C13		93.38	92.67	100.0	100.0	97.34	100.0	99.88	100.0	99.77
C14		97.03	99.49	99.70	99.22	93.29	98.56	99.41	99.03	98.55
C15		77.47	82.63	97.12	91.68	95.78	88.56	94.61	98.00	95.12
C16		99.19	100.0	99.89	99.94	100.0	98.86	100.0	100.0	100.0
OA (%)		90.23	93.53	96.86	97.94	97.98	97.46	97.62	97.39	98.80
AA (%)		94.39	96.25	98.92	98.70	98.53	98.74	98.37	98.88	99.19
Kappa × 100		89.09	92.79	96.49	97.70	97.75	97.18	97.34	97.09	98.66

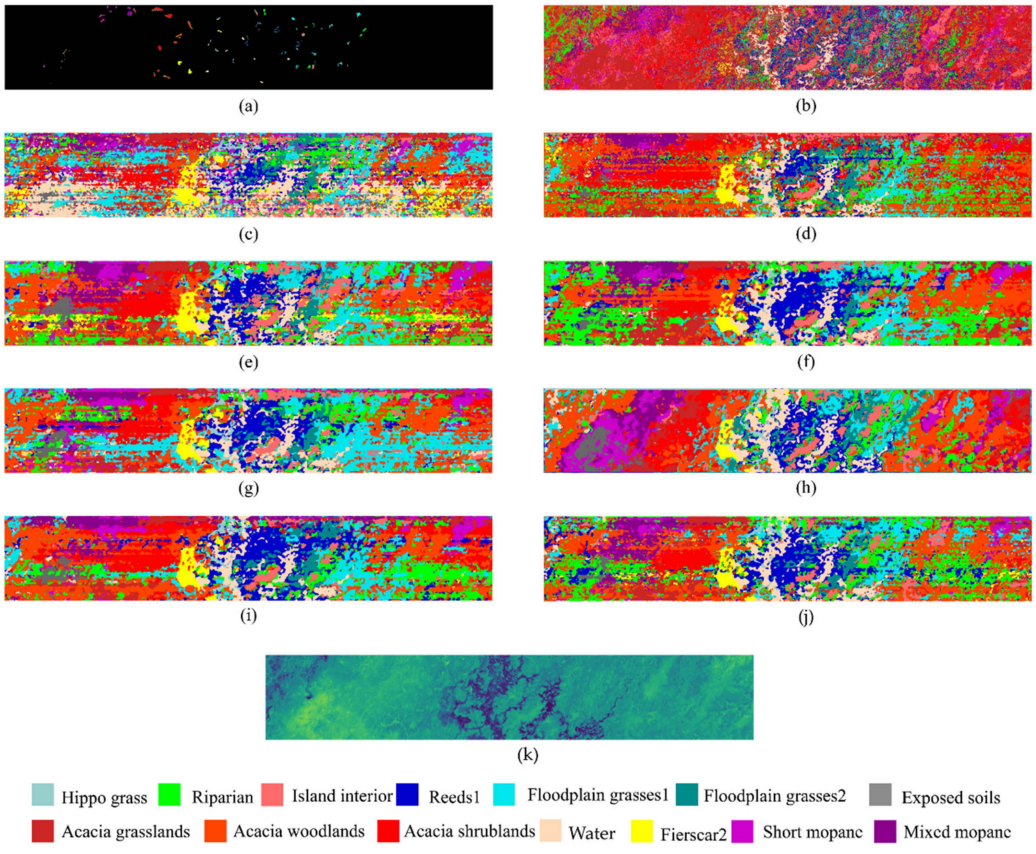


Figure 9. Full-factor classification maps for the BS dataset. (**a**) Ground-truth. (**b**) SVM. (**c**) HYSN. (**d**) SSRN. (**e**) FDSS. (**f**) DBMA. (**g**) DBDA. (**h**) PCIA. (**i**) SSGC. (**j**) OSDN. (**k**) False-color image.

