applied sciences

MDPI

Article

Hyperspectral Image Classification Method Based on Morphological Features and Hybrid Convolutional Neural Networks

Tonghuan Ran [1,2], Guangfeng Shi [1,*], Zhuo Zhang [2], Yuhao Pan [2] and Haiyang Zhu [2]

1 College of Mechanical and Electric Engineering, Changchun University of Science and Technology, Changchun 130022, China; ranth@ccut.edu.cn
2 School of Mechatronic Engineering, Changchun University of Technology, Changchun 130012, China; zhuo_0311@sina.com (Z.Z.); panyuhao_0819@sina.com (Y.P.); zhuhaiyang_1555@sina.com (H.Z.)
* Correspondence: sgfcust@sina.com; Tel.: +86-0431-85582693

Abstract: The exploitation of the spatial and spectral characteristics of hyperspectral remote sensing images (HRSIs) for the high-precision classification of earth observation targets is crucial. Convolutional neural networks (CNNs) have good classification performance and are widely used neural networks. Herein, a morphological processing (MP)-based HRSI classification method and a 3D–2D CNN are proposed to improve HRSI classification accuracy. Principal component analysis is performed to reduce the dimensionality of the HRSI cube, and MP is implemented to extract the spectral–spatial features of the low-dimensional HRSI cube. The extracted features are concatenated with the low-dimensional HRSI cube, and the designed 3D–2D CNN framework completes the classification task. Residual connections and an attention mechanism are added to the CNN structure to prevent gradient vanishing, and the scale of the control parameters of the model structure is optimized to guarantee the model's feature extraction ability. The CNN structure uses multiscale convolution, involving depthwise separable convolution, which can effectively reduce the amount of parameter calculation. Two classic datasets (Indian Pines and Pavia University) and a self-made dataset (My Dataset) are used to compare the performance of this method with existing classification techniques. The proposed method effectively improved classification accuracy despite its short classification time.

Keywords: convolutional neural nets; hyperspectral imaging; mathematical morphology; principal component analysis

check for updates

Citation: Ran, T.; Shi, G.; Zhang, Z.; Pan, Y.; Zhu, H. Hyperspectral Image Classification Method Based on Morphological Features and Hybrid Convolutional Neural Networks. *Appl. Sci.* **2024**, *14*, 10577. https://doi.org/10.3390/app142210577

Academic Editor: Douglas O'Shaughnessy

Received: 23 October 2024
Revised: 13 November 2024
Accepted: 14 November 2024
Published: 16 November 2024

1. Introduction

Hyperspectral remote sensing image technology refers to a comprehensive sensing technique that acquires and analyzes the data and information of ground objects without direct contact with distant research targets and regions through specific devices. Remote sensing technology is characterized by its scientificity, practicability, and advancement. It is an interdisciplinary subject integrating mathematics, computer science, geography, and other disciplines. A hyperspectral remote sensing image (HRSI) is a three-dimensional (3D) structure containing two-dimensional (2D) spatial information and rich spectral information. Making full use of such spectral and spatial information can significantly improve the accuracy of HRSI classification [1]. Hyperspectral remote sensing is widely used for disaster monitoring [2], mineral exploration [3], precision agriculture [4], and military reconnaissance [5], and HRSI classification is an important part of these applications. Classical classification methods include random forest [6], support vector machine (SVM) [7], K-nearest neighbor [8], and logistic regression [9] approaches. However, these traditional machine learning methods only consider the spectral features of HRSIs; the importance of the image's spatial features is ignored, and the classification results are not ideal. Many spatial feature extraction methods have been proposed to solve this problem, such as the use of morphological features [10], texture features [11], and edge preservation filters [12]. The high spatial resolution of hyperspectral images allows for a very small number of hybrid pixels and provides clear boundaries between different objects [13]. Spatial features, such as morphological features, can provide high classification accuracy.

Recent studies show that the classification accuracy of hyperspectral images using deep learning is high in practice [14]. Deep learning was developed on the basis of machine learning (through the study of artificial neural networks); a machine mimics the information processing structure of the human brain [15]. Deep learning often involves multiple levels of neural network structures, and a neural network automatically learns from a large number of samples during stepwise training to obtain deep features from the data [16]. As typical models in the field of deep learning, convolutional neural networks (CNNs) are vital in the field of computer vision. A CNN is a deep feedforward neural network with a convolutional structure in which neurons can respond to cells in a region with a central point; CNNs process large images well and are widely used in the field of image processing [17]. Vaddi et al. [18] used probabilistic principal component analysis (PCA) and Gabor filters to extract spectral and spatial features, respectively, and fused these features into their designed 2D CNN framework for classification. Ahmad et al. [19] proposed a fast 3D CNN model to extract spatial–spectral features and improve hyperspectral image classification performance. Yang et al. [20] proposed a deep CNN with a two-branch structure; they used one-dimensional (1D) convolution and 2D convolution to extract spectral and spatial features, respectively, and combined them for classification. Driven by ResNet, Zhong et al. [21] designed a spatial–spectral residual network (SSRN) to classify hyperspectral images and obtained good results. Wang et al. [22] proposed an end-to-end fast dense spectral–spatial convolution network framework. Different convolution kernels were used to extract multiscale spectral–spatial features, and features at multiple scales were extracted. Both 1D and 2D convolution can be used to extract spectral and spatial features, but the spectral–spatial relationship is ignored. Three-dimensional convolution can directly extract spectral–spatial features, but the use of 3D convolution itself complicates calculations. Therefore, Roy et al. [23] proposed a model that stacks 3D and 2D convolutional layers to make full use of spectral and spatial features to improve classification accuracy. CNN-based methods have achieved good results in the field of hyperspectral image classification, but the contribution of the feature map outputs of each convolutional layer of classification is different. The performance and generalization ability of a model should be improved to enable the neural network to focus selectively on important information in the input. Hu et al. [24] constructed a so-called squeeze incentive network and achieved remarkable results in the 2017 Large-Scale Visual Identity Challenge classification competition. Sergio R et al. [25] proposed a new machine learning-based tool called VULMA (Vulnerability Analysis using Machine Learning) that is able to capture the key features of buildings in existing inventory starting with a simple photo of the building. Their approach gave us a lot of ideas.

We found that it is difficult for the traditional hyperspectral remote sensing image classification method to distinguish some highly similar objects effectively in the face of complex objects. Inspired by the above research, we designed an HRSI classification method based on morphological processing (MP) and a 3D–2D CNN to improve the classification accuracy of hyperspectral remote sensing images and the feature discrimination ability of the network:

1. A new HRSI classification framework was designed. This framework consists of PCA, MP, CNNs, residual connections, and an attention mechanism.
2. A new 3D–2D CNN model was designed. This model combines 3D convolution for extracting spatial and spectral features and 2D convolution for extracting spatial features only. Such a combination effectively improves the classification accuracy of HRSIs.

3. Combining residual connections and the attention mechanism establishes a multiscale residual attention module to refine feature mapping.
4. The 3D–2D CNN structure uses multiscale convolution composed of depthwise separable convolution (DSC), which can effectively reduce the amount of parameter calculation and prevent overfitting.

2. Problem Formulation

An HRSI is a data cube with two spatial dimensions and one spectral dimension. An HRSI can be expressed as $O \in R^{h \times w \times b}$, where O is the original HRSI; h and w are the spatial height and width of the HRSI, respectively; and b is the number of bands. The first p principal components ($O_p \in R^{h \times w \times p}$) are retained after PCA dimensionality reduction. The first J principal components are selected for binarization, and the spatial features ($O_{p_J} \in R^{h \times w \times j}$) of the binarized data are extracted via MP. O_p and O_{p_J} are concatenated to obtain $O_A \in R^{h \times w \times A}$, $A = p + 3j$. The input patch is $O_{patch} \in R^{s \times s \times A}$; finally, the created HRSI patch is fed to the 3D–2D CNN.