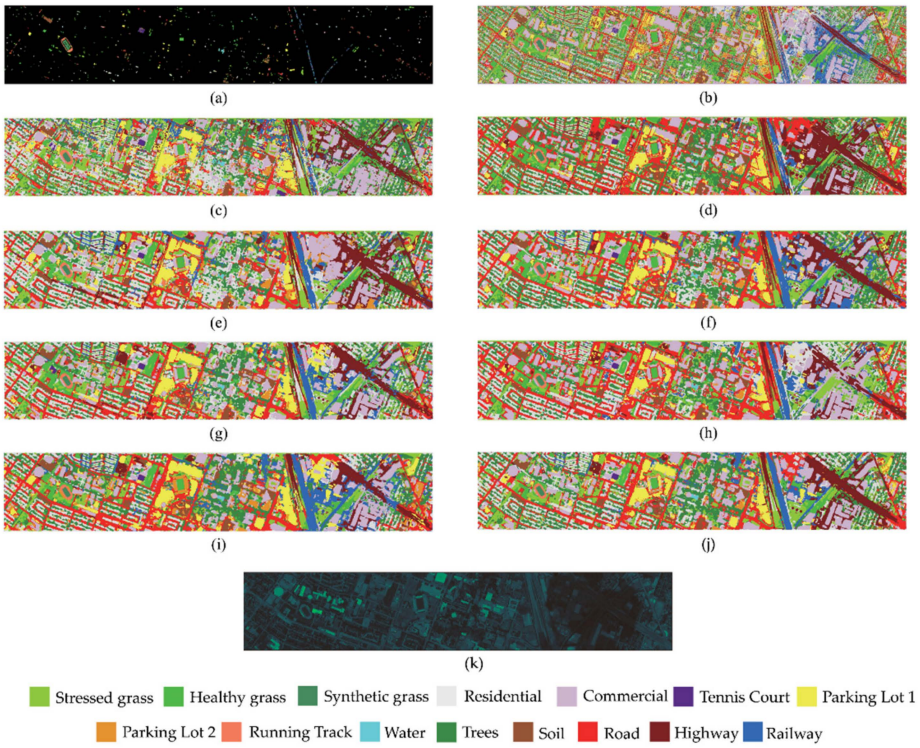


Figure 10. Full-factor classification maps for the HS dataset. (**a**) Ground-truth. (**b**) SVM. (**c**) HYSN. (**d**) SSRN. (**e**) FDSS. (**f**) DBMA. (**g**) DBDA. (**h**) PCIA. (**i**) SSGC. (**j**) OSDN. (**k**) False-color image.

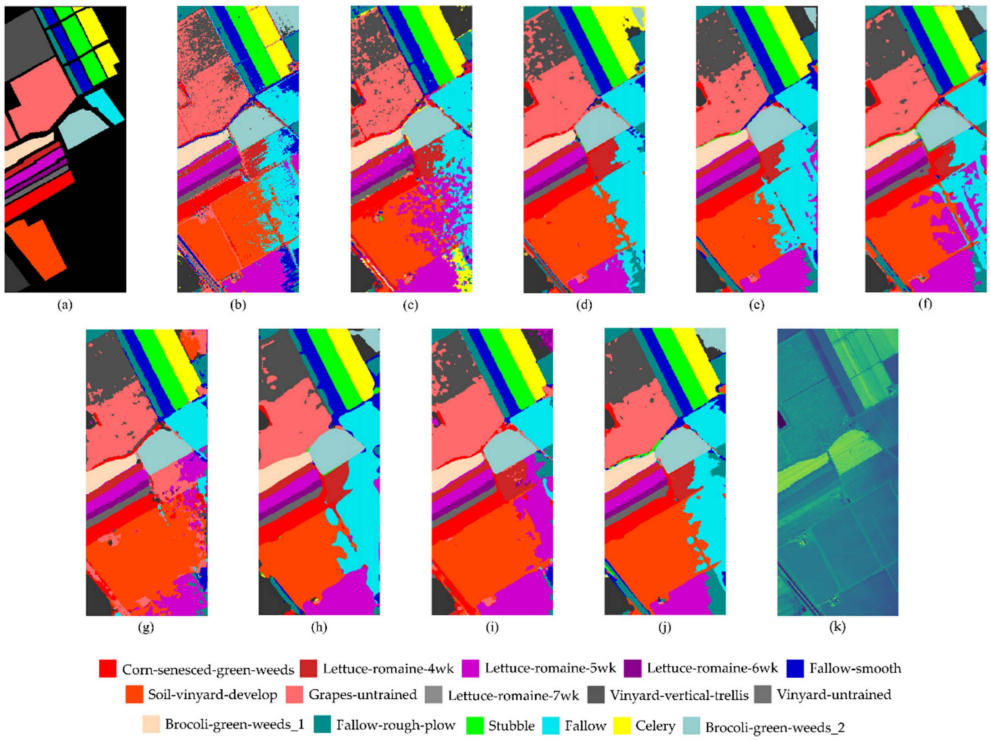


Figure 11. Full-factor classification maps for the SA dataset. (**a**) Ground-truth. (**b**) SVM. (**c**) HYSN. (**d**) SSRN. (**e**) FDSS. (**f**) DBMA. (**g**) DBDA. (**h**) PCIA. (**i**) SSGC. (**j**) OSDN. (**k**) False-color image.

5. Discussion

5.1. Comparison of Different Spatial Patch Size

In this subsection, we explore the effect between the spatial patch size and the classification accuracy of the proposed network. In general, if the spatial patch size is too small, it will not be enough to contain rich spatial features, and the classification performance might be decreased. Conversely, if the spatial patch size is too large, it will contain more mixed pixels and increase the computational cost. Therefore, an appropriate spatial patch size should be determined by classification accuracy and computational cost. Figure 12 depicts the *OA* with different spatial patch sizes ranging from 3 to 13 with a 2-pixel interval. According to Figure 12, with the increase in the spatial patch size, the classification accuracy of the five datasets gradually increased, and the best *OA* was acquired when the patch size was 7×7. This phenomenon indicates that more and more spatial features were included in the data cubes as the spatial size increased. Thus, the classification results were improved to some extent. However, if the space size increased, the *OA* of most datasets will show a decreasing trend. In conclusion, to balance both the *OA* and computational costs, we used 7×7 as the spatial patch size on the five datasets.