The flow of the proposed HRSI classification method is illustrated in Figure 1. First, PCA is used to reduce the dimensionality of the HRSI cube, and the first p principal components are extracted from the low-dimensional HRSI cube. The first j principal components are binarized, and the spatial features of the binary data are extracted through MP. Then, the low-dimensional HRSI cube and spatial features are concatenated. Finally, the designed 3D–2D CNN framework is used to complete classification.

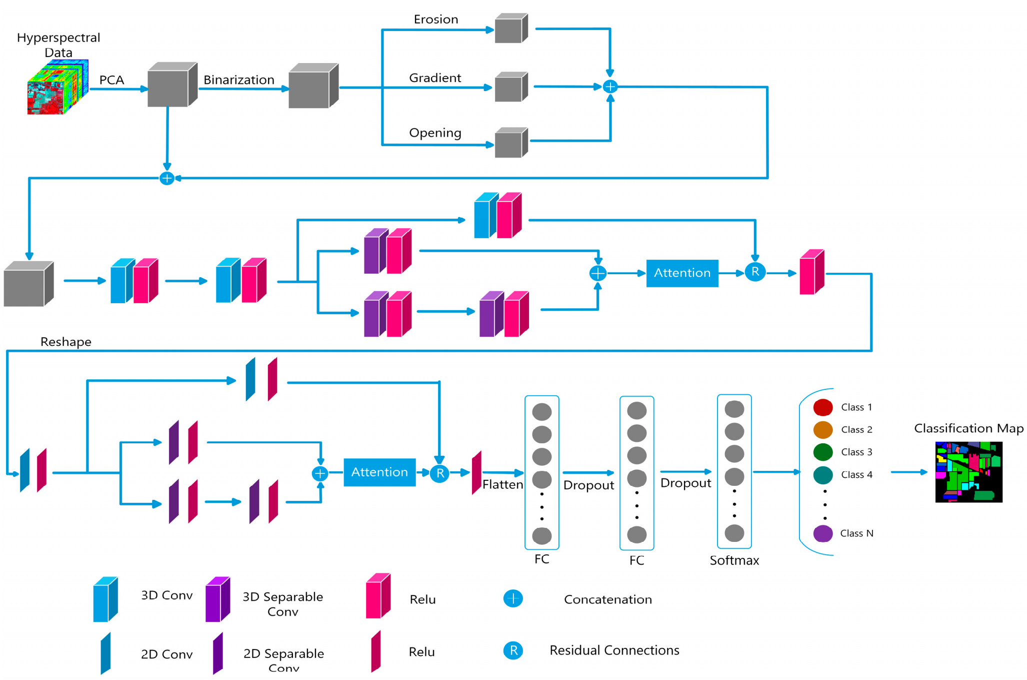

Figure 1. Network structure.

2.1. PCA

PCA reduces the dimensionality of high-dimensional data by converting them into a low-dimensional subspace. The use of PCA to reduce the dimensionality of an original hyperspectral image can effectively accelerate feature extraction [26]. The pixels of the HRSI data cube are expressed as vectors $t_i = [t_1, t_2, t_3, \ldots \ldots, t_x]_i^T$, and the average value of the pixel vectors is as follows:

$$t_{avg} = \frac{1}{s}\sum_{i=1}^{n} [t_1,t_2,t_3 \cdots \cdots t_x]_i^T, \tag{1}$$

where $s = r \times c$, where r is the row of the pixel vector, and c is the column of the pixel vector. The formula for the covariance matrix is as follows:

$$\sigma = \frac{1}{s}\sum_{i=1}^{n}(t_i - s)(t_i - s)^T. \tag{2}$$

The feature decomposition of the covariance matrix is as follows:

$$\sigma = V\Lambda V^T, \tag{3}$$

where Λ is a diagonal matrix composed of eigenvalues, and V is an orthogonal matrix, with the corresponding eigenvector being a column. The linear conversion of the original HRSI yields the following data after dimensionality reduction:

$$y_i = V^T t_i (i = 1, 2, 3 \ldots). \tag{4}$$

The rows of V^T are arranged according to the eigenvalues (from large to small), and the selected first p rows and pixel t_i are multiplied to obtain the PCA spectral band composed of most of the original HRSI information.

2.2. Binarization Process

Binarization is the process of converting the input value from 0 to 1. First, the data are converted into a range of 0 to 255 via the following formula:

$$G_{ij} = \frac{I_{ij} - \min(I)}{I_{ij}} * 255, \tag{5}$$

where I is the input image, and I_{ij} is the pixel value of I at the ij position. After the range is converted, a threshold must be selected as follows:

$$Th = \frac{\sum_{i=0}^{h-1}\sum_{j=1}^{w-1} G_{ij}}{h * w * j}, \tag{6}$$

where h and w are the height and width of the input data, respectively, and j is the number of bands that are binary. The value is 1 when G_{ij} is greater than or equal to Th and 0 when it is less than Th. The formula is as follows:

$$B_{ij} = \begin{cases} 1, & if \quad G_{ij} \geq Th \\ 0, & if \quad G_{ij} < Th \end{cases}. \tag{7}$$

2.3. MP

Morphological analysis was proposed by Serra et al. [27] in 1982 to collect information, such as image shapes and boundaries, using structural elements. This method captures the spatial morphology of land-cover types and eliminates the interference between different types. The two basic morphological operations are erosion and dilation. Their formulas are as follows [13].

Erosion:

$$(BI \ominus g)(i,j) = \min[BI(i+m),(j+n) - g(m,n)], \tag{8}$$

Dilation (an erosion dual operator):

$$(BI \oplus g)(i,j) = \min[BI(i+m),(j+n) - g(m,n)], \tag{9}$$

where $BI(i,j)$ is a binary image; $g(m,n)$ is a structural element; and and are the symbols for erosion and dilation, respectively.

Structural elements:

$$SE(i,j) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{10}$$

Opening:

$$(BI \cdot g)(i,j) = (BI \ominus g \oplus g)(i,j) \tag{11}$$

Gradient:

$$G(i,j) = (BI \oplus g - BI \ominus g)(i,j) \tag{12}$$

The proposed framework uses the erosion, opening, and gradient operations. Erosion removes pixels from the object's boundary; the opening operation eliminates small areas of the image; the gradient operation provides boundary information for objects.

2.4. CNNs

A 2D CNN performs convolution through a 2D convolution kernel, which can move in two directions on a 2D plane, and the convoluted features improve nonlinear expression through the activation function. The output features are as follows:

$$f_j^l = a\left(\sum_{i \in S_j} f_i^{l-1} * \omega_{ij}^l + b_j^l\right), \tag{13}$$

where f_j^l is the output feature after convolution, a is the activation function, S_j is the set of pixel features, f_i^{l-1} is the feature map of the previous layer, * is the convolution operation, ω_{ij}^l is the convolution kernel weight of the i and j positions of the l layer, and b_j^l is the bias.

According to the above formula, the feature value formula for each pixel is as follows:

$$\lambda_{ij}^{xy} = a\left(\sum_{c=0} \sum_{m=0}^{M_i-1} \sum_{n=0}^{N_i-1} \omega_{ijc}^{mn} u_{(i-1)c}^{(x+m)(x+n)} + b_{ij}\right), \tag{14}$$

where λ_{ij}^{xy} is the eigenvalue of the pixel; xy is the position of the pixel in the eigenvalue; xy is the position of the eigenvalue in the eigengram; a is the activation function; c is the index value of the number of eigenvalues; M_i and N_i represent the size of the convolution kernel of the ith layer; ω_{ijc}^{mn} is the weight of the corresponding position; $u_{(i-1)c}^{(x+m)(x+n)}$ represents the $i-1$ layer, the cth feature map, and the eigenvalue at the $(x+m)(x+n)$ position; and b_{ij} is the bias.

A 3D CNN performs convolution through a 3D convolution kernel, which can move in three directions: height, width, and channel. It can simultaneously combine the spatial and spectral features of an HRSI, making full use of the structural characteristics of the image. According to the feature representation of the above 2D convolution, the eigenvalue λ_{ij}^{xyz} of the xyz pixel of the jth feature map is deduced for layer i, as follows:

$$\lambda_{ij}^{xyz} = a\left(\sum_{c=0} \sum_{m=0}^{M_i-1} \sum_{n=0}^{N_i-1} \sum_{l=0}^{L_i-1} \omega_{ijc}^{mnl} u_{(i-1)c}^{(x+m)(y+n)(z+l)} + b_{ij}\right), \tag{15}$$

where M_i, N_i, and L_i are the sizes of the i-layer convolution kernels.

2.5. DSC

DSC was proposed by Howard et al. [28] and used in MobileNetV1. DSC integrates standard convolution into depthwise convolution, which is used to extract features from input channels, and pointwise convolution is combined with the depthwise convolution output [29]. Compared with standard convolution, DSC significantly reduces the computation effort. The ratio of the number of 2D DSC parameters to the number of standard 2D convolution parameters and the amount of computation are as follows:

$$P_{2D}^{r} = \frac{P_{2D}^{DSC}}{P_{2D}^{CNN}} = \frac{k_{2D}k_{2D}I + IO}{k_{2D}k_{2D}IO} = \frac{1}{O} + \frac{1}{k_{2D}^{2}}, \tag{16}$$

$$C_{2D}^{r} = \frac{C_{2D}^{DSC}}{C_{2D}^{CNN}} = \frac{k_{2D}k_{2D}Ixy + IOxy}{k_{2D}k_{2D}IOxy} = \frac{1}{O} + \frac{1}{k_{2D}^{2}}, \tag{17}$$

where P_{2D}^{r} is the ratio of the number of 2D DSC parameters to the number of standard 2D convolution parameters, P_{2D}^{DSC} is the number of 2D DSC parameters, P_{2D}^{CNN} is the number of standard 2D convolution computations, k_{2D} is the size of the convolution kernel, the input size is $x \times y$, I is the number of channels of the input feature map, and O is the number of output channels. C_{2D}^{r} is the ratio of the number of 2D DSC computations to the number of standard 2D convolution computations, C_{2D}^{DSC} is the number of 2D DSC computations, C_{2D}^{CNN} is the number of standard 2D convolution computations, and the ratio of the number of 3D DSC computations to the number of standard 3D convolution computations is as follows:

$$P_{3D}^{r} = \frac{P_{3D}^{DSC}}{P_{3D}^{CNN}} = \frac{k_{3D}k_{3D}k_{3D}I + IO}{k_{3D}k_{3D}k_{3D}IO} = \frac{1}{O} + \frac{1}{k_{3D}^{3}}, \tag{18}$$

$$C_{3D}^{r} = \frac{C_{3D}^{DSC}}{C_{3D}^{CNN}} = \frac{k_{3D}k_{3D}k_{3D}Ixyz + IOxyz}{k_{3D}k_{3D}k_{3D}IOxyz} = \frac{1}{O} + \frac{1}{k_{3D}^{3}}, \tag{19}$$

where P_{3D}^{r} is the ratio of the number of 3D DSC parameters to the number of standard 3D convolution parameters, P_{3D}^{DSC} is the number of 3D DSC parameters, P_{3D}^{CNN} is the number of standard 3D convolution computations, k_{3D} is the size of the convolution kernel, the input size is $x \times y \times z$, I is the number of channels of the input feature map, and O is the number of output channels. C_{3D}^{r} is the ratio of the number of 3D DSC computations to the number of standard 3D convolution computations, C_{3D}^{DSC} is the number of 3D DSC computations, and C_{3D}^{CNN} is the number of standard 3D convolution computations. The above formulas show that DSC can effectively reduce the number of parameters and the amount of calculation.

2.6. Residual Connections

In deep learning, as the network depth increases, accuracy tends to be saturated and then degrades rapidly, resulting in poor network training. Residual networks were proposed to address such network degradation. The inputs and outputs of residual cells are denoted as follows [30]:

$$x_{l} = F_{l}(x_{l-1}) + x_{l-1}, \tag{20}$$

where F is the residual function, x_{l-1} is the input of the element, and x_{l} is the output of the element.

2.7. Attention Mechanism

Squeeze-and-excitation (SE) blocks are new building blocks introduced by Hu et al. [24] to improve network performance by explicitly modeling the interdependencies between the characteristic channels of network evolution and introducing an attention mechanism between the channels. SE blocks can map the input X ($X \in R^{H \times W \times C}$) to U ($U \in R^{H \times W \times C}$). For any given F_{tr} transformation, F_{tr} is regarded as a simple convolution operation, and the input is represented as $V = [v_1, v_2, \ldots, v_C]$. V_c represents the parameters of the cth convolution kernel, and the output of F_{tr} is $U = [u_1, u_2, \ldots, u_C]$. The formula is as follows:

$$\mathbf{u}_C = \mathbf{v}_C * \mathbf{X} = \sum_{s=1}^{C'} \mathbf{v}_C^s * \mathbf{x}^s, \tag{21}$$

where $*$ denotes convolution, $v_c = \left[v_c^1, v_c^2, \ldots v_c^C\right]$, and $X = \left[x^1, x^2 \ldots x^C\right]$.

$Z \in R^C$ is obtained via the global average pooling of the feature U on the spatial dimension $H \times W$, where the c element of z is as follows:

$$z_C = \mathbf{F}_{sq}(\mathbf{u}_C) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_C(i,j) \tag{22}$$

The channel dependencies are fully obtained to use the information of the squeeze operation. A simple gating mechanism with sigmoid activation is implemented as follows:

$$\mathbf{s} = \mathbf{F}_{ex}(\mathbf{z}, \mathbf{W}) = \sigma(g(\mathbf{z}, \mathbf{W})) = \sigma(\mathbf{W}_2 \delta(\mathbf{W}_1 \mathbf{z})), \tag{23}$$

where σ is the sigmoid activation function, and δ is the ReLU activation function. $W_1 \in R^{\frac{C}{r} \times C}$, $W_2 \in R^{\frac{C}{r} \times C}$. Activation swap is used to rescale the feature map and obtain the final output of the block.

$$\widetilde{x}_C = \mathbf{F}_{\text{scale}}(\mathbf{u}_C, s_C) = s_C \cdot \mathbf{u}_C, \tag{24}$$

where $\widetilde{X} = \left[\widetilde{x}_1, \widetilde{x}_2 \ldots \widetilde{x}_C\right]$, $u_c \in R^{W \times H}$, and $F_{scale}(u_C, s_C)$ is the product of the channel.

3. Experiments and Discussion

3.1. Dataset Description

In this paper, three hyperspectral image datasets [31] (Indian Pines [IP], Pavia University [PU], and the self-made dataset "My Dataset") are used to validate the proposed method. Figures 2–4 show diagrams of the true categories of the hyperspectral images.

IP: This dataset includes 145 × 145 pixels with 200 spectral bands with a spatial resolution of approximately 20 m at wavelengths of 0.5 to 2.5 µm. The image contains 16 different land-cover categories.

PU: This dataset includes 610 × 340 pixels with 200 spectral bands with a spatial resolution of approximately 1.3 m at wavelengths of 0.43 to 0.86 µm. The image contains nine different land-cover categories.

My Dataset: This dataset includes 182 × 217 pixels with 135 spectral bands at wavelengths of 0.3 to 0.9 µm. The image contains seven different land-cover categories.

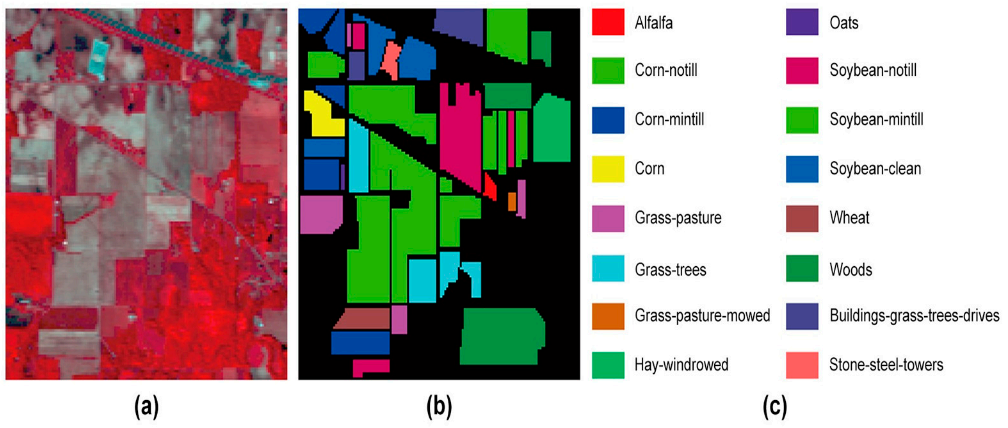

Figure 2. IP dataset: (**a**) false-color image, (**b**) ground-truth image, and (**c**) class names.