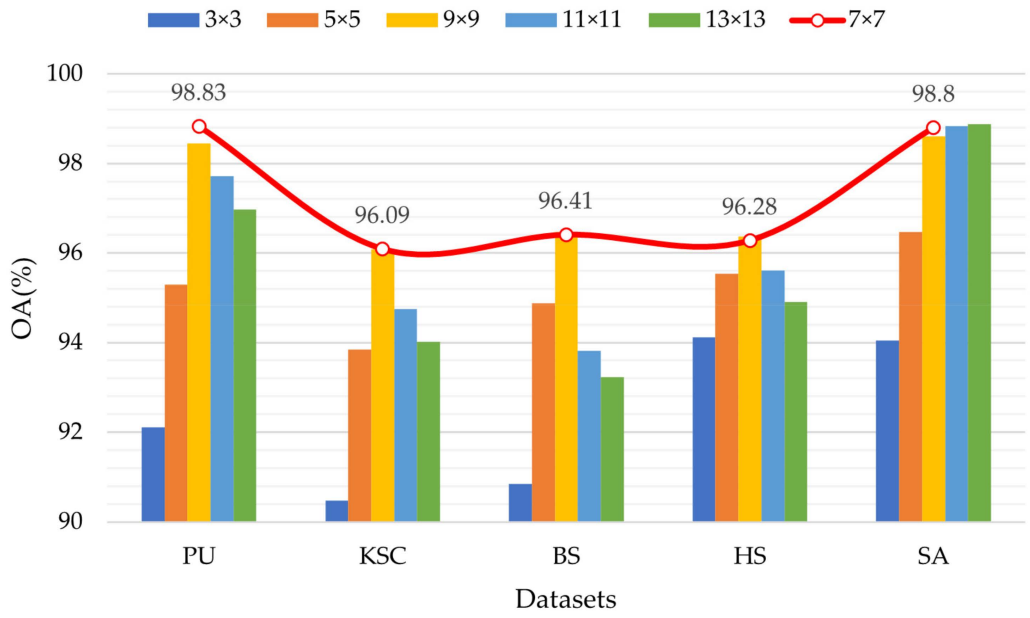


Figure 12. Comparison of *OA* using different spatial window sizes for the five datasets.

5.2. Comparison of Different Training Sample Proportions

It is well known that deep learning is a data-driven approach. In this subsection, we randomly chose 1%, 1.5%, 2%, 3%, 5%, 7%, 9%, and 60% of training samples from each dataset to explore the classification performance of different models with different training sample proportions. As shown in Figure 13, when the training samples were sufficient, these models maintained classification results above 99% on all five datasets. However, obtaining enough training samples is a time-consuming and labor-intensive task. Therefore, one of the motivations of our proposed OSDN was to obtain a good classification result in a few training samples. Compared with other methods, our proposed OSDN consistently maintained the most significant *OA* in various proportions of training samples. Especially with insufficient training samples, our proposed OSDN achieved the highest *OA* on the five datasets.

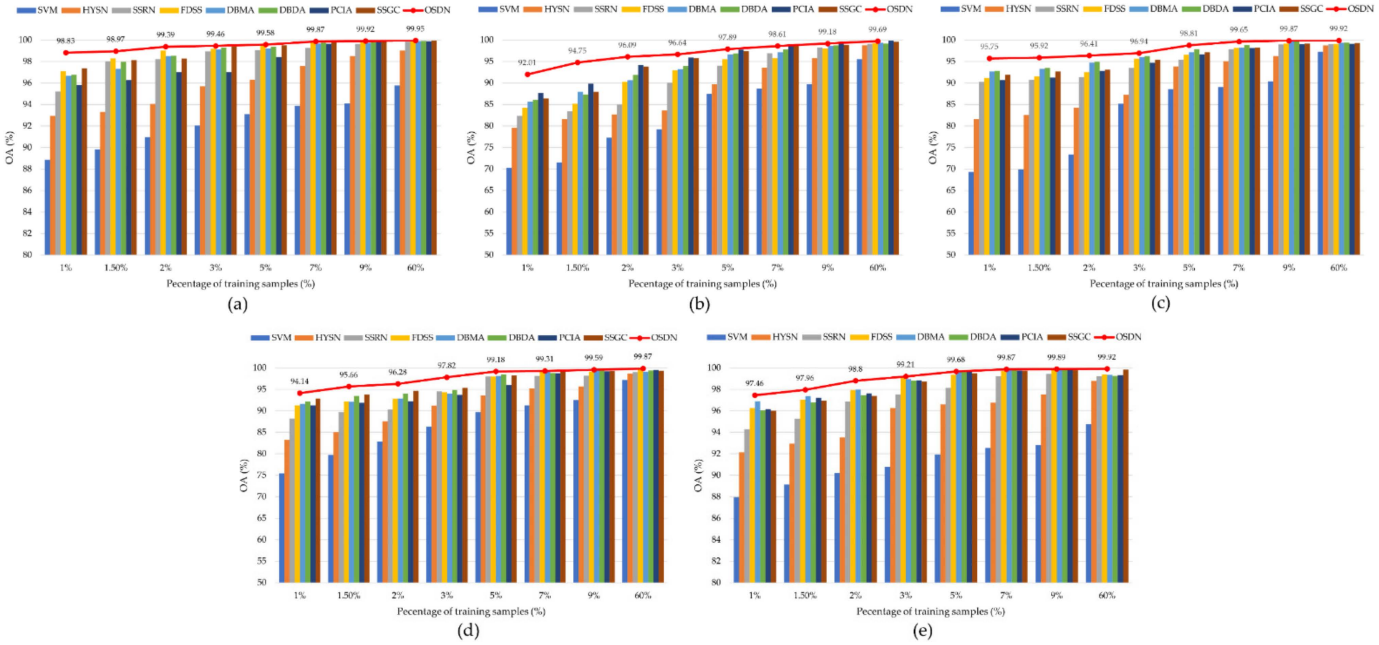


Figure 13. Comparison of *OA* using different training sample proportions for the five datasets: (**a**) PU, (**b**) KSC, (**c**) BS, (**d**) HS, and (**e**) SA.