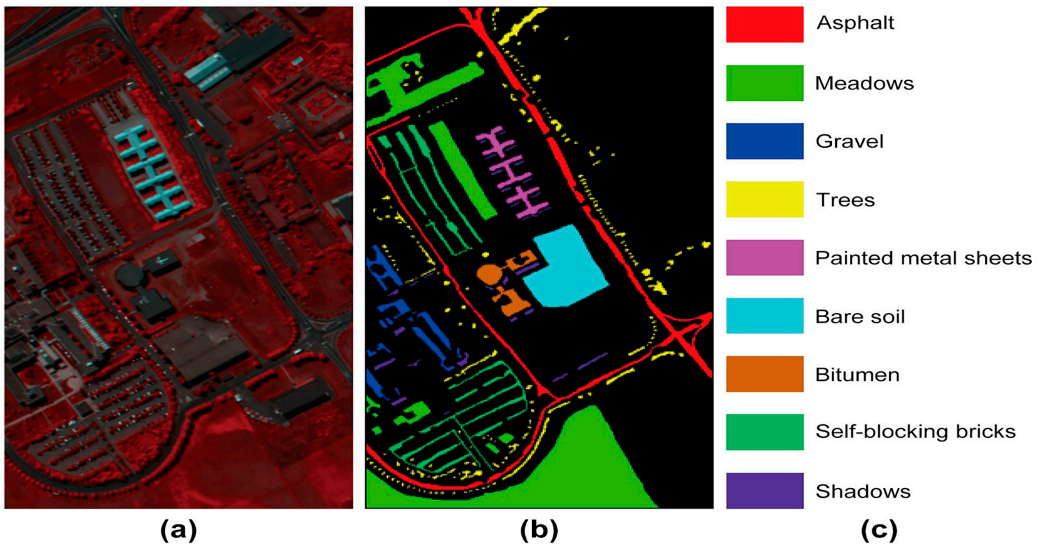

Figure 3. PU dataset: (**a**) false-color image, (**b**) ground-truth image, and (**c**) class names.

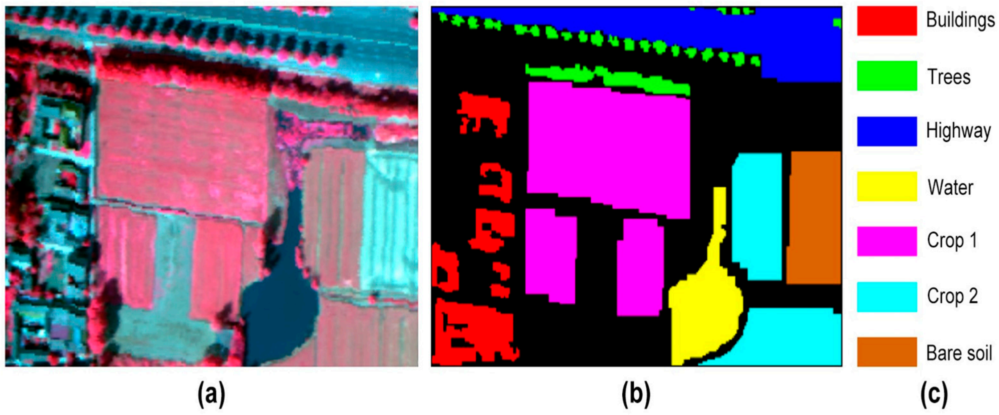

Figure 4. My Dataset: (**a**) false-color image, (**b**) ground-truth image, and (**c**) class names.

3.2. Parameter Setting

The proposed model divides the total input into the training set (30%) and the test set (70%). All datasets are optimized using the Adam optimizer, the learning rate is 0.001, the learning rate decay is (0.9, 0.999), the training batch size is 256, 100 repetitions are performed, the loss function is the cross-entropy loss function, and the patch size is 21 $\times$ 21. For IP and My Dataset, the first two principal components are selected using 14 principal components and a binarization process. For PU, the first principal component is selected using seven principal components and a binarization process (Tables 1–3).

Table 1. Framework structure of IP dataset.

Layer (Type)	Kernel Size	Stride	Padding	Output Shape
Input layer				Input: [(21,21,20,1)]
Conv3D_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_1: (19,19,18,8)
Conv3D_2	(3,3,3)	(1,1,1)	(0,0,0)	Out_2: (17,17,16,16)
Separable_Conv3D_3_1_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_3_1_1: (15,15,14,32)
Conv3D_3_1_2	(1,1,1)	(1,1,1)	(0,0,0)	Out_3_1_2: (15,15,14,16)
Separable_Conv3D_3_2_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_3_2_1: (15,15,14,32)
Conv3D_3_2_2	(1,1,1)	(1,1,1)	(0,0,0)	Out_3_2_2: (15,15,14,32)
Separable_Conv3D_3_2_3	(3,3,3)	(1,1,1)	(0,0,0)	Out_3_2_3: (13,13,12,32)
Conv3D_3_2_4	(1,1,1)	(1,1,1)	(1,1,1)	Out_3_2_4: (15,15,14,16)
Concatenate_1 (Out_3_1_2, Out_3_2_4)				Out_C_1: (15,15,14,32)

Table 1. *Cont.*

Layer (Type)	Kernel Size	Stride	Padding	Output Shape
Residual Connection_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_R_1: (15,15,14,32)
Add (Out_C1, Out_R1)				Out_A1: (15,15,14,32)
Reshape (Out_A1)				Out_Re: (15,15,448)
Separable_Conv2D_4_1_1	(3,3)	(1,1)	(0,0)	Out_4_1_1: (13,13,64)
Conv2D_4_1_2	(1,1)	(1,1)	(0,0)	Out_4_1_2: (13,13,32)
Separable_Conv2D_4_2_1	(3,3)	(1,1)	(0,0)	Out_4_2_1: (13,13,64)
Conv2D_4_2_2	(1,1)	(1,1)	(0,0)	Out_4_2_2: (13,13,64)
Separable_Conv2D_4_2_3	(3,3)	(1,1)	(0,0)	Out_4_2_3: (11,11,64)
Conv2D_4_2_4	(1,1)	(1,1)	(1,1)	Out_4_2_4: (13,13,32)
Concatenate_2 (Out_4_1_2, Out_4_2_4)				Out_C_2: (13,13,64)
Attention				Out_SE: (13,13,64)
Residual Connection_2	(3,3)	(1,1)	(0,0)	Out_R_2: (13,13,64)
Add (Out_C_2, Out_R_2)				Out_A_2: (13,13,64)
Flatten				Out_F: (10,816)
Linear_1				Out_L_1: (256)
Dropout_1				Out_D_1: (256)
Linear_2				Out_L_2: (128)
Dropout_2				Out_D_2: (128)
Linear_3				Out_L_3: (16)
Total Parameters: 3,270,656				

Table 2. Framework structure of My Dataset.