5.3. Comparison of Computational Cost and Complexity

One of the purposes of this article is to reduce the computational cost and complexity of the proposed network. Therefore, Table 16 compares the number of parameters, floating-point operations (FLOPs), training time, and testing time of different methods for five datasets. The FLOPs is an indicator to evaluate the model's complexity, which is used to measure the computational cost of the model. All methods are counted in the state of the best accuracy and are trained with the same samples. Generally, from Table 16, it can be found that the proposed OSDN achieved good results on all four metrics. Specifically, HYSN had the largest number of parameters and highest FLOPs compared with other DL-based methods, owing to its deeper network structure. Compared with SSRN, although FDSS achieved good classification results (see Figure 14), it had more parameters and FLOPs than SSRN due to its dense connections. In addition, it is worth noting that the DBMA, DBDA, PCIA, and SSGC had a similar feature extraction backbone. Among these four methods, DBMA had the maximum number of parameters and the highest FLOPs since its attention module contained fully connected operations. Furthermore, DBDA, PCIA, and SSGC had roughly the same parameters. However, PCIA took lower FLOPs because of its multiscale pyramidal feature extraction block and iterative attention module. Our proposed OSDN required the least number of parameters and lowest FLOPs on the five datasets due to its lightweight one-shot dense block and effective PAM module. In addition, since SVM contained fewer parameters, it did not consume much time for training and testing. As for the time efficiency of OSDN, it is very competitive with other similar comparative models (i.e., DBMA, DBDA, PCIA, and SSGC). Finally, combining Table 16 and Figure 14 with the above analysis, we can conclude that the proposed OSDN presents satisfying classification accuracy with less computational cost and complexity.

Table 16. The number of parameters, FLOPs, training time, and testing time of different methods for five datasets.

	Model	SVM	HYSN	SSRN	FDSS	DBMA	DBDA	PCIA	SSGC	OSDN
PU	Parameters (M)	/	1.37	0.21	0.34	0.32	0.20	0.23	0.19	0.05
	FLOPs (M)	/	71.41	48.47	39.55	74.43	32.72	26.50	32.51	21.18
	Training time (s)	5.19	15.14	40.47	43.62	39.41	37.00	35.25	37.21	30.19
	Testing time (s)	0.96	5.38	8.79	12.97	10.41	11.56	11.89	10.97	7.53
KSC	Parameters (M)	/	2.03	0.31	0.93	0.52	0.33	0.35	0.32	0.07
	FLOPs (M)	/	123.55	83.27	86.47	128.67	56.15	44.35	55.95	36.36
	Training time (s)	0.67	12.97	21.49	29.03	22.76	17.10	27.70	17.02	15.80
	Test time (s)	0.06	1.05	1.52	1.79	1.53	1.39	1.25	1.64	1.08
BS	Parameters (M)	/	1.76	0.27	0.65	0.44	0.28	0.30	0.27	0.06
	FLOPs (M)	/	101.82	68.77	64.92	106.07	46.39	36.91	46.18	30.04
	Training time (s)	0.51	4.50	9.32	15.01	11.72	10.06	13.22	9.53	5.64
	Testing time (s)	0.04	1.01	1.33	1.64	1.99	1.78	1.81	1.66	1.15
HS	Parameters (M)	/	1.74	0.27	0.63	0.43	0.27	0.29	0.26	0.06
	FLOPs (M)	/	100.38	67.81	63.80	104.56	45.74	36.42	45.53	29.62
	Training time (s)	3.21	9.67	20.91	21.85	24.63	22.18	22.17	23.37	13.39
	Testing time (s)	0.57	1.89	2.03	2.52	2.78	2.85	3.75	2.83	1.23
SA	Parameters (M)	/	2.47	0.39	1.53	0.66	0.42	0.44	0.41	0.08
	FLOPs (M)	/	158.31	106.47	126.12	164.82	71.77	56.25	71.57	42.26
	Training time (s)	41.46	59.77	222.32	424.52	298.42	272.88	326.39	264.63	120.11
	Testing time (s)	6.36	10.77	12.47	16.34	20.15	28.21	27.61	26.23	15.81

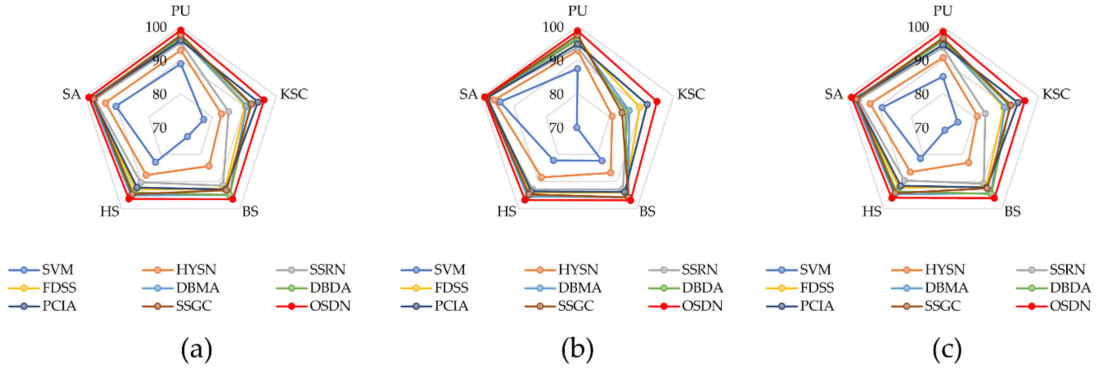


Figure 14. Classification results at different methods on the five datasets. (**a**) *OA*. (**b**) *AA*. (**c**) *Kappa*.