Layer (Type)	Kernel Size	Stride	Padding	Output Shape
Input layer				Input: [(21,21,20,1)]
Conv3D_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_1: (19,19,18,8)
Conv3D_2	(3,3,3)	(1,1,1)	(0,0,0)	Out_2: (17,17,16,16)
Separable_Conv3D_3_1_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_3_1_1: (15,15,14,32)
Conv3D_3_1_2	(1,1,1)	(1,1,1)	(0,0,0)	Out_3_1_2: (15,15,14,16)
Separable_Conv3D_3_2_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_3_2_1: (15,15,14,32)
Conv3D_3_2_2	(1,1,1)	(1,1,1)	(0,0,0)	Out_3_2_2: (15,15,14,32)
Separable_Conv3D_3_2_3	(3,3,3)	(1,1,1)	(0,0,0)	Out_3_2_3: (13,13,12,32)
Conv3D_3_2_4	(1,1,1)	(1,1,1)	(1,1,1)	Out_3_2_4: (15,15,14,16)
Concatenate 1 (Out_3_1_2, Out_3_2_4)				Out_C_1: (15,15,14,32)
Attention1				Out_SE1: (15,15,14,32)
Residual Connection_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_R_1: (15,15,14,32)
Add (Out_C1, Out_R1)				Out_A1: (15,15,14,32)
Reshape (Out_A1)				Out_Re: (15,15,448)
Separable_Conv2D_4_1_1	(3,3)	(1,1)	(0,0)	Out_4_1_1: (13,13,64)
Conv2D__4_1_2	(1,1)	(1,1)	(0,0)	Out_4_1_2: (13,13,32)
Separable_Conv2D_4_2_1	(3,3)	(1,1)	(0,0)	Out_4_2_1: (13,13,64)
Conv2D_4_2_2	(1,1)	(1,1)	(0,0)	Out_4_2_2: (13,13,64)
Separable_Conv2D_4_2_3	(3,3)	(1,1)	(0,0)	Out_4_2_3: (11,11,64)
Conv2D_4_2_4	(1,1)	(1,1)	(1,1)	Out_4_2_4: (13,13,32)
Concatenate_2 (Out_4_1_2, Out_4_2_4)				Out_C_2: (13,13,64)
Attention2				Out_SE2: (13,13,64)
Residual Connection_2	(3,3)	(1,1)	(0,0)	Out_R_2: (13,13,64)
Add (Out_C_2, Out_R_2)				Out_A_2: (13,13,64)
Flatten				Out_F: (10,816)
Linear_1				Out_L_1: (256)
Dropou_1				Out_D_1: (256)
Linear_2				Out_L_2: (128)
Dropout_2				Out_D_2: (128)
Linear_3				Out_L_3: (7)
Total Parameters: 3,269,495				

Table 3. Framework structure of PU dataset.

Layer (Type)	Kernel Size	Stride	Padding	Output Shape
Input layer				Input: [(21,21,10,1)]
Conv3D_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_1: (19,19,8,8)
Conv3D_2	(3,3,3)	(1,1,1)	(0,0,0)	Out_2: (17,17,6,16)
Separable_Conv3D_3_1_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_3_1_1: (15,15,4,32)
Conv3D_3_1_2	(1,1,1)	(1,1,1)	(0,0,0)	Out_3_1_2: (15,15,4,16)
Separable_Conv3D_3_2_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_3_2_1: (15,15,4,32)
Conv3D_3_2_2	(1,1,1)	(1,1,1)	(0,0,0)	Out_3_2_2: (15,15,4,32)
Separable_Conv3D_3_2_3	(3,3,3)	(1,1,1)	(0,0,0)	Out_3_2_3: (13,13,2,32)
Conv3D_3_2_4	(1,1,1)	(1,1,1)	(1,1,1)	Out_3_2_4: (15,15,4,16)
Concatenate_1 (Out_3_1_2, Out_3_2_4)				Out_C_1: (15,15,4,32)
Residual Connection_1	(3,3,3)	(1,1,1)	(0,0,0)	Out_R_1: (15,15,4,32)
Add (Out_C1, Out_R1)				Out_A1: (15,15,4,32)
Reshape (Out_A1)				Out_Re: (15,15,128)
Separable_Conv2D_4_1_1	(3,3)	(1,1)	(0,0)	Out_4_1_1: (13,13,64)
Conv2D_4_1_2	(1,1)	(1,1)	(0,0)	Out_4_1_2: (13,13,32)
Separable_Conv2D_4_2_1	(3,3)	(1,1)	(0,0)	Out_4_2_1: (13,13,64)
Conv2D_4_2_2	(1,1)	(1,1)	(0,0)	Out_4_2_2: (13,13,64)
Separable_Conv2D_4_2_3	(3,3)	(1,1)	(0,0)	Out_4_2_3: (11,11,64)
Conv2D_4_2_4	(1,1)	(1,1)	(1,1)	Out_4_2_4: (13,13,32)
Concatenate_2 (Out_4_1_2, Out_4_2_4)				Out_C_2: (13,13,64)
Attention				Out_SE: (13,13,64)
Residual Connection_2	(3,3)	(1,1)	(0,0)	Out_R_2: (13,13,64)
Add (Out_C_2, Out_R_2)				Out_A_2: (13,13,64)
Flatten				Out_F: (10,816)
Linear_1				Out_L_1: (256)
Dropout_1				Out_D_1: (256)
Linear_2				Out_L_2: (128)
Dropout_2				Out_D_2: (128)
Linear_3				Out_L_3: (9)
Total Parameters: 3,059,193				

The proposed method is compared with various existing classification methods: an SVM [32], a 2D CNN [33], a 3D CNN [34], an SSRN [21], and HybridSN [23].

All experiments are performed on Python, and the experimental computing platform is configured with an Intel Core i7-12700k (Santa Clara, CA, USA), an Nvidia GeForce GTX 3070Ti (Santa Clara, CA, USA), and 16 GB of RAM (Microsoft Corporation, Redmond, WA, USA).

The kappa (κ) value, overall accuracy (OA), and average accuracy (AA) are used to evaluate the performance of the classification methods.

3.3. Parameter Analysis

The model parameter settings are analyzed. Based on a large number of experiments, several parameters that considerably affect the experimental results are selected for analysis: the size of the input patch; proportion of dropout probability values; and effectiveness of the morphological features, residual connections, and attention mechanism.

3.3.1. Effect of Patch Size on Accuracy

Patch sizes 17, 19, 21, 23, and 25 are compared experimentally. Excessively large patches increase the amount of computation and may cause overfitting, whereas too small patches reduce accuracy. Table 4 shows the performance of the different-sized patches on the three datasets. The highest accuracy is obtained at a patch size of 21.

Table 4. Effect of different-sized patches on accuracy.

Dataset		17 × 17	19 × 19	21 × 21	23 × 23	25 × 25
IP	OA	99.41 ± 0.19	99.54 ± 0.1	99.64 ± 0.08	99.59 ± 0.13	99.57 ± 0.16
	AA	99.07 ± 0.53	99.16 ± 0.36	99.26 ± 0.32	99.13 ± 0.33	99.16 ± 0.32
	K	99.32 ± 0.22	99.46 ± 0.11	99.59 ± 0.09	99.54 ± 0.15	99.56 ± 0.09
UP	OA	99.96 ± 0.02	99.96 ± 0.03	99.97 ± 0.02	99.95 ± 0.02	99.93 ± 0.04
	AA	99.91 ± 0.05	99.90 ± 0.05	99.92 ± 0.04	99.87 ± 0.03	99.84 ± 0.07
	K	99.94 ± 0.03	99.95 ± 0.04	99.96 ± 0.02	99.94 ± 0.02	99.91 ± 0.05
My Dataset	OA	99.83 ± 0.04	99.82 ± 0.03	99.84 ± 0.03	99.83 ± 0.05	99.81 ± 0.03
	AA	99.64 ± 0.08	99.67 ± 0.07	99.70 ± 0.05	99.70 ± 0.06	99.66 ± 0.05
	K	99.80 ± 0.05	99.78 ± 0.03	99.81 ± 0.04	99.79 ± 0.05	99.77 ± 0.02

3.3.2. Effect of Dropout Probability Values on Accuracy

A dropout layer prevents the model from relying too much on some local features by randomly discarding the parameter values of some neurons, thereby enhancing model robustness and preventing overfitting. Table 5 shows the effects of different dropout values on accuracy.

Table 5. Effect of different dropout probability values on accuracy.

Dataset		25%	30%	35%	40%	45%
IP	OA	99.58 ± 0.06	99.50 ± 0.08	99.64 ± 0.08	99.61 ± 0.12	99.60 ± 0.08
	AA	99.13 ± 0.43	99.18 ± 0.32	99.26 ± 0.32	99.11 ± 0.5	99.15 ± 0.36
	K	99.52 ± 0.07	99.43 ± 0.07	99.59 ± 0.09	99.56 ± 0.14	99.55 ± 0.08
UP	OA	99.84 ± 0.11	99.88 ± 0.11	99.97 ± 0.02	99.95 ± 0.02	99.97 ± 0.02
	AA	99.71 ± 0.23	99.81 ± 0.15	99.92 ± 0.04	99.90 ± 0.06	99.92 ± 0.05
	K	99.79 ± 0.15	99.85 ± 0.14	99.96 ± 0.02	99.94 ± 0.03	99.96 ± 0.02
My Dataset	OA	99.82 ± 0.02	99.82 ± 0.04	99.84 ± 0.03	99.83 ± 0.03	99.80 ± 0.08
	AA	99.64 ± 0.05	99.66 ± 0.07	99.70 ± 0.05	99.68 ± 0.05	99.62 ± 0.17
	K	99.77 ± 0.03	99.78 ± 0.05	99.81 ± 0.04	99.80 ± 0.03	99.75 ± 0.08

3.3.3. Effectiveness of MP

MP provides effective spatial features for the model. As seen in Table 6, the model without MP has inferior results compared with the morphologically functional model.

Table 6. Effectiveness of morphological processing (MP), residual connections, and attention mechanism.

Dataset		Proposed Model	No MP	No Res	No Attention
IP	OA	99.64 ± 0.08	99.57 ± 0.11	99.48 ± 0.13	99.51 ± 0.14
	AA	99.26 ± 0.32	99.11 ± 0.46	99.13 ± 0.43	99.12 ± 0.5
	K	99.59 ± 0.09	99.47 ± 0.13	99.40 ± 0.14	99.45 ± 0.15
UP	OA	99.97 ± 0.02	99.95 ± 0.02	99.95 ± 0.02	99.94 ± 0.07
	AA	99.92 ± 0.04	99.86 ± 0.06	99.89 ± 0.06	99.90 ± 0.06
	K	99.96 ± 0.02	99.93 ± 0.03	99.94 ± 0.03	99.92 ± 0.09
My Dataset	OA	99.84 ± 0.03	99.81 ± 0.03	99.79 ± 0.04	99.79 ± 0.11
	AA	99.70 ± 0.05	99.65 ± 0.06	99.56 ± 0.13	99.63 ± 0.08
	K	99.81 ± 0.04	99.78 ± 0.04	99.74 ± 0.06	99.74 ± 0.13

3.3.4. Effectiveness of Residual Connections

The residual connections can prevent network degradation, and their effectiveness is evident in Table 6.

3.3.5. Effectiveness of Attention Mechanism

Experiments without the use of the attention mechanism are conducted on the three datasets. The results, shown in Table 6, demonstrate the effectiveness of the attention mechanism.

3.4. Classification Results and Analysis

As shown in Figures 5–7, My Net achieves the highest classification accuracy across all three datasets, as reflected in the OA, AA, and κ metrics. In contrast, the SVM classification yields the lowest performance. Although the 2D CNN method performs better than the traditional SVM, it only extracts spatial features for each pixel. Consequently, it exhibits relatively poor performance when tasked with simultaneously extracting both spectral and spatial features.

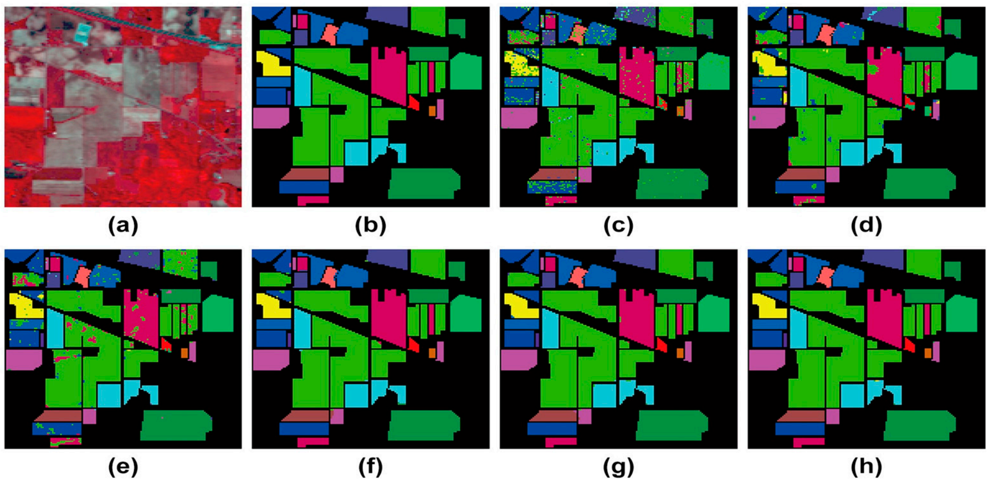

Figure 5. IP dataset classification results: (**a**) false-color image, (**b**) ground-truth image, (**c**) support vector machine (SVM), (**d**) two-dimensional convolutional neural network (2D CNN), (**e**) three-dimensional CNN (3D CNN), (**f**) spatial–spectral residual network (SSRN), (**g**) HybridSN, and (**h**) My Net.

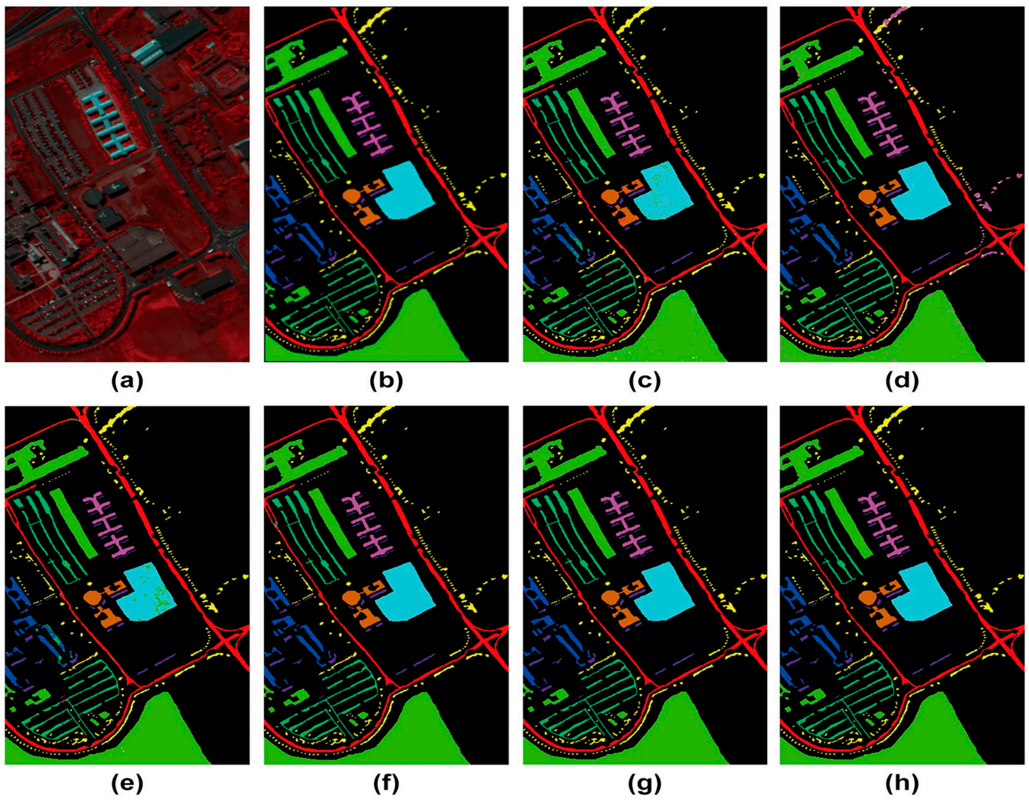

Figure 6. PU dataset classification results: (**a**) false-color image, (**b**) ground-truth image, (**c**) SVM, (**d**) 2D CNN, (**e**) 3D CNN, (**f**) SSRN, (**g**) HybridSN, and (**h**) My Net.