5.4. Comparison of Different Dense Connections

To verify the effectiveness and lightness of the proposed one-shot dense block (OSDB), we compared it with two other classical dense blocks, namely, the dense block (DB) and the weak dense block (WDB) [53], as shown in Figure 15. Note that the overall structure of the proposed OSDN remained unchanged; only the feature extraction blocks of OSDN were replaced by DB and WDB, respectively. The number of parameters, FLOPs, and *OA* of the three dense blocks is listed in Tables 17–21. The experimental results show that OSDB had fewer parameters and FLOPs; meanwhile, the *OA* was acceptable on the five datasets. From these five tables, although DB achieved the highest *OA*, it had a large number of parameters and FLOPs, which increased the complexity of the model. In addition, since WDB only retained the skip connection between the two *Conv* layers in DB, the parameters and FLOPs were reduced to some extent. However, the *OA* was also reduced. Lastly, our proposed OSDN not only connected the subsequent feature maps at once but also incorporated the residual connections. It enabled the proposed OSDB to maintain accuracy and reduce the amount of computation. In conclusion, although our proposed OSDB did not achieve the best for all indicators, it is acceptable and reasonable from the motivation of reducing computation cost and complexity.

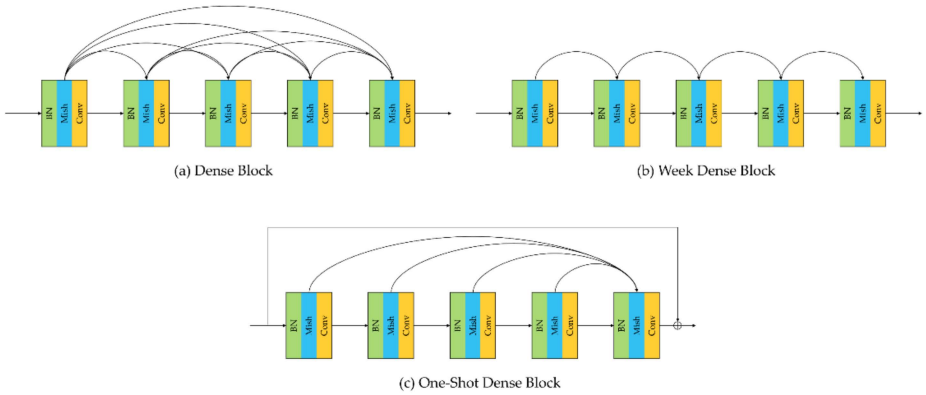


Figure 15. Different dense blocks. (**a**) Dense block. (**b**) Week dense block. (**c**) One-shot dense block.

Table 17. The number of parameters, FLOPs, and *OA* of different dense blocks on the PU dataset.

Dataset	Block	Parameters (M)	FLOPs (M)	*OA* (%)
PU	DB	0.29	49.49	98.99
	WDB	0.05	25.13	98.02
	OSDB	0.05	21.18	98.83

Table 18. The number of parameters, FLOPs, and *OA* of different dense blocks on the KSC dataset.

Dataset	Block	Parameters (M)	FLOPs (M)	*OA* (%)
KSC	DB	0.47	84.88	96.74
	WDB	0.07	43.12	95.93
	OSDB	0.07	36.36	96.09

Table 19. The number of parameters, FLOPs, and *OA* of different dense blocks on the BS dataset.

Dataset	Block	Parameters (M)	FLOPs (M)	*OA* (%)
BS	DB	0.41	70.14	96.89
	WDB	0.07	35.63	96.28
	OSDB	0.06	30.04	96.41

Table 20. The number of parameters, FLOPs, and *OA* of different dense blocks on the HS dataset.

Dataset	Block	Parameters (M)	FLOPs (M)	*OA* (%)
HS	DB	0.40	69.15	96.93
	WDB	0.07	35.13	96.11
	OSDB	0.06	29.62	96.28

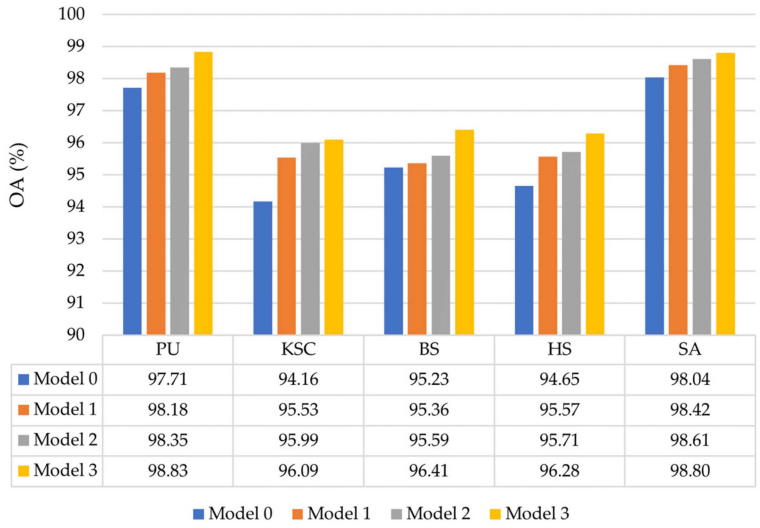
Table 21. The number of parameters, FLOPs, and *OA* of different dense blocks on the *SA* dataset.