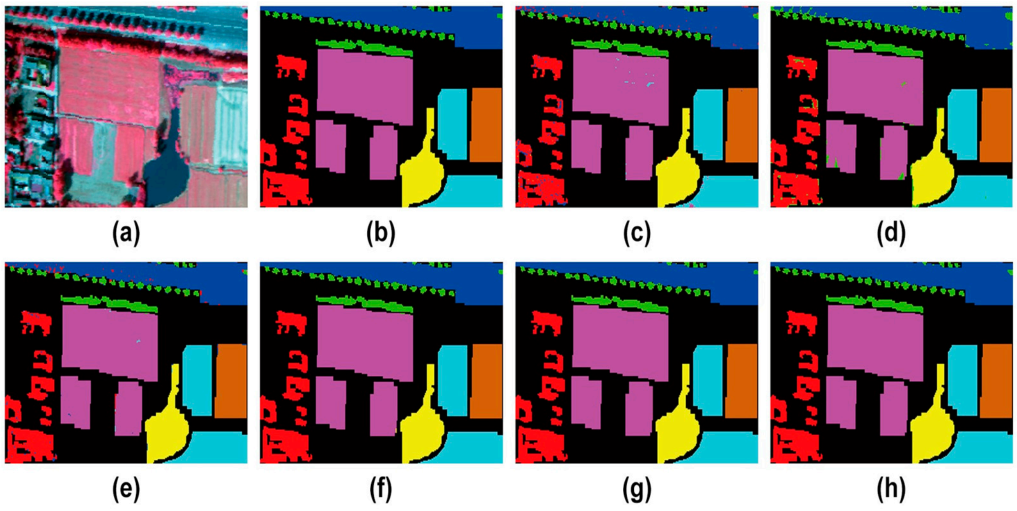

Figure 7. My Dataset classification results: (**a**) false-color image, (**b**) ground-truth image, (**c**) SVM, (**d**) 2D CNN, (**e**) 3D CNN, (**f**) SSRN, (**g**) HybridSN, and (**h**) My Net.

As presented in Tables 7–9, the 3D CNN model outperforms the 2D CNN model in terms of classification accuracy. However, the 3D CNN is computationally expensive and suffers from increased model complexity, leading to a higher risk of overfitting. To balance classification accuracy with computational efficiency, this study employs a multiscale convolutional layer consisting of two 3D convolution layers and a DSC layer. This configuration allows for the simultaneous extraction of spectral and spatial features while effectively reducing overfitting due to model complexity. Compared to the 3D CNN, the SSRN enhances accuracy by stacking multiple spectral residual blocks and spatial residual blocks for feature extraction.

Table 7. Classification accuracy results on IP dataset.

Class	SVM	2D CNN	3D CNN	SSRN	HybridSN	Proposed Model
1	83.21	76.3	100	100	100	100
2	74.83	82.5	77.92	98.79	99.30	99.50
3	81.05	86.9	91.25	100	99.83	99.83
4	78.72	63.54	91.84	98.96	100	98.81
5	74.65	89.63	98.92	99.20	99.11	100
6	92.5	99.03	97.99	99.31	99.80	100
7	95.31	77.41	100	100	100	100
8	84.7	100	96.97	100	100	100
9	96.83	65.31	100	100	100	100
10	72.04	81.93	80.6	99.36	99.85	99.85
11	77.51	90.65	86.44	99.80	99.59	99.88
12	85.6	84.25	90.74	98.54	98.33	99.04
13	84.61	99.36	97.62	94.25	98.59	100
14	97.54	98.68	97.64	99.12	99.66	99.89
15	93.61	88.29	94.44	97.77	99.26	98.18
16	73.4	99.63	100	100	95.31	100
OA	85.45 ± 2.43	90.36 ± 0.33	89.31 ± 0.38	99.20 ± 0.39	99.42 ± 0.36	99.64 ± 0.08
AA	78.33 ± 3.11	88.62 ± 0.92	91.68 ± 0.55	98.85 ± 0.54	98.93 ± 0.41	99.26 ± 0.32
κ	83.21 ± 2.51	89.92 ± 0.59	87.82 ± 0.53	99.1 ± 0.47	99.36 ± 0.35	99.59 ± 0.09

Table 8. Classification accuracy results on PU dataset.

Class	SVM	2D CNN	3D CNN	SSRN	HybridSN	Proposed Model
1	95.31	99.77	98.04	99.85	100	99.98
2	96.64	100	97.03	99.98	99.95	99.99
3	83.5	99.75	95.08	99.93	100	100
4	95.36	100	99.67	99.91	99.91	99.91
5	99.43	51.93	100	100	100	100
6	89.69	99.80	99.63	100	100	100
7	88.23	99.25	96.72	100	99.68	100
8	87.39	95.93	92.22	98.81	99.92	99.92
9	99.81	100	99.65	99.55	99.55	99.85
OA	95.01 ± 0.21	96.63 ± 0.21	97.27 ± 0.13	99.85 ± 0.09	99.92 ± 0.04	99.97 ± 0.02
AA	93.41 ± 0.53	95.56 ± 0.36	96.22 ± 1.23	99.76 ± 0.33	99.80 ± 0.14	99.92 ± 0.04
κ	92.86 ± 0.39	95.40 ± 0.51	96.36 ± 0.21	99.79 ± 0.15	99.90 ± 0.05	99.96 ± 0.02

Table 9. Classification precision results on My Dataset.

Class	SVM	2D CNN	3D CNN	SSRN	HybridSN	Proposed Model
1	79.38	100	95.48	100	100	100
2	97.41	68.23	99.26	97.65	98.54	99.37
3	95.43	99.89	97.38	99.59	99.42	99.45
4	99.10	100	99.84	99.79	100	100
5	95.59	100	99.92	100	99.98	100
6	95.36	100	99.24	100	100	100
7	99.5	100	100	100	100	100
OA	95.43 ± 0.55	96.92 ± 0.24	98.79 ± 0.21	99.78 ± 0.13	99.81 ± 0.04	99.84 ± 0.03
AA	93.21 ± 2.63	97.52 ± 0.45	98.50 ± 0.53	99.64 ± 0.28	99.67 ± 0.1	99.70 ± 0.05
κ	95.29 ± 0.63	97.52 ± 0.24	98.52 ± 0.43	99.72 ± 0.25	99.76 ± 0.07	99.81 ± 0.04

As seen in Table 10, the SVM, which has a simple structure, takes less time than the other classification methods, which use neural networks. Compared with the 3D CNN, the 2D CNN requires less time to train or test data. Unlike the 3D CNN, My Net uses both 3D and 2D convolutional layers to reduce model complexity. Compared with HybridSN, My Net needs a slightly shorter training time on IP and My Dataset but requires slightly more time on PU.

Table 10. Model training and testing times.

Dataset	Time	IP	PU	My Dataset
SVM	Train(s)	2.11	5.32	3.94
	Test(s)	1.03	2.31	1.84
2D CNN	Train(m)	1.12	1.31	1.53
	Test(s)	0.9	1.52	1.61
3D CNN	Train(m)	3.21	8.17	16.63
	Test(s)	8.51	13.04	30.26
SSRN	Train(m)	4.73	6.93	14.11
	Test(s)	5.89	11.36	25.13
HybridSN	Train(m)	2.98	3.65	7.43
	Test(s)	1.98	2.06	4.92
My Net	Train(m)	2.64	3.99	6.56
	Test(s)	1.23	1.63	2.91

4. Conclusions

In this paper, we propose a hyperspectral image classification method that combines convolutional neural networks (CNNs), depth-separable convolution (DSC), multiscale convolution, residual connections, attention mechanisms, and max pooling (MP). By employing multiscale feature extraction and splicing hyperspectral data after PCA dimensionality reduction, the model is capable of simultaneously extracting spectral and spatial features, thereby enhancing classification accuracy. The introduced attention mechanism effectively models the interdependencies between different channels, further improving network performance. The use of residual connections and dropout layers effectively mitigates the issues of gradient vanishing and overfitting, thereby enhancing the model's generalization ability.