Dataset	Block	Parameters (M)	FLOPs (M)	*OA* (%)
SA	DB	0.60	108.48	99.01
	WDB	0.09	55.12	98.75
	OSDB	0.08	42.26	98.80

5.5. Ablation Analysis toward the Attention Module

This subsection describes an ablation analysis performed on the attention module on five datasets. For a fair comparison, all the networks were trained with the same hyperparameters and samples, as described in Section 4.3. As shown in Figure 16, "Model 0" represents that the PAM was not used in the OSDN; "Model 1" and "Model 2" represent that

spatial-only PAM and channel-only PAM were used in the proposed network, respectively; and "Model 3" represents both spatial-only PAM and channel-only PAM being used in the OSDN. According to the results, we can observe that both Model 1 and Model 2 effectively improved the *OA* over Model 0 on the five datasets. It is worthwhile to note that even though the OA of Model 0 was already very high on the SA dataset, Model 1, Model 2, and Model 3 improved the *OA* by 0.38%, 0.57%, and 0.76%, respectively. The experimental results consistently show that compared with using Module 1 alone or Module 2 alone, Model 3 achieved the best *OA* on all datasets. Furthermore, we further analyzed the impact of the attention module (Model 3) on the computational cost of the OSDN. After extensive experiments, we observed that the computation times before and after incorporating Model 3 into the OSDN were ≈0.0051 and ≈0.0064 s, respectively. In addition, the FLOPs and parameters of Model 3 were 0.04 M and 0.001 M, respectively. Therefore, the introduced attention mechanism did not degrade the computational cost and complexity of the OSDN. At the same time, the introduced attention module can select the important channel and spatial features to improve the classification performance of OSDN.



	PU	KSC	BS	HS	SA
Model 0	97.71	94.16	95.23	94.65	98.04
Model 1	98.18	95.53	95.36	95.57	98.42
Model 2	98.35	95.99	95.59	95.71	98.61
Model 3	98.83	96.09	96.41	96.28	98.80

Figure 16. *OA* (%) of OSDN with different attention models on five datasets.

6. Conclusions

In this article, we aimed to construct an OSDN to solve the current problems of high complexity and inadequate feature extraction of CNN-based HSI classification models in the case of small training samples. By incorporating one-shot dense block, the number of parameters and computational cost of the network were significantly reduced while guaranteeing an excellent feature extraction ability. Moreover, to fully extract refined and discriminative features, the polarized AMs were introduced in the proposed OSDN. Compared to other previous AMs used in HSI classification models, the polarized AMs can maintain high channel and spatial resolution during the training process. In addition, some advanced techniques, including the BN layer, the Mish activation function, the cosine annealing learning rate, the dropout layer, and early stop operation, were used in the OSDN to prevent overfitting and accelerate network convergence.

The experiments demonstrated the effectiveness of two crucial parts in the OSDN, namely, one-shot dense block and polarized AMs. Moreover, several state-of-the-art models, such as HYSN, SSRN, FDSS, DBMA, DBDA, PCIA, and SSGC, were used for comparison on five HSI datasets. In the case of a few training samples, the classification results consistently demonstrated that the OSDN not only accurately predicted the labeled samples but also reasonably predicted the unlabeled samples. At the same time, compared with other comparison models, the proposed OSDN is an efficient lightweight model, which can achieve good classification performance with less computational cost even under limited training samples. In our future work, we will investigate more effective and

lightweight models to extract discriminative features for HSI classification. Finally, the code developed for OSDN is available at https://github.com/HaiZhu-Pan/OSDN (accessed on 5 May 2022).

Author Contributions: Conceptualization, H.P. and M.L.; data curation, M.L.; formal analysis, H.P., M.L. and H.G.; methodology, H.P. and M.L.; software, M.L.; validation, M.L. and H.P.; writing—original draft, M.L.; writing—review and editing, H.P., H.G. and L.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 62071084, and the Fundamental Research Funds in Heilongjiang Provincial Universities, grant number 135509116.

Data Availability Statement: The Pavia University, Kennedy Space Center, Botswana, and Salinas Valley datasets are available online at http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes (accessed on 5 May 2022). The Houston 2013 dataset is available at https://www.grss-ieee.org/resources/tutorials/data-fusion-tutorial-in-spanish/ (accessed on 5 May 2022).

Acknowledgments: The authors would like to thank the handing editor and anonymous reviewers for their insights and comments.

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

References

1. Zhong, Y.; Cao, Q.; Zhao, J.; Ma, A.; Zhao, B.; Zhang, L. Optimal decision fusion for urban land-use/land-cover classification based on adaptive differential evolution using hyperspectral and LiDAR data. *Remote Sens.* **2017**, *9*, 868. [CrossRef]
2. Lu, B.; Dao, P.D.; Liu, J.; He, Y.; Shang, J. Recent advances of hyperspectral imaging technology and applications in agriculture. *Remote Sens.* **2020**, *12*, 2659. [CrossRef]
3. Lorenz, S.; Salehi, S.; Kirsch, M.; Zimmermann, R.; Unger, G.; Vest Sørensen, E.; Gloaguen, R. Radiometric correction and 3D integration of long-range ground-based hyperspectral imagery for mineral exploration of vertical outcrops. *Remote Sen.* **2018**, *10*, 176. [CrossRef]
4. Audebert, N.; Le Saux, B.; Lefèvre, S. Deep learning for classification of hyperspectral data: A comparative review. *IEEE Geosci. Remote Sens.* **2019**, *7*, 159–173. [CrossRef]
5. Shahshahani, B.M.; Landgrebe, D.A. The effect of unlabeled samples in reducing the small sample size problem and mitigating the Hughes phenomenon. *IEEE Trans. Geosci. Remote Sens.* **1994**, *32*, 1087–1095. [CrossRef]
6. Li, S.; Song, W.; Fang, L.; Chen, Y.; Ghamisi, P.; Benediktsson, J.A. Deep learning for hyperspectral image classification: An overview. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 6690–6709. [CrossRef]
7. Zhang, Y.; Cao, G.; Li, X.; Wang, B.; Fu, P. Active semi-supervised random forest for hyperspectral image classification. *Remote Sens.* **2019**, *11*, 2974. [CrossRef]
8. Ma, L.; Crawford, M.M.; Tian, J. Local manifold learning-based k-nearest-neighbor for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2010**, *48*, 4099–4109. [CrossRef]
9. Kuo, B.-C.; Ho, H.-H.; Li, C.-H.; Hung, C.-C.; Taur, J.-S. A kernel-based feature selection method for SVM with RBF kernel for hyperspectral image classification. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2013**, *7*, 317–326. [CrossRef]
10. Kang, X.; Xiang, X.; Li, S.; Benediktsson, J.A. PCA-based edge-preserving features for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 7140–7151. [CrossRef]
11. Bruce, L.M.; Koger, C.H.; Li, J. Dimensionality reduction of hyperspectral data using discrete wavelet transform feature extraction. *IEEE Trans. Geosci. Remote Sens.* **2002**, *40*, 2331–2338. [CrossRef]
12. Falco, N.; Benediktsson, J.A.; Bruzzone, L. A study on the effectiveness of different independent component analysis algorithms for hyperspectral image classification. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 2183–2199. [CrossRef]
13. Jia, S.; Wu, K.; Zhu, J.; Jia, X. Spectral–spatial Gabor surface feature fusion approach for hyperspectral imagery classification. *IEEE Trans. Geosci. Remote Sens.* **2018**, *57*, 1142–1154. [CrossRef]
14. Jia, S.; Hu, J.; Zhu, J.; Jia, X.; Li, Q. Three-dimensional local binary patterns for hyperspectral imagery classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 2399–2413. [CrossRef]
15. Gu, Y.; Liu, T.; Jia, X.; Benediktsson, J.A.; Chanussot, J. Nonlinear multiple kernel learning with multiple-structure-element extended morphological profiles for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 3235–3247. [CrossRef]
16. Cao, X.; Zhou, F.; Xu, L.; Meng, D.; Xu, Z.; Paisley, J. Hyperspectral image classification with Markov random fields and a convolutional neural network. *IEEE Trans. Image Process.* **2018**, *27*, 2354–2367. [CrossRef]
17. Wang, P.; Fan, E.; Wang, P. Comparative analysis of image classification algorithms based on traditional machine learning and deep learning. *Pattern Recognit. Lett.* **2021**, *141*, 61–67. [CrossRef]