The experimental results on three real datasets demonstrate that the proposed method achieves excellent classification accuracy. However, the model exhibits high computational complexity, and there remains a risk of overfitting in certain cases, particularly with small sample datasets. Future research could focus on improving the model's performance and applicability by simplifying the network architecture, optimizing computational efficiency, and expanding to additional datasets.

Author Contributions: Conceptualization, T.R. and G.S.; methodology, T.R.; software, Z.Z.; validation, T.R., G.S. and Z.Z.; formal analysis, T.R.; investigation, Y.P.; resources, Z.Z.; data curation, Z.Z.; writing—original draft preparation, T.R.; writing—review and editing, G.S.; visualization, H.Z.; supervision, G.S.; project administration, T.R.; funding acquisition, T.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Science and Technology Development Key Projects of Jilin Province, grant number 20230201107GX.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Acknowledgments: The authors would like to thank Key Laboratory of Micro-Nano and Ultra-Precision Manufacturing of Jilin Province (NO. 20140622008JC) for providing helpful experimental equipment related to this work.

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

References

1. Saidi, S.; Idbraim, S.; Karmoude, Y.; Masse, A.; Arbelo, M. Deep-Learning for Change Detection Using Multi-Modal Fusion of Remote Sensing Images: A Review. *Remote Sens.* **2024**, *16*, 3852. [CrossRef]
2. Liu, B.; Li, T. A Machine-Learning-Based Framework for Retrieving Water Quality Parameters in Urban Rivers Using UAV Hyperspectral Images. *Remote Sens.* **2024**, *16*, 905. [CrossRef]
3. Tang, L.; Werner, T.T. Global mining footprint mapped from high-resolution satellite imagery. *Commun. Earth Environ.* **2023**, *4*, 134. [CrossRef]
4. Wang, C.; Liu, B.; Liu, L.; Zhu, Y.; Hou, J.; Liu, P.; Li, X. A review of deep learning used in the hyperspectral image analysis for agriculture. *Artif. Intell. Rev.* **2021**, *54*, 5205–5253. [CrossRef]
5. Cannaday, A.B.; Davis, C.H.; Bajkowski, T.M. Detection of Camouflage-Covered Military Objects Using High-Resolution Multi-Spectral Satellite Imagery. In Proceedings of the IGARSS 2023—2023 IEEE International Geoscience and Remote Sensing Symposium, Pasadena, CA, USA, 16–21 July 2023; IEEE: New York, NY, USA, 2023; pp. 5766–5769.
6. Belgiu, M.; Drăguţ, L. Random forest in remote sensing: A review of applications and future directions. *ISPRS J. Photogramm.* **2016**, *114*, 24–31. [CrossRef]
7. Amrani, M.; Chaib, S.; Omara, I.; Jiang, F. Bag-of-visual-words based feature extraction for SAR target classification. In *Ninth International Conference on Digital Image Processing (ICDIP 2017), Hong Kong, China, 19–22 May 2017*; SPIE: Bellingham, WA, USA, 2017; Volume 10420.
8. Liu, Q.; Liu, C. A novel locally linear KNN method with applications to visual recognition. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *28*, 2010–2021. [CrossRef]
9. Saha, D.; Manickavasagan, A. Machine learning techniques for analysis of hyperspectral images to determine quality of food products: A review. *Curr. Res. Food Sci.* **2021**, *4*, 28–44. [CrossRef]
10. Liu, B.; Guo, W.; Chen, X.; Gao, K.; Zuo, X.; Wang, R.; Yu, A. Morphological attribute profile cube and deep random forest for small sample classification of hyperspectral image. *IEEE Access* **2020**, *8*, 117096–117108. [CrossRef]
11. Pan, H.; Liu, M.; Ge, H.; Chen, S. Semi-supervised spatial–spectral classification for hyperspectral image based on three-dimensional Gabor and co-selection self-training. *J. Appl. Remote Sens.* **2022**, *16*, 028501. [CrossRef]
12. Kang, X.; Duan, P.; Li, S. Hyperspectral image visualization with edge-preserving filtering and principal component analysis. *Inf. Fusion.* **2020**, *57*, 130–143. [CrossRef]

13. Kumar, V.; Singh, R.S.; Dua, Y. Morphologically dilated convolutional neural network for hyperspectral image classification. *Signal Process Image Commun.* **2022**, *101*, 116549. [CrossRef]
14. Li, Q.; Wang, Q.; Li, X. Exploring the relationship between 2D/3D convolution for hyperspectral image super-resolution. *IEEE Trans. Geosci. Remote Sens.* **2021**, *59*, 8693–8703. [CrossRef]
15. Akodad, S.; Bombrun, L.; Xia, J.; Berthoumieu, Y.; Germain, C. Ensemble learning approaches based on covariance pooling of CNN features for high resolution remote sensing scene classification. *Remote Sens.* **2020**, *12*, 3292. [CrossRef]
16. Ghamisi, P.; Plaza, J.; Chen, Y.; Li, J.; Plaza, A.J. Advanced spectral classifiers for hyperspectral images: A review. *IEEE Geosci. Remote Sens. Mag.* **2017**, *5*, 8–32. [CrossRef]
17. Yu, S.; Jia, S.; Xu, C. Convolutional neural networks for hyperspectral image classification. *Neurocomputing* **2017**, *219*, 88–98. [CrossRef]
18. Vaddi, R.; Manoharan, P. Hyperspectral image classification using CNN with spectral and spatial features integration. *Infrared Phys. Technol.* **2020**, *107*, 103296. [CrossRef]
19. Ahmad, M.; Khan, A.M.; Mazzara, M.; Distefano, S.; Ali, M.; Sarfraz, M.S. A fast and compact 3-D CNN for hyperspectral image classification. *IEEE Geosci. Remote Sens. Lett.* **2020**, *19*, 1–5. [CrossRef]
20. Yang, J.; Zhao, Y.Q.; Chan, C.W. Learning and transferring deep joint spectral-spatial features for hyperspectral classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 4729–4742. [CrossRef]
21. 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]
22. 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]
23. 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]
24. Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Wu, E. Squeeze-and-excitation networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 2011–2023. [CrossRef] [PubMed]
25. Sergio, R.; Angelo, C.; Valeria, L.; Giuseppina, U. Machine-learning based vulnerability analysis of existing buildings. *Autom. Constr.* **2021**, *132*, 103936.
26. Yuan, Y.; Jin, M. Multi-type spectral spatial feature for hyperspectral image classification. *Neurocomputing* **2022**, *492*, 637–650. [CrossRef]
27. Serra, J. *Image Analysis and Mathematical Morphology*; Academic Press: Cambridge, UK, 1982.
28. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
29. Lin, C.; Wang, T.; Dong, S.; Zhang, Q.; Yang, Z.; Gao, F. Hybrid convolutional network combining 3D depthwise separable convolution and receptive field control for hyperspectral image classification. *Electronics* **2022**, *11*, 3992. [CrossRef]
30. Ghaderizadeh, S.; Abbasi-Moghadam, D.; Sharifi, A.; Tariq, A.; Qin, S. Multiscale dual-branch residual spectral–spatial network with attention for hyperspectral image classification. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 5455–5467. [CrossRef]
31. Yang, J.; Du, B.; Zhang, L. From center to surrounding: An interactive learning framework for hyperspectral image classification. *ISPRS J. Photogramm.* **2023**, *197*, 145–166. [CrossRef]
32. Melgani, F.; Bruzzone, L. Classification of hyperspectral remote sensing images with support vector machines. *IEEE Trans. Geosci. Remote Sens.* **2004**, *42*, 1778–1790. [CrossRef]
33. Makantasis, K.; Karantzalos, K.; Doulamis, A.; Doulamis, N. Deep supervised learning for hyperspectral data classification through convolutional neural networks. In Proceedings of the 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, Italy, 26–31 July 2015; IEEE Publications: New York, NY, USA.
34. Hamida, A.B.; Benoit, A.; Lambert, P.; Amar, C.B. 3-D deep learning approach for remote sensing image classification. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 4420–4434. [CrossRef]