18. Lateef, F.; Ruichek, Y. Survey on semantic segmentation using deep learning techniques. *Neurocomputing* **2019**, *338*, 321–348. [CrossRef]
19. Zhang, L.; Zhang, L.; Du, B. Deep learning for remote sensing data: A technical tutorial on the state of the art. *IEEE Geosci. Remote Sens.* **2016**, *4*, 22–40. [CrossRef]
20. Chen, Y.; Lin, Z.; Zhao, X.; Wang, G.; Gu, Y. Deep learning-based classification of hyperspectral data. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 2094–2107. [CrossRef]
21. Hu, W.; Huang, Y.; Wei, L.; Zhang, F.; Li, H. Deep convolutional neural networks for hyperspectral image classification. *J. Sens.* **2015**, *2015*, 258619. [CrossRef]
22. Yu, C.; Zhao, M.; Song, M.; Wang, Y.; Li, F.; Han, R.; Chang, C.-I. Hyperspectral image classification method based on CNN architecture embedding with hashing semantic feature. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2019**, *12*, 1866–1881. [CrossRef]
23. Chen, Y.; Jiang, H.; Li, C.; Jia, X.; Ghamisi, P. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 6232–6251. [CrossRef]
24. Yu, S.; Jia, S.; Xu, C. Convolutional neural networks for hyperspectral image classification. *Neurocomputing* **2017**, *219*, 88–98. [CrossRef]
25. Mei, S.; Ji, J.; Geng, Y.; Zhang, Z.; Li, X.; Du, Q. Unsupervised spatial–spectral feature learning by 3D convolutional autoencoder for hyperspectral classification. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 6808–6820. [CrossRef]
26. Roy, S.K.; Krishna, G.; Dubey, S.R.; Chaudhuri, B.B. HybridSN: Exploring 3-D–2-D CNN feature hierarchy for hyperspectral image classification. *IEEE Geosci. Remote Sens. Lett.* **2019**, *17*, 277–281. [CrossRef]
27. Zhang, H.; Li, Y.; Jiang, Y.; Wang, P.; Shen, Q.; Shen, C. Hyperspectral classification based on lightweight 3-D-CNN with transfer learning. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 5813–5828. [CrossRef]
28. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
29. Zhong, Z.; Li, J.; Luo, Z.; Chapman, M. Spectral–spatial residual network for hyperspectral image classification: A 3-D deep learning framework. *IEEE Trans. Geosci. Remote Sens.* **2017**, *56*, 847–858. [CrossRef]
30. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
31. Wang, W.; Dou, S.; Jiang, Z.; Sun, L. A fast dense spectral–spatial convolution network framework for hyperspectral images classification. *Remote Sens.* **2018**, *10*, 1068. [CrossRef]
32. Tian, Z.; Zhan, R.; Hu, J.; Wang, W.; He, Z.; Zhuang, Z. Generating anchor boxes based on attention mechanism for object detection in remote sensing images. *Remote Sens.* **2020**, *12*, 2416. [CrossRef]
33. You, Q.; Jin, H.; Wang, Z.; Fang, C.; Luo, J. Image captioning with semantic attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4651–4659.
34. Atoum, Y.; Ye, M.; Ren, L.; Tai, Y.; Liu, X. Color-wise attention network for low-light image enhancement. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 506–507.
35. Fang, B.; Li, Y.; Zhang, H.; Chan, J.C.-W. Hyperspectral images classification based on dense convolutional networks with spectral-wise attention mechanism. *Remote Sens.* **2019**, *11*, 159. [CrossRef]
36. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7132–7141.
37. Li, J.; Cui, R.; Li, B.; Song, R.; Li, Y.; Dai, Y.; Du, Q. Hyperspectral image super-resolution by band attention through adversarial learning. *IEEE Trans. Geosci. Remote Sens.* **2020**, *58*, 4304–4318. [CrossRef]
38. Roy, S.K.; Dubey, S.R.; Chatterjee, S.; Chaudhuri, B.B. FuSENet: Fused squeeze-and-excitation network for spectral-spatial hyperspectral image classification. *IET Image Process.* **2020**, *14*, 1653–1661. [CrossRef]
39. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
40. Ma, W.; Yang, Q.; Wu, Y.; Zhao, W.; Zhang, X. Double-branch multi-attention mechanism network for hyperspectral image classification. *Remote Sens.* **2019**, *11*, 1307. [CrossRef]
41. Li, R.; Zheng, S.; Duan, C.; Yang, Y.; Wang, X. Classification of hyperspectral image based on double-branch dual-attention mechanism network. *Remote Sens.* **2020**, *12*, 582. [CrossRef]
42. Fu, J.; Liu, J.; Tian, H.; Li, Y.; Bao, Y.; Fang, Z.; Lu, H. Dual attention network for scene segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3146–3154.
43. Shi, C.; Liao, D.; Zhang, T.; Wang, L. Hyperspectral Image Classification Based on 3D Coordination Attention Mechanism Network. *Remote Sens.* **2022**, *14*, 608. [CrossRef]
44. Li, Z.; Cui, X.; Wang, L.; Zhang, H.; Zhu, X.; Zhang, Y. Spectral and spatial global context attention for hyperspectral image classification. *Remote Sens.* **2021**, *13*, 771. [CrossRef]
45. Cao, Y.; Xu, J.; Lin, S.; Wei, F.; Hu, H. GCNet: Non-local networks meet squeeze-excitation networks and beyond. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, Long Beach, CA, USA, 16–17 June 2019; pp. 1971–1980.
46. Shi, H.; Cao, G.; Ge, Z.; Zhang, Y.; Fu, P. Double-Branch Network with Pyramidal Convolution and Iterative Attention for Hyperspectral Image Classification. *Remote Sens.* **2021**, *13*, 1403. [CrossRef]

47. Liu, H.; Liu, F.; Fan, X.; Huang, D. Polarized self-attention: Towards high-quality pixel-wise regression. *arXiv* **2021**, arXiv:2107.00782.
48. Misra, D. Mish: A self regularized non-monotonic activation function. *arXiv* **2019**, arXiv:1908.08681.
49. Liu, D.; Han, G.; Liu, P.; Yang, H.; Sun, X.; Li, Q.; Wu, J. A Novel 2D-3D CNN with Spectral-Spatial Multi-Scale Feature Fusion for Hyperspectral Image Classification. *Remote Sens.* **2021**, *13*, 4621. [CrossRef]
50. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
51. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
52. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.
53. Ge, Z.; Cao, G.; Shi, H.; Zhang, Y.; Li, X.; Fu, P. Compound Multiscale Weak Dense Network with Hybrid Attention for Hyperspectral Image Classification. *Remote Sens.* **2021**, *13*, 3305. [CrossRef]